Refinements to the General Methodology Behind Strapdown Airborne Gravimetry

AE 8900 MS Special Problems Report
Space Systems Design Lab (SSDL)
Guggenheim School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA

Author:
Kevin Lee Seywald

Advisor:
Prof. Brian C. Gunter

August 5, 2016

Refinements to the General Methodology Behind Strapdown Airborne Gravimetry

Kevin L. Seywald* and Brian C. Gunter†

AE 8900 Special Problems Report

Measuring Earth's gravitational field has important applications in fields ranging from geodesy to exploration geophysics. Gravity field disturbances are typically no more than 100 mGal, hence requiring extremely precise sensors. The estimation of error sources inherent in these sensors, such as bias, scale factor, and drift rate, significantly improve the accuracy of these measurements, allowing for more precise gravity estimates. This research builds upon prior work using a strapdown Inertial Navigation System (INS) paired with Global Positioning Systems (GPS) for airborne platforms. In order to test and validate the processing algorithms, various simulated test cases were created. Several refinements were made to the traditional approach found in the literature, making the process more robust. Most notably, an analytical solution was developed for the quaternion integration problem, which is typically implemented using numerical methods. The analytical solution limits the integration error to machine precision, and removes any error propagation. Furthermore, the error equations implemented in the Kalman Filter were refined such that they better capture the true dynamics of the error-states. These changes to the existing methodology were validated by the proposed algorithm's ability to accurately estimate the parameters used to generate the simulated flight data.

Nomenclature

t	Time; independent variable unless stated otherwise
x	Dependent variable; $x = [x_1, .., x_n]^T \in \mathbb{R}^n$ unless stated otherwise
ω^c_{ab}	Angular rate of the b-frame with respect to the a-frame, expressed in c-frame coordinates
C^a_b	transformation matrix *from* the a-frame *to* the b-frame
$\|\| \cdot \|\|$	Euclidean norm

Subscript

a	Accelerometer
g	Gyroscope
wg	"With gravity", used to describe acceleration data. a_{wg} describes total kinematic acceleration
ng	"Non gravitational", used to describe acceleration data. a_{ng} describes what the accelerometers can sense

Superscript

i	Expressed in inertial-frame coordinates
b	Expressed in body-frame coordinates
n	Expressed in navigation-frame coordinates
e	Expressed in Earth-Centered Earth Fixed frame coordinates
$\wedge$	Calculated using measured data
$\sim$	Sensed data
$\vee$	True data

*Master's Student, Space Systems Design Laboratory, Georgia Institute of Technology

†Assistant Professor, Space Systems Design Laboratory, Georgia Institute of Technology

I. Background and Motivation

Earth's gravity field contains important information about Earth's distribution of mass, which is particularly of value to the fields of geodesy and geophysics. Knowledge of Earth's gravity field can be used to discover variations in density, useful for exploring mineral and petrochemical deposits,[11] or precisely defining the shape of the Earth.[10] Periodic gravimetric surveys can also lend insight into changes of the distribution of mass over time, applicable, for example, to surveying the changes of the ice caps,[12] or perhaps even to forewarn of an impending volcanic eruption.

Many methods exist for measuring gravity, from either ground-based, airborne, or satellite platforms. Measurements taken closer to the surface of the Earth are preferable as satellite based systems can only resolve the mid to long wavelength (>100km) gravity signals. While stationary ground-based gravimetry provides the highest accuracy, they are point measurements by nature, and require significant effort to adequately sample large areas.[4] A dynamic platform provides a faster and potentially more cost-effective alternative. The focus of this study will be on airborne applications, in particular those related to Unmanned Aerial Vehicles (UAVs), which offer a viable and cost-effective option for gravimetry.

A strapdown IMU setup is necessary in order to employ small UAVs, since gimbaled systems, which provide a platform constantly perpendicular to the gravity vector, are both bulky and expensive. Gimbal-mounted sensors were the norm up until 1990, when Wei and Schwarz[8] developed an algorithm for processing strapdown inertial data. In the strapdown configuration, the sensors are mounted to the body of the vehicle itself, as opposed to being mounted to a stabilized platform. Hence the data from the sensors are output in body-frame coordinates. Although the foundation for strapdown gravimetry had been developed back in 1990, the technology was not advanced enough for tasks that demanded high accuracy. Even as recently as 2010, the work of Inacio[9] concluded that accelerometers needed to improve by a factor of three before an accuracy of +/- 0.5 mGal ($1Gal = 1cm/s^2$) could be achieved, the necessary level for natural resource exploration.

Other studies have confirmed that sensor technology is on the cusp of being sufficiently accurate for the purpose of strapdown gravimetry. In 2011, Li tested the accuracy of a navigation-grade IMU and a tactical grade IMU by comparing their results against control data obtained from the Gravity-Lidar Study of 2006 and the NGS database.[5] Using the same method as described in his 2007 dissertation[4] (the main source of dynamics equations for this paper), estimates for the North, East, and Down components of the gravity disturbance vector were made. When compared against the gravity vector's down component from the NGS data, the tactical grade IMU's estimates fluctuated +/- 15 mGal about the control data, an unacceptable amount of error for precise geodetic purposes. The navigation grade IMU's estimates, on the other hand, erred by at most 3 mGal. This level of accuracy is nearly sufficient for natural resource exploration, one of the more demanding applications of gravimetry, which requires errors be no greater than 2 mGal.[9] While Li does not state what the actual bias and scale factor of the two IMUs are, only that the bias and scale factor of the tactical grade IMU were about one order of magnitude higher than that of the navigation grade IMU, the conclusion can be made that precise gravimetry is possible using an airborne strapdown IMU setup, as long as the IMU is "navigation-grade".

Properly measuring the differences in Earth's gravity requires highly accurate sensors, as disturbances in the gravity vector commonly range up to 100 mGal in each axis. Today's commercial accelerometers typically have biases around $4mg$ [a] (g being standard gravity, or $\sim 9.8m/s^2$), equivalent to 39.2 mGal. Hence, the use of filters to estimate the errors inherent in these sensors is crucial to obtaining a reliable estimate for the gravity disturbance. The capability to accurately estimate an accelerometer's bias of 50 mGal, using the previous example, to within 95% of its true value would decrease the error to only 1.96 mGal (ignoring all other sources of error), just small enough to be acceptable for use in subsurface resource exploration. The precise modeling of the error dynamics such that bias, scale factor, drift rate, and noise are estimated to the greatest possible accuracy could therefore be critical in determining whether a UAV-mounted strapdown IMU can become a feasible source of accurate, fast, and cost-effecting gravity data.

This study takes a critical look at the equations underlying the IMU data integration and the Kalman Filter(KF) dynamics as expressed in Jekeli[3] and Li.[4] Various refinements were made to the approach of Li and Jekeli to make the estimation of the various parameters more numerically stable, as well as to increase the accuracy of the KF estimates. These corrections, with their derivations, are presented here alongside

[a] Specifications for Honeywell's QA3000 accelerometer found at *https : //aerospace.honeywell.com/*, last accessed August 1, 2016

the original equations for reference. While there was insufficient time for this research to be applied to a simulation attempting to measure gravity disturbances, it is hoped that this work might be used to increase the accuracy of strapdown IMU measurements, and that it might serve as a foundation for future research.

II. Methodology

An Extended Kalman Filter (EKF) was initially assembled according to the algorithms outlined in Li,[4] specifically equations 3.2.19 and 3.2.22. 3.2.19 outlines the dynamics of the error states, and 3.2.22 shows how the position and velocity calculated by integrating the gyroscope and accelerometer data is differenced with the GPS data and used as the measurement for the KF.

In order to test and verify that the EKF was working as intended, the EKF needed to return the same parameters that were used to generate the sensed data. For example, if an accelerometer were assigned a bias of b and scale factor of k, then the EKF was expected to return bias and scale factor estimates within a reasonable neighborhood of b and k. The process behind generating the simulated data is outlined below.

A. Data Simulation

To help the reader see how the following sections all fit together, a brief summary of the data simulation is given here before jumping into the equations. First, true position and angle data are defined as functions of time. The position functions are defined in the inertial frame, while the angular rates are defined in the body frame. The true position is used to calculate true total acceleration and velocity in the inertial frame. Then, bias, scale factor, and noise are added to both the acceleration and angular rate data sets; this becomes the "sensed" data set, or what the INS would output. Finally, the true inertial position and velocity is transformed from the inertial frame to the Earth-Centered-Earth-Fixed (ECEF) frame to generate the GPS data set.

1. Generation of the INS Data

First, the biases and scale factors are defined. Let $y(t)$ be any state that is measured and used as an input. Then we assume that the sensed value of that state, $\tilde{y}(t)$, is related to the actual value of the state, $\check{y}(t)$, as follows:

$$\tilde{y}(t) = \check{y}(t) \cdot (1 + k) + b + \epsilon \tag{1}$$

where k is the scale factor, b is the bias, and ϵ is white noise with a known covariance. Note that while it may be fine to assume the bias and scale factor are constants over short periods, they do, in fact, change with time, typically modeled with a random walk process about a fixed constant. However, for the purpose of testing whether or not a Kalman Filter can reconstruct the biases and scale factors used to generate the simulated data, modeling them as dynamic is not done here.

The subscripts a and g are used to distinguish between accelerometer and gyroscope parameters, respectively. For example, b_a denotes accelerometer bias, and k_g denotes gyroscope scale factor. Representative values for these parameters are provided in Table 1, taken from Jekeli[3](page 195).

Table 1. Representative Parameter Values

Accelerometer:	bias repeatability	220 mgal
	scale factor error	300 ppm
	white noise	40 mgal/$\sqrt{Hz}$
Gyroscope:	bias repeatability	0.005°/hr
	scale factor error	10 ppm
	white noise	0.002°/$\sqrt{hr}$

The true position, $\check{p}$, in the inertial frame and true angular rates, $\check{\omega}$, in the body frame are also defined by the user as functions of time, with each axis receiving its own function.

$$\begin{pmatrix} \breve{p}_1^i(t) \\ \breve{p}_2^i(t) \\ \breve{p}_3^i(t) \end{pmatrix} = \begin{pmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \end{pmatrix} \quad , \quad \begin{pmatrix} \breve{\omega}_{ib,1}^i(t) \\ \breve{\omega}_{ib,2}^i(t) \\ \breve{\omega}_{ib,3}^i(t) \end{pmatrix} = \begin{pmatrix} f_4(t) \\ f_5(t) \\ f_6(t) \end{pmatrix} \tag{2}$$

Differentiating position yields velocity, v, and differentiating again yields acceleration. Note that this yields the total acceleration, a, meaning that it includes both the acceleration due to specific forces as well as acceleration due to gravity.

$$\begin{pmatrix} \breve{a}_{wg,1}^i(t) \\ \breve{a}_{wg,2}^i(t) \\ \breve{a}_{wg,3}^i(t) \end{pmatrix} = \begin{pmatrix} \ddot{\breve{p}}_1^i(t) \\ \ddot{\breve{p}}_2^i(t) \\ \ddot{\breve{p}}_3^i(t) \end{pmatrix} = \begin{pmatrix} \ddot{f}_1(t) \\ \ddot{f}_2(t) \\ \ddot{f}_3(t) \end{pmatrix} \tag{3}$$

We next assign the initial attitude such that we may use the angle data to integrate forward from that orientation. Attitude can be initialized by measuring the Euler angles between the body and navigation frames. Those initial Euler angles can be used to calculate the transformation from the body to the navigation frame, C_b^n

$$C_b^n = \begin{pmatrix} cos(\gamma)cos(\beta) & cos(\gamma)sin(\beta)sin(\alpha) + sin(\gamma)cos(\alpha) & -cos(\gamma)sin(\beta)cos(\alpha) + sin(\gamma)sin(\alpha) \\ -sin(\gamma)cos(\beta) & -sin(\gamma)sin(\beta)sin(\alpha) + cos(\gamma)cos(\alpha) & sin(\gamma)sin(\beta)cos(\alpha) + cos(\gamma)sin(\alpha) \\ sin(\beta) & -cos(\beta)sin(\alpha) & cos(\beta)cos(\alpha) \end{pmatrix} \tag{4}$$

Then the initial C_b^i is given by

$$C_b^i = C_n^i \cdot C_b^n \tag{5}$$

where $C_n^i = C_e^i \cdot C_n^e$,

$$C_e^i = \begin{pmatrix} cos(\omega_e(t)) & -sin(\omega_e(t)) & 0 \\ sin(\omega_e(t)) & cos(\omega_e(t)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6}$$

where

$$\omega_e \approx \begin{pmatrix} 0 \\ 0 \\ 7.3e-5 \end{pmatrix} \tag{7}$$

and

$$C_n^e = \begin{pmatrix} -sin(\phi)cos(\lambda) & -sin(\lambda) & -cos(\phi)cos(\lambda) \\ -sin(\phi)sin(\lambda) & cos(\lambda) & -cos(\phi)sin(\lambda) \\ cos(\phi) & 0 & -sin(\phi) \end{pmatrix} \tag{8}$$

where λ and ϕ are the longitude and latitude, respectively, and can be calculated from the position relative to the Earth using

$$\phi = atan2\left(p_3^e, \sqrt{(p_1^e)^2 + (p_2^e)^2}\right) \tag{9}$$

$$\lambda = atan2\left(p_2^e, p_1^e\right) \tag{10}$$

where $p^e = [p_1^e, p_2^e, p_3^e]^T$ is the position expressed in ECEF coordinates. After establishing the initial attitude, the sensed gyro data is generated by adding the error sources to the true data as follows:

$$\begin{pmatrix} \tilde{\omega}^i_{ib,1}(t) \\ \tilde{\omega}^i_{ib,2}(t) \\ \tilde{\omega}^i_{ib,3}(t) \end{pmatrix} = \begin{pmatrix} [\breve{\omega}^i_{ib,1}(t)] \cdot (1 + k_{g,1}) + b_{g,1} + \epsilon_{g,1} \\ [\breve{\omega}^i_{ib,2}(t)] \cdot (1 + k_{g,2}) + b_{g,2} + \epsilon_{g,1} \\ [\breve{\omega}^i_{ib,3}(t)] \cdot (1 + k_{g,3}) + b_{g,3} + \epsilon_{g,1} \end{pmatrix} \tag{11}$$

Next, the true attitude is found by integrating the true angle data. The angle data must first be transformed into quaternions and then integrated, since quaternions provide a robust means of representing rotations that is not subject to singularities as with Euler angles. The relationship between quaternion dynamics and the angular rates is given by:

$$\dot{q}(t) = \frac{1}{2} A(t) \cdot q(t) \tag{12}$$

$$q_0 = q(t_0) \tag{13}$$

where

$$A(t) = \begin{pmatrix} 0 & \omega_1(t) & \omega_2(t) & \omega_3(t) \\ -\omega_1(t) & 0 & \omega_3(t) & -\omega_2(t) \\ -\omega_2(t) & -\omega_3(t) & 0 & \omega_1(t) \\ -\omega_3(t) & \omega_2(t) & -\omega_1(t) & 0 \end{pmatrix} \tag{14}$$

and

$$\omega(t) = \begin{pmatrix} \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \end{pmatrix} \tag{15}$$

q_0 is found by converting the initial C^i_b matrix into a quaternion using

$$q_0 = \begin{pmatrix} a_0 \\ b_0 \\ c_0 \\ d_0 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \cdot (1 + (C^i_b)_{1,1} + (C^i_b)_{2,2} + (C^i_b)_{3,3})^{\frac{1}{2}} \\ \frac{1}{4a_0} \cdot ((C^i_b)_{2,3} - (C^i_b)_{3,2}) \\ \frac{1}{4a_0} \cdot ((C^i_b)_{3,1} - (C^i_b)_{1,3}) \\ \frac{1}{4a_0} \cdot ((C^i_b)_{1,2} - (C^i_b)_{2,1}) \end{pmatrix} \tag{16}$$

Numerous algorithms of varying accuracy exist for integrating the quaternions. Jekeli[3] introduces a second and third order algorithm, which are defined for the general case: the gyroscopes are outputting the angular velocity of the body frame with respect to the inertial frame, expressed in the body frame, ω^b_{ib}, and the transformation between the body-frame and some arbitrary a-frame, C^a_b, is desired. It is important to note here that gyroscopes do not, in fact, directly measure angular velocity. Instead, the change in angle is measured, and together with the sampling rate of the gyroscope, an approximation can be made for the angular velocity. This distinction is noteworthy as the following analytical solution, as well as the numerical solutions proposed by Jekeli, depend upon the change in angles as inputs, and not the angular rates.

However, in the specific case where C^i_b is the desired solution, and not some arbitrary C^a_b, an analytical solution exists, given as follows:

$$q(t) = \begin{cases} \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times4} + sin\left(\frac{\|\theta(t)\|}{2}\right) \cdot \frac{B(t)}{\|\theta(t)\|} \right] \cdot q_0 & \text{for} \quad \|\theta(t)\| \neq 0 \\ \left[I_{4\times4} \right] \cdot q_0 & \text{for} \quad \|\theta(t)\| = 0 \end{cases} \tag{17}$$

where

$$\theta(t) = \begin{pmatrix} \theta_1(t) \\ \theta_2(t) \\ \theta_3(t) \end{pmatrix} = \begin{pmatrix} \int_{t_0}^{t} \omega_1(\tau)\, d\tau \\ \int_{t_0}^{t} \omega_2(\tau)\, d\tau \\ \int_{t_0}^{t} \omega_3(\tau)\, d\tau \end{pmatrix} \tag{18}$$

$$\left\|\theta(t)\right\| = \sqrt{\theta_1(t)^2 + \theta_2(t)^2 + \theta_3(t)^2} \tag{19}$$

and

$$B(t) = \int_{t_0}^{t} A(\tau)\, d\tau = \begin{pmatrix} 0 & \theta_1(t) & \theta_2(t) & \theta_3(t) \\ -\theta_1(t) & 0 & \theta_3(t) & -\theta_2(t) \\ -\theta_2(t) & -\theta_3(t) & 0 & \theta_1(t) \\ -\theta_3(t) & \theta_2(t) & -\theta_1(t) & 0 \end{pmatrix} \tag{20}$$

The proof of this analytical solution is found in section III.

After integrating the quaternion to find $q(t) = [a, b, c, d]^T$, it is converted back into a Directional Cosine Matrix (DCM) via

$$C_b^i = \begin{pmatrix} a^2 + b^2 - c^2 - d^2 & 2 \cdot (bc + ad) & 2 \cdot (bd - ac) \\ 2 \cdot (bc - ad) & a^2 - b^2 + c^2 - d^2 & 2 \cdot (cd + ab) \\ 2 \cdot (bd + ac) & 2 \cdot (cd - ab) & a^2 - b^2 - c^2 + d^2 \end{pmatrix} \tag{21}$$

Now that the transformation matrix is known at every time t, we move on to how the sensed accelerometer data is generated. Accelerometers only sense accelerations caused by specific forces, not those caused by gravitational fields, so before we add the biases and scale factors to create the sensed data, we must first subtract gravity and transform into the body frame.

$$\check{a}_{ng}^i(t) = \check{a}_{wg}^i(t) - \check{g}^i(t) \tag{22}$$

$\check{g}^i$ is modeled as

$$\check{g}^i \approx -(\frac{kM}{r^2}) \cdot \boldsymbol{n} \tag{23}$$

where

$$r^2 = (\check{p}_1^i)^2 + (\check{p}_2^i)^2 + (\check{p}_3^i)^2 \tag{24}$$

$\boldsymbol{n} = \check{p}^i / r$ is a unit vector pointing radially outward from the center of the Earth, k is the gravitational constant, and M is the mass of the Earth ($kM \approx 3.986 \times 10^{14} m^3/s^2$).

Next, $\check{a}_{ng}^i$ is converted to the body frame.

$$\check{a}_{ng}^b(t) = C_b^i(t) \cdot \check{a}_{ng}^i(t) \tag{25}$$

$\check{a}_{ng}^b$ is essentially what the accelerometers would output if they were perfectly precise. Combining $\check{a}_{ng}^b$ with the bias and scale factor, we finally have the sensed accelerations.

$$\begin{pmatrix} \tilde{a}_1^b(t) \\ \tilde{a}_2^b(t) \\ \tilde{a}_3^b(t) \end{pmatrix} = \begin{pmatrix} [\check{a}_1^b(t)] \cdot (1 + k_{a,1}) + b_{a,1} + \epsilon_{a,1} \\ [\check{a}_2^b(t)] \cdot (1 + k_{a,2}) + b_{a,2} + \epsilon_{a,1} \\ [\check{a}_3^b(t)] \cdot (1 + k_{a,3}) + b_{a,3} + \epsilon_{a,1} \end{pmatrix} \tag{26}$$

Equation (26) together with the sensed angular rates in equation (11), list all the outputs of the INS. All that is left is to generate the simulated GPS data.

2. Generation of the GPS Data

For the purposes of this simulation, GPS is treated as a nearly perfect measurement, only corrupted by white noise. Thus, GPS data is simply simulated as

$$v_{GPS} = \check{v}^i + \epsilon_{GPS} \tag{27}$$

$$p_{GPS} = \check{p}^i + \epsilon_{GPS} \tag{28}$$

The standard deviation of this white noise was assumed to be on the order of 2-3 cm. Note that this is assuming that the raw GPS data has already been processed by a separate algorithm, which is not covered in this paper. The noise magnitudes used to generate the sensed data are provided in table 2.

Table 2. Noise Used to Generate Data

Sensor	Noise Magnitude
Accelerometer	$10^{-5}(m/s^2)$
Gyroscope	$10^{-5}(rad/s)$
GPS	$10^{-2}(m)$

B. Data Processing

Once the accelerometer and gyroscope data are received, they must be integrated in order to calculate inertial position. As a brief summary of the process, the sensed angular rates are first used to calculate $\hat{C}_b^i$, which is then used to convert the accelerations into the inertial frame, giving us $\hat{a}_{ng}^i$. Summing this with an estimate of gravity, we get $\hat{a}_{wg}^i$, which is integrated to find velocity, $\hat{v}^i$, and then position in the inertial frame, $\hat{p}^i$. The difference between the position and velocity calculated from INS data and that received from the GPS is fed into the EKF as the measurement. By weighing the measurements against the propagated state using their respective uncertainties, the EKF calculates the most probable state, consisting of position error, velocity error, orientation error, biases, and scale factors. Comparing the estimated state against the known true data allow for an assessment of the EKF's performance.

1. Velocity and Position Integration

Before we can derive position from the accelerometer and gyro data, the accelerations must be converted to the inertial frame and an estimate of gravity must be made. Acceleration at time $t-1$, and hence gravity at time $t-1$, must be known in order to propagate position from time $t-1$ to t. Essentially, we must reconstruct the total inertial acceleration at each point in time according to

$$\hat{a}_{wg}^i(t-1) = \hat{a}_{ng}^i(t-1) + \hat{g}^i(t-1) \tag{29}$$

$$= \hat{C}_b^i(t-1) \cdot \tilde{a}^b(t-1) + \hat{g}^i(t-1) \tag{30}$$

Therefore, we need $\hat{C}_b^i$ at every time step. $\hat{C}_b^i$ is calculated the same way that we calculated C_b^i in the data simulation process, except that sensed data is used instead of the true data. First, the initial attitude is determined using equations (4) through (10), then converted to a quaternion using equation (16). Next, using equation (17), the quaternion is propagated forward to each time step, and then converted back to a DCM using equation (21). This gives us $\hat{C}_b^i(t)$ for all t and allows us to transform the sensed accelerations from the body frame into the inertial frame, giving us $\tilde{a}_{ng}^i(t)$.

Next, gravity must be estimated. Gravity is evaluated as function of position according to

$$\hat{g}^i(t-1) = -\left[(\frac{kM}{r^2}) \cdot \boldsymbol{n} \right]_{\hat{p}^i(t-1)} \tag{31}$$

Thus, all components of equation (28) are now known, and $\hat{a}_{wg}^i(t-1)$ can be calculated. Finally, $\hat{v}^i$ and $\hat{p}^i$ can be calculated by plugging $\hat{a}_{wg}^i(t-1)$ in to a fixed-step Ordinary Differential Equation (ODE) solver.

2. Kalman Filter

The Kalman Filter, first introduced by Rudolph E. Kalman in 1960, has been around for over half a century. An attempt at a clear and simple explanation is made here. Readers looking for more information are encouraged to see Simon's *Optimal State Estimation*.[6]

The Kalman Filter can be broken down into two main portions: prediction and update. In the prediction phase, a model of the state dynamics is used to propagate the state forward through time. In the update

phase, the KF receives a measurement that is in some way related to the state; this allows the KF to calculate the state that would have made the measurement possible. Based on the covariances of the propagated state estimate and the measurement, the KF assigns weights to each and calculates the most probable state, a weighted average of the two.

The prediction phase is driven by the dynamics of the model, how the state changes with time. This relationship is used to solve for the state transition matrix, $\Phi(t, t_0)$, which is in turn used to propagate both the state and the state covariance matrices forward. The dynamics model used in the author's code, shown below, closely follows the equations found in Li,[4] with the exception of a few changes. The differences between (33) and Li's equation (3.2.19) are discussed more in-depth in section III.

First we define our state vector. Since we are trying to solve for the biases and scale factors, the state vector contains δx^i and $\delta \dot{x}^i$, not x^i and $\dot{x}^i$. We are also interested in the orientation error, as well as the biases and scale factors. The state vector is therefore defined as

$$X = \begin{pmatrix} \delta x^i \\ \delta \dot{x}^i \\ \psi^i \\ b_a \\ b_g \\ k_a \\ k_g \end{pmatrix} \tag{32}$$

To begin solving for the state transition matrix, $\Phi(t, t_0)$, we find the relationship between $\dot{X}$ and X, defined below.

$$\frac{d}{dt}\begin{pmatrix} \delta x^i \\ \delta \dot{x}^i \\ \psi^i \\ b_a \\ b_g \\ k_a \\ dr \end{pmatrix} = \begin{pmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ \Gamma^i & 0 & [\check{C}_b^i \delta a^b \times] + [\check{a}_{ng}^i \times] & \hat{C}_b^i & 0 & \hat{C}_b^i \cdot [\check{a}_{ng}^b] & 0 \\ 0 & 0 & 0 & 0 & -\check{C}_b^i & 0 & [t - t_0] \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \delta x^i \\ \delta \dot{x}^i \\ \psi^i \\ b_a \\ b_g \\ k_a \\ dr \end{pmatrix}$$
$$+ \begin{pmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \hat{C}_b^i & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\check{C}_b^i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{pmatrix} \cdot \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \\ \epsilon_7 \end{pmatrix} \tag{33}$$

where

$$[t - t_0] = diag\{[t - t_0, t - t_0, t - t_0]\} \tag{34}$$

This can be rewritten, for simplicity, as

$$\frac{d}{dt}X(t) = F(t) \cdot X(t) + G(t) \cdot \bar{\epsilon}(t) \tag{35}$$

The above linear, first-order system of differential equations has a solution of the form

$$x(t) = \Phi(t, t_0) \cdot x(t_0) + \int_{t_0}^{t} \Phi(t, t') \cdot G(t') \cdot x(t') dt' \tag{36}$$

where Φ is the state transition matrix and satisfies

$$\frac{d}{dt}\Phi(t,t') = F(t)\cdot\Phi(t,t') \tag{37}$$

which has the solution

$$\Phi(t,t') = e^{F(t-t')} \tag{38}$$
$$= I + F(t-t') + \frac{1}{2!}(F(t-t'))^2 + \frac{1}{3!}(F(t-t'))^3 + ... \tag{39}$$
$$= \sum_{n=0}^{\infty}\frac{(F(t-t'))^n}{n!} \tag{40}$$

$\Phi_{k,k-1}$ propagates the state forward according to the following:

$$X_k = \Phi_{k,k-1}X_{k-1} + w_k \tag{41}$$

where

$$w_k \sim (0, Q_k) \tag{42}$$

With the state transition matrix defined, we now move on to describe the update phase, in which a measurement is received. In this case, the measurement is defined as the difference between the position and velocity derived from the INS data and the position and velocity received from the GPS. The relationship between the measurements and the state remain unchanged from Li's paper. They are given as

$$z_k = H_k\cdot X_k + v_k \tag{43}$$

where

$$z_k = \begin{pmatrix}\delta x^i\\ \delta\dot{x}^i\end{pmatrix} = \begin{pmatrix}\hat{p}^i - p^i_{GPS}\\ \hat{v}^i - v^i_{GPS}\end{pmatrix} \tag{44}$$

$$H = \begin{pmatrix}I & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & I & 0 & 0 & 0 & 0 & 0\end{pmatrix} \tag{45}$$

and

$$v_k \sim (0, R_k) \tag{46}$$

Next, the KF algorithm must be initialized. Since the state vector X is comprised of the error states, the estimate of the initial state, $\hat{X}(t_0)$, is assigned a 21×1 vector of zeros, unless there is some prior knowledge about the parameters available. If the parameters associated with the sensors are known to some extent, either from the manufacturer or from prior tests, then those values may be assigned in the initial state estimate. The covariance matrix of these initial estimates becomes $\hat{P}(t_0)$.

Now that we have Φ, H, Q, R, and the initial state and state covariance defined, we can implement the EKF according to the following algorithm:

Predict

$$\hat{X}^-_k = \Phi_{k,k-1}\hat{X}^+_{k-1} \tag{47}$$
$$\hat{P}^-_k = \Phi_{k,k-1}\hat{P}^+_{k-1}\Phi^T_{k,k-1} + G_{k-1}Q_{k-1}G^T_{k-1} \tag{48}$$

Update

$$K_k = \hat{P}^-_k H^T_k (H_k\hat{P}^-_k H^T_k + R_k)^{-1} \tag{49}$$
$$\hat{X}^+_k = \hat{X}^-_k + K_k(z_k - H_k\hat{X}^-_k) \tag{50}$$
$$\hat{P}_k = (I - K_kH_k)\hat{P}^-_k \tag{51}$$

Note that the frequency of the INS data is likely to be significantly less than that of the GPS signal, meaning that the *Update* portion of the KF code will not take place every time step. During those periods in between measurements, the state is merely propagated forward using the INS data, and at the end of every *Predict* phase, $\hat{X}_k^+ = \hat{X}_k^-$, and $\hat{P}_k^+ = \hat{P}_k^-$.

The final result of the KF will be an estimate $\hat{X}(t)$ at every time t in the integration interval. Since $\hat{X} = [\delta x^i, \delta \dot{x}^i, \psi^i, b_a, b_g, k_a, k_g]^T$, and $\delta x = \hat{x}^i - \check{x}^i$, the filter's estimate for true position and velocity can be calculated using

$$v_{KF}^i = \hat{v}^i - \delta \dot{x}^i \tag{52}$$

$$p_{KF}^i = \hat{p}^i - \delta x^i \tag{53}$$

To verify that the KF has been coded properly, compare $v_{KF}^i(t)$ with $\check{v}^i(t)$, and $x_{KF}^i(t)$ with $\check{x}^i(t)$, as well as the estimated biases, scale factors, and drift rate against the known true values. Although, instead of comparing the entire time series, it may be simpler to check the KF's final estimates of b_a, k_a, b_g, and k_g with their corresponding true values, since it will take the KF a number seconds to converge to the proper value. On the other hand, plotting the whole time series of these parameters may lend insight to those having convergence issues.

III. Refinements

The vast majority of the algorithm described above is not original work. In order to clarify the innovations presented in this paper, deviations from previous works are described here, along with their justification.

A. Innovations

1. Quaternion Integration

The refinement to Jekeli's numerical solution to the quaternion differential equation, mentioned in section 1, is a significant upgrade to existing algorithms. It provides an analytical solution to a problem which others have been approximating numerically. Furthermore, since the solution is formulated such that it is propagated from the initial quaternion, and not from the previously calculated quaternion, it is not subject to propagation error; errors remain capped at machine precision no matter the length of the integration interval. Of course, errors in the estimation of the initial quaternion will be propagated as well, however this is the case for both the analytical solution as well as numerical methods. Lastly, the analytical solution requires at most the same computation time as most numerical approximations. Accurate Taylor Series approximations demand the calculation of a high number of terms, and Runge-Kutta approximations require the evaluation of the function at the beginning, middle, and end of the integration interval; the analytical solution is requires only the evaluation of a single equation.

The analytical solution, as expressed in equation (69), provides the quaternion q_s^t describing the rotation from some arbitrary s-frame to another arbitrary t-frame, as long as the angular velocity ω_{ts}^s is known. Note that the sub- and superscripts on q are not standard notation, and are used here briefly only to emphasize that q, as used in these following equations, describes a rotation between two reference frames, analagous to the transformation matrix C_s^t. This research is primarily concerned with the strapdown methodology, in which gyroscopes are mounted directly to body, yielding the angular velocity of the body frame with respect to the inertial frame expressed in body frame coordinates, ω_{ib}^b. In an effort to keep the following equations uncluttered, the sub- and superscripts on ω and θ have been dropped, i.e. for the rest of this section,

$$\omega(t) = \omega_{ib}^b(t) \tag{54}$$

and

$$\theta(t) = \int_{t_0}^{t} \omega_{ib}^b(\tau) d\tau \tag{55}$$

While it is unlikely that the analytical solution presented here has never before been expressed, the author was unable to find any usage of or reference to it in relevant literature. Thus, the analytical solution

is restated here in the form of a new theorem, along with its proof. But first, another theorem is necessary in order to lay the groundwork.

Theorem 1. *Consider the following linear homogeneous initial value problem*

$$\dot{x}(t) = A(t)\, x(t) \tag{56}$$

$$x(t_0) = x_0 \tag{57}$$

where $t \in \mathbb{R}$ is the independent variable, $x \in \mathbb{R}^n$ denotes the state, and $A \in \mathbb{R}^{n,n}$ is continuous and bounded, i.e., for any interval $I \subseteq \mathbb{R}$, $\exists\, M \geq 0$ such that

$$\big\|A(t)\big\| \leq M \quad \forall t \in I \tag{58}$$

Then the initial value problem (56), (57) has the unique solution

$$x(t) = e^{\int_{t_0}^{t} A(\tau) d\tau} x_0 \tag{59}$$

Proof. By the Picard-Lindelöf Theorem (see Theorem 1.1 of[1]), the existence and uniqueness of the solution are guaranteed if

i the right-hand side of (56) is continuous in t

ii the right-hand side of (56) is continuous in x

iii the right-hand side of (56) is uniformly Lipschitz bounded with respect to x

Conditions (i) and (ii) are satisfied if A(t) is continuous. As for (iii), a function $f(x,t) : \mathbb{R}^n \times I \to \mathbb{R}^n$ is uniformly Lipschitz bounded on the interval $I \subseteq \mathbb{R}$ if there exists a constant L such that

$$\big\|f(x,t) - f(y,t)\big\| \leq L \cdot \|x - y\| \quad \forall x, y \in \mathbb{R}^n \ \textit{and} \ \forall t \in I \tag{60}$$

We have

$$\begin{aligned} \big\|A(t)x(t) - A(t)y(t)\big\| &= \big\|A(t)(x(t) - y(t))\big\| \\ &\leq \big\|A(t)\big\| \cdot \big\|x(t) - y(t)\big\| \\ &\leq M \cdot \big\|x(t) - y(t)\big\| \end{aligned} \tag{61}$$

Therefore, the right-hand side of (56) is uniformly Lipschitz bounded on the entire real axis, as long as A(t) is bounded on the entire real axis. Hence, conditions (i), (ii), and (iii) are all satisfied, and the Picard-Lindelöf Theorem guarantees the existence and uniqueness of a solution to (56) and (57) for any $t_0 \in \mathbb{R}, x_0 \in \mathbb{R}^n$. Now we just need to show that (58) satisfies (56) and (57). Inserting $t = t_0$ into (59):

$$x(t_0) = e^{\int_{t_0}^{t_0} A(\tau) d\tau} \cdot x_0 = e^0 \cdot x_0 = x_0 \tag{62}$$

The solution candidate (59) therefore satisfies the initial condition (57). Next, we differentiate (59) with respect to time

$$\frac{d}{dt}x(t) = \frac{d}{dt}\left(e^{\int_{t_0}^{t} A(\tau)\, d\tau} \cdot x_0\right) \tag{63}$$

$$\overset{ChainRule}{=} \left[\frac{d}{dt}\int_{t_0}^{t} A(\tau)\, d\tau\right] \cdot e^{\int_{t_0}^{t} A(\tau)\, d\tau} \cdot x_0$$

$$= A(t) \cdot e^{\int_{t_0}^{t} A(\tau)\, d\tau} \cdot x_0 \tag{64}$$

$$= A(t) \cdot x(t)$$

The solution candidate (59) therefore satisfies the differential equation (56) ■

We now apply this theorem to the quaternion differential equation, defined as follows:

Definition III.1.

$$\dot{q}(t) = \frac{1}{2} A(t) \cdot q(t) \tag{65}$$

$$q(t_0) = q_0 \tag{66}$$

where

$$A(t) = \begin{pmatrix} 0 & \omega_1(t) & \omega_2(t) & \omega_3(t) \\ -\omega_1(t) & 0 & \omega_3(t) & -\omega_2(t) \\ -\omega_2(t) & -\omega_3(t) & 0 & \omega_1(t) \\ -\omega_3(t) & \omega_2(t) & -\omega_1(t) & 0 \end{pmatrix} \tag{67}$$

and

$$\omega(t) = \begin{pmatrix} \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \end{pmatrix} \tag{68}$$

is continuous and bounded.

Theorem 2. *The unique solution of the initial value problem (65), (66), (67) of Definition III.1 is given by*

$$q(t) = \begin{cases} \left[cos\left(\frac{\left\|\theta(t)\right\|}{2} \right) \cdot I_{4\times 4} + sin\left(\frac{\left\|\theta(t)\right\|}{2} \right) \cdot \frac{B(t)}{\left\|\theta(t)\right\|} \right] \cdot q_0 & for \quad \left\|\theta(t)\right\| \neq 0 \\ \left[I_{4\times 4} \right] \cdot q_0 & for \quad \left\|\theta(t)\right\| = 0 \end{cases} \tag{69}$$

where

$$\theta(t) = \begin{pmatrix} \theta_1(t) \\ \theta_2(t) \\ \theta_3(t) \end{pmatrix} = \begin{pmatrix} \int_{t_0}^{t} \omega_1(\tau)\, d\tau \\ \int_{t_0}^{t} \omega_2(\tau)\, d\tau \\ \int_{t_0}^{t} \omega_3(\tau)\, d\tau \end{pmatrix} \tag{70}$$

$$\left\|\theta(t)\right\| = \sqrt{\theta_1(t)^2 + \theta_2(t)^2 + \theta_3(t)^2} \tag{71}$$

and

$$B(t) = \int_{t_0}^{t} A(\tau)\, d\tau = \begin{pmatrix} 0 & \theta_1(t) & \theta_2(t) & \theta_3(t) \\ -\theta_1(t) & 0 & \theta_3(t) & -\theta_2(t) \\ -\theta_2(t) & -\theta_3(t) & 0 & \theta_1(t) \\ -\theta_3(t) & \theta_2(t) & -\theta_1(t) & 0 \end{pmatrix} \tag{72}$$

Proof. Since $\omega(t)$, appearing in the system matrix $A(t)$ defined in equation (67), is assumed to be continuous and bounded, the initial value problem of Definition III.1 meets the conditions of Theorem 1. Hence the solution of problem (65), (66), (67) of Definition III.1 exists and is unique for all $t_0 \in \mathbb{R}$, $q_0 \in \mathbb{R}^4$, and, from equation (59), can be expressed in the form

$$q(t) = e^{\int_{t_0}^{t} \frac{1}{2} A(\tau) d\tau} \cdot q_0 \tag{73}$$

or

$$q(t) = e^{\frac{B(t)}{2}} \cdot q_0 \tag{74}$$

Since $A(t)$ is bounded, it is clear that $B(t)$ is bounded. Hence the exponential $e^{\frac{B(t)}{2}}$ is well-defined and its Taylor series expansion converges absolutely for all $t \in \mathbb{R}$. Substituting the Taylor expansion for the exponential in (74) yields

$$q(t) = \left[\sum_{n=0}^{\infty} \frac{1}{n!}\left(\frac{B(t)}{2}\right)^n\right] \cdot q_0 \tag{75}$$

Since the series converges absolutely we can rearrange terms without altering the value of the sum. Hence

$$q(t) = \left[\left(\sum_{n=0}^{\infty} \frac{1}{(2n)!}\left(\frac{B(t)}{2}\right)^{2n}\right) + \left(\sum_{n=0}^{\infty} \frac{1}{(2n+1)!}\left(\frac{B(t)}{2}\right)^{2n+1}\right)\right] \cdot q_0 \tag{76}$$

Here we have merely segregated the series into terms of even and odd powers of $B(t)$. Taking the square of $B(t)$ defined in (72), we get

$$B^2(t) = B(t) \cdot B(t) = (-\theta_1(t)^2 - \theta_2(t)^2 - \theta_3(t)^2) \cdot I_{4\times 4} \tag{77}$$

Therefore, we have for all integers n,

$$B^{2n}(t) = (-\theta_1(t)^2 - \theta_2(t)^2 - \theta_3(t)^2)^n \cdot I_{4\times 4} \tag{78}$$

$$B^{2n+1}(t) = (-\theta_1(t)^2 - \theta_2(t)^2 - \theta_3(t)^2)^n \cdot B(t) \tag{79}$$

and hence with $\|\theta(t)\|$ as defined in (71)

$$B^{2n}(t) = (-1)^n \|\theta(t)\|^{2n} \cdot I_{4\times 4} \tag{80}$$

$$B^{2n+1}(t) = (-1)^n \|\theta(t)\|^{2n} \cdot B(t) \tag{81}$$

Substituting (80) and (81) into (76) yields

$$q(t) = \left[\left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} \cdot \frac{\|\theta(t)\|^{2n}}{2^{2n}}\right) \cdot I_{4\times 4} + \left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} \cdot \frac{\|\theta(t)\|^{2n}}{2^{2n+1}}\right) \cdot B(t)\right] \cdot q_0 \tag{82}$$

For $\|\theta(t)\| = 0$, (82) reduces to

$$q(t) = q_0 \text{ for } \|\theta(t)\| = 0 \tag{83}$$

This proves the second row on the right-hand side of equation (69).
For $\|\theta(t)\| \neq 0$, we can divide by $\|\theta(t)\|$ and rewrite (82) as

$$q(t) = \left[\left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} \cdot \frac{\|\theta(t)\|^{2n}}{2^{2n}}\right) \cdot I_{4\times 4} + \left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} \frac{\|\theta(t)\|^{2n+1}}{2^{2n+1}}\right) \cdot \frac{B(t)}{\|\theta(t)\|}\right] \cdot q_0 \tag{84}$$

Recognizing the infinite sums appearing in (84) as the Taylor series expansions of sine and cosine, we can rewrite (84) as

$$q(t) = \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times 4} + sin\left(\frac{\|\theta(t)\|}{2}\right) \cdot \frac{B(t)}{\|\theta(t)\|}\right] \cdot q_0 \tag{85}$$

■

While equation (69) contains a separate equation for when $\theta(t) = 0$, it is reasonable to question the function's behavior in the neighborhood of zero. An argument could be made that since equation (69) can be restated as

$$q(t) = \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times 4} + sin\left(\frac{\|\theta(t)\|}{2}\right) \cdot \frac{B(t)}{\frac{\|\theta(t)\|}{2}} \cdot \frac{1}{2}\right] \cdot q_0 \qquad \text{for} \|\theta(t)\| \neq 0 \tag{86}$$

$$= \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times 4} + sinc\left(\frac{\|\theta(t)\|}{2}\right) \cdot \frac{B(t)}{2}\right] \cdot q_0 \qquad \text{for} \|\theta(t)\| \neq 0 \tag{87}$$

and the sinc function ($sinc(x) = sin(x)/x$) is a continuous and well-defined function along the entire real axis ($sinc(0) = 1$), that there is no need for approximations in the neighborhood of zero. Therefore, since $sinc(0) = 1$, and B is a zero matrix when $\theta = 0$, equation (69) is sufficient. For those who prefer an alternative that avoids dividing by small numbers, a Taylor Series approximation is provided here (for an interesting example of an issue that may occur due to small numbers, see section 6 in the Appendix).

$$q(t) = \begin{cases} \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times4} + sin\left(\frac{\|\theta(t)\|}{2}\right) \cdot \frac{B(t)}{\|\theta(t)\|}\right] \cdot q_0 & \text{for} \quad \|\theta(t)\| \geq \epsilon \\ \left[cos\left(\frac{\|\theta(t)\|}{2}\right) \cdot I_{4\times4} + \left(\sum_{n=0}^{k} \frac{(-1)^n}{(2n+1)!} \frac{\|\theta(t)\|^{2n}}{2^{2n+1}}\right) \cdot B(t)\right] \cdot q_0 & \text{for} \quad \|\theta(t)\| < \epsilon \end{cases} \tag{88}$$

Equation (88) is left in this form such that the reader may tailor the level of precision to their own requirements. For those who wish to maintain the same level of precision as granted by the analytical solution, i.e. machine precision, k must be chosen such that the order of the approximation, $(2k + 1)$, multiplied by the negative of the order of ϵ, must be greater than or equal to the number of floating digits. For example, on typical programs storing 16 digits, a choice of $\epsilon = 10^{-4}$ and $k = 2$ is sufficient to ensure the maximum possible accuracy, since $(2 \cdot 2 + 1) \cdot 4 = 20 > 16$.

2. Algorithm Comparison

The analytical solution presented in equation (69) is compared here against the "Second-Order Algorithm" and "Third-Order Algorithm" found in Jekeli.[3] Interestingly, Jekeli's Second-Order Algorithm, reproduced below, is nearly identical to equation (69).

$$\hat{q}(t_l) = \left[cos\left(\frac{1}{2}|\delta\beta_l|\right) I + \frac{1}{|\delta\beta_l|} sin\left(\frac{1}{2}|\delta\beta_l|\right) B_l\right] \hat{q}_{l-1} \tag{89}$$

where

$$|\delta\beta_l| = \sqrt{\delta\beta_l^T \delta\beta_l} \tag{90}$$

$$\delta\beta_l = \int_{\delta t} \omega_{ab}^b(t)dt \tag{91}$$

$$= \int_{\delta t} \omega_{ib}^b(t)dt - C_a^b(t_l)\omega_{ia}^a(t_l)\delta t \tag{92}$$

$$= \delta\theta_l - C_a^b(t_l)\omega_{ia}^a(t_l)\delta t \tag{93}$$

$\delta\beta$ is analogous to θ used in equation (70), and the only other difference is that Jekeli's Second-Order Algorithm calculates $q(t_l)$ by propagating forward from $q(t_{l-1})$, whereas equation (69) calculates $q(t_l)$ by propagating forward from $q(t_0)$ (more on this difference later). However, equation (89) was not treated as the true analytical solution. Jekeli developed the Second-Order Algorithm as a method for finding C_b^a, a being any arbitrary frame, when only the angular rates ω_{ib}^b are available. Also, in his derivation, Jekeli makes the assumption that the angular rates are constant over the integration interval. These factors lead Jekeli to treat the Second-Order Algorithm as merely an approximate solution, which he then attempts to improve upon this by developing his Third-Order Algorithm, reproduced below.

$$\hat{q}_l = [I + \frac{1}{12}(\hat{B}_l + 4\hat{B}_{l-1} + \hat{B}_{l-2}) + \frac{1}{12}(I + \frac{1}{4}\hat{B}_l)\hat{B}_{l-1}\hat{B}_{l-2} + \frac{1}{12}\hat{B}_l(\hat{B}_{l-1} - \frac{1}{2}\hat{B}_{l-2})]\hat{q}_{l-2} \tag{94}$$

where

$$\hat{B}_l = \begin{pmatrix} 0 & 3(\delta\beta_1)_l - (\delta\beta_1)_{l-1} & 3(\delta\beta_2)_l - (\delta\beta_2)_{l-1} & 3(\delta\beta_3)_l - (\delta\beta_3)_{l-1} \\ -3(\delta\beta_1)_l + (\delta\beta_1)_{l-1} & 0 & 3(\delta\beta_3)_l - (\delta\beta_3)_{l-1} & -3(\delta\beta_2)_l + (\delta\beta_2)_{l-1} \\ -3(\delta\beta_2)_l + (\delta\beta_2)_{l-1} & -3(\delta\beta_3)_l + (\delta\beta_3)_{l-1} & 0 & 3(\delta\beta_1)_l - (\delta\beta_1)_{l-1} \\ -3(\delta\beta_3)_l + (\delta\beta_3)_{l-1} & +3(\delta\beta_2)_l - (\delta\beta_2)_{l-1} & -3(\delta\beta_1)_l + (\delta\beta_1)_{l-1} & 0 \end{pmatrix} \tag{95}$$

$$\hat{B}_{l-1} = \begin{pmatrix} 0 & (\delta\beta_1)_{l-1} + (\delta\beta_1)_l & (\delta\beta_2)_{l-1} + (\delta\beta_2)_l & (\delta\beta_3)_{l-1} + (\delta\beta_3)_l \\ -(\delta\beta_1)_{l-1} - (\delta\beta_1)_l & 0 & (\delta\beta_3)_{l-1} + (\delta\beta_3)_l & -(\delta\beta_2)_{l-1} - (\delta\beta_2)_l \\ -(\delta\beta_2)_{l-1} - (\delta\beta_2)_l & -(\delta\beta_3)_{l-1} - (\delta\beta_3)_l & 0 & (\delta\beta_1)_{l-1} + (\delta\beta_1)_l \\ -(\delta\beta_3)_{l-1} - (\delta\beta_3)_l & (\delta\beta_2)_{l-1} + (\delta\beta_2)_l & -(\delta\beta_1)_{l-1} - (\delta\beta_1)_l & 0 \end{pmatrix} \tag{96}$$

$$\hat{B}_{l-2} = \begin{pmatrix} 0 & 3(\delta\beta_1)_{l-1} - (\delta\beta_1)_l & 3(\delta\beta_2)_{l-1} - (\delta\beta_2)_l & 3(\delta\beta_3)_{l-1} - (\delta\beta_3)_l \\ -3(\delta\beta_1)_{l-1} + (\delta\beta_1)_l & 0 & 3(\delta\beta_3)_{l-1} - (\delta\beta_3)_l & -3(\delta\beta_2)_{l-1} + (\delta\beta_2)_l \\ -3(\delta\beta_2)_{l-1} + (\delta\beta_2)_l & -3(\delta\beta_3)_{l-1} + (\delta\beta_3)_l & 0 & 3(\delta\beta_1)_{l-1} - (\delta\beta_1)_l \\ -3(\delta\beta_3)_{l-1} + (\delta\beta_3)_l & +3(\delta\beta_2)_{l-1} - (\delta\beta_2)_l & -3(\delta\beta_1)_{l-1} + (\delta\beta_1)_l & 0 \end{pmatrix} \tag{97}$$

and

$$\delta\beta_l = \delta\theta_l - C_a^b(t_{l-1})\omega_{ia}^a(t_{l-1})\frac{\Delta t}{2} \tag{98}$$

$$\delta\beta_{l-1} = \delta\theta_{l-1} - C_a^b(t_{l-1})\omega_{ia}^a(t_{l-1})\frac{\Delta t}{2} \tag{99}$$

The performance of the analytical, second-order, and third-order algorithms were benchmarked by comparing them against the existing transformation between Euler angles $[\alpha, \beta, \gamma]^T$ and quaternion q, given by

$$q = \begin{pmatrix} cos(\alpha/2)cos(\beta/2)cos(\gamma/2) + sin(\alpha/2)sin(\beta/2)sin(\gamma/2) \\ sin(\alpha/2)cos(\beta/2)cos(\gamma/2) - cos(\alpha/2)sin(\beta/2)sin(\gamma/2) \\ cos(\alpha/2)sin(\beta/2)cos(\gamma/2) + sin(\alpha/2)cos(\beta/2)sin(\gamma/2) \\ cos(\alpha/2)cos(\beta/2)sin(\gamma/2) - sin(\alpha/2)sin(\beta/2)cos(\gamma/2) \end{pmatrix} \tag{100}$$

The following plots, figures 1, 2, and 3 demonstrate the differences between the three methods. Jekeli's Second-Order Algorithm significantly out-performs the Third-Order Algorithm, and although being nearly identical to the analytical solution, it is still subject to error propagation. This is due to the Second-Order Algorithm being propagated from the last calculated quaternion whereas the analytical solution propagates from the initial quaternion at every step. While it may seem like a trivial difference, it is the reason that the analytical solution's error remains capped at machine precision, and remains so regardless of how long the simulation is run.

3. General Case

Once the quaternion defining the rotation from the body frame to the inertial frame has been calculated, it is then converted into the transformation matrix C_b^i, as described in equation (21). This algorithm can easily be expanded to a solution for C_b^a, a being some arbitrary reference frame, as long as C_i^a can be calculated without the use of unknown variables. Luckily, this is the case for the majority of the common reference frames, such as the ECEF frame and the navigation frame. For example, to calculate C_b^e, we first apply the analytical quaternion solution to find C_b^i. Next, C_i^e is found using

$$C_e^i = \begin{pmatrix} cos(\omega_e(t)) & -sin(\omega_e(t)) & 0 \\ sin(\omega_e(t)) & cos(\omega_e(t)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{101}$$

where ω_e is the angular velocity of Earth about its z-axis ($\omega_e \approx 7.292 \times 10^{-5} rad/s$). Next,

$$\begin{aligned} C_i^e &= (C_e^i)^{-1} \\ &= (C_e^i)^T \end{aligned} \tag{102}$$

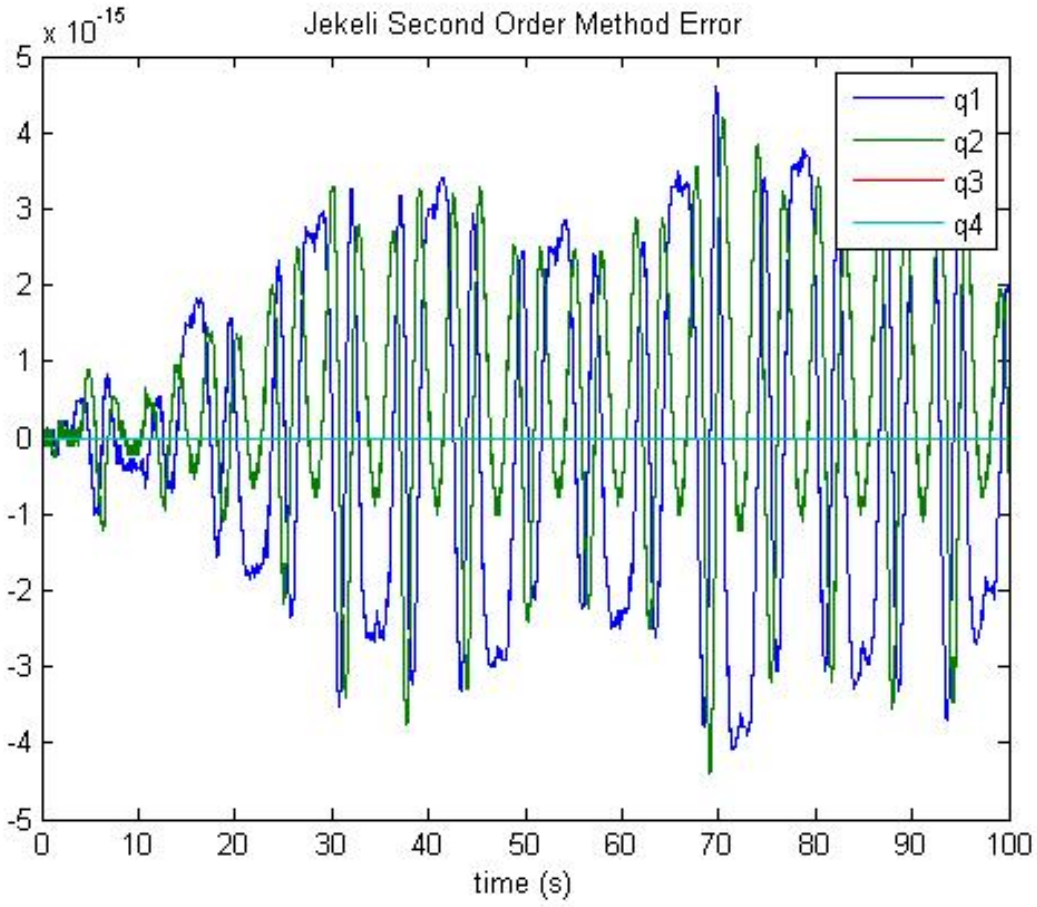

Figure 1. Error between the quaternions generated using Jekeli's second-order algorithm and equation (100)

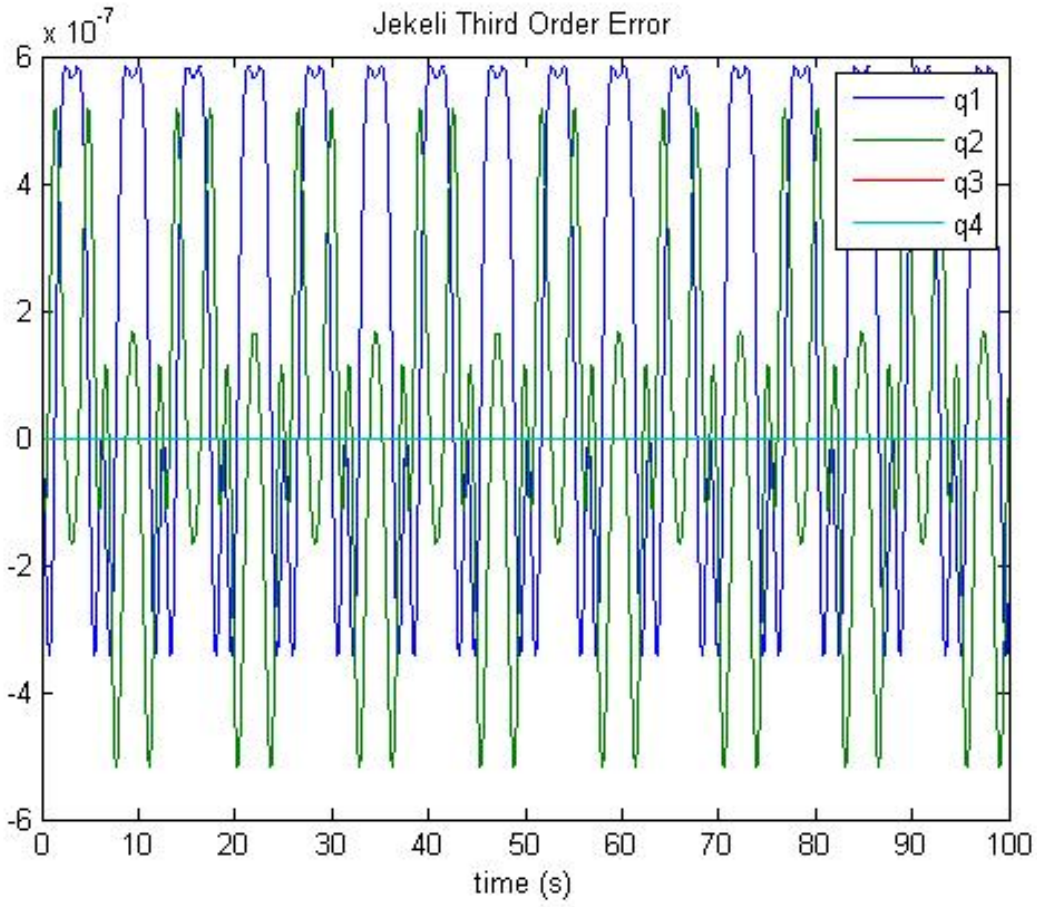

Figure 2. Error between the quaternions generated using Jekeli's third-order algorithm and equation (100)

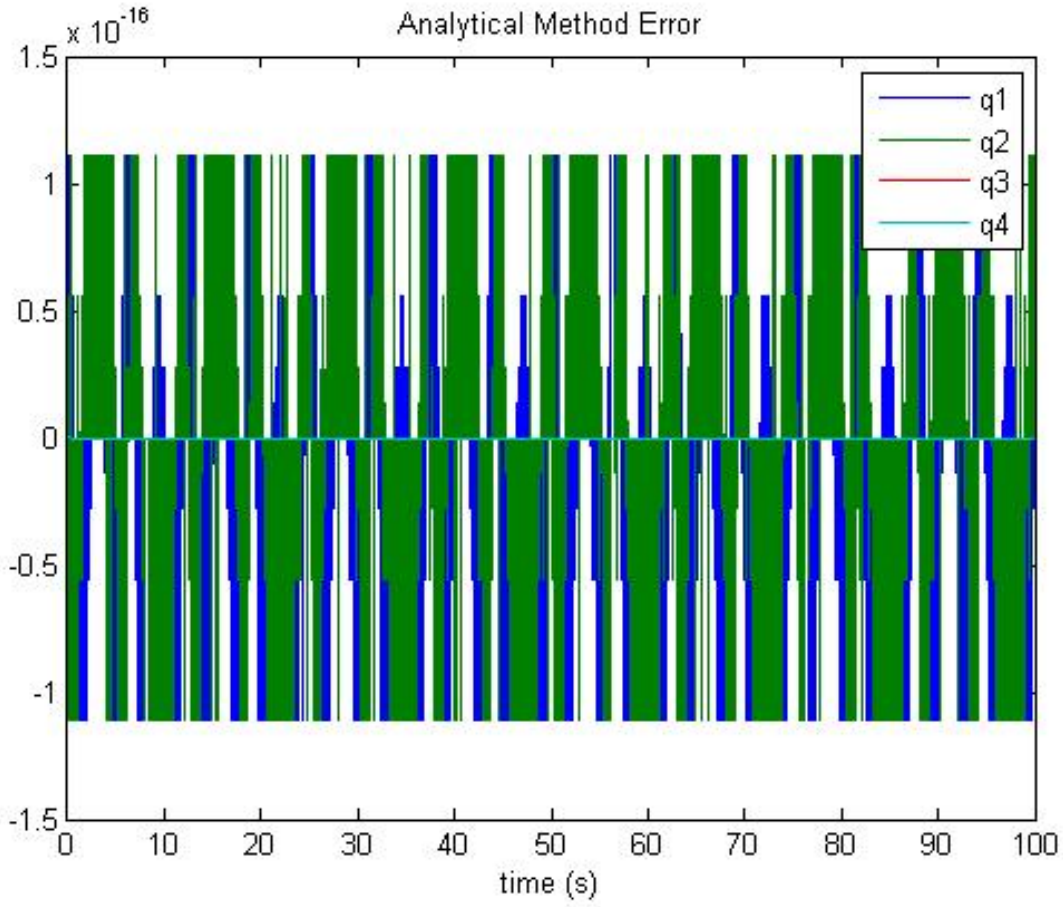

Figure 3. Error between the quaternions generated using equation (69) and equation (100)

The second line of (102) follows from the fact that all transformation matrices are orthogonal by definition. Finally,

$$C_b^e = C_i^e \cdot C_b^i \tag{103}$$

Similarly, C_b^n can be calculated with

$$C_b^n = C_e^n \cdot C_i^e \cdot C_b^i \tag{104}$$

where $C_e^n = (C_n^e)^T$, and C_n^e can be calculated using equation (8).

The transformation matrices used here to convert from the inertial frame to the n-frame or e-frame can both be calculated using known variables. C_i^e is only a function of time, and C_i^n is a function of latitude, longitude (which can be calculated using equations (9) and (10)), and time. Thus, the algorithm presented here for calculating C_b^i can be easily expanded to an algorithm for calculating the transformation matrix between the b-frame and a number of various reference frames, such as C_b^n and C_b^e.

B. Accuracy Improvement for Φ

A Taylor Series expansion was implemented to solve for Φ, the state transition matrix.

$$\Phi(t,t') = e^{F(t-t')} \tag{105}$$

$$= I + F(t-t') + \frac{1}{2!}(F(t-t'))^2 + \frac{1}{3!}(F(t-t'))^3 + ... \tag{106}$$

$$= \sum_{i=0}^{\infty} \frac{(F(t-t'))^i}{i!} \tag{107}$$

This differs from the solution provided by Li,[4]

$$\Phi_{k,k-1} = I + F \cdot \Delta t \tag{108}$$

which is the first order approximation to the solution. Evaluating a greater number terms of the Taylor series will yield a higher accuracy estimate of the true value of Φ, although it can quickly become computationally expensive. The author found the fifth-order approximation to yield accurate results within a reasonable run time.

C. KF Error Equations

1. *Change in δa^i*

The following expression for δa^i was derived,

$$\delta a^i = \hat{C}_b^i \delta a^b + \check{a}_{ng}^i \times \Psi^i \tag{109}$$

differing from the original found in Li's dissertation:[4]

$$\delta a^i = C_b^i \delta a^b + a^i \times \Psi^i \tag{110}$$

Noting that Jekeli and Li do not distinguish a true value with their notation, it is assumed that the C_b^i used here denotes the true value, $\check{C}_b^i$, and not the transformation calculated from measured data, $\hat{C}_b^i$.

Proof. We begin with a few of the equations that we will be making use of:

$$\delta a = \tilde{a}_{wg} - \check{a}_{wg} \tag{111}$$

$$\tilde{a}^b = \quad \check{a}^b \cdot (1 + k_a) + b_a + \epsilon \tag{112}$$

$$\hat{C}_b^i \approx [I - \Psi^i] \cdot \check{C}_b^i \tag{113}$$

Equation (113) is only approximately true as its derivation assumes small angles between the true and the calculated inertial frames.

Starting from equation (111),

$$\begin{aligned}
\delta a^i &= \tilde{a}^i_{wg} - \check{a}^i_{wg} \\
&= \tilde{a}^i_{ng} + \tilde{g}^i - (\check{a}^i_{ng} + \check{g}^i) \\
&= \hat{C}^i_b \cdot \tilde{a}^b_{ng} + \tilde{g}^i - \check{a}^i_{ng} - \check{g}^i \\
&= \hat{C}^i_b \cdot ([\check{a}^b_{ng}] \cdot (1 + k_a) + b_a + \epsilon) - \check{a}^i_{ng} + \delta g_i \\
&= \hat{C}^i_b \cdot \check{a}^b_{ng} + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon - \check{a}^i_{ng} + \delta g_i \\
&\approx [I - \Psi^i] \cdot \check{C}^i_b \cdot \check{a}^b_{ng} + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon - \check{a}^i_{ng} + \delta g_i \\
&= [I - \Psi^i] \cdot \check{a}^i_{ng} + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon - \check{a}^i_{ng} + \delta g_i \\
&= [I - \Psi^i - I] \cdot \check{a}^i_{ng} + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon + \delta g_i \\
&= [-\psi\times] \cdot \check{a}^i_{ng} + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon + \delta g_i \\
&= [\check{a}^i_{ng}\times]\psi + \hat{C}^i_b \cdot [\check{a}^b_{ng}] \cdot k_a + \hat{C}^i_b \cdot b_a + \hat{C}^i_b \cdot \epsilon + \delta g_i
\end{aligned} \tag{114}$$

This final equation can be rewritten to match more closely to Li's:

$$\begin{aligned}
\delta a^i &= [\check{a}^i_{ng}\times]\psi + \hat{C}^i_b([\check{a}^b_{ng}] \cdot k_a + b_a + \epsilon) + \delta g_i \\
&= \hat{C}^i_b \delta a^b + \check{a}^i_{ng} \times \psi + \delta g_i
\end{aligned} \tag{115}$$

■

Equation 115 contains the δg^i term, whereas equation (110) does not, since δg^i, at this point, is assumed to be unknown and negligible. δg^i is only included in this derivation for thoroughness.

The difference between (110) and (109) impacts the error equation dynamics. Specifically, the expression for $\partial \delta a^i / \partial \psi^i$ must be changed.

Using equation (110),

$$\begin{aligned}
\frac{\partial}{\partial \psi^i} \delta a^i &= \frac{\partial}{\partial \psi^i} (\check{C}^i_b \delta a^b + a^i \times \psi^i) \\
&= \frac{\partial}{\partial \psi^i} (\check{C}^i_b \delta a^b + [a^i\times]\psi^i) \\
&= [a^i\times]
\end{aligned}$$

Using equation (109),

$$\begin{aligned}
\frac{\partial}{\partial \psi^i} \delta a^i &= \frac{\partial}{\partial \psi^i} (\hat{C}^i_b \delta a^b + a^i \times \psi^i) \\
&\approx \frac{\partial}{\partial \psi^i} ([I - \Psi^i]\check{C}^i_b \delta a^b + [a^i\times]\psi^i) \\
&= \frac{\partial}{\partial \psi^i} (\check{C}^i_b \delta a^b - \Psi^i \check{C}^i_b \delta a^b + [a^i\times]\psi^i) \\
&= \frac{\partial}{\partial \psi^i} (\check{C}^i_b \delta a^b + [\check{C}^i_b \delta a^b\times]\psi^i + [a^i\times]\psi^i) \\
&= [\check{C}^i_b \delta a^b\times] + [a^i\times]
\end{aligned} \tag{116}$$

2. Change in Gyroscope Error Model

The gyroscope error model provided in Jekeli as well as in Li is given as

$$\delta\omega_{ib}^{b} = [\omega_{ib}^{b}] \cdot k_g + b_g + \epsilon \tag{117}$$

where $[\omega_{ib}^{b}] = diag\{\omega_{ib}^{b}\}$, which implies

$$\tilde{\omega}_{ib}^{b} = [\omega_{ib}^{b}] \cdot (1 + k_g) + b_g + \epsilon \tag{118}$$

(Again, note that Li did not distinguish the true value with some notation, and it is assumed here that ω denotes the true value)

When this model was implemented in the KF, the accelerometer bias and scale factor, as well as the gyroscope bias, were correctly recovered. However, the gyroscope scale factor never converged to its true value. Many changes were implemented in an attempt to fix this issue, such as tweaking the magnitudes of the initial covariance matrix's components corresponding to scale factors and biases, as well as running the program with various values for the Q and R matrices. None of these changes allowed the scale factor to be properly estimated. This is a topic for further investigation, as it is unknown what the issue here was. Possible causes are unobservability, truncation errors, or errors in the equations used to generate data.

The approach found to address this issue involved modifying the original error model and introducing the following:

$$\delta\omega_{ib}^{b} = [t - t_0] \cdot dr + b_g + \epsilon \tag{119}$$

where $[t - t_0] = (t - t_0) \cdot I_{3\times 3}$. This can be restated as

$$\tilde{\omega}_{ib}^{b} = \check{\omega}_{ib}^{b} + [t - t_0] \cdot dr + b_g + \epsilon \tag{120}$$

In other words, the scale factor was removed and replaced with a drift rate, dr. This change, together with the one mentioned in section 1, altered the KF error equations in the following manner:

$$\frac{d}{dt}\begin{pmatrix} \delta x^i \\ \delta \dot{x}^i \\ \psi^i \\ b_a \\ b_g \\ k_a \\ dr \end{pmatrix} = \begin{pmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ \Gamma^i & 0 & [\check{C}_b^i \delta a^b \times] + [\check{a}_{ng}^i \times] & \hat{C}_b^i & 0 & \hat{C}_b^i \cdot [\check{a}_{ng}^b] & 0 \\ 0 & 0 & 0 & 0 & -\check{C}_b^i & 0 & -\check{C}_b^i [t - t_0] \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \delta x^i \\ \delta \dot{x}^i \\ \psi^i \\ b_a \\ b_g \\ k_a \\ dr \end{pmatrix}$$
$$+ \begin{pmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \hat{C}_b^i & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\check{C}_b^i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \\ \epsilon_7 \end{pmatrix} \tag{121}$$

The second line of equation (121) follows directly from (114). The third line can be derived from

$$\dot{\Psi}^i = -\check{C}_b^i \delta\omega \tag{122}$$

Note, this equation appears in Jekeli and Li as

$$\dot{\Psi}^i = -C_b^i \delta\omega \tag{123}$$

where the lack of a sub- or super-script here denotes the true value. Hence,

$$\begin{aligned}\dot{\Psi}^i &= -\check{C}_b^i(dr\cdot(t-t_0)+b_g+\epsilon)\\ &= -\check{C}_b^i\cdot dr\cdot(t-t_0)-\check{C}_b^i\cdot b_g-\check{C}_b^i\cdot\epsilon\end{aligned} \tag{124}$$

which can be used to verify the third line of (121). The fourth through seventh lines all follow from the assumption that b_a, b_g, k_a, and dr are constants.

The corresponding error equations found in Li's dissertation are provided here to make the alterations more apparent:

$$\begin{aligned}\frac{d}{dt}\begin{pmatrix}\delta x^i\\ \delta\dot{x}^i\\ \psi^i\\ b_a\\ b_g\\ k_a\\ k_g\end{pmatrix} &= \begin{pmatrix}0 & I & 0 & 0 & 0 & 0 & 0\\ \Gamma^i & 0 & [a^i\times] & C_b^i & 0 & C_b^i\cdot[\tilde{a}^b] & 0\\ 0 & 0 & 0 & 0 & -C_b^i & 0 & -C_b^i\cdot[\omega_{ib}^b]\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}\cdot\begin{pmatrix}\delta x^i\\ \delta\dot{x}^i\\ \psi^i\\ b_a\\ b_g\\ k_a\\ k_g\end{pmatrix}\\ &+\begin{pmatrix}I & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & C_b^i & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & -C_b^i & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & I & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & I & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & I & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & I\end{pmatrix}\cdot\begin{pmatrix}\epsilon_1\\ \epsilon_2\\ \epsilon_3\\ \epsilon_4\\ \epsilon_5\\ \epsilon_6\\ \epsilon_7\end{pmatrix}\end{aligned} \tag{125}$$

IV. Results

When put together as described above, with data generated using the values for noise found in table 2 and the values for biases, scale factors, and drift rate found in table 3, the KF is able to accurately estimate the parameters used to generate the sensed data, as demonstrated in the figures 4 and 5. Notice how the estimates start to diverge slightly after the 12 second mark; this is believed to be due to the violation of the small angle assumption, which is addressed more in-depth in the Further Research section. Table 3 provides a more precise look at the final estimated values in comparison to the true values.

Table 3. Estimated Parameters versus True Values

Parameter	Final Estimate	True Value	Error
$b_a(1)$	0.0004	0.0010	6.304×10^{-3}
$b_a(2)$	0.0032	0.0030	2.477×10^{-4}
$b_a(3)$	0.0057	0.0050	6.891×10^{-4}
$b_g(1)$	0.0009	0.0010	1×10^{-4}
$b_g(2)$	0.0056	0.0040	1.6×10^{-3}
$b_g(3)$	-0.0063	-0.0060	3×10^{-4}
$k_a(1)$	-0.0053	-0.0030	2.3×10^{-3}
$k_a(2)$	0.0055	0.0070	1.5×10^{-3}
$k_a(3)$	-0.0030	-0.0020	1.0×10^{-3}
$dr(1)$	0.0008	0.0008	3.17×10^{-5}
$dr(2)$	-0.0013	-0.0009	3.602×10^{-4}
$dr(3)$	0.0004	0.0002	2.088×10^{-4}

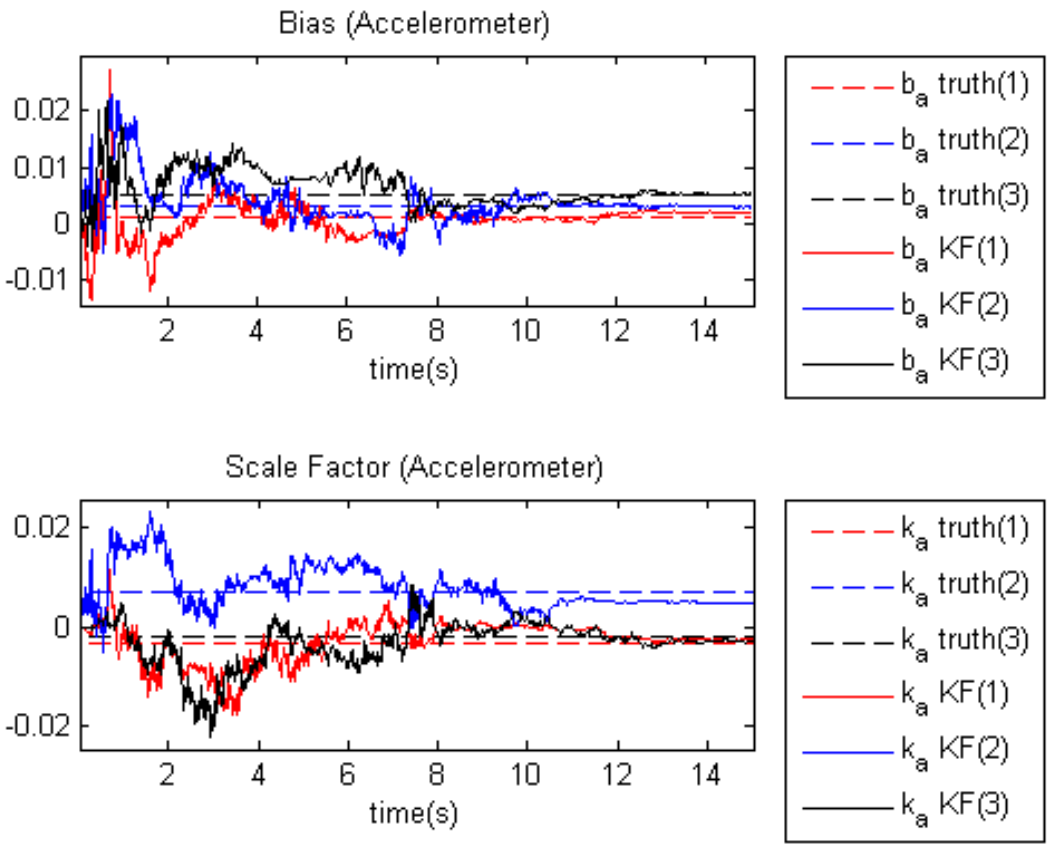

Figure 4. KF estimates of b_a and k_a compared against their true values

After estimating the parameters, an estimate of the true body-frame accelerations can be made using

$$\breve{a}^b(i) = \left(\hat{a}^b(i) - b_a(i)\right) / \left(1 + k_a(i)\right) \quad , \quad i = 1, 2, 3 \tag{126}$$

The accuracy of the corrected accelerations can then be compared with with the accuracy of the original measurements in order to determine the overall benefit of the KF. Figure 6 compares the errors between the measured and true accelerations with the error between the KF-corrected and the true accelerations, showing the norm of the error has been reduced from around $12{\times}10^{-3}$ to $4{\times}10^{-3}$ m/s^2 ($\sim$ 4 mGal), a reduction in error of approximately 65%.

While not quite at the accuracy level required for the more sensitive applications of gravimetry, such as subsurface resource exploration, the reduction in error is significant. More importantly, the results presented confirm the efficacy of the alterations to existing methodology. The changes made to the error dynamics equations, along with the analytical solution to the quaternion integration problem, constitute reliable and validated innovations.

The next step to be taken is to apply this research to gravity estimation, either by incorporating the gravitational disturbance, δg, into the KF's state vector, or by subtracting the sensed accelerations from the accelerations obtained by finite-differencing the GPS data. These topics, along with a number of additional improvements to the algorithm presented in this paper, are discussed in section VI.

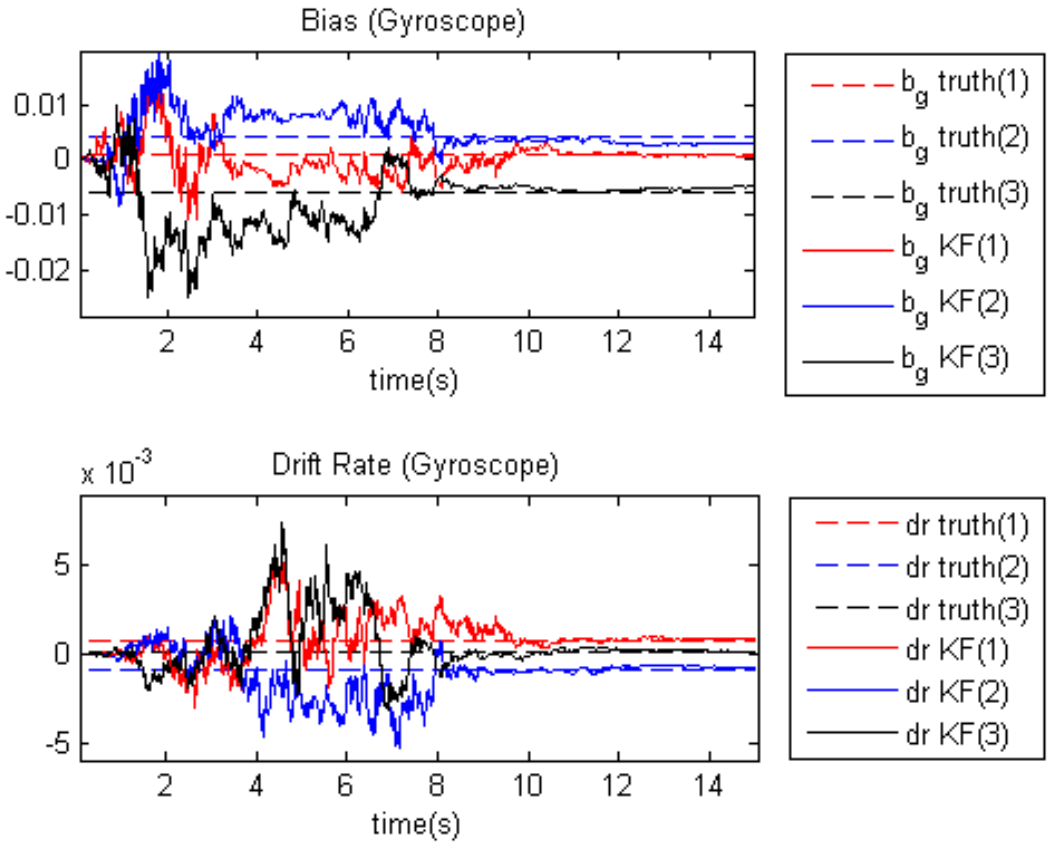

Figure 5. KF estimates of b_g and dr compared against their true values

V. Conclusion

This paper outlines a method for the estimation of the various error sources inherent in accelerometers and gyroscopes. Several innovations and refinements were made to the existing methodology behind strapdown gravimetry, such as the replacement of numerical approximations to the quaternion integration problem with an analytical solution, as well as significant changes to the error dynamics equations implemented in the Kalman Filter. The algorithm outlined in this paper demonstrated its ability to reduce the error in an accelerometer from around 12 mGal to 4 mGal, a reduction of around 65%. The success of this methodology warrants further research into the application of this work to the field of gravity estimation.

VI. Further Research

A. Model Improvements

The current model is relatively basic and has made a number of simplifying assumptions. Research that builds off of this work could possibly increase the precision of the KF by removing those assumptions and introducing more realistic models.

1. Small Angle Assumption

The derivation of the error equation dynamics found in (114) relies on the small angle assumption, as stated in (113). The substitution of $[I-\Psi^i]\cdot C_b^i$ in for $\hat{C}_b^i$ is only approximately true. $[I-\Psi^i]$ is used to approximate the transformation from the true inertial frame (call this *i-true*) to the inertial frame calculated using sensed

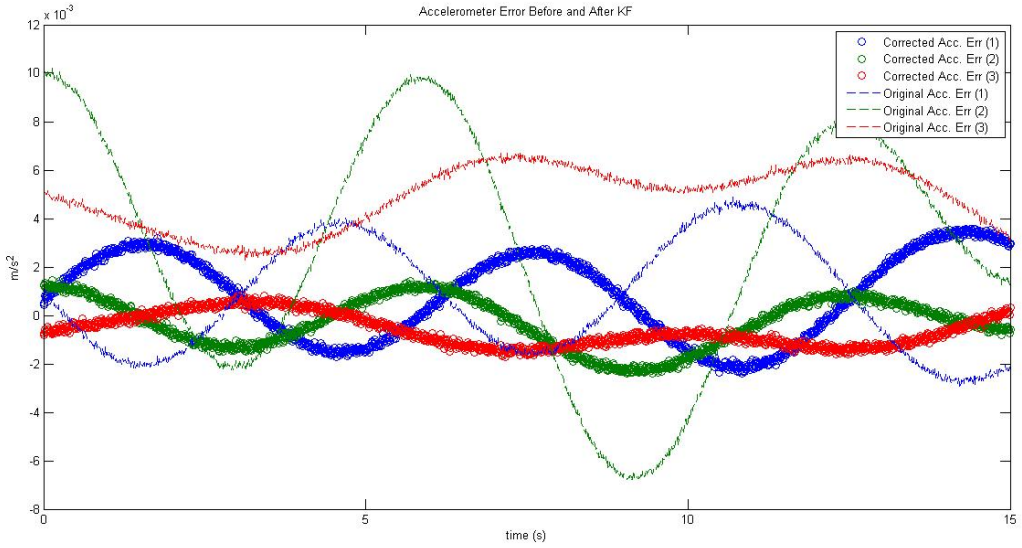

Figure 6. KF estimates of b_a and k_a compared against their true values

data (call this *i-false*). This can be seen more clearly by writing

$$\begin{aligned}\hat{C}_b^i &= C_b^{i-false} \\ &= C_{i-true}^{i-false} \cdot C_b^{i-true} \\ &= C_{i-true}^{i-false} \cdot \check{C}_b^i \\ &\approx [I - \Psi^i] \cdot \check{C}_b^i \end{aligned} \tag{127}$$

where $\Psi^i = [\psi^i \times]$, ψ^i being the orientation error between the *i-true* and *i-false*. The derivation of the approximation can be found in equations (1.22) through (1.24) on page 13 of Jekeli's text.

Due to integration and modeling errors, the calculated inertial frame will steadily diverge from the true inertial frame, especially given the presence of a drift rate in the gyroscopes. Eventually, the angles will grow large enough to make the error dynamics no longer hold true.

Removing the small angle assumption would have multiple benefits. The first is that the KF could be run for longer flights without the the error dynamics falling apart. More importantly, removing the approximation makes the model more accurate and increases the precision of the KF's estimates. Removing the small angle assumption makes the error dynamics equations slightly more messy, but with precision being such a driving factor, it seems worth the trouble. This solution was explored by the author but was not implemented in the main project due to time constraints. A code detailing the process is left for others to potentially build upon.

2. GPS Error Model

The simulation described in this paper treats GPS as a near-perfect measurement of position and velocity. Of course, GPS is subject to its own host of error sources, such as clock error, ionospheric effects, and multi-path distortions. If desired, these parameters could be added to the KF's error states and estimated, provided that the appropriate changes are made in the error dynamics equations.

3. Alignment Error

Another simplifying assumption made was that alignment error was treated as perfectly zero. It would be interesting to see what level of alignment precision would be needed in order to estimate the error parameters to a certain desired accuracy.

4. Thermal Effects

Thermal effects were completely ignored in this simulation. The author recognizes that thermal considerations have a dramatic impact on the performance of the IMU. However, this is a topic beyond the scope of the current study, and thus all sensors were assumed to be operating at a nominal room temperature.

5. Bias and Scale Factor Modeling

The data simulation code described in this paper treats biases and scale factors as constants. While this is a fine approximation for short simulations, if this study were to be expanded to hour long flights, it may be more appropriate to include a random walk component, as in

$$b(t) = b + \mu(t) \tag{128}$$

where b is a constant, and $\mu(t)$ is a random variable of known variance.

6. Gravity Estimation

The algorithm presented here lends itself to the pursuit of high accuracy gravity estimates. One method for doing so would be to simulate slight disturbances to the gravity vector, then attempt to recover those perturbations by adding the δg term to the KF's state vector, similar to how bias and scale factor were recovered. However, care must be taken to set up the equations such all elements of the state vector remain observable. Also, the error dynamics used in the KF will be dependent upon the method used to simulate true gravity. For example, gravity modeled as a constant vector always pointing towards the center of the Earth would have different partials with respect to the other elements of the state vector, such as orientation error, compared to gravity modeled as a function of inertial position.

Appendix

Corrections

The following are various small, but important, corrections to the equations of Jekeli and Li found during the course of this study.

Correction 1

Equation (2.5.5) in Li is given as

$$\begin{aligned}\hat{v}_l^i = \hat{v}_{l-2}^i + \frac{1}{6}(&\hat{C}_b^i(l-2)(3\delta v_{l-1}^b - \delta v_l^b) \\ &+ 4\hat{C}_b^i(l-1)(\delta v_{l-1}^b + \delta v_l^b)\hat{C}_b^i(l)(3\delta v_l^b - \delta v_{l-1}^b)) \\ &+ \hat{f}(x^i, C_i^n, \Omega_{ie}^i, g^i)_{t=t_{l-2}}\Delta t\end{aligned} \tag{129}$$

A " + " is missing before the last C_b^i and should instead be

$$\begin{aligned}\hat{v}_l^i = \hat{v}_{l-2}^i + \frac{1}{6}(&\hat{C}_b^i(l-2)(3\delta v_{l-1}^b - \delta v_l^b) \\ &+ 4\hat{C}_b^i(l-1)(\delta v_{l-1}^b + \delta v_l^b) + \hat{C}_b^i(l)(3\delta v_l^b - \delta v_{l-1}^b)) \\ &+ \hat{f}(x^i, C_i^n, \Omega_{ie}^i, g^i)_{t=t_{l-2}}\Delta t\end{aligned} \tag{130}$$

Correction 2

Equation (3.2.4) in Li is given as

$$\Psi^i = -C_b^i \delta\omega_{ib}^i \tag{131}$$

The Ψ is missing its dot. The equation should read

$$\dot{\Psi}^i = -C_b^i \delta\omega_{ib}^i \tag{132}$$

Li then substitutes equation (131) into

$$\frac{d}{dt}\delta\dot{x}^i = \Gamma^i \delta x^i + C_b^i \delta a^b + a^i \times \Psi^i + \delta g^i \tag{133}$$

giving

$$\frac{d}{dt}\delta\dot{x}^i = \Gamma^i \delta x^i + C_b^i \delta a^b - a^i \times C_b^i \delta\omega_{ib}^i + \delta g^i \tag{134}$$

Luckily, equations (131) and (134) are not referenced in the rest of Li's dissertation, and the suspected typo does not propagate any further.

Correction 3

On page 119 of Jekeli's text, equation (4.45) is given as

$$\omega(t) = \omega_{l-2} + \dot{\omega}_{l-2}(t - t_{l-2}) + O(\Delta t^2), \qquad |t - t_{l-2}| \leq \Delta t \tag{135}$$

Jekeli intends to apply this approximation only on the interval from t_{l-2} to t_l, and not from t_{l-4} to t_l, hence the equation should read:

$$\omega(t) = \omega_{l-2} + \dot{\omega}_{l-2}(t - t_{l-2}) + O(\Delta t^2), \qquad t \in [t_{l-2}, t_l] \tag{136}$$

Example: Numerical Issue due to Small Numbers

Consider the expression

$$f(x) = \frac{x}{1 + x - 1} \tag{137}$$

Analytically, this expression is well-defined for all values of x that are non-zero. Only if x is precisely zero we have a problem, and $f(0)$ is not defined. But for $x \neq 0$ the result is always the same, $f(x) = 1$, no matter how close x is to the singularity $x = 0$. In summary

$$f(x) = \begin{cases} 1 & \text{for} \quad x \neq 0 \\ \text{undefined} & \text{for} \quad x = 0 \end{cases} \tag{138}$$

Numerically, however, the situation is very different in the neighborhood of the singularity $x = 0$ due to numerical cancellation. If $f(x)$ is evaluated according to $f(x) = \frac{x}{(1+x)-1}$, i.e. if the order of operations of $f(x)$ is programmed as

$$\text{evaluation method 2:} \begin{cases} \text{step 1:} & a = 1 + x \\ \text{step 2:} & num = x \\ \text{step 3:} & denom = a - 1 \\ \text{step 4:} & f = num/denom \end{cases} \tag{139}$$

the numerical result goes very wrong in the neighborhood of $x = 0$. Note that, for small positive numbers x, the sequence of the numerical operations of lines 1 and 3 in the evaluation method 2 does not reproduce x precisely, as it would if calculations were performed analytically. Since the computer stores only 16 digits, all digits of the intermediate result a of line 1 smaller than 10^{-16} are simply chopped off and deleted. That means, for $x = 0.1234567890123456e - 13$, we get $a = 1.000000000000012$. The evaluation of line 3 then yields denom=0.12e-13, and the numerical result for $f(x)$ from line 4 is $f = 0.1234567890123456e - 13/0.12e - 13 = 1.02880657510288$. Similarly, for $x = 0.1234567890123456e - 14$, the numerical result is $f = 0.1234567890123456e - 14/0.1e - 14 = 1.234567890123456$. And for $x = 0.1234567890123456e - n$ with $n > 14$, the numerical result is $f = 0.1234567890123456e - 14/0 = Inf$.

References

[1]Hartman, Philip, "Ordinary Differential Equations", Birkhauser, Boston, Basel, Stuttgart, 1982.

[2]Hale, Jack K."Ordinary Differential Equations", Krieger Publishing Company, Malabar, Florida, 1980.

[3]Jekeli, Christopher, "Inertial Navigation Systems with Geodetic Applications", de Gruyter, Berlin, 2001.

[4]Li, Xiaopeng, "Moving Base INS/GPS Vector Gravimetry on a Land Vehicle", Ohio State University, 2007.

[5]Li, Xiaopeng. J Geod (2011) 85: 597. doi:10.1007/s00190-011-0462-2

[6]Simon, Dan, "Optimal State Estimation", John Wiley and Sons Inc., 2007.

[7]Schwarz, Klaus-Peter. "Simultaneous determination of position and gravity from INS/DGPS." Festschrift Univ.-Prof. Dr.-Ing. Prof. hc Günter Seeber anlässlich seines 65 (2006): 137-144.

[8]Wei, M., and Klaus-Peter Schwarz. "A Strapdown Inertial Algorithm Using an Earth-Fixed Cartesian Frame." Navigation 37.2 (1990): 153-167.

[9]Inacio, Pedro. A Sensitivity Study into Strapdown Airborne Gravimetry. Master's Thesis. TU Delft, Delft University of Technology, 2010.

[10]VanRoek, Petr, and Zdenek Martinec. "The Stokes-Helmert scheme for the evaluation of a precise geoid." (1994).

[11]Bishop, Christopher. "Interpretation and Modelling of the Pedirka Basin (central Australia) using Magnetics, Gravity, Well-log and Seismic data." ASEG Extended Abstracts 2012.1 (2012): 1-4.

[12]Harig, Christopher, and Frederik J. Simons. "Accelerated West Antarctic ice mass loss continues to outpace East Antarctic gains." Earth and Planetary Science Letters 415 (2015): 134-141.