Navigation Models Used for OKSI's Project Involving Crossbow AHRS, an EKF, and Video Position Fixes*

Thomas H. Kerr III
TeK Associates

Abstract

The requisite Navigation Models are documented herein for OKSI's project involving **Crossbow** AHRS400CD-200 and an Extended Kalman Filter (which TeK Associates will provide in MatLab) with Navaid Position Fixes from OKSI's Vision Video approach. This documentation exercise was performed so that coding in MatLab can be expedited and easily cross-checked when everything that is needed is explicitly spelled out. This documentation is also convenient to avail OKSI with our models slightly beforehand for their scrutiny and approval (or for OKSI to intervene and redirect us, if necessary).

Keywords: Earth Model, Navigation Equations, Nav EKF Models

1 Introduction

Earth Models (from [4]) are reviewed in Sec. 2. More Earth models used in the Navigation Calculations to be performed on AHRS sensors to convert the results into an INS (from [4] [1]) are presented in Sec. 3. Mathematical Models (from [8]) for two different Extended Kalman Filter (EKF) options (settling on use of only the full state) are summarized in Sec. 4, including appropriate one-sigma parameter values, as specified in [3]. Our innovative way to check for sufficient proximity of the resulting ground track (obtained from our processing) to GPS, as a gauge of truth, is in Sec. 5, as well as a standard view sans covariances. Details of appropriately converting to a discrete-time formulation in Tek's software implementation are provided in Sec. 6.

2 Earth Models

Plumb bob gravity Local gravity or "Plumb bob" gravity (i.e., the sum of acceleration due to inverse square mass attraction and centripetal acceleration, known as the *Coriolis* acceleration, due to the rotation of the earth):

$$\mathbf{g}_l = \mathbf{g} - \omega_{ie} \times [\omega_{ie} \times \mathbf{r}] \approx \mathbf{g} - \frac{\Omega^2 (R_0 + h)}{2} \begin{bmatrix} \sin 2L \\ 0 \\ 1 + \cos 2L \end{bmatrix}, \tag{1}$$

where, in the above, the Latitude, L, and altitude, h, are functions of time (and in all that follows below). The fixed values of Ω_{ie}, R_0, and $\mathbf{g}$ are provided further below as well.

3 Navigation Calculations

3.1 Direction Cosine Matrix (DCM) in terms of Euler Angles:

$$\mathbf{C}_b^n = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} = \begin{bmatrix} \cos(P)\cos(\psi) & -cos(R)\sin(\psi) + \sin(R)\sin(P)\cos(\psi) & \sin(R)\sin(\psi) + \cos(R)\sin(P)\cos(\psi) \\ \cos(P)\sin(\psi) & cos(R)\cos(\psi) + \sin(R)\sin(P)\sin(\psi) & -\sin(R)\cos(\psi) + \cos(R)\sin(P)\sin(\psi) \\ -\sin(P) & \sin(R)\cos(P) & \cos(R)\cos(P) \end{bmatrix}, \tag{2}$$

where P, R, and ψ are (instantaneous) pitch, roll, and yaw, respectively consistent with [5, p. 8], and are the physical angles that would be measured by angular pick-offs between a set of three gimbals in a stable INS platform. (We chose to use DCM herein rather than the alternative quaternion option for conveying attitude. Both options have benefits and drawbacks [7].)

*TeK Associates, 9 Meriam St., Suite 7-R, Lexington, MA 02420, Contract No. 2012-1, funded by OptoKnowledge Systems, Inc., 19805 Hamilton Ave., Torrance, CA 90502.

[1] While the purpose of [4] is to derive results for various options, our purpose here is merely to document the final results for the particular option that we selected to follow (of the many options offered in [4]). The convention that we use here is always spelled out where it is introduced since [5] unsuccessfully tries to "be all things to all people" yet ignores the landmark clarifications made in [6] between ϕ, θ, and ψ (e.g., gyro misalignment angles in inertial-frame-to-computer-frame: ψ) for INS navigation that predates the notation conveyed in [5], where, understandably, standard notation previously arose in various contributory fields as they developed using the very same symbols to mean different things, depending on the field. As a further reason for me originally refraining from falling in line (but being more compliant now), notice that [5, Abbreviated Terms, p. 7] fails to associate **EO** with Electro-Optical and, instead, says less obvious "Exterior Orientation"! Another example is "g" on p. 8 not being gravity and "GM" is a symbol but not as arises in $g = GMm/R^2$. No "A" for Angstroms, nor "nm" for nanometers (nor nautical miles), nor "c" for speed-of-light. Risk is likely alienation of experts whose conventions are overlooked in [5]. For more, please see 2^{nd} footnote on page 10 herein.

3.2 Euler Angles in terms of Direction Cosine Matrix entries:

$$\begin{aligned} \text{Roll} : R &= \arctan\left[\frac{c_{32}}{c_{33}}\right], \\ \text{Pitch} : P &= \arctan\left[-c_{31}\right], \\ \text{Yaw} : \psi &= \arctan\left[\frac{c_{21}}{c_{11}}\right]. \end{aligned} \tag{3}$$

3.3 The Transport Rate (i.e., the turn rate of the local geographic frame with respect to [wrt] Earth-fixed frame):

$$\omega_{en}^{n} = \begin{bmatrix} \frac{v_E}{(R_0+h)} \\ -\frac{v_N}{(R_0+h)} \\ -\frac{v_E \tan L}{(R_0+h)} \end{bmatrix}, \tag{4}$$

where v_E and v_N are the East and North (instantaneous) components of velocity, respectively, and R_0 is mean radius of the Earth.

The Turn rate of the Earth (wrt the local geographic [2] frame) [3]:

$$\omega_{ie}^{n} = \begin{bmatrix} \Omega \cos L \\ 0 \\ -\Omega \sin L \end{bmatrix}, \tag{5}$$

where, in the above,

$$\Omega \equiv 7.292115 \times 10^{-5} \text{ radians/sec} = 15.041067^{\text{o}}/\text{hour}, \tag{6}$$

$$R_0 \equiv 6378137.0 \text{ meters}, \tag{7}$$

are the rotation rate and radius of the earth, respectively. TeK Associates briefly considered using only a Spherical Earth approximation for simplicity but instead opted to use the more detailed WGS 84 oblate spheroid model with median radius:

$$R_M = \frac{a \cdot \left(1-\epsilon^2\right)}{\left(1-\epsilon^2 \sin^2 L\right)^{\frac{3}{2}}}, \tag{8}$$

and normal radius:

$$R_N = \frac{a}{\left(1-\epsilon^2 \sin^2 L\right)^{\frac{1}{2}}}, \tag{9}$$

where $\epsilon = 0.08181919$ and $a \equiv R_0$ [11, pp. 7-10]. Ref. [8, Sec. 2.3.2.1] emphasizes that full precision must be used for these constants. A useful and well-respected approximation for the acceleration of gravity at a particular altitude [11, p. 57] is:

$$\mathbf{g}(h) = \frac{g(0)}{\left(1+h/R_0\right)^2}, \tag{10}$$

$$g(0) = 9.780318\left(1 + 5.3024 \times 10^{-3} \sin^2 L - 5.9 \times 10^{-6} \sin^2 2L\right) \text{ meters/sec}^2. \tag{11}$$

The capital letters N, E, D correspond to a right handed coordinate system (North, East, Down) erected at the center of gravity (c.g.) of the aircraft, where the AHRS/INS is assumed to be located (as assumed in [2]). The aircraft Euler Angles (of Pitch, Roll, Yaw, respectively) nominally align with N, E, D in straight and level flight when the aircraft is flying due North and at constant altitude. The altitude or height above the Earth geoid is denoted as h, where $h = 0$ corresponds to sea level.

To complete the equations that must be simultaneously solved to provide INS navigation (in Fig. 2), the relevant direction cosine matrix evolves in time as the solution to:

$$\frac{d\mathbf{C}_n^b}{dt} = \mathbf{C}_n^b \Omega_{nb}^b, \tag{12}$$

and

$$\omega_{nb}^b = \omega_{ib}^b - \mathbf{C}_n^b \left[\omega_{ie}^n + \omega_{en}^n\right]. \tag{13}$$

[2] The **Geographic** frame is also frequently called the **Geodetic** frame [11, p. 19].

[3] There is an alternative formulation of Eq. 11 herein as found in [11, Eq. 1.23, p. 10] (involving a square root but likely merely a result of substituting trigonometric identities) and attributed to the Defense Mapping Agency WGS 84. We, likewise, also ignore the mass of the atmosphere herein in determining the effective gravity.

3.4 Ground speed expressed in navigation coordinates:

$$\frac{d\mathbf{v}_e^n}{dt} = \mathbf{f}^n - [2\omega_{ie}^n + \omega_{en}^n] \times \mathbf{v}_e^n + \mathbf{g}_l^n, \tag{14}$$

where, in the above, the first term on the right is the specific force acting on the vehicle, as measured by a triad of single input perfectly orthogonal accelerometers mounted at the center of gravity within the vehicle and the second long term on the right is the appropriate correction for the acceleration caused by the vehicle's velocity over the surface of a rotating Earth (denoted as the "*Coriolis*" acceleration [see Fig. 3]).

Component-wise, North, East, and Down velocity in the local geographic reference frame wrt Earth are solved by integrating the following wrt time:

$$\frac{dv_N}{dt} = f_N - 2\Omega \cdot v_E \cdot \sin(Lat) + \frac{\left(v_N \cdot v_D - v_E^2 \tan(Lat)\right)}{R_0 + h} + \xi \cdot g, \tag{15}$$

$$\frac{dv_E}{dt} = f_E + 2\Omega \left(v_N \cdot \sin(Lat) + v_D \cdot \cos(Lat)\right) + \frac{v_E}{R_0 + h}\left(v_D + v_N \cdot \tan(Lat)\right) - \eta \cdot g, \tag{16}$$

$$\frac{dv_D}{dt} = f_D - 2\Omega \cdot v_E \cdot \cos(Lat) - \frac{\left(v_E^2 + v_N^2\right)}{R_0 + h} + g, \tag{17}$$

where ξ and η, in the above, are angular deflections in the direction of the local gravity vector with respect to the local vertical due to gravity anomalies. Comparing Eqs. 15 to 17 to Fig. 2, notice that the velocities in the three orthogonal directions N, E, D are obtained by performing the indicated integration of each and that the associated N, E, D positions are obtained by an additional integration of each component. Ref. [9, Sec. 4.3.6] shows that Simpson's Rule (a 3$^{\text{rd}}$order method) should be used for the integration. Since we do not explicitly know the small angular deflections which are location dependent, take $\xi = \eta = 0$.

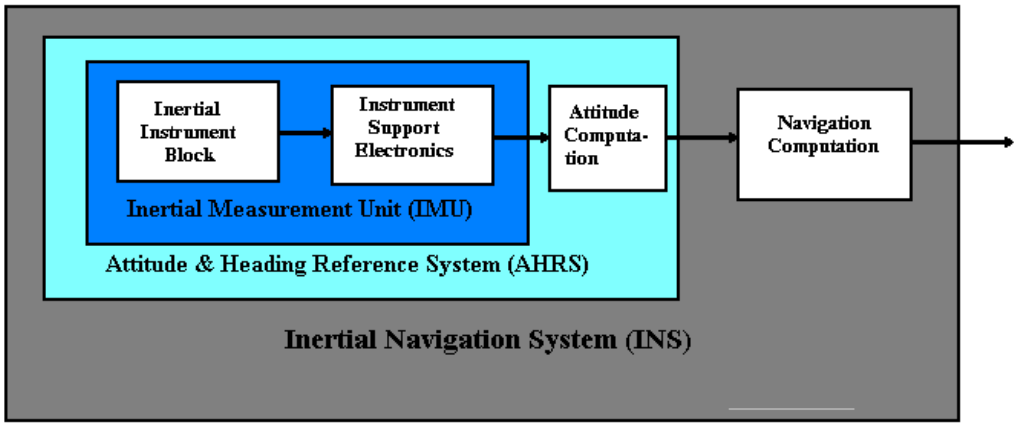
Latitude ($Lat \equiv \phi$), Longitude ($Long \equiv \lambda$), and Height (h) above the surface of the Earth are solved by integrating the following wrt time:

$$\frac{d\phi}{dt} = \frac{dLat}{dt} = \frac{v_N}{(R_0 + h)}, \tag{18}$$

$$\frac{d\lambda}{dt} = \frac{dLong}{dt} = \frac{v_E \cdot \sec(Lat)}{(R_0 + h)} = \frac{v_E \cdot \sec(\phi)}{(R_0 + h)}, \tag{19}$$

$$\frac{dh}{dt} = -v_D. \tag{20}$$

The additional superstructure addressed here is necessary to proceed from being a mere AHRS to being an INS using the same measured inputs from its gyros and accelerometers (a perspective that is provided in Fig. 1).



Strapdown Inertial Navigation System Building Blocks [4, Fig. 9.1, p. 264]

Figure 1: Block diagram signifying how to proceed from being a mere AHRS to obtain an INS

Increasing visual detail (in overview block diagram form) into exactly what must be explicitly provided (by TeK Associates within this MatLab software in order to proceed as we have indicated) is conveyed in Figs. 2 to 3 below.

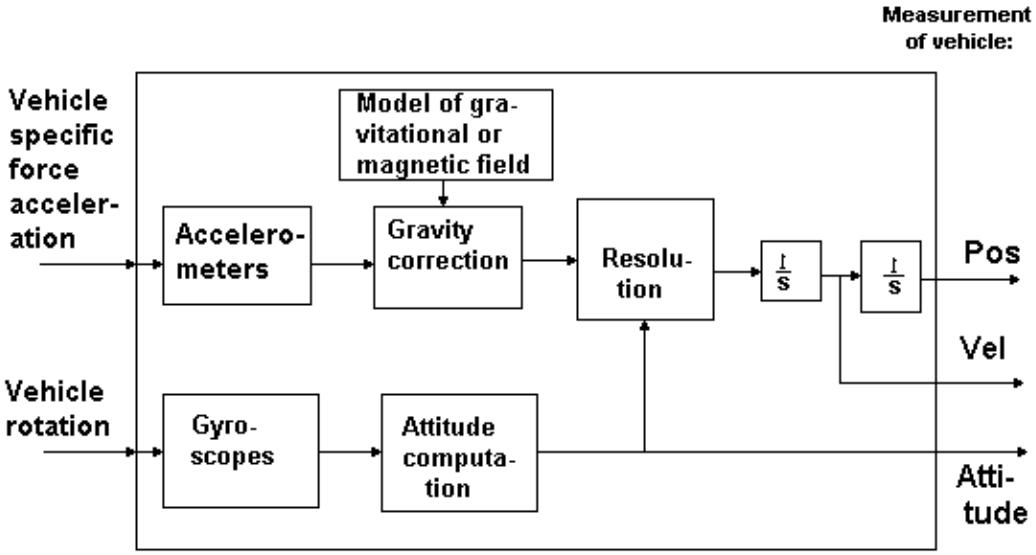


Figure 2: Simplified view of what TeK Associates is to provide, corresponding to the right most box of Fig. 1

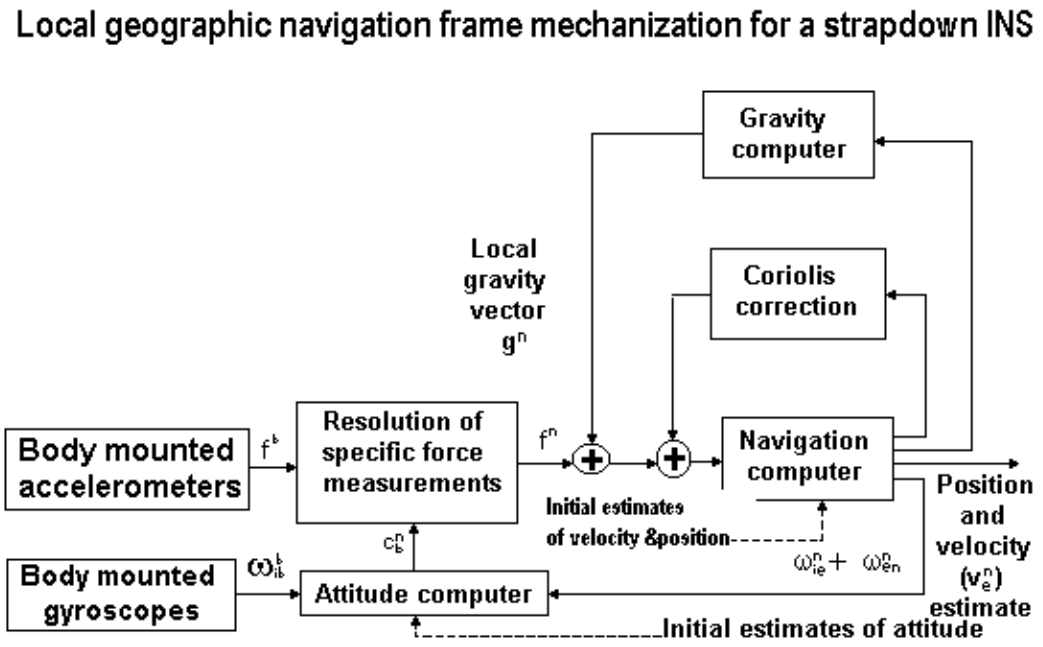


Figure 3: Detailed end result that TeK Associates will implement to emulate an INS

Once we have an INS [4] (admittedly, not a very accurate INS but an INS nonetheless), it is easier for us to specify an appropriate Kalman Filter to match this situation. The Kalman filter specified in [1, Chapt. 3] was for an INS but used the most general expression for inverse squared gravity, which is likely overkill for our UAV application. Ref. [1] also had inconsistent notation for Longitude between its Fig. 2-1, Eq. 2.5, and Fig. 2-2. A magnetometer is present in OKSI's application but not in [1]. We will return to this aspect in more detail in Sec. 4.

4 Mathematical Models for Two Extended Kalman Filter Options

The mathematical model of Gregory Andrews' 2008 Masters Thesis at Northeastern University ECE Department, with the states defined as in [1, Eq. 3.14] has a

- 1^{st} block being 3 components of position (in whatever frame but eventually to be converted to roll, pitch, and heading angle (yaw) to be compatible with the ARHS400 block diagram in lower right hand side Figure on 1^{st} page of **Crossbow** high level description [3]),
- 2^{nd} block being 3 components of velocity,
- 3^{rd} block being angular rate (as determined from the gyros),
- 4^{th} block being 3 components of acceleration (i.e., 3 bias states, if scale factor error is included, that would add 3 additional states),
- 5^{th} block being 3 components of gravity and maybe 3 more states for accelerometer biases (or, alternatively, as a magnetic field, which is handled entirely as an associated observation matrix [8, Sec. 10.5.1.3], [4, 107-112]),

for a total of at most 15 states (plus an additional 3 bias states for gyros and an additional 3 bias states for accelerometers) represented here. Use of 21 states is not too unwieldy but these particular states are not totally compatible with the actual AHRS400CD-200 that is used in our application.

We needed some of the additional states discussed above since Crossbow accelerometers have biases and scale factor errors too but gyro biases are believed to be more dominant in their adverse effect and are therefore sought to be identified by explicit modeling for compensation (and perhaps other constituent structures may need to be modeled in the plant or process dynamics), as a convention that is depicted in the TASC textbook [12, p. 168, p. 289, Figs. 3.8-3 & -4] and in [13, Chap. 6] and in [8, Sec. 11.6.2]. This particular 18 to 21 state model above, as prescribed in [1, Eq. 3.14], appears to be very reasonable for this problem if the corresponding system were in fact an INS instead of the current AHRS, which uses a magnetometer to stabilize or damp the effective vertical channel (that, otherwise, would have errors that grow without bound if no damping were present [15]). TeK Associates assumes that magnetometer measurements have already been handled adequately internal to the Crossbow AHRS, as we had also assumed for the proper handling of temperature compensation.

Overview Computational Impact of Number of States Used: for any vector state variable model, the computational burden of a real-time Kalman Filter goes as the cube of the total state size. The storage requirements for the real-time error model (i.e., the matrices that describe the gyro, accelerometer, and magnetometer dynamics) goes as the square of the state size used.

We originally thought that we needed to know more about the magnetometer internal operation to understand how to properly model the dynamic description to appear in the system matrix. For this aspect, OKSI provided TeK Associate with [14] in a timely manner.

TeK Associates assumes that the whole motivation for using the vision updates is to seek viable alternatives when the environment is GPS-denied due to presence of jammers or other interference (e.g., HEMP). In the last 10 years, so many platforms rely on low accuracy INS assuming that GPS will always be there to aid it at a high rate. That is not the case for a more realistic application scenario and a more sophisticated enemy, as I have warned the Navigation community about since 1997 [21], [22]. (However, one can use GPS in peace-time to gauge just how well this approach will work, as is being done for this project.)

http://www.helimx.com/article/layman%E2%80%99s-guide-attitude-heading-reference-systems-ahrs

http://en.wikipedia.org/wiki/Attitude_and_heading_reference_system

Excellent Block Diagram representations of an appropriate Extended Kalman Filter structure and alternative approximate simplifications as a linearized Kalman Filter are portrayed in [11, pp. 195-197] [5] roughly corresponding to Figs. 2 and 3 herein; however, we will only show the final equivalent state variable error representations below in terms of the underlying matrices corresponding to the associated Kalman filter that we selected to use (from [8, Sec. 11.5, pages 396-400]).

As indicated in our earlier e-mail, the book that appears to be most useful to us for the Kalman Filter development aspect of this project is [8] (2008). It also specifically considers AHRS on [8, page 353], where it devotes an entire chapter to this topic including a simplified block diagram in [8, Fig. 10.1], Kinematic Model [8, Sec 10.1], Sensor models [8, Sec. 10.2], AHRS Mechanization Equations [8, Sec. 10.4], Error Models [8, Sec. 10.5], Magnetometer Analysis [8, Sec. 10.5.1.3]. I originally leaned toward use of the simplified reduced state error model of [8, Eq. 11.89] (with logical rationale provided ahead of it by saying

[4] By documenting in this systematic manner, TeK Associates gained insight into other information that we need from OKSI in order to proceed, namely, initial velocity, position, and orientation [11, Sec. VII, p. 7], that OKSI had already conveyed to TeK within the Excel spreadsheet information.

[5] Conveyed separately as an e-mail pdf attachment transmitted on 29 Nov. 2012.

that he will ignore the vertical channel since coupling with it is slight and vertical channel must have compensation otherwise it would go unstable; so, instead, assume that it has already been properly attended to). Resulting state variable error model is only 7 states, as enumerated on [8, p. 400]. The author of Ref. [8] received the Engineering Vice President's Best Technical Publication Award in 1990 while he was at Draper Laboratory (1989-1994). (From an e-mail correspondence in March 2013, I learned that he was largely self-taught in the navigation area.) We upgrade to a more general higher dimensional model (further below) so that we may handle maneuvers without worry about effects of aircraft banking affecting and again cross-correlating vertical and horizontal channels (since we will handle both simultaneously instead of separately).

Please consider the standard state variable representation of the error states associated with an INS, which is of the following continuous-time form:

$$\delta\dot{x}(t) = f(x,t)\delta x(t) + G(x,t)w(t)\,, \tag{21}$$

where $w(t)$ is a zero mean White Gaussian Noise (WGN) random process, with statistics $N(0,Q)$ that will be specified and Gaussian random vector initial condition $N(0,P_0)$. In what follows, we will reflect the continuous-time version of this error model representing the inherent dynamics of an INS. The specifics of the required conversion to the discrete-time formulation for actual software implementation is addressed in Sec. 6.

For a magnetometer [8, Sec. 10.5.1.3], the associated measurement matrix is of the form (also denoted as H_2 in our software):

$$H_m = [-[\mathbf{m}^n\times]\,,\mathbf{0},\mathbf{0}]\,, \tag{22}$$

where, in the above, we used the following standard **summarizing cross product notation** for a vector of the form $\omega = [\omega_1,\omega_2,\omega_3]^{\mathrm{T}}$ in terms of a skew-symmetric matrix as

$$\omega\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{23}$$

Please notice that the definition of this important cross product notation has a typo in [1, Eq. 29]. Part of this documentation exercise herein is to spot and correct typos that would otherwise hurt us if committed to code. When position is not available, the magnetometer measurement matrix reduces to (as also denoted as H_2 in our software):

$$H_m = [\mathrm{diag}\,(0,0,m_e)\,,\mathbf{0},\mathbf{0}]\,, \tag{24}$$

(compare to [14, Eq. 11]), which contradicts the above by using a different structure for the position components and an Identity Matrix [14, Eq. 12] as the observation matrix for the camera[6]. Ref. [8] derives his results from first principles (but still omits the diag that appears in Eq. 24 above). Ref.[14, Eq. 11] sometimes obtains results from things that come afterwards. I am not sure of the status of [14] as a peer-reviewed publication to be relied upon.) Why some form of vertical damping is needed for an INS to prevent the error in the vertical channel from going unbounded and a new alternative technique for doing so is discussed in [15]. Use of a magnetometer is a more recent way of doing so. The corresponding measurement noise covariance matrix for the magnetometer within the particular AHRS being utilized was offered by OKSI and speculated to be of the proper order of magnitude (from [50] since it was assumed by OKSI that the manufacturer likely used one of their own manufactured magnetometers; however, which particular magnetometer was not identified but OKSI estimates upper bounds of about 2 nT/rt-Hz) therefore:

$$R_m = \begin{bmatrix} (2.0\mathrm{nt/rt-Hz})^2 & 0 & 0 \\ 0 & (2\mathrm{nt/rt-Hz})^2 & 0 \\ 0 & 0 & (2\mathrm{nt/rt-Hz})^2 \end{bmatrix} = \begin{bmatrix} 4.0(\mathrm{nt/rt-Hz})^2 & 0 & 0 \\ 0 & 4.0(\mathrm{nt/rt-Hz})^2 & 0 \\ 0 & 0 & 4.0(\mathrm{nt/rt-Hz})^2 \end{bmatrix}. \tag{25}$$

An appealing argument [7] is offered on [8, p. 400] to justify their extremely low dimensional reduced-order 7-state INS error model (ϕ being Latitude and λ being Longitude and ϵ being gyro drift-rates below):

$$\begin{aligned} \delta\mathbf{x} &= [\delta\phi,\delta\lambda,\delta v_N,\delta v_E,\epsilon_N,\epsilon_E,\epsilon_D]^{\mathbf{T}}\,, \\ &= [\mathrm{deg},\mathrm{deg},m/sec,m/sec,\mathrm{deg}/hr,\mathrm{deg}/hr,\mathrm{deg}/hr]^{\mathbf{T}}\ (\text{corresponding units}), \end{aligned} \tag{26}$$

[6] TeK Associates needs an associated effective measurement covarance matrix from OKSI for the imaging measurement as well as an effective measurement covaiance matrix R for the magnetometer measurement in Eqs. 22 and 24, above. The 1^{st} request is met in Eq. 56 while the 2^{nd} request can be ignored by assuming that everything about it is already adequately handled by AHRS. If that assumption is not true, then we may have a problem by not having this aspect completely defined.

[7] Argument is that the vertical channel is definitely unstable unless it is properly compensated so assume that it has been properly handled, then vertical channel can be ignored as having little effect on the remainder which is only slightly coupled (back into the vertical channel). Banking aircraft maneuvers again cross-couples both channels too much to ignore.

with associated continuous-time state dynamics matrix:

$$F = \begin{bmatrix} 0 & 0 & \frac{1}{R_0} & 0 & 0 & 0 & 0 \\ \frac{-\rho_D}{\cos\phi} & 0 & 0 & \frac{1}{R_0 \cos\phi} & 0 & 0 & 0 \\ F_{41} & 0 & k_D & 2\omega_D & 0 & f_D & -f_E \\ F_{51} & 0 & F_{54} & F_{55} & -f_D & 0 & f_N \\ -\Omega_D & 0 & 0 & \frac{-1}{R_0} & 0 & \omega_D & -\omega_E \\ 0 & 0 & \frac{1}{R_0} & 0 & -\omega_D & 0 & \omega_N \\ F_{91} & 0 & 0 & \frac{\tan\phi}{R_0} & \omega_E & -\omega_N & 0 \end{bmatrix}, \tag{27}$$

that we tie into here as the 10 state Kalman filter model consisting of the following states:

$$\begin{aligned} \delta\mathbf{x} &= [\delta\phi, \delta\lambda, \delta v_N, \delta v_E, \epsilon_N, \epsilon_E, \epsilon_D, \text{gyro random walk}_N, \text{gyro random walk}_E, \text{gyro random walk}_D]^{\mathbf{T}}, \\ &= [\text{deg}, \text{deg}, m/sec, m/sec, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr]^{\mathbf{T}} \text{ (corresponding units)}, \end{aligned} \tag{28}$$

with associated continuous-time state dynamics matrix:

$$F = \left[\begin{array}{cc:c:cc:ccc:ccc} 0 & 0 & \vdots & \frac{1}{R_0} & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \frac{-\rho_D}{\cos\phi} & 0 & \vdots & 0 & \frac{1}{R_0\cos\phi} & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ F_{41} & 0 & \vdots & k_D & 2\omega_D & \vdots & 0 & f_D & -f_E & \vdots & 0 & 0 & 0 \\ F_{51} & 0 & \vdots & F_{54} & F_{55} & \vdots & -f_D & 0 & f_N & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ -\Omega_D & 0 & \vdots & 0 & \frac{-1}{R_0} & \vdots & 0 & \omega_D & -\omega_E & \vdots & 1 & 0 & 0 \\ 0 & 0 & \vdots & \frac{1}{R_0} & 0 & \vdots & -\omega_D & 0 & \omega_N & \vdots & 0 & 1 & 0 \\ F_{91} & 0 & \vdots & 0 & \frac{\tan\phi}{R_0} & \vdots & \omega_E & -\omega_N & 0 & \vdots & 0 & 0 & 1 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \end{array}\right], \tag{29}$$

where the necessary F_{jk} blocks are provided in [8, Table 11.1, p. 397], and are echoed back below:

$$F_{41} = -2\Omega_N \cdot v_E - \frac{\rho_N \cdot v_E}{\cos^2\phi}, \tag{30}$$

$$F_{51} = 2\left(\Omega_N \cdot v_N + \Omega_D \cdot v_D\right) + \frac{\rho_N \cdot v_N}{\cos^2\phi}, \tag{31}$$

$$F_{54} = -\left(\omega_D + \Omega_D\right), \tag{32}$$

$$F_{55} = k_D - \rho_E \cdot \tan\phi, \text{ (k_D is defined in Eq. 45 below)} \tag{33}$$

$$F_{91} = \Omega_N + \frac{\rho_N}{\cos^2\phi}. \tag{34}$$

or, after the indicated substitutions, the above associated continuous-time dynamics matrix becomes:

$$F = \left[\begin{array}{ccc:ccc:ccc:ccc:ccc} 0 & 0 & \vdots & \frac{1}{R_0} & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \frac{-\rho_D}{\cos\phi} & 0 & \vdots & 0 & \frac{1}{R_0\cos\phi} & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ -2\Omega_N \cdot v_E - \frac{\rho_N \cdot v_E}{\cos^2\phi} & 0 & \vdots & k_D & 2\omega_D & \vdots & 0 & f_D & -f_E & \vdots & 0 & 0 & 0 \\ 2\left(\Omega_N \cdot v_N + \Omega_D \cdot v_D\right) + \frac{\rho_N \cdot v_N}{\cos^2\phi} & 0 & \vdots & -(\omega_D + \Omega_D) & [k_D - \rho_E \cdot \tan\phi] & \vdots & -f_D & 0 & f_N & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ -\Omega_D & 0 & \vdots & 0 & \frac{-1}{R_0} & \vdots & 0 & \omega_D & -\omega_E & \vdots & 1 & 0 & 0 \\ 0 & 0 & \vdots & \frac{1}{R_0} & 0 & \vdots & -\omega_D & 0 & \omega_N & \vdots & 0 & 1 & 0 \\ \Omega_N + \frac{\rho_N}{\cos^2\phi} & 0 & \vdots & 0 & \frac{\tan\phi}{R_0} & \vdots & \omega_E & -\omega_N & 0 & \vdots & 0 & 0 & 1 \\ \cdots & \cdots & & \cdots & \cdots & & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & \vdots & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \end{array}\right], \tag{35}$$

The other fairly obvious option for a Kalman filter model would be to not omit the vertical channel, and so would contain the two additional states (ignored above), where the entirety corresponds to the Full INS Error Model of [8, Eq. 11.80, p. 396]. The more general Full INS Error Model is likely needed in maneuvering flights while the reduced-order model will probably suffice for straight and level flight. The more general form should accommodate both. The System dynamics matrix for the Full INS Error Model has the following structure for the associated continuous-time F matrix:

$$\left[\begin{array}{ccc:ccc:ccc:ccc:ccc} 0 & 0 & \frac{\rho_E}{R_0} & \vdots & \frac{1}{R_0} & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \frac{-\rho_D}{\cos\phi} & 0 & \frac{-\rho_N}{R_0\cos\phi} & \vdots & 0 & \frac{1}{R_0\cos\phi} & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & -1 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ F_{41} & 0 & F_{43} & \vdots & k_D & 2\omega_D & -\rho_E & \vdots & 0 & f_D & -f_E & \vdots & 0 & 0 & 0 \\ F_{51} & 0 & F_{53} & \vdots & F_{54} & F_{55} & F_{56} & \vdots & -f_D & 0 & f_N & \vdots & 0 & 0 & 0 \\ -2v_E \cdot \Omega_D & 0 & F_{63} & \vdots & 2\rho_E & -2\Omega_N & 0 & \vdots & f_E & -f_N & 0 & \vdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ -\Omega_D & 0 & \frac{\rho_N}{R_0} & \vdots & 0 & \frac{-1}{R_0} & 0 & \vdots & 0 & \omega_D & -\omega_E & \vdots & 1 & 0 & 0 \\ 0 & 0 & \frac{\rho_E}{R_0} & \vdots & \frac{1}{R_0} & 0 & 0 & \vdots & -\omega_D & 0 & \omega_N & \vdots & 0 & 1 & 0 \\ F_{91} & 0 & \frac{\rho_D}{R_0} & \vdots & 0 & \frac{\tan\phi}{R_0} & 0 & \vdots & \omega_E & -\omega_N & 0 & \vdots & 0 & 0 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \end{array}\right], \tag{36}$$

that we tie into here as the 12 state Kalman filter model consisting of the following states (ϕ being Latitude and λ being Longitude and h being altitude above earth ellipsoid below):

$$\begin{array}{rcl} \delta\mathbf{x} & = & [\delta\phi, \delta\lambda, \delta h, \delta v_N, \delta v_E, \delta v_D, \epsilon_N, \epsilon_E, \epsilon_D, \text{gyro random walk}_N, \text{gyro random walk}_E, \text{gyro random walk}_D]^{\mathbf{T}}, \\ & = & [\text{deg}, \text{deg}, m, m/sec, m/sec, m/sec, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr, \text{deg}/hr]^{\mathbf{T}} \text{ (corresponding units)}, \end{array} \tag{37}$$

and, additionally, from [8, Table 11.1, p. 397]:

$$\Omega_N = \omega_{ie} \cdot \cos\phi, \tag{38}$$

$$\Omega_D = -\omega_{ie} \cdot \sin\phi, \tag{39}$$
$$\rho_N = \frac{v_e}{R_e}, \tag{40}$$
$$\rho_E = \frac{-v_n}{R_e}, \tag{41}$$
$$\rho_D = \frac{-v_e \cdot \tan\phi}{R_e}, \tag{42}$$
$$\omega_N = \Omega_N + \rho_N, \tag{43}$$
$$\omega_E = \rho_E, \tag{44}$$
$$\omega_D = \Omega_D + \rho_D, \tag{45}$$
$$k_D = \frac{v_d}{R_e}, \tag{46}$$

where the additional necessary F_{jk} blocks are also provided in [8, Table 11.1, p. 397], and are echoed back here below for completeness (and to allow easy cross-checking with code implementation to reduce errors that would otherwise be more likely):

$$F_{43} = \rho_E \cdot k_D - \rho_N \cdot \rho_D, \tag{47}$$
$$F_{53} = -\rho_E \cdot \rho_D - k_D \cdot \rho_N, \tag{48}$$
$$F_{63} = \rho_N^2 + \rho_E^2 - 2\frac{g}{R_0}, \tag{49}$$
$$F_{56} = \omega_N + \Omega_N. \tag{50}$$

The above F_{ij} were excellent candidates for becoming user-defined functions in MatLab, as convenient summaries that are easy to cross-check and also confirm in intermediate printouts of the above system matrix. The above model set forth in [8] was checked against possible errors and corrections (previously noted by others) as listed on Prof. Jay A. Farrell's official Web Site:
http://www.ee.ucr.edu/~farrell/AidedNavigation that I was alerted to by Prof. Jay A. Farrell himself (in a personal e-mail correspondence) and simultaneously by Scott Foes (OKSI). No such errors were found listed there for this Chapter 11 that was relied upon so heavily here. That was a welcome relief. (The book [8] had said that such corrections would reside with the publisher, but Prof. Farrell had found it to be more convenient to handle them himself on his own personal University of California - Riverside Web Site.)

Unfortunately, we did not have the luxury of extra time available to pursue other more recent modern approaches offered in [26] for "tuning the filter" but this aspect is not critical but secondary.

The proper initial conditions to use for the Kalman filter covariance **(not yet consistent in Eq. 50 here nor adequately complete in [3] to allow us to fill in all required information about $1-\sigma$ values in the following)**:

$$\begin{aligned} P &= \text{diag}\left[(2.5\,\text{deg})^2, (2.5\,\text{deg})^2, (?\ m)^2, ?\ m^2/sec^2, ?\ m^2/sec^2, ?\ m^2/sec^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2\right] \\ &= \text{diag}\left[6.25\,\text{deg}^2, 6.25\,\text{deg}^2, ?\ m^2, ?\ m^2/sec^2, ?\ m^2/sec^2, ?\ m^2/sec^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2, ?\ \text{deg}^2/hr^2\right]. \end{aligned} \tag{51}$$

but despite being incompletely specified here, a work-around exists because of the theory discussed next that allows us to use unity for unknown or unsure entries and run EKF until it converges (enough) with frequent virtual video position fixes (more plentiful than actually available) being used in this manner merely to home in on realistic initial covariances to use in getting started after these computationally obtained results are available from this alternate route. Use of covariance analysis alone could possibly suffice for this aspect as well.

The proper initial conditions to use for the Kalman filter state estimates:

$$\hat{x} = [0\ \text{deg}, 0\ \text{deg}, 0\ m, 0\ m/sec, 0\ m/sec, 0\ m/sec, 0\ \text{deg}/hr, 0\ \text{deg}/hr, 0\ \text{deg}/hr, 0\ \text{deg}/hr, 0\ \text{deg}/hr, 0\ \text{deg}/hr]^T. \tag{52}$$

The *obsevability* and *controllability* of INS error models has already been established in [23], [24] and, moreover, even if the error models were only *detectable* and *stabilizable* (as weaker more easily met conditions than *obsevability* and *controllability*), Ref. [25] demonstrates that initial guesses still converge to the correct values as time elapses. Full *obsevability* and *controllability* guarantee exponentially asymtotically fast convergence (as long as the guess for the initial covariance is positive definite, as assured when matrix is taken to be diagonal with all entries positive). Also, by computing over a reasonable (but not necessarily long) time epoch conveniently provides a non-diagonal covariance to use as an initial start in subsequent simulations and with actual data.)

Proper process noise gain matrix and covariance intensity matrix, respectively, for zero mean Gaussian White Noise are:

$$G^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and (53)

$$Q = \begin{bmatrix} (4.5\,\mathrm{deg}/\sqrt{\mathrm{hr}})^2 & 0 & 0 \\ 0 & (4.5\,\mathrm{deg}/\sqrt{\mathrm{hr}})^2 & 0 \\ 0 & 0 & (4.5\,\mathrm{deg}/\sqrt{\mathrm{hr}})^2 \end{bmatrix} = \begin{bmatrix} 20.25\,\mathrm{deg}^2/\mathrm{hr} & 0 & 0 \\ 0 & 20.25\,\mathrm{deg}^2/\mathrm{hr} & 0 \\ 0 & 0 & 20.25\,\mathrm{deg}^2/\mathrm{hr} \end{bmatrix},$$

which are both used to form the matrix GQG^T that arises in the Covariance Propagate Step addressed in great detail in Sec. 6.

Robert M. Roger's book (3rd Edition, 2007) Ref. [10] has a short discussion of AHRS on [10, page 281], and a miniaturized block diagram is in [10, Fig. 11.3] (depicting requisite filter model and truth model for simulation). However, one needs a magnifying glass to read it (as I used). The discussion about driver programs was useful to TeK Associates but other sources are much more germane to this project (from our point of view). However, this was a useful early reminder that TeK Associates also needs to provide a truth model simulator in MatLab to support our bootstraping effort in cross-checking correct Kalman-like filter response before using it on actual data.

The 2004 2nd Edition book [4], has a Chapter 9 on Strapdown System Technology and [4, Fig. 9.1] displays the more limited functionality of an AHRS as compared to the further processing needed in a full INS. Other useful insights (for OKSI's project) are provided in [4, Chap. 12]. In particular, Sec. 12.6 reminds us of motion dependence of strapdown system performance and Sec. 12.6.1 depicts the maneuver-dependent error terms. Next useful discussion is in Sec. 13.6 and, in particular, INS aiding using ground tracker measurements in [4, Fig. 13.15] The measurement equation is provided in [4, Eq. 13.6] and the appropriate observation matrix H is provided in [4, Eq. 13.6] (it is nonlinear). This reference did emphasize the various calibrations that a strapdown system undergoes upon start-up. This particular OKSI project assumes that the necessary calibrations have already been successfully performed.

4.1 Measurement Model for OKSI Video Image Position Fix

The sensor model for the OKSI Video Image position fix is of the following standard form:

$$z(t_k) = H_1 \delta x(t_k) + v(t_k), \tag{54}$$

where $z(t_k)$ is the actual measurement realization at discrete-time instant t_k, H_1 is the Observation Matrix, and $v(t_k)$ is a particular realization of the zero mean White Gaussian Noise (WGN), with distribution $N(0, R_1)$ that corrupts the measurement.

The Observation Matrix in the above is of the following structural form:

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{55}$$

From the bottom row of a recently conveyed Excel file (corresponding to 15th hour and 51.149 minutes), the following information was obtained for estimated location:

$$Est\_x = -1109.41; Est\_y = 4313.024; Est\_z = 128.4346, \tag{56}$$

where TeK Associates assumes that units are the same as in an earlier Excel spreadsheet that it received from OKSI **(Cross-check units)**. TeK Associates may need to ignore attitude update this time around but will definitely perform position update. Any attempted attitude update needs additional analysis and implementation beyond the scope of what is offered herein[8].

$$R_1 = \begin{bmatrix} (6.8725\,\mathrm{deg})^2 & 0 & 0 \\ 0 & (2.19176\,\mathrm{deg})^2 & 0 \\ 0 & 0 & (55.45258\,\mathrm{deg})^2 \end{bmatrix} = \begin{bmatrix} 47.2313\,\mathrm{deg}^2 & 0 & 0 \\ 0 & 4.8038\,\mathrm{deg}^2 & 0 \\ 0 & 0 & 3074.9886\,\mathrm{deg}^2 \end{bmatrix}, \tag{57}$$

where the units for each are degrees [9] for roll, pitch, and yaw, respectively. While this was the associated measurement covariance for a particular location, it is assumed (by TeK Associates) that this same covariance intensity matrix is a reasonable representative approximation that is adequate for every OKSI Video-Image position fix (unless we are told otherwise by OKSI).

[8] In considering how to best modify the equations that contain (and propagate in time) the attitude information contained in the Direction Cosine Matrix (DCM), as currently conveyed in Eqs. 2, 3, 12 herein, and as included in all the other KF equations that depend on this, I wanted to be able to compare in some way the accuracy conveyed within the covariance that accompanies the attitude information that OKSI Video Navigation provides with the accuracy of attitude inherent in the DCM (which is not further explicitly called out here) but does affect our Kalman-like Filter by how it is implemented. I recall a fairly new (within the last 10 years) approach that has been championed by Technion University, mainly for NASA-like applications, that may be appropriate for this type of application as well. A Matrix Kalman-like filter is setup for the DCM itself. This represents an "also" and not an "instead of" what we already have herein and, as such, is currently considered to be beyond the scope of this current work. What we currently implement here is a more familiar vector-matrix Kalman like filter with mere position fixes (that should suffice to compensate for gyro drift that is rampant in less expensive gyros). The new initiative just mentioned is a Matrix-Matrix version of a Kalman-like filter that is less straight forward and invokes Kronecker products in its theoretical derivation at a high level but ultimately simplifies into fairly familiar-looking expressions in implementation that similarly correspond to a Propagate Step and an Update Step. Matrix KF material that we provide references for here, as [35]-[46], were compiled by TeK Associates a year ago and the technique was primarily pioneered by Daniel Choukroun, B. S. (Summa cum Laude), M.S., Ph.D. (1997, 2000, 2003), post-doc (UCLA), currently an Assistant Professor at Delft University of Technology, Netherlands. Of course, his stellar committee also deserves credit too, especially the (late) Prof. Itzhack Bar-Itzhack and Prof. Yaakov Oshman for focusing his work on useful and challenging practical applications.

[9] While calculus requires use of radians with trigonometric functions and so does [5], we adhere to degrees for angle, as adhered to in [2], [3]. With respect to this aspect, we must be especially careful with MatLab implementations of trigonometric functions (that expect radians) to get it right. Will correct units in Eq. 50 using $R_1' = TR_1T^T$ to get units right after converting to proper coordinate system that matches how we want it.

Disclaimer: I had speculated on a possible need to revise or back petal if there were cracks or holes recognized to be in the dike. (I now adhere to [5].) Better to get it right early on and not have to extricate errors from the code and then put something else in. This whole documentation exercise is important to keep things under control and to maintain the sanity of everyone involved. Yes, I am sweating the coding too because I need to finish it and get it to OKSI ASAP. Better to have something that works as it should. My starting place was in modifying the requisite building blocks of [28] to be what this project needs[10]. I augmented this with some subroutines from [29] and [30]. Certain newer transformations are recognized [31] to be more efficient implementations than that in [29]. Also see [7].

Intermediate validation (specifically, unit testing) of the performance of this Kalman filter implementation in MatLab was accomplished using the results of [32] and [33] and test cases provided in the user manual of [28].

I have verified the simulator that I used to generate simulated measurement data (of known structure and values) and am using it as input measurement data to compare to what the Kalman-like filter outputs from its processing. I am currently simulating data that is available at a periodic rate (first).

Next steps: I am not yet running actual real data (that you folks at OKSI provided me with) through the Kalman-like filter. I have used a known entity first (from above mentioned simulator) as a test case and have favorably compared outputs to exactly what was simulated.

I will alter the structure of handling input measurement data so that it does not need to be periodic but can take input data and handle it as it comes and cause processing to stop or cease when no more measurement data is available. I side-stepped any need to use the callback feature in MatLab to implement this aspect of "handle-it-as-it-comes" since MatLab allows one to view the length of the data record beforehand and act accordingly in processing since these data records have been observed to not have any missing entries (otherwise more sophisticated methods would be needed). The initial test to confirm that this modification is successful will still be to use the same simulated periodic measurement data on this new structural change and expect to obtain the exact same Kalman filter outputs (otherwise something is clobbered).

After successfully passing these milestones with simulated data and not necessarily periodic measurements, I can move on to process the data that OKSI gave me **that is the main goal of this exercise and this project**. I have displayed the intermediate steps too (that perhaps no one else is interested in except me because they were necessary to get to the goal and have confidence in the final results).

5 Creative Suggestions for Performing the Tests of Proximity (along with standard direct comparison plots as main outputs for decisions)

TeK Associates plans to invoke some creative aspects from [18] to [20] and then use these ideas where it should logically occur during comparisons of our filter's outputs (routine estimates and associated covariances, which can be creatively specialized into an $m-\sigma^2$ covariance ellipsoid centered about its specific associated estimates and compared for sufficient proximity to corresponding GPS indicated state estimates encompassed by its associated $m-\sigma^2$ covariance ellipsoid [GPS output comes from its own internal filter too!]). The degree of proximity is minimum size of scalar "m" for which the two ellipses (or ellipsoids) do overlap [we may merely take a fixed $m=1$ or $m=2$ or $m=3$ or maybe $m=1.5$]. Ref. [19] provides an easy implementation (in one line of mathematical operations that are especially MatLab-compatible for actual implementation). Reference [18] offers a way to numerically determine whether two ellipsoids (or ellipses) of the following form:

$$(x-x_1)^T\left(\frac{1}{2}\right)P_1^{-1}(x-x_1)=1 \text{ and } (x-x_2)^T\left(\frac{1}{2}\right)P_2^{-1}(x-x_2)=1 \text{ either overlap or not,} \tag{58}$$

where $x_1 \neq x_2$ and the matrices A and B in the above are symmetric $A=A^T$, $B=B^T$ and positive definite: $A>0$, $B>0$. Reference [20] further emphasizes the simplification offered in Ref. [19] rather than use our own ellipsoid overlap test of 30 years ago, which does not look to be a perfect fit to OKSI's application but the more recent test of overlap [18] [not our own, but with our own additional simplifications, which make it easier to apply without having to decompose into several different cases] does appear to be an exact fit. Our particular simplification makes the following useful observation: The recent test for two ellipsoid overlap in [18], was observed in [19] to correspond to numerically solving the symmetric generalized eigenproblem **(using a Choleski factorization and the symmetric QR algorithm)** since

$$\lambda A\underline{x}=B\underline{x} \Leftrightarrow [\lambda A-B]\,\underline{x}=0 \Rightarrow \underline{x}^T[\lambda A-B]\,\underline{x}=0 \Leftrightarrow \underline{x}^T A\left[\lambda I-A^{-1}B\right]\underline{x}=0, \tag{59}$$

(the last relationship on the right hand side of the above being the cornerstone of [18]), where the useful simplifying directions of implication (from left to right) depicted here above were first revealed in [19] and further emphasized in [20, Introduction, 4^{th} paragraph]. Notice that the eigenvalue/eigenvector problem for the expression on the far right hand side of Eq. 58 involves finding eigenvalues for a non-symmetric matrix (which can be complex and therefore needs to be decomposed into the various cases considered in [18]), while the eigenvalue/eigenvector problem for the expression on the far left hand side of Eq. 58 involves finding eigenvalues for a symmetric matrix (which are always real) and is therefore a simplification since no further

[10] As a precedent, I have used modules from this software package before in 2003 in expediting results in MatLab code. Although it does not initially conform to the algorithmic software architecture that is sought here, it is clear and simple enough to easily repackage the puzzle pieces to be the architecture sought for this project, as long as changes are made carefully and adequately cross-checked.

decomposition is required. It is the symmetric generalized eigenproblem that is so compatible with implementation in MatLab. For completeness here we again summarize the manipulations constituting a complete implementation of the approach of [18] (as also revealed in a more direct form in [19, Eqs. 4 and 5]), we have that:

$$x^T \overbrace{MS_1M^T}^{A} x = 0 \text{ for } S_1 \triangleq \begin{bmatrix} (\frac{1}{2})P_1^{-1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{60}$$

with translation offset:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\tilde{x}_1 & -\tilde{x}_2 & \tilde{x}_3 1 & \end{bmatrix}, \tag{61}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{bmatrix}, \tag{62}$$

and

$$x^T \overbrace{S_2}^{B} x = 0, \text{ and } S_2 \triangleq \begin{bmatrix} (\frac{1}{2})P_2^{-1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \text{ with no offset,} \tag{63}$$

(without any loss of generality, because coordinate origin can always be moved to perform this numerical test at the location of the second possible offset, thus causing it to be zeroed out). After Eqs. 59 and 62 are combined, the test consists of solving for λ in

$$x^T A \left[\lambda I_{4\times 4} - A^{-1}B\right] x = 0 \tag{64}$$

to determine whether the underlying two 3-dimensional ellipsoids of primary interest overlap (or not), depending on how the corresponding numerical evaluation turns out.

This Two Ellipsoid Proximity test (or overlap test) would be performed as a function of time, as an easy to interpret overview and summary (of either success or failure of proximity in 3-dimensions or in 2-dimensions, if confined merely to ground track considerations). By making "m" big enough, failure or non-overlap can be made into overlap (success) but the value of "m" has a certain probability of containment associated with it for a proper tie-in with "reality" that can not/should not be overlooked or ignored. This is a good thing.

The above is merely pre-planning now for use of these potential devices to be invoked further downstream (this month). If OKSI (or their customer) does not want an uncertainty covariance to be associated with the GPS outputs used as a gauge of"truth" then this approach offers considerable simplification in the above described methodology by reducing to a test of a specified point for containment within an uncertainty ellipse about a specified center. By taking the difference between the test point and the center point, the test degenerates into an even simpler test based on the following fairly straightforward theoretical interpretations and approximations: for an ideal mean of the estimator as

$$E\left[\hat{x}(t_i)\right] = x_{true}(t_i) \approx x_{\text{GPS}}(t_i), \tag{65}$$

with corresponding covariance:

$$P(t_i) \triangleq E\left[(\hat{x}(t_i) - x_{true}(t_i))(\hat{x}(t_i) - x_{true}(t_i))^T\right] \approx E\left[(\hat{x}(t_i) - x_{\text{GPS}}(t_i))(\hat{x}(t_i) - x_{\text{GPS}}(t_i))^T\right], \tag{66}$$

and now define the following coordinate transformation (that is time-varying in general but with the time index being suppressed here for clarity) by letting

$$y \triangleq P^{-1/2}(\hat{x} - x_{\text{GPS}}), \tag{67}$$

then

$$E\left[y\right] = P^{-1/2}E\left[(\hat{x} - x_{\text{GPS}})\right] = \underline{0}, \tag{68}$$

and

$$E\left[yy^T\right] = P^{-1/2}E\left[(\hat{x} - x_{\text{GPS}})(\hat{x} - x_{\text{GPS}})\right]P^{-1/2} = P^{-1/2}PP^{-1/2} = I_{n\times n}. \tag{69}$$

Moreover, by using the following scalar measure

$$\ell \triangleq y^T y = y_1^2 + y_2^2 + \cdots + y_n^2, \tag{70}$$

one obtains the following insight into its underlying statistics:

$$Prob\left[(\hat{x} - x_{true})^T P^{-1}(\hat{x} - x_{true}) \le K^2\right] = Prob\left[y^T y \le K^2\right] = Prob\left[\ell \le K^2\right] = Prob\left[\chi_n^2 \le K^2\right], \tag{71}$$

where the last expression may be evaluated by look-up tables for the Chi-square distribution with n degrees-of-freedom and thus related to the proceeding equalities that have been established here.

For more details, please see http://www.tekassociates.biz/PriorSummer2003TeKCovarianceFidelityProposal.pdf (discussed in Section 6 within this link just cited). (A typo occurred in the document of this link where the transpose was placed on the wrong vector. It is corrected herein as Eq. 65 above.)

Ref. [16] alerts us to the fact that even when everything is done correctly, vibrations and flexure in the vehicle can cause big problems with strapdown systems; however, [16] indicates how to reduce such adverse effects.

A recent theoretical investigation [34] demonstrates great promise for future estimator designs for INS-related applications but independent detailed supporting evaluations of favorable estimator behavior in applications remain to be performed to validate optimistic early claims.

6 Some Critical Details of the Software Implementation

In the interest of full disclosure of what TeK Associates has implemented in the software, especially since OKSI has recently asked regarding these details and I am not finished until I have completely documented the answer, I will detail the conversion of the continuous-time version of the state variable model of Eq. 21 into the discrete-time version actually implemented in TeK Associates' software. This is an important consideration and should be properly answered since this is where implementers frequently go astray in properly making this bridge from continuous-time to discrete-time. This is why I will carefully lay it out here below to inform OKSI about how TeK Associates handled this important aspect.

Again, please consider the standard state variable representation of the error states associated with an INS in Eq. 21, which is now simplified somewhat by the linearization of [8, Eqs. 11.79, 11.80, 11.104, 11.105, and Table 11.1] to be of the following continuous-time form:

$$\delta\dot{x}(t) = F(t)\delta x(t) + Gw(t) \text{ with } E\left[w(t)w^T(\tau)\right] = Q\delta(t-\tau), \tag{72}$$

where the detailed structure of $F(t)$ is as defined in Eqs. 29-33, 35-52 for continuous-time and where $w(t)$ is a zero mean White Gaussian Noise (WGN) random process, with statistics $N(0, Q)$ that were specified along with Gaussian random vector initial condition $N(0, P_0)$ in Sec. 4. The last $\delta(t-\tau)$ on the far right in Eq. 71 is a Dirac delta impulse function with well-known properties. In what follows, we will provide the discrete-time version of this error model.

Eventually, we will obtain a final result for the corrected output of the INS as being: $[x(t) - \delta\hat{x}(t)]$, where $x(t)$ is obtained from our augmenting computations in Sec. 2 and 3 (that cause an AHRS be an INS) and the *correction* $\delta\hat{x}(t)$ is computed by solving the Eq. 71 above, as provided by the TeK Associates' Kalman-like filter.

In reality, since everything is being provided using digital computer computations, the more correct restatement of the above sentence is that we will obtain a final result for the *corrected* output of the INS as being: $[x(t_k) - \delta\hat{x}(t_k)]$, where $x(t_k)$ is obtained from our augmenting computations (that cause an AHRS be an INS) and the correction $\delta\hat{x}(t_k)$ is computed by solving a slightly modified discrete-time form corresponding to Eq. 71 above, as provided by the TeK Associates discrete-time Kalman-like filter formulation.

As is standard practice in Inertial Navigation applications, for convenience in subsequent notation, the δ is dropped from $\delta x(t)$ to yield the following simpler appearing version of Eq. 71:

$$\dot{x}(t) = F(t)x(t) + Gw(t) \text{ with } E\left[w(t)w^T(\tau)\right] = Q\delta(t-\tau), \tag{73}$$

(which is still understood to be in terms of $\delta x(t)$ but minus the more unwieldy notation in front of $x(t)$). Eq.72 has a solution of the following familiar form:

$$x(t) = \Phi(t,0)x(0) + \int_0^t \Phi(t,\tau)Gw(\tau)d\tau\ , \tag{74}$$

where the associated transition matrix $\Phi(t,s)$ satisfies:

$$\frac{\partial\Phi(t,s)}{\partial t} = F(t)\Phi(t,s) \text{ where } \Phi(t,t) = I_{n\times n} \text{ for all } t. \tag{75}$$

Notice that the above solution can be rewritten in the following slightly different form going from lower limit time = s (instead of from lower limit time = 0) as:

$$x(t) = \Phi(t,s)x(s) + \int_s^t \Phi(t,\tau)Gw(\tau)d\tau\ . \tag{76}$$

Under the assumption that the lower limit s is relatively close to the upper limit t, and further under the assumption that over a relatively short time interval from s to t, that the matrix $F(s)$ is essentially constant, then we can invoke the following simplification for the transition matrix to be merely the matrix exponential[11]:

$$\Phi(t,s) = \exp\left[F(s)(t-s)\right] \text{ where for } t = s, \text{ the requisite boundary condition is satisfied as } \exp\left[0_{n\times n}\right] = I_{n\times n} \text{ for all } t. \tag{77}$$

[11] While MatLab offers several different ways for calculation the matrix exponential with good accuracy and precision, we approximate it in the code using the first three primary terms in its Taylor series approximation $[I + F + 0.5F^2]$ to speed up the calculations. Using merely a two term approximation as $[I + F]$ is not unusual in some applications.

within the following representation of Eq. 75:

$$x(t) = \exp\left[\overbrace{F(s)}^{\text{treated as constant}} (t-s)\right] x(s) + \int_s^t \exp\left[\overbrace{F(s)}^{\text{treated as constant}} (t-\tau)\right] Gw(\tau)d\tau \ . \tag{78}$$

which simplifies to

$$x(t) = \exp\left[F(s)(t-s)\right]x(s) + \exp\left[F(s)t\right]\int_s^t \exp\left[-F(s)\tau\right]Gw(\tau)d\tau \ . \tag{79}$$

If we were going to deal with a fixed time-step Δ throughout, then within the above, we would let

$$s = k\Delta \tag{80}$$

and

$$t = (k+1)\Delta \tag{81}$$

and obtain the following familiar intermediate result:

$$x(k+1) = \exp\left[F(k)\Delta\right]x(k) + \int_{k\Delta}^{(k+1)\Delta} \exp\left[F(k)((k+1)\Delta - \tau)\right]Gw(\tau)d\tau. \tag{82}$$

A common practice is to instead work with the discrete-time equivalent of continuous-time white process noise [12], [32, p. 196], defined as:

$$W(k) \triangleq \int_{k\Delta}^{(k+1)\Delta} \exp\left[F(k)((k+1)\Delta - \tau)\right]Gw(\tau)d\tau, \tag{83}$$

then Eq. 81 simplifies as:

$$x(k+1) = \exp\left[F(k)\Delta\right]x(k) + W(k), \tag{84}$$

where $W(k)$ has process noise covariance matrix Q_d [12], where:

$$Q_d(k) = \exp\left[F(k)\Delta\right]\left[\int_0^\Delta e^{-F(k)\tau}GQG^T e^{-F^T(k)\tau}\right]\exp\left[F^T(k)\Delta\right] \ \text{(an exact expression)}. \tag{85}$$

In practical applications, we must sometimes make use of simplifying expedient approximations such as the following:

$$Q_d(k) = \begin{cases} \exp\left[F(k)\Delta\right]\left[\int_0^\Delta e^{-F(k)\tau}GQG^T e^{-F^T(k)\tau}\right]\exp\left[F^T(k)\Delta\right] & \text{(an exact expression),} \\ \text{or} & \\ GQG^T\Delta & \text{(an expedient approximation).} \end{cases} \tag{86}$$

The **Covariance Propagate Step** of the Kalman-like Filter is:

$$P(j+1,j) = \exp\left[F(j)\Delta\right]P(j,j-1)\exp\left[F^T(j)\Delta\right] + GQG^T\Delta \tag{87}$$

The above discussion was a necessary prelude to show how this changes for measurements coming in between main time steps next so that the magnitude of the discrete-time equivalent to the continuous-time white noise is modified appropriately to correspond.

Eq. 83 (without the presence of the process noise term $W(k)$) corresponds to what is used in the Propagate Step or Prediction Step of the Kalman-like filter estimate between measurements [13], the Update Step which then follows when there is a sensor measurement present incorporates the effect of the measurement received at that time instant of reception (which should correspond exactly to the time endpoint of the last Prediction Step). Focusing on the approximate expression above, please notice that it is a function of the scalar time interval Δ. For a OKSI Video Nav position fix that occurs between two main propagate steps, the appropriately scaled expression to use for Q_d in the Covariance Propagate Equation is:

$$\mu\Delta GQG^T, \tag{88}$$

and, subsequently, the value to use is:

$$(1-\mu)\Delta GQG^T, \tag{89}$$

[12] Obtained by post-multiplying Eq. 82 by its transpose with time index k replaced by time index j, then taking expectation throughout and interchanging the order of integration and expectation and invoking the sifting property of the Dirac delta impulse function.

[13] For this application, the more numerous shorter time steps are for $Delta = 10$ millisec. Rather than take big time steps within the Propagation Step, small steps are used so that the assumption that the $F(s)$ being essentially constant is met and appropriately also incrementally captures the time-varying nature of $F(t)$ over the entire time epoch (which can be considerable).

for an explicit fixed value of μ, where $0 \leq \mu \leq 1$. Therefore, the **Covariance Propagate Step** of the Kalman-like Filter for a measurement that falls between the usuall small time-steps is:

$$P(j+1,j) = \exp\left[F(j)\mu\Delta\right]P(j,j-1)\exp\left[F^T(j)\mu\Delta\right] + GQG^T\mu\Delta, \tag{90}$$

and after the instantaneous Update Step, the subsequent **Covariance Propagate Step** of the Kalman-like Filter should be:

$$P(j+1,j) = \exp\left[F(j)(1-\mu)\Delta\right]P(j,j)\exp\left[F^T(j)(1-\mu)\Delta\right] + GQG^T(1-\mu)\Delta, \tag{91}$$

before reverting back to use Eq. 86 for the usual predominant time step Δ. Notice that both Eqs 89 and 90 are of the same basic form as Eq. 86 except that they use a different appropriate scaling down of Δ. If the exact form in Eq. 85 were used, it would take too long to perform the indicated integrations within the exact calculation for the cases of $\mu\Delta$ and $(1-\mu)\Delta$. Similar modifications must be invoked for the Estimator Propagate Step as well to accommodate $\mu\Delta$ and $(1-\mu)\Delta$. This is what is needed to appropriately handle the different measurement sample rates of this application. An additional case that should be checked for is when/if both measurements arrive simultaneously to avoid introducing double the effect that should be there. If both periodic rates were "perfect" but without synchronized start times so there is a constant off-set, we could just figure this spacing out once and assume that it stays the same throughout. However, if the system hic-cupped, then it would be off. A more conservative approach that doesn't really take much longer is to calculate the appropriate Δ to use from the time-stamp of each sample and the time stamp of its next subsequent sample. In this way, we will always use the correct value even if the system hic-cups or if off-set changes from time to time or if there are missing or skipped rows. Better to be safe than sorry.

Explanation of Processing Structure also included in the associated MatLab code: Linearized Kalman Filter for OKSI including further preprocessing of AHRS to become an INS (rather than merely an AHRS) to which the Linearized Kalman Filter then corresponds. All inputs are assumed to be periodic but not necessarily synchronized.

The IMU within the AHRS (being a Crossbow AHRS400CD-200), is a strapdown mechanized MEMS, and the model utilized for the Kalman-like Filter herein is the full IS Error Model of Eq. 11.80, p. 396 of Jay A. Farrell's text -book; with matrix entries as spelled out in Table 11.1 p. 397 (and with gyro random-walk error terms included [beyond what is shown in Eq. 11.80] in each of the pertinent input channels of the 3 gyros). This VERSION was coded by Thomas H. Kerr III (TeK Associates) and much was carried over from his preliminary work on this code that he performed for this project on his other more limited capability 32-bit Vista OS Laptop PC. This cur- rent version is running on a 64-bit Windows 7 OS Laptop PC and, as such, can accomodate much longer input files (that were provided by OKSI).

To speed up MatLab processing, rateRef is defined herein first and set to be a zero matrix that will be appropriately filled in as processing progresses. The dimension of rateRef will be altered to accommodate processing associated with both IMU and OKSI Video position fixes together.

This code assumes that OKSI Video Position Fix measurements are available periodically and that there are no missing rows or columns.

The IMU data is also assumed to be periodic but more frequently available at a higher rate (with a shorter period than OKSI Video data fixes) and that there are no missing rows or columns.

Programmer: Thomas H. Kerr III, Ph.D. in E.E., TeK Associates, Lexington, Massachusetts, 24 February 2013.

The total number of "Kalman Update Steps" is exactly the number of OKSI Video Position fixes (in time) availed as Input Data (obtained from OKSI).

The total number of Kalman Propagate Steps is the number of ARHRS data (in time) PLUS 2 times the number of OKSI Video Position fixes (assuming none are exactly simultaneous with the times of the AHRS data (we built in a cross-check just in case it is so that we still do the right thing). This AHRS input data was also obtained from OKSI.

We also have GPS data to be used as "TRUTH" (that can be input now but ordinarily would not be used until all the Kalman-like Filter Processing has been completed over the entire time epoach covered by both the AHRS and OKSI Video position fix data. There is no reason why GPS TRUTH data would be synchronous with AHRS or OKSI Video Position fix data without being explicitly forced to be so. In order to get an "apples-to-apples" comparison of final Navigation results to test for proximity to what GPS indicates is TRUTH, it would have to be done at a common time for both. This can be accomplished by interpolating GPS data to match the same points in time that the processed Nav data is available. A reasonable question to ask is why not instead interpolate the computed Navigation data to match the times at which GPS TRUTH data is available. The quick answer is that if we want to use some of the creative ideas of Section 5 of the progress Report to gauge proximity then the associated computed covariances are also needed within the comparison (as indicated in Section 5). Since covariances correspond to ellipsoids with principal axes that move around with time, it is not obvious how to interpolate within covariances properly so interpolation needs to be performed on the GPS data as the most convenient option. The GPS input data was also obtained from OKSI.

Contributing to the above decision to interpolate GPS truth data to the desinated comparison times, recall from Sec. 6 of TeK Associates Progress Report that the comupted Nav solution is to be added to (or subtracted from) the delta solution outputed by the Kalman-like filter in order to get the properly corrected Navigation solution to which the GPS data is to be compared for proximity.

The most straight forward way to performing a proximity test to GPS Truth would be to wait until all the navigation data has been processed for the entire time epoch of interest. However, that would require that we also output whole covariances for the quantities that are to participate in the comparison (specifically a 3 by 3 covariance for Lat, Long, and Altitude) for each time at which a comparison is to take place. Rather than output so much time-tagged covariance data and wait to apply the

proximity tests alluded to above, we instead opt to apply it incrementally as we proceed and output all the results as a plot at the end. This appears to be the most expedient path that avoids larger output files and any additional processing.

Helpful comments from an independent outside navigation expert [14] (obtained via an informal e-mail): "I have read only Chapter 10 of Jay Farrell's book [8], which (as far as I know) contains the only book-treatment of the AHRS topic. I am more familiar with the treatment in [4] (Eds. 1 and 2). The Farrell treatment [8] appears similar to [4], but I cannot say the equations are correct because I haven't explicitly checked them, unlike those in [4]. I have re-derived and checked the equations in [4] and I believe they are correct with the following two exceptions: (1) The G matrix in Eq. 12.27, page 344 of [4] Ed. 2 needs three more rows of zeros in the bottom; and (2) the block diagram in Figure 12.3, page 346, has stuff missing, namely a -dL*Omega sin L term that should appear in the upper left summer, and dalpha, dbeta, and dv_E terms going into the summer at the bottom of the figure. But, the F-matrix in [4, p. 345], to which the block diagram corresponds, is correct as it stands with no omissions present in the F-matrix unlike what occurs in the associated block diagram of the figure.

Strapdown navigation error models are all time-varying, except when the platform is at rest, because motion will change the orientation of the input axes of the gyros and accelerometers relative to the local-level frame, because no explicit effort is expended in trying to keep these input axes in any particular orientation in a strapdown mechanizations, unlike what occurs for local-level mechanizations. Consequently, varying velocity and varying acceleration will introduce time variation into the model. This variation can change quickly, i.e., the velocity and acceleration of an aircraft can/will change instantaneously. This variation is in addition to the dependence on Latitude, which also changes, but Latitude changes very slowly, so it can sometimes be treated as a constant. Our old error analysis for local-level is different from a strapdown mechanization because in the old SINS, the gyro and accelerometer input axes were actively kept oriented along a local-level (wander) frame, so the C_B$\hat{\text{L}}$ DCM remains essentially constant (except for the wander frame variation about the vertical).

As always, what error model to use or what terms to retain in the navigation model depends on the application. For example, if your application requires the KF model to be valid for a significant fraction of 24 hours, then the full-blown model of [4] page 345 or Farrell Chapter 11 is required. Probably that is not the case for you, as that would only be meaningful for very high quality inertial systems like the kind that are used in submarines. If your application requires the KF model to be valid for, say, 4 hours (¡¡24 hrs), then you can throw away the Earth-rare and Foucault loops, and retain only two decoupled Schuler loops (N&E) in your Nav model. That is the situation for the F-matrix on page 351 and the phi-matrix on page 353 of [4] Ed. 2, page 353. But notice that those two matrices assume zero acceleration. Additional terms should be introduced if f_E is not zero. Also notice that this state-space model is only for the North-velocity Schuler loop, and an additional similar but different model would be required for the East Schuler loop. The 4-hour model could be useful, for example, if GPS is denied because of jamming, or if the GPS measurements are intermittent because of blockage of the GPS antenna (e.g., within the valleys of obscuring mountain ranges).

If the application requires a KF model that is valid for over shorter periods of time, then the models could be further simplified. For example, see [4, 354ff]. That may be the case if GPS is available at a high data rate, although single-antenna GPS provides position only, so the attitude dynamics, not directly observable by single-antenna GPS may need to be retained." [TeK Associates: Please excuse this last sentence. This expert was not informed beforehand that we were not utilizing GPS other than as a truth reference for OKSI's application. I did not mention OKSI, nor AHRS, nor Terrain maps, nor Visual position fixes. I confined my questions to be of a general nature regarding our two main navigation references [4], [8] and whether he was aware of any blatant errors present in them. His comments helped me although they were in an informal abbreviated form in an e-mail, as a quick answer. In recording his comments here, I filled in the few implied blanks so that it would be more readable to others. Only when I included his comments in this report did I notice that additional words of elaboration would help its readability.]

Vagaries of the data: I looked at the GPS data entries: 1136 rows. It wasn't exactly in the format that I expected but I believe that it is good enough. Altitude is in meters (as I had requested), Lat and Long are in degrees and minutes. Time is in hhmmsss. I worry that time resolution may not be fine enough since every 100 milliesec and every 10 millisec are the two data rate periods. The time format for GPS data is hhmmsss or two digits for hours, two digits for minutes, three digits for seconds leaving only 1/10 sec as the least significant digit for time (100 millisec).

From my initial experience with OKSI data, I learned that one needs to be careful of the filenames upon which the actual data is conveyed. I need to be compatible with MatLab filename conventions (while allowing OKSI to recognize the names of files that they sent me. Unfortunately, I can not allow ";" and "-" to appear in any MatLab filenames and it is better to lead with alphanumeric characters and trail with exclusively numeric designators or date and time separated by underscores to cause filenames to contain only contiguous characters as one "word".

Test Conditions: aircraft moves about 4.4 meters in 0.1 second = 44 meters/sec = 44 x (3.6) km/hr = 158.4 km/hr = $\frac{158.4\text{miles}}{1.609\text{hr}}$ = 98.425 miles/hr.

Video is collected at 30 Hz, and analyzed to produce 6 DOF at some frequency less than 30 Hz (i.e., 10 Hz).

So while inertial data are produced (and recorded) at close to 100 Hz (every 10 msec or so), you do get Video 6 DOF roughly at 10 Hz or every 100 msec.

The inertial and video measurements are nonsynchrnous, so you may need to watch the time stamps and "insert" the video 6DOF between the two appropriate inertial.

it will be most efficient if we deliver to you three separate text files where the data in the files will be delimited by a tab, comma, or space. Sample files are attached.

[14] From Dr. Jorge I. Galdo, a past coworker from 40+ years ago who also overlapped with me at Lincoln Laboratory in the mid to late 1980's.

File #1 - 6DOF and 6×6 error matrix from our video navigation algorithm with time stamps; tab delimited; time is local (Pacific Standard Time); File #2 - AHRS data logged during flight with time stamps; space delimited; time is local (Pacific Standard Time); File #3 - GPS data logged during flight with time stamps; comma delimited; time is UTC.

The time stamp entries for each data set are not synchronized. It will be up to TeK Associates to read the time of each entry and handle as needed. I think the AHRS data is fairly self-explanatory. The video navigation data and the GPS data need a bit of explanation.

In the video nav data, the 6×6 error matrix has been reorganized into a 36×1 vector – the headings describe the position from the original 6×6 matrix. For example, V(x) stands for Variance(x); C(x,y) stands for Covariance(x,y). Right now all of the variance terms are 1 and all of the covariance terms are 0.

The GPS file contains a new position entry 0.5 seconds. The time, latitude, and longitude data follow the GPGGA heading. So, the first GPGGA entry in this file is:

GPGGA,224742.00,3415.7664006,N,11824.8400308,W,1,05,9.5,343.72,M,,,,*1F

Where,

The time is: 22 hours, 47 minutes, and 42.00 seconds The latitude is: 34, 15.7664006', North The longitude is: 118, 24.8400308', West

Interpretation of GPS data file is found in: http://aprs.gids.nl/nmea/#gga

I am currently faced with an airborne application involving a strapdown mechanized INS. I have INS data available at a fairly fast (high) rate (but navaid sensor data at a lower rate) and know that my system is time-varying so I attempt to linearize or re-linearize the system matrix at (using) each available sample of INS data to capture the time-varying nature but still allow a matrix exponential approximation to suffice for each small incremental time step (so that the approximation remains close to true). I used or retain the first three terms of the Taylor series approximation to represent the corresponding matrix exponential.

From the following reference [49], we have that:

- 0.310.58 gauss the Earth's magnetic field at its surface.
- Another unit conversion that may be useful is 1 gauss = 10-4 kg C-1 s-1.
- One gauss is defined as one maxwell per square centimeter; it equals 110-4 tesla (or 100 T).
- The gauss, abbreviated as G, is the cgs unit of measurement of a magnetic field B, which is also known as the "magnetic flux density" or the "magnetic induction".

References

[1] Andrews, G. L., "Implementation considerations for vision-aided inertial navigation," Electrical and Computer Engineering Masters Thesis, Northeastern University, Boston, MA, 2008.

[2] *AHRS400 Series Users Manual,* Document Part #: 7430-004-04 Rev. A MEMSIC, Inc. Milpitas, CA. 2010.

[3] *AHRS400 Attitude & Heading Reference System,* (1 sigma parameter values for AHRS400CD-200), Document Part #: 6020-0025-12 Rev B, pp. 64-65, Crossbow Technology, Inc. San Jose, CA (no publication date was apparent).

[4] Titterton, D. H., and J.L. Weston, *Strapdown Inertial Navigation Technology,* Peter Peregrinus Ltd., Stevenage, Herts, England, UK, 2004.

[5] NGA STANDARDIZATION DOCUMENT: *Pushbroom/Whiskbroom Sensor Model Metadata Supporting Precise Geopositioning (2009-07-21)*, Version 1.0, NGA.SIG.0003_1.0, NATIONAL CENTER FOR GEOSPACIAL INTELLIGENCE STANDARDS, National Geospatial Intelligence Agency, 21 July 2009.

[6] Britting, K. R., *Inertial Navigation Systems Analysis*, Wiley-Interscience, New York, 1971.

[7] Kuipers, Jack B., *QUATERNIONS and ROTATION SEQUENCES: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*, Princeton University Press, Princeton, NJ, 1999.

[8] Farrell, J. A., *Aided Navigation: GPS with High Rate Sensors,* McGraw-Hill, NY, 2008.

[9] Jekeli, C., *Inertial Navigation Systems with Geodetic Applications,* Walter deGruyter GmbH & Co. KG, Berlin, Ger., 2001.

[10] Rogers, R. M., *Applied Mathematics in Integrated Navigation Systems,* AIAA Education Series, AIAA, pp. 99-108, Reston, VA., 2000.

[11] Chatfield, A. B. (Ed.), *Fundamentals of High Accuracy Inertial Navigation,* Paul Zarchan, Editor-in-Chief, AIAA Progress in Astronautics and Aeronautics Series, AIAA, 1997.

[12] Gelb, A. (Ed.), *Applied Optimal Estimation,* MIT Press, Cambridge, MA, 1974.

[13] Maybeck, P., *Stochastic Models, Estimation and Control,* Vol. 1, Academic Press, NY, 1979.

[14] James Goppert, Brandon Wampler, "Development of a Real-Time Navigation System using a Magnetometer/ Vision aided Low-Cost IMU," 30 April 2009 (no indication of where published or of authors' affiliations).

[15] Jaewon Seo, Jang Gyu Lee, Chan Gook Park, "A New Error Compensation Scheme for INS Vertical Channel," *IFAC Proceedings* 2004.

[16] Savage, P. G., "Strapdown Sculling Algorithm Design for Sensor Dynamic Amplitude and Phase-Shift Error," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 6, pp. 1718-1729, Nov.-Dec. 2012.

[17] Kyungsuk Lee, Jason M. Kriesel, Nahum Gat, "Autonomous Airborne Video-Aided Navigation," *Navigation: Journal of the Institute of Navigation*, Vol. 57, No. 3, pp. 163-173, Fall 2010.

[18] Alfano, S., Greer, M. L., "Determining if Two Solid Ellipsoids Intersect," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 1, pp. 106-110, Jan.-Feb. 2003.

[19] Kerr, T. H., "Comments on 'Determining if Two Solid Ellipsoids Intersect'," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 1, pp. 189-190, Jan.-Feb. 2005.

[20] Kerr, T. H., "Integral Evaluation Enabling Performance Trade-offs for Two Confidence Region-Based Failure Detection," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 3, pp. 757-762, May-Jun. 2006.

[21] Kerr, T. H., "Further Critical Perspectives on Certain Aspects of GPS Development and Use, *Proceedings of 57th Annual Meeting of the Institute of Navigation*, pp. 592-608, Albuquerque, NM, 9-13 Jun. 2001.

[22] Kerr, T. H., "Vulnerability of Recent GPS Adaptive Antenna Processing (and all STAP/SLC) to Statistically Non-Stationary Jammer Threats, *Proceedings of SPIE*, Session 4473: Tracking Small Targets, pp. 62-73, San Diego, CA, 29 Jul.-3 Aug. 2001.

[23] Bar-Itzhack, I. Y., and Berman, N., "Control Theoretic Approach to Inertial Navigation Systems," *AIAA Journal of Guidance, Control, and Dynamics,* Vol. 11, No. 3, pp. 237-245, May-June 1988.

[24] Bar-Itzhack, I. Y., and Goshen-Meskin, D., "Observability Studies of Inertial Navigation Systems," *Proceedings of AIAA Journal of Guidance, Navigation, and Control Conference,* Part 2, pp. 1283-1289, Boston, MA, 14-16 Aug. 1989.

[25] Aasnaes, H. B., Kailath, T., "Initial-Condition Robustness of Linear Least Square Filtering Algorithms," *IEEE Transactions on Automatic Control*, Vol. 19, No. 4, pp. 393-397, Aug. 1974.

[26] Powell, T. D., "Automated Tuning of an Extended Kalman Filter Using the Downhill Simplex Algorithm," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 5., pp. 901-908, Sep.-Oct. 2002.

[27] Karabork, Alper, *Terrain Aided Navigation,* Electrical and Electronics Engineering Master of Science Thesis, Graduate School of Natural and Applied Sciences of Middle East Technical University, September 2010. http://scholar.google.com/scholar?start=40&q=related:UJ_i7VZdJ5gJ:scholar.google.com/&hl=en&as_sdt=0,22

[28] Lupash, Lawrence, *MatLab Kalman Filtering Software Toolbox*, User's Guide and Reference Manual, Version 3, L3NAV Systems, 1992-2011.

[29] Lupash, Lawrence, *Coordinate Tranaformation Software Toolbox for MatLab*, User's Guide and Reference Manual, Version 2.02, L3NAV Systems, 1998-2006.

[30] *MatLab Aerospace Toolbox*, (Units conversions and transforming of coordinate systems and spatial representations).

[31] Zhu, J,, "Conversion of Earth-Centered Earth-Fixed Coordinates to Geodetic Coordinates," *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 30, No., pp. 957-961, 1994. (Better than in [29] as identified in National Missile Defense but using results in [29], by selecting best 1 of 3 options, should be good enough for our needs herein.)

[32] Kerr, T. H., "Numerical Approximations and Other Structural Issues in Practical Implementations of Kalman Filtering," a chapter (pp. 193-220) in *Approximate Kalman Filtering*, edited by Guanrong Chen, World Scientific, NY, 1993.

[33] Kerr, T. H., "Exact Methodology for Testing Linear System Software Using Idempotent Matrices and Other Closed-Form Analytic Results," *Proceedings of SPIE*, Session 4473: Tracking Small Targets, pp. 142-168, San Diego, 29 July-3 Aug. 2001.

[34] Zamani, M., Trumpf, J., Mahony, R., "Near-Optimal Deterministic Filtering on the Rotation Group," *IEEE Trans. on Automatic Control*, Vol. 56, No. 6, pp. 1411-1414, June 2011.

[35] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., Oshman, Y., "Kalman Filtering for Matrix Estimation, *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 42, No. 1, pp. 147-159, Jan. 2006 (A linear Matrix Kalman filter for DCM. DCM Refinement #1).

[36] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., Oshman, Y., "Direction Cosine Matrix Estimation from Vector Observations Using a Matrix Kalman Filter, *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 1-11, Aug. 2003 (A linear Matrix Kalman Filter for DMC using either vector or matrix measurement updates. DCM Refinement #2).

[37] Choukroun, D., "A Novel Quaternion Kalman Filter using GPS Measurements, *Proceedings of ION GPS*, Portland, OR, pp. 1117-1128, 24-27 Sep. 2002 (An alternative viewpoint. Quaternion Refinement #1).

[38] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., Oshman, Y., "Kalman Filtering for Matrix Estimation, *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 42, No. 1, pp. 147-159, Jan. 2006 (Quaternion Refinement #2).

[39] Choukroun, D., Bar-Itzhack, I. Y., Oshman, Y., "Novel Quaternion Kalman Filter, *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 42, No. 1, pp. 174-190, Jan. 2006 (Quaternion Refinement #3).

[40] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., Oshman, Y., "Direction Cosine Matrix Estimation From Vector Observations Using A Matrix Kalman Filter, *Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, TX, pp. 1-11, 11-14 August 2003 (DCM Refinement #3).

[41] Choukroun, D., "Ito Stochastic Modeling for Attitude Quarternion Filtering, *Proceedings of Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P. R. China, pp. 733-738, 16-18 Dec. 2009 (Quaternion Refinement #4).

[42] Cheng, Y., Landis Markley, F., Crassidis, J. L. Oshman, Y., "Averaging Quaternions, *Advances in the Astronautical Sciences series*, Vol. 127, American Astronautical Society, AAS paper No. 07-213, 2007.

[43] Landis Markley, F., "Attitude Filtering on SO(3), *Advances in the Astronautical Sciences series*, Vol. 122, American Astronautical Society, AAS paper No. 06-460, 2006.

[44] Cheng, Y., Crassidis, J. L., and Landis Markley, F., "Attitude Estimation for Large Field-of-View Sensors, *Advances in the Astronautical Sciences series*, Vol. 122, American Astronautical Society, AAS paper No. 06-462, 2006.

[45] Landis Markley, F., "Attitude Estimation or Quaternion Estimation?, *Advances in the Astronautical Sciences series*, Vol. 115, American Astronautical Society, AAS paper No. 03-264, 2003 (Critical and thorough Analysis of 3 different EKFs vs. use of Technion MKF. MKF was improved.)

[46] Reynolds, R., Landis Markley, F., Crassidis, J. L., "Asymptotically Optimal Attitude and Rate Bias Estimation with Guaranteed Convergence, *Advances in the Astronautical Sciences series*, Vol. 132, American Astronautical Society, AAS paper No. 08-286, 2008.

[47] Kopka, H., Daly, P. W., *A Guide to LaTeX: document preparation for beginners and advanced users*, Addison-Wesley Publishing Company, Reading, MA, 1993.

[48] Etter, D. M., *Engineering Problem Solving with MatLab*, 2nd Edition, Prentice Hall, Upper Saddle River, NJ, 1997.

[49] http://www.engineeringtoolbox.com/acceleration-converter-d_1284.html

[50] http://www.memsic.com/magnetic-sensors/