Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms

Paul G. Savage*  
Strapdown Associates, Inc., Maple Plain, Minnesota 55359

This series of two papers provides a rigorous comprehensive approach to the design of the principal software algorithms utilized in modern-day strapdown inertial navigation systems: integration of angular rate into attitude, acceleration transformation/integration into velocity, and integration of velocity into position. The algorithms are structured utilizing the two-speed updating approach originally developed for attitude updating in which an analytically exact equation is used at moderate speed to update the integration parameter (attitude, velocity, or position) with input provided from a high-speed algorithm measuring dynamic angular rate/acceleration effects within the parameter update time interval (coning for attitude updating, sculling for velocity updating, and sculling (writer’s terminology) for high-resolution position updating). The algorithm design approach accounts for angular rate/specific force acceleration measurements from the strapdown system inertial sensors as well as rotation of the navigation frame used for attitude referencing and velocity integration. This paper, Part 1, defines the overall design requirement for the strapdown inertial navigation integration function and develops direction cosine and quaternion forms for the attitude updating algorithms. Part 2 (Savage, P. G., “Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms,” Journal of Guidance, Control, and Dynamics [to be published]) deals with design of the velocity and position integration algorithms. Although Parts 1 and 2 often cover fundamental inertial navigation concepts, the material presented is intended for use by the practitioner who is already familiar with basic inertial navigation concepts.

Nomenclature

\[ A, A_1, A_2, A_3 = \text{arbitrary coordinate frames} \]
\[ C_{A_1}^{A_2} = \text{direction cosine matrix that transforms a vector from its } A_2 \text{ frame projection form to its } A_1 \text{ frame projection form} \]
\[ I = \text{identity matrix} \]
\[ q_{A_1} = \text{attitude quaternion that transforms a quaternion vector from its } A_2 \text{ frame component form to its } A_1 \text{ frame component form} \]
\[ q_{A_1}^{A_2} = \text{attitude quaternion } q_{A_1}^{A_2} \text{ conjugate having the same first element as } q_{A_1} \text{ but with the negative of elements 2–4 in } q_{A_1} \]
\[ q_1 = \text{identity quaternion having 1 for the first element and zero for the remaining three} \]
\[ V = \text{vector without specific coordinate frame designation} \]
\[ V^A = \text{column matrix with elements equal to the projection of } V \text{ on frame } A \text{ axes} \]
\[ V^{A \times} = \text{skew symmetric (or cross product) form of } V^A, \text{ represented by the square matrix} \]
\[
\begin{bmatrix}
0 & -V_{ZA} & V_{YA} \\
V_{ZA} & 0 & -V_{XA} \\
-V_{YA} & V_{XA} & 0
\end{bmatrix}
\]
where \( V_{XA}, V_{YA}, V_{ZA} \) are the components of \( V^A \); matrix product of \( V^{A \times} \) with another \( A \)-frame vector equals the cross product of \( V^A \) with the vector in the \( A \) frame
\[ V_q^A = \text{quaternion four vector equivalent to } V^A, \]
\[ \omega_{A_1 A_2} = \text{angular rate of coordinate frame } A_2 \text{ relative to coordinate frame } A_1 \text{; when } A_1 \text{ is the inertial } I \text{ frame, } \omega_{I A_1 A_2} = \text{the angular rate measured by angular rate sensors mounted on frame } A_2 \]

I. Introduction

Inertial navigation is the process of calculating position by integration of velocity and computing velocity by integration of total acceleration. Total acceleration is calculated as the sum of gravitational acceleration, plus the acceleration produced by applied nongravitational forces (known as specific force acceleration). An inertial navigation system (INS) consists of a navigation computer for the integration function, a precision clock for timing the integration operations, an accelerometer assembly for measuring the specific force acceleration, a gravitation model software resident in the navigation computer for calculating gravitational acceleration as a function of calculated position, and an attitude reference for defining the angular orientation of the accelerometer triad as part of the velocity calculation. In a modern day INS, the attitude reference is provided by a software integration function residing in the INS computer using inputs from a three-axis set of inertial angular rate sensors. The angular rate sensor and accelerometer triads are mounted to a common rigid structure within the INS chassis to maintain alignment between each inertial sensor. Such an arrangement has been denoted as a strapdown INS because of the rigid attachment of the inertial sensors within the chassis, hence, to the vehicle in which the INS is mounted.

The primary functions executed in the INS computer are the angular rate into attitude integration function (denoted as attitude integration), use of the attitude data to transform measured acceleration into a suitable navigation coordinate frame where it is integrated into velocity (denoted as velocity integration), and integration of the navigation frame velocity into position (denoted as position integration). Thus, three integration functions are involved, attitude, velocity, and position, each of which requires high accuracy to assure negligible error compared to inertial sensor accuracy requirements.

From a historical perspective, since the basic strapdown inertial navigation concept was originally formulated in the 1950s, strapdown analysts have primarily focused on the design of algorithms for the attitude integration function. Invariably, the design approaches were driven by the capabilities and limitations of contemporary
flight computer technology. In the late 1950s and the 1960s, two approaches were pursued by strapdown analysts (in various organizations) for the attitude integration function: high-speed attitude updating, e.g., 10–20 kHz, using first-order digital algorithms, and lower-speed attitude updating, e.g., 50–100 Hz, using higher-order algorithms. The high-speed approach was promoted as a means for accurately accounting for high-frequency angular rate components that can rectify into systematic three-dimensional attitude change; however, computer technology of that time period was only capable of executing simplified first-order equations of limited accuracy for the attitude updating algorithms. In contrast, the higher-order algorithm proponents touted improved analytical accuracy compared to first-order algorithms; however, the improved accuracy was degraded due to the associated increase in executable operations per attitude update cycle and, hence, a slower attitude update rate to satisfy contemporary computer throughput limitations. Tradeoffs between the two approaches were clouded by the emergence of the attitude quaternion as the preferred approach for the analytical form of the computed attitude parameter (vs the traditional direction cosine matrix attitude representation). For the algorithms investigated during that time period, the quaternion showed improved accuracy in high-frequency angular rate environments.

In 1966, the writer proposed a new two-speed approach for the attitude integration function whereby the attitude updating operation is divided into two parts: a simple high-speed, first-order algorithm portion coupled with a more complex moderate-speed, higher-order algorithm portion whose input was provided by the high-speed algorithm. The simplified high-speed portion accounted for high-frequency angular oscillations within the attitude update cycle that can rectify into systematic attitude buildup (traditionally denoted as coning). Taken together, the combined accuracy of the two-speed approach was equivalent to operating the higher-order algorithm at the high-speed rate (for improved accuracy); however, due to the simplicity of the high-speed algorithm, the combined computer throughput requirement was no greater than for original high-speed, first-order attitude updating algorithms. The utility of the Ref. 6 two-speed algorithm design approach was limited by its reliance on a Picard-type recursive integration of the continuous form attitude rate differential equation in which both the moderate- and high-speed algorithms were generated simultaneously. The complexity of the analytical recursive integration design process limited expansion of the higher-order, moderate-speed algorithm (to only second order in Ref. 6, which was considered acceptable at that time).

In an unrelated design activity, Jordan in 1969 suggested a two-speed approach for the strapdown attitude updating function in which the analytical formulation at the onset was based on two separate controller algorithm portion whose input was provided by the (exact) higher-order attitude updating algorithm based on input attitude change, and a simplified high-speed, second-order integration algorithm that measured the attitude change input for the moderate-speed algorithm. In 1971, Bortz extended the Jordan concept to have the high-speed calculation based on a differential equation that, when integrated, measures the exact attitude change input to the exact attitude updating algorithm. The exact moderate-speed-determination algorithm can be structured to any specified order of accuracy by truncation of two trigonometric coefficients. In practice, simplified forms of the Bortz attitude change differential equation are used for the high-speed function. References 7 and 8 therein provided a more general form of the two-speed attitude updating approach in which the moderate-speed, higher-order algorithm and high-speed, simplified algorithm can be independently tailored to meet particular application requirements. (Interestingly, Ref. 8 proposed an analog implementation for a simplified version of the high-speed algorithm.) A secondary benefit derived from the Ref. 7 and 8 two-speed approach (proposed using direction cosines for the exact moderate-speed attitude update operation) is that the moderate-speed portion can also be formulated with an analytically exact, closed-form quaternion updating algorithm using the identical high-speed input approach for the direction cosine updating. Thus, the new two-speed approach has equal accuracy for either direction cosine or quaternion updating, both of which derive from analytically exact, closed-form equations (assuming that Taylor series expansion for trigonometric coefficients is carried out to comparable accuracy order).

Most modern-day strapdown INSs for aircraft utilize attitude updating algorithms based on a two-speed approach. The repetition rate for the moderate-speed algorithm portion, e.g., 50–200 Hz, is typically designed, based on maximum angular rate considerations, to minimize power series truncation error effects in the moderate- and high-speed algorithms. The repetition rate for the high-speed algorithm, e.g., 1–4 kHz for an aircraft INS with 1 n m·sec 50 percent radial position error rate, is designed, based on the anticipated strapdown inertial sensor assembly vibration environment, to accurately account for vibration-induced coning effects. Continuing two-speed attitude algorithm development work has centered on variations for the high-speed integration function. Originally conceived as a simple first-order algorithm, today’s high-speed attitude algorithms have taken advantage of increased throughput capabilities in modern-day computers and become higher order for improved accuracy (Refs. 9–11 and 12, Sec. 7.1). While the attitude updating function has been evolving to its current form, very little parallel work has been published on the development of the companion strapdown INS algorithms for acceleration transformation/velocity integration and position integration (the subject of the Ref. 13, Part 2, paper).

This paper, Part 1, defines the overall design requirement for the strapdown inertial navigation integration function and describes a comprehensive design process for developing the attitude integration algorithms based on the two-speed approach. The material presented in this condensed version of Ref. 12, Sec. 7.1 (which is an expansion of material in Ref. 9), emphasizing a more rigorous analytical formulation and the use of exact closed-form equations, where possible, for ease in computer software documentation/validation (which is also consistent with modern-day flight computer technology). Included in the attitude algorithm design process is a rigorous treatment of methods for accounting for navigation coordinate frame rotation during the attitude update time periods.

The paper is organized as follows. Section II provides background material regarding coordinate frames and attitude parameters used. Section III provides a complete set of typical strapdown inertial navigation attitude, velocity, and position equations in continuous differential equation format, which serves as a framework for the equivalent algorithm design process. Section IV develops the two-speed attitude integration algorithm (for both direction cosine and quaternion formulations including navigation frame rotation effects) in a generic form for the high-speed portion and describes a particular form to illustrate the design of one of the classical high-speed, second-order computing algorithms. A tabular reference summary of the attitude integration algorithms is presented in Sec. V. Section VI provides a general discussion of the process followed in selecting algorithms for a particular application and establishing their execution rates. Concluding remarks are provided in Sec. VII.

Finally, it is important to recognize that although the original intent of the two-speed approach was to overcome throughput limitations of early computer technology (1965–1975), that limitation is rapidly becoming insignificant with continuing rapid advances in modern high-speed computers. This provides the motivation to return to a simpler single-speed algorithm structure whereby all computations are executed at a repetition rate that is sufficiently high to accurately account for multiplicative high-frequency angular rate and acceleration/rectification effects. The two-speed structure presented in both Parts 1 and 2 is compatible with compression into such a single-speed format as explained in the particular sections where the algorithms are formulated.

II. Coordinate Frames and Attitude Orientation Relationships

This section defines the coordinate frames used in this paper and generically describes the properties of the direction cosine matrix, the attitude quaternion, and the rotation vector, attitude parameters utilized to represent the angular relationship between two coordinate frames.
A. Coordinate Frame Definitions

A coordinate frame is an analytical abstraction defined by three consecutively numbered (or lettered) unit vectors that are mutually perpendicular to one another in the right-hand sense. It can be visualized as a set of three perpendicular lines (axes) passing through a common point (origin) with the unit vectors emanating from the origin along the axes. In this paper, the physical locations of the coordinate frame origins are arbitrary. A vector's components (or projections) in a particular coordinate frame equal the dot product of the vector with the coordinate frame unit vectors. The vectors used in this paper are classified as free vectors and, hence, have no preferred location in coordinate frames in which they are analytically described.

The coordinate frames are defined as follows.

1) The E frame is the Earth fixed coordinate frame used for position location definition. It is typically defined with one axis parallel to the Earth polar axis with the other axes fixed to the Earth and parallel to the equatorial plane.

2) The N frame is the nonrotating inertial coordinate frame used as a reference for angular rotation measurements. Particular orientations selected for the N frame are as follows. (a) The X axis is parallel to the downward vertical, and (b) the Y and Z axes are parallel to the Earth polar axis with the other axes fixed to the Earth and parallel to the equatorial plane.

B. Attitude Parameter Definitions

The direction cosine matrix is defined as a square matrix whose columns are an orthogonal set of unit vectors, each equal to a unit vector along a coordinate axis of frame A2 as projected onto the axes of coordinate frame A1:

\[ C_{A1}^{A2} = \begin{bmatrix} u_{A1}^{A2} & u_{A2}^{A2} & u_{A3}^{A2} \end{bmatrix} \] (1)

where \( u_{A1}^{A2} \) is the unit vector along A2 frame axis i projected on coordinate frame A1 axes.

From this basic definition it can be demonstrated that the element in row i, column j of \( C_{A1}^{A2} \) equals the cosine of the angle between frame A1 axis i and frame A2 axis j, that is, the transpose of \( C_{A1}^{A2} \) equals its inverse, the columns of \( C_{A1}^{A2} \) transpose equal frame A1 axis unit vectors projected on frame A2 axes, and the product of \( C_{A1}^{A2} \) with a vector projected on frame A2 axes equals the components of the vector projected on frame A1 axes (and the converse for \( C_{A2}^{A1} \) transpose):

\[ V_{A1} = C_{A1}^{A2} V_{A2}; \quad V_{A2} = (C_{A1}^{A2})^T V_{A1} = C_{A2}^{A1} V_{A1} \] (2)

Equations (2) can be used to derive the direction cosine matrix chain rule:

\[ C_{A2}^{A1} = C_{A1}^{A2} C_{A2}^{A1} \] (3)

The rotation vector defines an axis of rotation and magnitude for rotation about the axis. Imagine frame A1 being rotated from its starting attitude to a new attitude by rotation about the rotation vector through an angle equal to the rotation vector magnitude. Now call frame A2 the new attitude of frame A1. By this definition of frame A2, an arbitrarily defined rotation vector uniquely defines the attitude of frame A2 relative to the original frame A1 attitude. Conversely, for a given attitude of frame A2 relative to frame A1, a rotation vector can be defined that is consistent with this attitude. Thus, a rotation vector can be used to define the attitude of frame A2 relative to frame A1. Analytically, it can be shown (Refs. 4, 9, and 12, Sec. 3.2.2.1) that the relationship between the rotation vector and the direction cosine matrix is given by

\[ C_{A1}^{A2} = \begin{bmatrix} I + \frac{\sin \phi}{\phi} (\phi \times \sin (1 - \cos \phi)) \phi^2 \end{bmatrix} \] (4)

where \( \phi \) and \( \phi \) are the rotation vector and its magnitude. A unique property of the rotation vector is that it has identical components in the A1 and A2 frames (Ref. 12, Sec. 3.2.2.1); hence, \( \phi \) in Eq. (4) represents either \( \phi^{A1} \) or \( \phi^{A2} \).

The attitude quaternion is a four vector, i.e., four components, defined as a function of the rotation vector (Refs. 4 and 9; 12, Sec. 3.2.4; and 14, pp. 73–76)

\[ q_{A2}^{A1} = \begin{bmatrix} \cos 0.5 \phi \\ \sin 0.5 \phi \\ 0.5 \phi \end{bmatrix} \] (5)

From Eq. (5), it is easily verified that the sum of the squares of the \( q_{A2}^{A1} \) elements is unity. The coordinate frame transformation equations associated with \( q_{A2}^{A1} \) are in quaternion algebra (Refs. 4, 9, and 12, Sec. 3.2.4.1)

\[ V_{A1}^{q_{A2}} = q_{A2}^{A1} V_{A1}^{A2} q_{A2}^{A1}; \quad V_{A2}^{q_{A1}} = q_{A1}^{A2} V_{A2}^{A1} q_{A1}^{A2} = q_{A2}^{A1} V_{A2}^{A1} q_{A2}^{A1} \] (6)

Equations (6) can be used to derive the attitude quaternion chain rule,

\[ q_{A1}^{A2} = q_{A2}^{A1} q_{A1}^{A2} \] (7)

C. Attitude Parameter Rate Equations

The rates of change of the Sec. II.B attitude parameters (Refs. 4, 8, 9, and 12, Sec. 3.3) are given by

\[ \dot{C}_{A2}^{A1} = C_{A2}^{A1} (\omega_{A2}^{A1} \times) - (\omega_{A2}^{A1} \times) C_{A2}^{A1} \] (8)

\[ \dot{q}_{A2}^{A1} = 2q_{A2}^{A1} \phi \times \omega_{A2}^{A1} - \frac{1}{\phi} \omega_{A2}^{A1} \phi q_{A2}^{A1} \] (9)

\[ \dot{\phi} = \omega_{A2}^{A1} \phi + \frac{1}{\phi^2} 1 - \phi \sin \phi \cos \phi \] (10)

III. Continuous Form Strapdown Inertial Navigation Equations

The differential equations that define the primary operations typically performed in a strapdown inertial navigation system (Refs. 9; 12, Chap. 4; and 15, pp. 77–103 and 156–177) are given as follows.

**Attitude rate**

\[ \dot{C}_{B}^{E} = C_{B}^{E} (\omega_{B}^{E} \times) - (\omega_{B}^{E} \times) C_{B}^{E} \] (11)

or, alternatively,

\[ \dot{q}_{B}^{E} = 2q_{B}^{E} \phi \times \omega_{B}^{E} - \frac{1}{\phi} \omega_{B}^{E} \phi q_{B}^{E} \] (12)

**Local level frame rotation rate**

\[ \omega_{L}^{E} = C_{L}^{E} (\omega_{L}^{N} + \omega_{N}^{L} \times) \] (13)

\[ \omega_{N}^{E} = C_{N}^{E} \times \omega_{E}^{N} \] (14)

\[ \omega_{L}^{N} = F_{C} (u_{Z}^{N} \times V_{N}) + \mu_{Z}^{N} R_{Z}^{N} \] (15)

**Acceleration transformation**

\[ a_{SF}^{L} = C_{B}^{L} a_{SF}^{B} \] (16)

or, alternatively,

\[ a_{SF}^{L} = q_{B}^{L} a_{SF}^{B} q_{B}^{L} \] (17)

\[ a_{SF}^{N} = C_{L}^{N} a_{SF}^{L} \] (18)
Velocity rate
\[ \dot{g}^N = g^N - (\omega^N \times (\omega^N \times R^N)) \]
(19)
\[ v^N = \dot{a}_G + \dot{g}^N_p - (\omega^N \times (\omega^N \times v^N)) \]
(20)
Position rate
\[ C^E_{N} = C^E_{N}(\omega^E_{EN} \times) \]
(21)
\[ h = a^N_{ZN} \cdot v^N \]
(22)
where
\[ R = \text{position vector from Earth’s center to the INS} \]
\[ v = \text{velocity (rate of change of position) relative to the Earth defined analytically as the time derivative of} \]
\[ h = \text{altitude above the Earth defined as the distance from the INS to the Earth surface measured along a line from the INS that is perpendicular to a tangent plane on the Earth’s reference geoid}^{16} \text{ surface} \]
\[ F_C = \text{curvature matrix (3 × 3) that is a function of position} (C^E_{EN}, h) \text{ with elements 3,} \]
\[ \rho_{ZN} = \text{vertical component of} \omega^E_{EN}; \text{ the value selected for} \rho_{ZN} \text{ depends on the type of} \]
\[ \sigma_{SF} = \text{specific force acceleration defined as the acceleration relative to nonrotating inertial space produced by} \]
\[ g = \text{mass attraction gravitational acceleration or} \]
\[ g_p = \text{plumb-bob gravity or gravity, which, for a stationary INS, lies along the line of a plumb bob} \]

Analytical models for \( g \) can be found in Refs. 16; 17, Sec. 4.4; and 18, Sec. 6.3. See Ref. 12, Sec. 5.4.1, for \( N \) frame components of \( g_p \).

In performing the strapdown inertial navigation function, the strapdown INS computer integrates the latter attitude rate, velocity rate, and position rate equations using suitable integration algorithms.

The following points are worthy of note regarding the form of the latter navigation equations. Both direction cosine and quaternion attitude forms are shown for the body attitude rate/acceleration transformation operations. Either can be used in practice with virtually identical results. The velocity is defined relative to the Earth (\( E \) frame) and the velocity rate equation is written in the locally level defined \( N \) frame (for integration into velocity). This is typical for many terrestrial navigation applications, e.g., aircraft INS. Other coordinate frame options are also used for velocity definition and the velocity rate equation, e.g., for tactical and strategic missile guidance. The position rate equations define position as altitude plus the angular orientation of the \( N \) frame relative to the \( E \) frame [from which altitude and longitude can be extracted and \( R \) calculated (Refs. 12, Secs. 4.5.1 and 4.5.3, and 15, pp. 88, 89)]. Position can also be defined for the position rate equation as simply \( R \) [from which \( C^E_{EN} \) and \( h \) can be calculated (Ref. 12, Sec. 4.5.4)]. Attitude rate equation (22) appears trivial, but not necessarily when one considers a rotating oblate Earth model, a rotating \( N \) frame over the Earth, and the stated altitude definition. Reference 12, Secs. 4.4 and 5.5, shows that Eq. (22) is exact for a rotating oblate Earth model. If vertical channel gravity/divergence control is to be incorporated to prevent exponentially unstable vertical channel error growth, Eqs. (20) and (22) would include an additional vertical control term (Refs. 12, Sec. 4.4.1; 15, pp. 102–103, and 18, Sec. 10.3).

IV. Attitude Update Algorithms

In this section we develop algorithmic forms for direction cosine matrix rate equation (11) and attitude quaternion rate equation (12) suitable for integration in a digital computer. The algorithms will be structured using what is now the traditional two-speed approach in which analytically exact closed-form equations are applied for the basic attitude update function using inputs from a higher speed algorithm designed to measure attitude change over the basic attitude update cycle.

A. Attitude Direction Cosine Matrix

The updating algorithm for the \( C^E_{B} \) direction cosine matrix is designed to achieve the same numerical result at the attitude update times as would the formal continuous integration of the Eq. (11) \( C^E_{B} \) expression at the same time instant. The algorithm is constructed by envisioning the body \( B \) frame and local level \( L \) frame orientation histories in the digital updating world produced in Eq. (11) \( \omega^B_{EN} \) and \( \omega^B_L \) as being constructed of successive discrete orientations relative to nonrotating inertial space \( I \) at each update time instant. To be completely general, we also allow that \( C^E_L \) updating operations for \( L \) frame angular motion may not necessarily occur at the same time instant that \( C^E_B \) is updated for \( B \) frame motion, e.g., for a multirate digital computation loop structure where \( C^E_L \) is updated at a higher rate for \( B \) frame rotation than for \( L \) frame rotation. In the interests of minimizing computer throughput requirements, the software architecture might have \( L \) frame updates occurring 5–10 times slower than \( B \) frame updates. The nomenclature we adopt to describe the coordinate frame orientation history is as follows:

- \( B_{1}(m) \) = discrete orientation of the body \( B \) frame in nonrotating inertial space \( I \) at computer update time \( t_m \)
- \( m \) = computer cycle index for \( B \) frame angular motion updates to \( C^E_B \)
- \( L_{1}(m) \) = discrete orientation of the locally level \( L \) frame in nonrotating inertial space \( I \) at computer update time \( t_m \)
- \( n \) = computer cycle index for \( L \) frame angular motion updates to \( C^E_L \)

With these definitions, the general updating algorithm for \( C^E_B \) is constructed as follows using the Eq. (3) direction cosine matrix product chain rule:

\[ C^{L}_{B(m)} = C^{L}_{B(m-1)} \cdot C^{B}_{L(m-1)} \]
(23)
\[ C^{L}_{B(m)} = C^{L}_{B(m-1)} \cdot C^{B}_{B(m)} \]
(24)
where

- \( C^{L}_{B(m-1)} \) = \( C^E_B \) relating the \( B \) frame at time \( t_{m-1} \) to the \( L \) frame at time \( t_{m-1} \)
- \( C^{B}_{B(m-1)} \) = \( C^E_L \) relating the \( L \) frame at time \( t_{m-1} \) to the \( B \) frame at time \( t_{m} \)
- \( C^{L}_{B(m)} \) = direction cosine matrix that accounts for \( B \) frame rotation relative to inertial space from its orientation at time \( t_{m-1} \) to its orientation at time \( t_{m} \)
- \( C^{B}_{B(m-1)} \) = direction cosine matrix that accounts for \( L \) frame rotation relative to inertial space from its orientation at time \( t_{m-1} \) to its orientation at time \( t_{m} \)

The algorithm described by Eqs. (23) and (24) relates body \( B \) frame and local-level \( L \) frame orientations at separate times and provides for \( B \) and \( L \) frame inertial angular motion updates to \( C^E_B \) at different
update rates. Unlike the $B$ frame (which can be rotating dynamically at 200–300 deg/s), the inertial angular rate of the local level $L$ frame is generally small, equal to Earth’s rotation rate plus $L$ frame angular rate relative to the Earth (transport rate, which is typically never larger than a few Earth rates). Consequently, the $L$ frame update can generally be performed at a lower rate than the $B$ frame update with comparable accuracy. Note the update rate requirement for $B$ and $L$ frame motion is based, in part, on minimizing errors in the approximate high-speed algorithm used to measure attitude change (see Secs. IV.A.1 and IV.A.2). The $B$ and $L$ frame motion updates to $C_{Bl}$ are performed by the $C_{Bm}^{(m-1)}$ and $C_{Lm}^{(m-1)}$ terms in Eqs. (23) and (24), algorithms for which are derived separately next.

1. Body Frame Rotation

Equation (23) updates the $C_{Bl}$ frame attitude direction cosine matrix using $C_{Bm}^{(m-1)}$ to account for angular rate of the strapdown sensor (body) $B$ frame relative to nonrotating space $\omega_B^0$. The formal definition for $C_{Bm}^{(m-1)}$ is

$$C_{Bm}^{(m-1)} = 1 + \int_{t_{m-1}}^{t_m} C_{Bm}^{(m-1)} \, dt$$

(25)

where $B(t)$ is the $B$ frame attitude at an arbitrary time in the interval $t_{m-1}$ to $t_m$.

The $C_{Bm}^{(m-1)}$ matrix can also be expressed in terms of a rotation vector defining the frame $B_{lm}$ attitude relative to frame $B_{im-1}$. Applying Eq. (4) using Taylor series expansion for the coefficient terms obtains

$$C_{Bm}^{(m-1)} = 1 + \sin \phi_m \phi_m (\phi_m \times) + \frac{1 - \cos \phi_m}{\phi_m^2} (\phi_m \times) (\phi_m \times)$$

$$\sin \phi_m \phi_m = 1 - \frac{\phi_m^2}{3!} + \frac{\phi_m^4}{5!} - \cdots$$

$$\frac{1 - \cos \phi_m}{\phi_m^2} = \frac{1}{{2!}} \frac{\phi_m^2}{4!} + \frac{\phi_m^4}{6!} - \cdots$$

(26)

where $\phi_m$ is the rotation vector defining the frame $B_{lm}$ attitude relative to frame $B_{im-1}$ at time $t_m$. The $\phi_m$ rotation vector can be computed by treating $\phi$ as a general rotation vector defining the general $B$ frame attitude relative to frame $B_{im-1}$, for time greater than $t_{m-1}$. Then $\phi$ is calculated as the integral from $t_{m-1}$ of the general $\phi$ equation, with $\phi$ for Eq. (26) evaluated as the integral solution at time $t_m$. Treating frame $B_{im-1}$, for $\phi$ definition as the nonrotating inertial reference frame $I$, we obtain the following for the general $\phi$ expression by application of Eq. (10) with general frame $A_2$ replaced by body frame $B$ and general frame $A_1$ replaced by inertial frame $I$ for angular rate description:

$$\phi = \omega_B^0 + \frac{1}{2} \phi \times \omega_B^0 + \frac{1}{\phi^3} \left(1 - \frac{\phi \sin \phi}{2(1 - \cos \phi)}\right) \phi \times (\phi \times \omega_B^0)$$

(27)

where $\phi$ is the rotation vector defining the general attitude of frame $B$ relative to frame $B_{im-1}$, for time greater than $t_{m-1}$. Equation (27), commonly referred to as the Bortz equation, relates the change in $B$ frame attitude to the $B$ frame inertial angular rate $\omega_B^0$ that would be measured by strapdown angular rate sensors.

The attitude rotation vector $\phi_m$ for Eq. (26) is then obtained as the integral of Eq. (27) from time $t_{m-1}$, evaluated at time $t_m$

$$\phi(t) = \int_{t_{m-1}}^{t_m} \phi(t) \, dt, \quad \phi_m = \phi(t_m)$$

(28)

where $\tau$ is the running integration time variable. To reduce the number of computations involved in calculating $\phi$ with Eq. (27), simplifying assumptions are incorporated. For example, through a power series expansion, the scalar multiplier of the $\phi \times (\phi \times \omega_B^0)$ term in Eq. (27) can be approximated as

$$\frac{1}{\phi^3} \left(1 - \frac{\phi \sin \phi}{1 - \cos \phi}\right) \approx \frac{1}{12}$$

(29)

hence, Eq. (27) to second order in $\phi$ is given by

$$\phi \approx \omega_B^0 + \frac{1}{\phi} \times \omega_B^0 + \frac{1}{\phi^3} \phi \times (\phi \times \omega_B^0)$$

(30)

Through simulation and analysis (analytical expansion under hypothesized analytically definable angular motion conditions), it can be shown that to second-order accuracy in $\phi$

$$\frac{1}{\phi} \times (\phi \times \omega_B^0) \approx \frac{1}{\phi^3} \phi \times (\phi \times \omega_B^0)$$

(31)

where $\alpha(t) = \int_{t_0}^{t} \omega_B^0 \, dt$.

(32)

Equation (31) is extremely significant because it enables Eq. (27) to be simplified to second-order accuracy, i.e., in error to third order in $\phi$, by retaining only first-order terms. Thus, Eq. (27) becomes to second-order accuracy

$$\phi \approx \omega_B^0 + \frac{1}{\phi^3} \phi \times \omega_B^0$$

(33)

Substituting Eq. (33), Eq. (28) is given by

$$\phi_m = \int_{t_{m-1}}^{t_m} \left[\omega_B^0 + \frac{1}{\phi} \phi \times (\phi \times \omega_B^0)\right] \, dt$$

(34)

Finally, with Eq. (32) we obtain

$$\phi_m = \alpha_m + \beta_m$$

(35)

with

$$\alpha(t) = \int_{t_0}^{t} \omega_B^0 \, dt, \quad \alpha_m = \alpha(t_m)$$

(36)

$$\beta_m = \frac{1}{2} \int_{t_{m-1}}^{t_m} (\alpha(t) \times \omega_B^0) \, dt$$

(37)

where $\beta_m$ is the coning attitude motion from $t_{m-1}$ to $t_m$. The $\alpha_m$ term has been coined the coning term because it measures the effects of coning motion components present in $\omega_B^0$. Coning motion is defined as the condition whereby an angular rate vector is itself rotating. For $\omega_B^0$ exhibiting pure coning motion (the absolute magnitude being constant but the vector rotating), a fixed axis in the $B$ frame that is approximately perpendicular to the plane of the rotating $\omega_B^0$ vector will generate a conical surface as the angular rate motion ensues (hence, the term coning to describe the motion). Under coning angular motion conditions, $B$ frame axes perpendicular to $\omega_B^0$ appear to osculate (in contrast with nonconing or spinning angular motion in which axes perpendicular to $\omega_B^0$ rotate around $\omega_B^0$).

For situations where $\omega_B^0$ is not rotating, it is easily seen from Eq. (36) that $\alpha(t)$ will be parallel to $\omega_B^0$; hence, the cross product in the $\beta_m$ integrand will be zero and $\beta_m$ will be zero. Under these conditions, Eq. (34) reduces to the simplified form

$$\phi_m = \int_{t_{m-1}}^{t_m} \omega_B^0 \, dt$$

(38)

when $\omega_B^0$ is not rotating. Note that Eq. (37) also applies to the exact $\phi$, Eqs. (27) and (28) for a nonrotating $\omega_B^0$, i.e., without approximation. This is readily verified by observing from Eq. (27) that $\phi(t)$ will initially be aligned with $\omega_B^0$ as the $\phi(t)$ integration begins and will then remain parallel to $\omega_B^0$ because its cross products with $\phi(t)$ in the $\phi(t)$ expression will remain zero. Under these conditions, Eqs. (27) and (28) also reduce to Eq. (37).

Integrated angular rate and coning increment algorithms are discussed next. A discrete digital algorithm form of the $\alpha_m$ integrated
rate and $\beta_n$ coning expressions in Eq. (36) can be developed by considering $\beta_n$ to be the value at $t = t_n$ of the general function $\beta(t)$, where from Eq. (36)

$$\beta(t) = \frac{1}{2} \int_{t_{n-1}}^{t} (\alpha(\tau) \times \omega_{IB}^b) \, d\tau$$  \hspace{1cm} (38)

Let us now consider the integration of Eq. (38) as divided into a portion up to and after a general time $t_{n-1}$ within the $t_{n-1}$ to $t_n$ interval so that Eq. (38) is equivalently

$$\beta(t) = \beta(t_{n-1}) + \Delta \beta(t), \quad \beta_m = \beta(t_n)$$  \hspace{1cm} (39)

where $\beta(t_{n-1})$ is the value of $\beta(t)$ at $t = t_{n-1}$ and $l$ is the computer cycle index for $t = t_l$ cycle times. Note that by its definition, the $l$ cycle index is faster than the $m$ cycle index. We now define the next $l$ cycle time point $t_l$ within the $t_{n-1}$ to $t_n$ interval so that at $t_l$, Eq. (39), including initial conditions, become

$$\beta_l = \beta(t_{n-1}) + \Delta \beta, \quad \beta_m = \beta(t_l = t_n)$$  \hspace{1cm} (40)

$$\Delta \beta_l = \frac{1}{2} \int_{t_{n-1}}^{t_l} (\alpha(\tau) \times \omega_{IB}^b) \, d\tau$$

Through a similar process, the $\alpha(t)$ expression for Eq. (40) is obtained by manipulation of $\alpha(t)$ in Eqs. (36) as

$$\alpha(t) = \alpha(t_{n-1}) + \Delta \alpha(t), \quad \Delta \alpha(t) = \int_{t_{n-1}}^{t} \omega_{IB}^b \, d\tau$$  \hspace{1cm} (41)

$$\Delta \alpha_l = \Delta \alpha(t_l), \quad \alpha_l = \alpha(t_{n-1}) + \Delta \alpha_l \quad \alpha_m = \alpha_l(t_l = t_n)$$

$$\alpha_l = 0 \quad \text{at} \quad t = t_{n-1}$$

With Eqs. (41), Eqs. (40) are equivalently

$$\Delta \beta_l = \frac{1}{2}(\alpha(t_{n-1}) + \Delta \alpha_l) + \frac{1}{2} \int_{t_{n-1}}^{t} \alpha(\tau) \times \omega_{IB}^b \, d\tau$$  \hspace{1cm} (42)

$$\beta_l = \beta(t_{n-1}) + \Delta \beta_l, \quad \beta_m = \beta(t_l = t_n)$$

$$\beta_l = 0 \quad \text{at} \quad t = t_{n-1}$$

Equations (41) and (42) constitute the construct of a digital recursive algorithm at the computer cycle rate for calculating $\alpha_m$ and the $\beta_m$ coning term as a summation of changes in $\alpha, \beta$ over the $t_{n-1}$ to $t_l$ interval. It remains to determine a digital equivalent for the Eq. (42) integral term in $\Delta \beta_l$. Continuing work in attitude algorithm development has centered on the design of digital algorithms for evaluating the coning equation (42) integral term. In general, the methods utilized assume a general analytical form for the angular rate profile $\omega_{IB}^b$ in the $t_{n-1}$ to $t_l$ time interval, e.g., a truncated general polynomial in time. The Eq. (42) integral is then analytically determined as a function of the general rate profile coefficients, e.g., the polynomial coefficients. Finally, the coefficients for the angular rate profile are calculated to fit successive integrated angular rate increment measurements.

Equation (45) has been classified as a second-order algorithm for $\beta_m$ because it includes current and past cycle angular products in the $\Delta \beta_l$ equation. From the analysis leading to Eq. (44), the $l$ and $l - 1$ $\Delta \alpha$ product term in $\Delta \beta_l$, i.e., the $\bar{\alpha}$ term, stems from the approximation of linearly ramping angular rate in the $t_{l-1}$ to $t_l$ time interval. If the angular rate was approximated as a parabolically varying function of time, a third-order algorithm would result containing $l, l - 1$, and $l - 2 \Delta \alpha$ products. If the angular rate was approximated as a constant over $t_{l-1}$ to $t_l$, the $\bar{\alpha}$ term for $\Delta \beta_l$ in Eq. (45) would vanish, resulting in a first-order algorithm for $\beta_m$. Finally, if angular rates are slowly varying, we can approximate $\beta_m$ as being equal to zero. Alternatively (and more accurately), we can set the $l$ cycle rate equal to the $m$ cycle rate, which equals $\beta_m$ in Eq. (45) to $\Delta \beta_l$ calculated once at time $t_n$ (and noting from the initial condition defined in Eq. (41) that $\alpha_{n-1}$ would be zero). The latter algorithm was developed in Ref. 4. Note that setting the $l$ and $m$ rates equal can also be achieved by increasing the $m$ rate to match the $l$ rate. The result is a single, high-speed, higher-order algorithm with a simpler software architecture than the two-speed approach, but requiring more throughput. Continuing advances in the speed of modern-day computers may make this the preferred approach for the future.
where
\[ \text{d} \alpha = \text{diifferential angular rate increment, i.e., analytical representation of pulse output from strapdown angular rate sensors, } \omega^R_{\alpha} \text{ dt} \]
\[ \Delta \alpha = \text{summation of integrated angular rate output increments from angular rate sensors} \]

2. Local Level Frame Rotation

Equation (24) updates the \( C_{B}^{L} \) attitude direction cosine matrix using \( C_{L_{(a)}}^{L} \) to account for angular rate of the local-level coordinate frame relative to nonrotating space \( \omega_{L_{(a)}}^{R} \). The derivation for \( C_{L_{(a)}}^{L} \) directly parallels that used to determine \( C_{L_{(a-1)}}^{B} \) in Sec. IV.A.1. The formal definition for \( C_{L_{(a-1)}}^{L} \) is

\[ C_{L_{(a-1)}}^{L} = 1 + \int_{t_{a-1}}^{t_{a}} C_{L_{(a-1)}}^{L(t)} \text{ dt} \]

where \( L(t) \) is the \( L \) frame attitude at an arbitrary time in the interval \( t_{a-1} \) to \( t_{a} \).

The \( C_{L_{(a-1)}}^{L} \) matrix can also be expressed in terms of the rotation vector defining the frame \( L_{(a-1)} \) attitude relative to frame \( L_{(a-1)} \). Applying Eq. (4) with Taylor series expansion for the coefficient terms obtains

\[ C_{L_{(a-1)}}^{L} = 1 - \frac{\sin \zeta_{a}}{\zeta_{a}} (\zeta_{a} \times) + \frac{(1 - \cos \zeta_{a})}{\zeta_{a}^2} (\zeta_{a} \times)(\zeta_{a} \times) \]

\[ \frac{\sin \zeta_{a}}{\zeta_{a}} = 1 - \frac{\zeta_{a}^2}{3!} + \frac{\zeta_{a}^4}{4!} - \cdots \]

\[ (1 - \cos \zeta_{a}) \frac{\zeta_{a}^2}{\zeta_{a}^2} = \frac{\zeta_{a}^2}{2!} - \frac{\zeta_{a}^4}{4!} + \frac{\zeta_{a}^6}{6!} - \cdots \]

where \( \zeta_{a} \) is the rotation vector defining the frame \( L_{(a-1)} \) attitude at time \( t_{a-1} \) relative to the frame \( L_{(a-1)} \) attitude at time \( t_{a-1} \). Note in Eq. (49) that the sign for the \((\sin \zeta_{a})/\zeta_{a} (\zeta_{a} \times)\) term is negative in contrast with the similar term in the Eq. (26) \( C_{B_{(a-1)}}^{L} \) expression.

This is because the \( C_{L_{(a-1)}}^{L} \) matrix has the opposite phase sense from \( C_{B_{(a-1)}}^{L} \) [or \( C_{L_{(a-1)}}^{B} \) in Eq. (4)] in that \( C_{L_{(a-1)}}^{L} \) transforms vectors from \( L_{(a-1)} \) to \( L_{(a-1)} \), whereas \( C_{B_{(a-1)}}^{L} \) transforms vectors from \( B_{(a-1)} \) to \( B_{(a-1)} \). As such, the \( C_{L_{(a-1)}}^{L} \) form in Eq. (49) is the transpose of the Eq. (26) \( C_{B_{(a-1)}}^{L} \) expression form.

Because the \( t_{a-1} \) to \( t_{a} \) update cycle is relatively short, \( \zeta_{a} \) will be very small in magnitude. Because \( \omega_{L_{(a-1)}}^{R} \) is small and slowly changing over a typical \( t_{a-1} \) to \( t_{a} \) update cycle (due to small changes in velocity and position over this time period) the \( L \) frame rate vector \( \omega_{L_{(a-1)}}^{R} \) can be approximated as nonrotating. The result is that \( \zeta_{a} \) for Eq. (49) can be calculated as the integral of the simplified form of the Eq. (10) rotation vector rate equation where the cross-product terms are neglected,

\[ \zeta_{a} \approx \int_{t_{a-1}}^{t_{a}} \omega_{L_{(a-1)}}^{R} \text{ dt} \]

We note in passing that based on the smallness of \( \zeta_{a} \) as already discussed, Eq. (49) for \( C_{L_{(a-1)}}^{L} \) can also be simplified. For example, a second-order version (accurate to second order in \( \zeta_{a} \)) is from Eq. (49),

\[ C_{L_{(a-1)}}^{L} \approx 1 - (\zeta_{a} \times) + \frac{1}{2}(\zeta_{a} \times)(\zeta_{a} \times) \]

The computer memory/throughput advantages of utilizing a simplified form of Eq. (49) for \( C_{L_{(a-1)}}^{L} \) [such as Eq. (51)] are trivial for today’s modern computer technology compared to the disadvantages of increased software validation/documentation complexity and loss in accuracy. The accuracy loss is generally minor during navigation; however, it might not be negligible during initial alignment operations (prior to the start of inertial navigation) where the \( C_{L_{(a-1)}}^{B} \) matrix is used to apply tilt updates to \( C_{B}^{B} \) (Refs. 12, Sec. 6.1.2, and 15, pp. 120–121). Initial tilt alignment corrections to \( C_{B}^{B} \) can be fairly large, e.g., 0.1–1.0 deg. which can produce undesirable errors in \( C_{B}^{B} \) during the initial alignment process if too simplified a version of Eq. (49) is utilized. The closed-loop servo action of the initial alignment operations would eventually correct the resulting attitude error generated in \( C_{B}^{B} \); however, it could leave a residual orthogonality/normality error in the \( C_{B}^{B} \) rows (and columns). The result would be the requirement to include an orthogonality/normality correction algorithm (see Sec. IV.A.3) as an outer loop in the \( C_{B}^{B} \) update processing.

A discrete digital algorithm for the Eq. (50) \( \zeta_{a} \) integral can be constructed by first combining Eqs. (13) and (15) to obtain the \( \omega_{L_{(a-1)}}^{R} \) integrand and then approximating

\[ \omega_{L_{(a-1)}}^{R} \approx C_{L_{(a-1)}}^{L} \left[ \omega_{L_{(a-1)}}^{N} + \rho \omega_{L_{(a-1)}}^{N} \right] \]

where the subscript \( n \) is the value for \( t \) midway between times \( t_{a-1} \) and \( t_{a} \). Using Eq. (52) in Eq. (50) then obtains

\[ \zeta_{a} \approx C_{L_{(a-1)}}^{L} \left[ \omega_{L_{(a-1)}}^{N} \right] \]

with \( \omega_{L_{(a-1)}}^{N} \) evaluated using Eq. (14) and

\[ \begin{align*}
\Delta R_{m}^{N} & \approx \int_{t_{a-1}}^{t_{a}} v^{N} \text{ dt} \\
\end{align*} \]

where \( T_{m} \) is the number of computer \( m \) cycle update period \( t_{a} - t_{a-1} \) and \( j \) is the number of computer \( m \) cycles over the \( t_{a-1} \) to \( t_{a} \) \( m \)-cycle computer update period.

The subscript \( n \) is the term in Eq. (53) are all functions of position, which (from Part 2, Ref. 13) is updated following the attitude update at the \( n \)-cycle rate. Hence, to calculate these terms in Eq. (52), an approximate extrapolation formula must be used based on previously computed values of the \( n \) parameters. For example, a linear extrapolation formula using the last two computed values for \( n \) should be

\[ \zeta_{a} \approx \zeta_{a-1} + \frac{1}{2!(n-1)} \left( \zeta_{a} - \zeta_{a-1} \right) \]

In Part 2 (Ref. 13) we find that the \( v^{N} \) velocity update follows the attitude update. Therefore, current and past \( m \)-cycle values of \( v^{N} \) are available for evaluating the Eq. (54) integral for \( \Delta R_{m}^{N} \). Using a trapezoidal integration algorithm for Eq. (54) obtains

\[ \begin{align*}
\Delta R_{m}^{N} & \approx \frac{1}{2} \left( v_{m}^{N} + v_{m-1}^{N} \right) T_{m} \\
\end{align*} \]

where \( T_{m} \) is the computer \( m \) cycle update period \( t_{a} - t_{a-1} \).

Part 2 (Ref. 13) also develops a high-resolution version of \( \Delta R_{m}^{N} \) for precision position updating that accounts for dynamic angular rates and accelerations within the \( m-1 \) to \( m \) cycle update interval.

3. Normalization and Orthogonalization

From its basic definition in Sec. II.B, the columns (and rows) of \( C_{B}^{B} \) represent orthogonal unit vectors, which, therefore, should be unity in magnitude (normality condition) and mutually orthogonal to one another (orthogonality condition). In addition to the basic \( C_{B}^{B} \) update algorithms already described, a normalization and orthogonalization algorithm is frequently included to ensure that the \( C_{B}^{B} \) rows and columns remain normal and orthogonal. Factors that cause \( C_{B}^{B} \) orthogonality/normality errors and thereby reduce the computer wordlength for the total number of \( C_{B}^{B} \) algorithm update cycles expected, and insufficient number of terms carried in the Eqs. (26) and (49) Taylor series expansions (truncation error). It is important to note (Ref. 12, Sec. 3.4.1)
that orthogonality and normalization errors can only be produced from errors in the software implementation of Eqs. (23), (24), (26), and (49), not from errors in the algorithms feeding these equations or from inertial sensor input errors. The overall design/verification process for the $C_B^L$ integration algorithm software must assure error-free programming and acceptable roundoff/truncation error for the angular rate environment anticipated over the expected navigation time period, a readily achievable goal with today’s computer/software development technology. Nevertheless, inclusion of a $C_B^L$ orthogonality/normality correction algorithm has been traditionally employed in many strapdown navigation software packages for enhanced accuracy and to relax the more stringent requirement of not allowing any orthogonality/normalization error in the basic $C_B^L$ updating operations. The algorithms used for normalization/orthogonalization are based on the property that the transpose of a direction cosine matrix equals its inverse (see Sec. II.B); consequently, the product of $1$ times its transpose should be identity. Variations from this condition measure the orthogonality/normality error, which can then be used by a control algorithm in iterative fashion for correction (Refs. 9, 12, Secs. 7.1.1.3 and 15, pp. 216-218).

B. Attitude Quaternion

The updating algorithm for the $q_B^L$ attitude quaternion is designed to achieve the same numerical result at the attitude update times as would the formal continuous integration of the Eq. (12) $q_B^L$ expression at the same time instant. The updating algorithm for the $q_B^L$ attitude quaternion is developed following the identical procedure used for the $C_B^L$ updating derivation in Sec. IV.A. Thus, using the Eq. (7) attitude quaternion chain rule, we write

$$q_{B(m)}^{L(n-1)} = q_{B(m-1)}^{L(n-1)} q_{B(m-1)}^{L(m)}$$

where

$$q_{B(m-1)}^{L(m)} = q_B^L$$ relating the $B$ frame at time $t_{m-1}$ to the $L$ frame at time $t_{m-1}$

$$q_{B(m-1)}^{L(n)} = q_B^L$$ relating the $B$ frame at time $t_m$ to the $L$ frame at time $t_m$

$$q_{B(m-1)}^{L(n)} = \text{attitude quaternion that accounts for } B \text{ frame rotation relative to inertial space from its orientation at time } t_{m-1} \text{ to its orientation at time } t_m$$

$$q_{B(m-1)}^{L(n-1)} = \text{attitude quaternion that accounts for } L \text{ frame rotation relative to inertial space from its orientation at time } t_{n-1} \text{ to its orientation at time } t_n$$

The updates for $q_B^L$ are performed by $q_{B(m)}^{L(m-1)}$ and $q_{B(m-1)}^{L(m)}$ in Eqs. (57) and (58), algorithms for which are derived separately next.

1. Body Frame Rotation

Equation (57) updates the $q_B^L$ attitude quaternion using $q_{B(m)}^{L(m-1)}$ to account for angular rotation $\omega_B^L$ of the strapdown sensor (body) $B$ frame relative to nonrotating space. The formal definition for $q_{B(m)}^{L(m-1)}$ is

$$q_{B(m)}^{L(m-1)} = q_B^L = 1 + \int_{t_{m-1}}^{t_m} q_{B(m-1)}^{L(m-1)} dt$$

where $B(t)$ is the $B$ frame attitude at an arbitrary time in the interval $t_{m-1}$ to $t_m$.

The $q_{B(m-1)}^{L(m-1)}$ attitude quaternion can also be expressed in terms of a rotation vector defining the frame $B(m-1)$ relative to frame $B(m-1)$. Applying Eq. (5) with Taylor series expansion for the coefficient terms obtains

$$q_{B(m)}^{L(m-1)} = \begin{bmatrix} \cos 0.5 \phi_m \\ \sin 0.5 \phi_m \\ 0.5 \phi_m \\ 0 \end{bmatrix}$$

$$\sin 0.5 \phi_m = 1 - 0.5 \phi_m^2 + 0.5 \phi_m^4$$

$$\cos 0.5 \phi_m = 1 - 0.5 \phi_m^2 + 0.5 \phi_m^4$$

(60)

The $\phi_m$ rotation vector in Eq. (60) for attitude quaternion updating is identical to $\phi_m$ used in Sec. IV.A.1 for $C_B^L$ direction cosine matrix updating and is calculated using the identical algorithm provided by Eqs. (35), (41), and (42) or Eqs. (35), (43), and (47).

2. Local Level (L) Frame Rotation

Equation (58) updates the $q_B^L$ attitude quaternion using $q_{B(m)}^{L(n-1)}$ to account for angular rate of the local-level coordinate $L$ frame relative to nonrotation space $\omega_L^L$. The formal definition for $q_{B(m)}^{L(n-1)}$ is

$$q_{B(m)}^{L(n-1)} = q_1 + \int_{t_{n-1}}^{t_n} q_{B(n-1)}^{L(n-1)} dt$$

(61)

with $L(t)$ in Eq. (61) representing the $L$ frame attitude at an arbitrary time in the interval $t_{n-1}$ to $t_n$.

The $q_{B(m)}^{L(n-1)}$ attitude quaternion can also be expressed in terms of the rotation vector defining the frame $L(n-1)$ attitude relative to frame $L(n-1)$. Applying Eq. (5) with Taylor series expansion for the integral terms yields

$$q_{B(m)}^{L(n-1)} = \begin{bmatrix} \cos 0.5 \zeta_n \\ -\sin 0.5 \zeta_n \\ 0.5 \zeta_n \\ 0 \end{bmatrix}$$

$$\sin 0.5 \zeta_n = 1 - 0.5 (0.5 c_\zeta_n)^2 + 0.5 (0.5 c_\zeta_n)^4$$

$$\cos 0.5 \zeta_n = 1 - 0.5 (0.5 c_\zeta_n)^2 + 0.5 (0.5 c_\zeta_n)^4$$

(62)

The negative sign on $\zeta_n$ accounts for the opposite phase sense of $q_{B(m)}^{L(n-1)}$, which describes the frame $L(n-1)$ attitude relative to frame $L(n)$ compared with the rotation vector $\zeta_n$ phase sense, which describes the frame $L(n)$ attitude relative to frame $L(n-1)$. The $\zeta_n$ rotation vector in Eqs. (62) is identical to $\zeta_m$ used for $C_B^L$ direction cosine matrix updating and is calculated using the identical computational algorithm described in Sec. IV.A.2 and provided by Eqs. (53), (55), and (56).

An approximate form of Eqs. (62) that is comparable in accuracy to direction cosine updating Eq. (51) is readily obtained by substitution and truncation

$$q_{B(m)}^{L(n-1)} = \begin{bmatrix} 1 - 0.5 (0.5 c_\zeta_n)^2 \\ -0.5 \zeta_n \end{bmatrix}$$

(63)

The comments in Sec. IV.A.2 regarding the advisability of using the simplified Eq. (51) direction cosine local-level frame updating algorithm also apply regarding use of Eq. (63) for attitude quaternion updating rather than the complete Eqs. (62) form.

3. Normalization

To preserve the fundamental attitude quaternion normality characteristic discussed in Sec. II.B, a normalization algorithm is frequently incorporated as an outer-loop function in the $q_B^L$ attitude quaternion updating process. The discussion in Sec. IV.A.3 for direction cosine matrices regarding the need for a normalization/orthogonalization function is equally applicable for the attitude quaternion, the only exception being that orthogonalization has no meaning in the definition for the quaternion (as it does for the attitude direction cosine matrix); hence, the orthogonalization discussion in
Sec. IV.A.3 does not apply. If a quaternion normalization algorithm is to be utilized, it is based on comparing the magnitude of

\[ F \]

Faced with the multitude of potential strapdown inertial navigation algorithms to choose from, the software designer ... e.g., Ref. 12, Sec. 11.2. For the attitude algorithms discussed, simplified analytical error models can also be

Table 1 summarizes the algorithms described for the strapdown inertial navigation attitude integration function listed in the order they would be executed in the navigation computer. Table 1 lists the algorithm function, input parameters, output parameters, and equation number.

<table>
<thead>
<tr>
<th>Algorithm function</th>
<th>Input</th>
<th>Output</th>
<th>Equation number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrated B frame angular rate increments</td>
<td>[ \Delta \alpha_t ]</td>
<td>[ \alpha_t, \alpha_m ]</td>
<td>(41) or (46)</td>
</tr>
<tr>
<td>Coning increment</td>
<td>[ \Delta \alpha_t, \alpha_i ]</td>
<td>[ \beta_m ]</td>
<td>(42) or (47)</td>
</tr>
</tbody>
</table>

Normal-speed calculations for Earth related parameters

| N frame Earth rate components | \[ C_N^E \] | \[ \omega_N^E \] | (14) |
| Vertical transport rate component | \[ C_N^E \] | \[ \rho_{ZN} \] | Ref. 12, Sec. 4.6 |
| Curvature matrix | \[ C_N^E, h \] | \[ F_c \] | Ref. 12, Sec. 5.3 |

Normal-speed velocity calculations

| N frame velocity update | \[ v^N \] | Part 2 (Ref. 13) |

Normal-speed attitude calculations

| B frame rotation vector | \[ \alpha_m, \beta_m \] | (35) |
| B frame rotation matrix (for attitude direction cosine matrix updating) | \[ \phi_m \] | \[ C_{B_{(m-1)}} \] | (26) |
| B frame rotation quaternion (for attitude quaternion updating) | \[ \phi_m \] | \[ q_{B_{(m-1)}} \] | (60) |
| Attitude update for B frame rotation (direction cosine matrix form) | \[ C_{B_{(m-1)}}, C_{B_{(m-1)}} \] | \[ C_{B_{(m-1)}} \] | (23) |
| Attitude update for B frame rotation (quaternion form) | \[ q_{B_{(m-1)}}, q_{B_{(m-1)}} \] | \[ q_{B_{(m-1)}} \] | (57) |
| N frame position increment | \[ v^N \] | \[ \Delta R_{mN}^N \] | (56) |
| L frame rotation vector | \[ \omega_N^F, \rho_{ZN}, F_c, \Delta R_{mN}^N \] | \[ C_{L_{(m-1)}} \] | (53), (55) |
| L frame rotation matrix for attitude direction cosine matrix updating (exact form) | \[ \omega_{L_{(m-1)}} \] | \[ C_{L_{(m-1)}} \] | (49) |
| L frame quaternion for attitude quaternion updating (exact form) | \[ \omega_{L_{(m-1)}} \] | \[ q_{L_{(m-1)}} \] | (62) |
| Attitude update for L frame rotation (direction cosine matrix form) | \[ C_{L_{(m-1)}}, C_{L_{(m-1)}} \] | \[ C_{L_{(m-1)}} \] | (24) |
| Attitude update for L frame rotation (quaternion form) | \[ q_{L_{(m-1)}}, q_{L_{(m-1)}} \] | \[ q_{L_{(m-1)}} \] | (58) |
| Normalization and orthogonalization corrections (for attitude direction cosine matrix) | \[ C_{L_{(m-1)}} \] | \[ C_{L_{(m-1)}} \] | Sec. IV.A.3 |
| Normalization corrections (for attitude quaternion) | \[ q_{L_{(m-1)}} \] | \[ q_{L_{(m-1)}} \] | Sec. IV.B.3 |

Normal-speed position calculations

| Position direction cosine matrix and attitude update | \[ C_N^E, h \] | Part 2 (Ref. 13) |

V. Attitude Integration Algorithm Summary

Table 1 summarizes the algorithms described for the strapdown inertial navigation attitude integration function listed in the order they would be executed in the navigation computer. Table 1 lists the algorithm function, input parameters, output parameters, and equation number.

VI. Algorithm and Execution Rate Selection

Faced with the multitude of potential strapdown inertial navigation algorithms to choose from, the software designer must ultimately choose one set for the application at hand. The algorithms presented in this Part 1 and the subsequent Part 2 (Ref. 13) papers are but one version of many similar algorithms developed over the years by several authors. The process of selecting the algorithm set for a particular application should consider the allowable algorithm error under anticipated angular rates/accelerations/vibrations, the capability of the projected target navigation computer for the required algorithm execution rate, and the complexity of the design procedure for software validation/documentation with the selected algorithms.

Evaluation of candidate algorithm error characteristics is generally performed using computerized time-domain simulators that exercise the algorithms in particular groupings at their selected repetition rates. The simulators generate simulated strapdown inertial sensor angular rate/acceleration profiles for algorithm test input together with known navigation parameter solutions for algorithm output comparison, e.g., Ref. 12, Sec. 11.2. For the attitude algorithms discussed, simplified analytical error models can also be used to predict high-speed coning algorithm error under specified coning rates/amplitudes as a function of algorithm repetition rate (Refs. 9–11 and 12, Sec. 10). The coning rates/amplitudes must be derived either from empirical data or, more commonly, from analytical models of the sensor assembly mount imbalance and its response to external input vibration at particular frequencies (Ref. 12, Sec. 10). Frequency-domain simulators can be used to evaluate high-speed coning algorithm error under specified input vibration power spectral density profiles and sensor assembly mount imbalance as a function of algorithm repetition rate (Ref. 12, Sec. 10). For example, the coning algorithm described by Eqs. (46) and (47) can be shown by such simulators to have an error rate of 0.00037 deg/h when operated at a 2-kHz repetition rate under exposure to 7.6 g rms wideband random linear input vibration (flat 0.04 g^2/Hz density from 20 to 1000 Hz, then decreasing logarithmically to 0.01 g^2/Hz at 2000 Hz). The linear vibration generates a 0.0003-rad multiaxis angular oscillation of the sensor assembly with a corresponding coning rate of 9.9 deg/h due to the following typical sensor assembly mount characteristics selected as simulator input parameters: 50-Hz linear vibration mode undamped natural frequency, 0.125 linear vibration mode damping ratio, 71-Hz rotary vibration mode undamped natural frequency, 0.18 rotary vibration mode damping ratio, 5% sensor assembly mount mechanical isolator spring and damping imbalance, and 1.4% sensor assembly center of mass offset from mechanical e.g., mount center (percent of distance between isolators).

The capabilities of modern-day computer and INS software technology make it reasonable to specify that the attitude algorithm error be no greater than 5% of the equivalent error produced by the INS inertial sensors (whose cost increases dramatically with accuracy demands). For an INS with a 0.0007-deg/h angular rate sensor bias accuracy requirement (for a typical aircraft INS having 1 n mph 50 percentile radial position error rate), the 0.00037-deg/h coning algorithm error rate satisfies the 5% allowance.
So long as the selected integration algorithm is analytically valid, it can be improved in accuracy by increasing its repetition rate. Continuing computer technology advances (increasing speed and decreasing program memory cost), therefore, tend to diminish any advantages one algorithm might have over another (usually measured, primarily, by accuracy for a given repetition rate and, secondarily, by required program memory). Excessively high repetition rates are to be avoided, however (even if computer throughput allowances permit) to limit error buildup caused by computer finite wordlength effects and rectification of high-frequency multiaxis sensor errors (high-frequency error output from one inertial sensor that is frequency correlated with outputs from sensors in the other axes, denoted as pseudoconing error for the coning computation in Part 1 and pseudosculling error for the sculling part of the velocity calculation in Ref. 13, Part 2). The finite computer wordlength error effect is generally not a major factor with modern computer technology, typically having 64-bit double precision floating point wordlengths. The pseudoconing/sculling issue must be resolved on an individual design basis depending on the characteristics of high-frequency error effects anticipated from the inertial sensor assembly in its operational dynamic environment. A general ground rule to follow in coning/sculling algorithm repetition rate selection is to run the algorithms only as fast as required to accurately measure anticipated real multiaxis high-frequency angular rates/accelerations that can potentially rectify into real attitude/velocity change, but no faster, to minimize the likelihood of rectifying high-frequency sensor output error into attitude/velocity error buildup.

The ultimate selection of algorithms to be used in a particular application is generally made based on the previous experience of the responsible design engineer. The author has had long experience with the algorithms described and feels comfortable adapting them to any strapdown application. They are well defined analytically, can be programmed using a simple sequential software executive structure, readily lend themselves to straightforward validation procedures, and are easily adapted to the requirements and constraints of particular applications.

VII. Concluding Remarks

We have defined the overall requirement for the strapdown inertial navigation integration function (in the form of continuous differential equations) and developed the attitude integration algorithms based on the two-speed updating approach: an exact algorithm for moderate speed updating fed by a simplified high-speed algorithm. The high-speed algorithm contains a simple summing operation of angular rate sensor inputs plus an approximate coning motion integration function. Under conditions where the angular rate vector is not rotating, i.e., zero coning, the coning term becomes zero, the simple summing operation becomes an analytically exact representation of the attitude change, and the overall attitude update operation is error free. Where computer throughput restrictions are not at issue, the two-speed structure presented can be compressed into a single high-speed format by operating the moderate-speed algorithm at the high-speed rate. This general form for the two-speed attitude algorithm defines a framework for design of the velocity/position integration algorithms in Part 2 (Ref. 13) to have similar characteristics: analytical exactness under constant angular rate/specific force acceleration and using a small approximate high-speed computation to measure deviations from the latter condition (denoted as sculling for the velocity algorithm and sculling for the position algorithm).

A summary of the attitude integration algorithms developed is provided in Table 1 as a listing in the order they would be executed in the navigation computer. A similar table is provided in Part 2 (Ref. 13) for the velocity/position integration algorithms.

References
14Morse, P. M., and Feshbach, H., Methods of Theoretical Physics, McGraw–Hill, New York, 1953.
Strapdown Inertial Navigation Integration Algorithm Design
Part 2: Velocity and Position Algorithms
Paul G. Savage
Strapdown Associates, Inc., Maple Plain, Minnesota 55359

This series of two papers (Parts 1 and 2) provides a rigorous comprehensive approach to the design of the principal software algorithms utilized in modern-day strapdown inertial navigation systems: integration of angular rate into attitude, acceleration transformation/Integration into velocity, and integration of velocity into position. The algorithms are structured utilizing the two-speed updating approach originally developed for attitude updating; an analytically exact equation is used at moderate speed to update the integration parameter (attitude, velocity, or position) with input provided from a high-speed algorithm measuring rectified dynamic motion within the parameter update time interval (coning for attitude updating, sculling for velocity updating, and sculling writer's terminology) for high-resolution position updating. The algorithm design approach accounts for angular rate-specific force acceleration inputs from the strapdown system inertial sensors, as well as rotation of the navigation frame used for attitude referencing and velocity integration. The Part 1 paper (Savage, P. G., “Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms,” Journal of Guidance, Control, and Dynamics, Vol. 21, No. 1, 1998, pp. 19–28) defined the overall design requirement for the strapdown inertial navigation integration function and developed the attitude updating algorithms. This paper, Part 2, deals with design of the acceleration transformation/velocity integration and position integration algorithms. Although Parts 1 and 2 often cover basic concepts, the material presented is intended for use by the practitioner who is already familiar with inertial navigation fundamentals.

Nomenclature

\[ A, A_1, A_2 = \text{arbitrary coordinate frames} \]
\[ a_{SF} = \text{specific force defined as the acceleration relative to nonrotating inertial space produced by applied nongravitational forces, measured by accelerometers} \]
\[ C_{A_1}^{A_2} = \text{direction cosine matrix that transforms a vector from its } A_2 \text{ frame projection form to its } A_1 \text{ frame projection form} \]
\[ I = \text{identity matrix} \]
\[ V_A = \text{column matrix with elements equal to the projection of vector } V \text{ on frame } A \text{ axes} \]
\[ (V^A \times) = \text{skew symmetric (or cross product) form of } V^A \text{ represented by the square matrix} \]
\[
\begin{bmatrix}
0 & -V_{ZA} & V_{YA} \\
V_{ZA} & 0 & -V_{XA} \\
-V_{YA} & V_{XA} & 0
\end{bmatrix}
\]
where \( V_{XA}, V_{YA}, V_{ZA} \) are the components of \( V^A \); matrix product of \((V^A \times)\) with another \( A \) frame vector equals the cross product of \( V^A \) with the vector in the \( A \) frame
\[ \omega_{A_1 A_2} = \text{angular rate of coordinate frame } A_2 \text{ relative to coordinate frame } A_1; \text{ when } A_1 \text{ is the inertial } I \text{ frame, } \omega_{A_1 A_2} \text{ is the angular rate measured by angular rate sensors mounted on frame } A_2 \]

I. Introduction

A STRAPDOWN inertial navigation system (INS) is typically composed of an orthogonal three-axis set of inertial angular rate sensors and accelerometers providing data to the INS computer. The inertial sensors are directly mounted (strapdown) to the INS chassis structure in contrast with original INS technology that utilized an active multiaxis gimbal isolation mounting assembly to isolate the sensors from rotation. The principal software functions executed in the strapdown INS computer are the integration of sensed angular rate into attitude, transformation of accelerometer sensed specific force acceleration into a navigation coordinate frame, addition of software modeled gravity to the transformed specific force to calculate total acceleration, and double integration of total acceleration into velocity and position. The key element in the INS software design process is the development of repetitive digital algorithms that will flawlessly execute the attitude, velocity, and position digital integration functions in the presence of dynamic angular rate/specific force acceleration inputs.

As discussed in Part 1 (Ref. 1), most modern-day strapdown INSs utilize attitude updating algorithms based on a two-speed approach\(^4\); a higher-order updating algorithm is processed at moderate repetition rate using inputs from a high-speed algorithm. The moderate-speed routine can be represented by an exact closed-form attitude updating operation.\(^3,4\) The high-speed algorithm is designed to accurately account for multiaxis high-frequency angular motion between moderate speed algorithm updates that can rectify into systematic attitude change (traditionally denoted as coning). Originally conceived as a simple first-order algorithm,\(^7\) today's high-speed attitude algorithms have taken advantage of increased throughput capabilities in modern-day computers and become higher order for improved accuracy (Refs. 1; 5–7; and 8, Chap. 7). While the attitude updating function has been evolving to its current form, very little parallel work has been published on the development of the companion strapdown INS algorithms for specific force acceleration transformation/velocity integration and position integration, the subject of this paper.

The specific force transformation algorithm processes the inertial sensor data to calculate an integrated specific force increment in navigation coordinates over the velocity algorithm update time interval. The velocity is updated by adding the navigation frame specific force increment (plus an increment for gravity and coordinate frame rotation effects) to the previous velocity value. A key function of the transformation algorithms is to accurately account for attitude rotation (hence, rotation of the strapdown accelerometers) during the velocity update time period. In some applications, this has been achieved using a centering algorithm\(^8\) in which attitude data for the specific force transformation is updated at the center of the velocity update time interval (thereby introducing a staggered
attitude update/velocity update software architecture). The transformation operation then consists of integrating the accelerometer specific force output over the velocity update interval and transforming the integrated specific force increment to the navigation frame using attitude data at the center of the velocity update time interval. A variation of the latter approach updates the attitude at twice the velocity update rate so that the attitude and velocity updates are available for specific force increment transformation. Another variation calculates the attitude used for specific force transformation as the average of the computed attitude at the start and end of the velocity update time interval. A two-speed approach can also be used for specific force transformation/velocity integration in a dynamic environment that parallels the two-speed attitude integration approach (Refs. 5 and 8, Sec. 7.2). A high-speed algorithm is designed to account for high-frequency angular and linear oscillations that can rectify into systematic velocity buildup (traditionally denoted as sculling), and a moderate-speed algorithm executes the specific force transformation based on inputs from the high-speed algorithm.

In general, the specific force transformation/velocity integration algorithms have lacked the analytical sophistication of the attitude integration algorithms, being typically limited to first-order accuracy under maneuvering conditions. Virtually no specialized work has been reported for the inertial navigation position integration function. From the writer’s understanding, modern-day strapdown INSs typically generate position as a simple trapezoidal integration of velocity at an update rate equal to or lower than the velocity update frequency. For applications requiring precise position change data in a dynamic environment, such a rudimentary approach to position integration may prove inadequate.

This paper provides a comprehensive process for the design of strapdown inertial navigation specific force transformation, velocity integration, and position integration algorithms. The material presented is a condensation of Refs. 8, Secs. 7.2 and 7.3 (an expansion of material in Ref. 5), emphasizing a more rigorous analytical formulation and the use of exact closed-form equations where possible for ease in computer software documentation/validation. The velocity and position algorithms presented are structured using a two-speed computation format; the moderate-speed algorithm, e.g., 50–200 Hz, is designed to be exact under constant angular rate/specification force acceleration conditions during the moderate-speed update interval; the moderate-speed algorithms fed by a high-speed computation algorithm, e.g., 1–4 kHz, that accounts for dynamic variations from constant angular rate/specific force [sculling for the velocity algorithm and sculling (writer’s terminology) for the position algorithm]. Included is a rigorous treatment of navigation coordinate frame rotation during the integration update time periods.

This paper is organized as follows. Section II defines the coordinate frames utilized. Section III utilizes the Part I (Ref. 1) attitude algorithm derivation as a model to formulate two-speed specific force acceleration transformation/velocity integration algorithms. Section IV then uses Sec. III as a framework for the development of position updating algorithms in two forms: a traditional form based on trapezoidal integration and a two-speed high-resolution form. A tabular reference summary of the derived algorithms is presented in Sec. V. Section VI provides a general discussion of the process followed in selecting algorithms for a particular application and establishing their execution rates. Concluding remarks are provided in Sec. VII.

Finally, it is important to recognize that, whereas the original intent of the two-speed approach was to overcome throughput limitations of early computer technology (1965–1975), that limitation is rapidly becoming insignificant with continuing rapid advances in modern high-speed computers. This provides the motivation to eventually return to a simpler single-speed algorithm structure whereby all computations are executed at a repetition rate that is sufficiently high to accurately account for multitaxis high-frequency angular rate and specific force acceleration rectification effects. The two-speed structure presented in this paper and in Part I (Ref. 1) is compatible with compression into such a single-speed format as explained in the particular sections where the algorithms are formulated.

II. Coordinate Frames

A coordinate frame is an analytical abstraction defined by three consecutively numbered (or lettered) unit vectors that are mutually perpendicular to one another in the right-hand sense. It can be visualized as a set of three perpendicular lines (axes) passing through a common point (origin) with the unit vectors emanating from the origin along the axes. In this paper, the physical locations of the coordinate frame origins are arbitrary. A vector’s components (or projections) in a particular coordinate frame equal the dot product of the vector with the coordinate frame unit vectors. The vectors used in this paper are classified as free vectors and, hence, have no preferred location in coordinate frames in which they are analytically described.

The coordinate frames are defined as follows.

1) The E frame is the Earth-fixed coordinate frame used for position location definition. It is typically defined with one axis parallel to the Earth polar axis and with the other axes fixed to the Earth and parallel to the equatorial plane.

2) The N frame is the navigation coordinate frame having its Z axis parallel to the upward vertical at the local Earth surface referenced position location. It is used for integrating acceleration into velocity and for defining the angular orientation of the local vertical in the E frame.

3) The L frame is the locally level coordinate frame parallel to the N frame but with the Z axis parallel to the downward vertical and X and Y along N frame Y and X axes. It is used as the reference for describing the strapdown sensor coordinate frame orientation.

4) The B frame is the strapdown inertial sensor coordinate frame (body frame) with axes parallel to nominal right-handed orthogonal sensor input axes.

5) The I frame is the nonrotating inertial coordinate frame used as a reference for angular rate measurements. Particular orientations selected for the I frame are discussed in the sections where its orientation is pertinent to analytical operations.

III. Velocity Update Algorithms

In this section we develop algorithms for integrating the Ref. 1, Eq. (20), velocity rate equation using Ref. 1, Eqs. (16) and (18), for the specific force transformation term and using angular rates from Ref. 1, Eqs. (14) and (15), in the Coriolis acceleration term (angular rate products with velocity):

\[
\dot{\mathbf{v}}^N = C_N^C \mathbf{a}^B + g_P^N - \left( \omega_{EN}^N + 2 \omega_{IE}^N \right) \times \mathbf{v}^N \tag{1}
\]

\[
\omega_{IE}^N = C_N^C \omega_{IE}^C \tag{2}
\]

\[
\omega_{EN}^N = F_C \left( \mathbf{u}_{EN}^N \times \mathbf{v}^N \right) + \rho_{ZU} \mathbf{u}_{ZU}^N \tag{3}
\]

where \( \mathbf{v} \) is the velocity relative to the Earth defined analytically as the time derivative in the E frame of the position vector from Earth’s center to the INS, and \( g_P \) is plumb-bob gravity (or gravity) that, for a stationary INS, lies along the line of a plumb bob. \( F_C \) is a curvature matrix (3 x 3) that is a function of position having elements \( i,j \) and \( i,j \) equal to zero and the remaining elements symmetrical about the diagonal. For a spherical Earth model, the remaining elements of \( F_C \) are zero off the diagonal and equal the reciprocal of the radial distance from the Earth’s center to the INS on the diagonal. For an oblate Earth model, the remaining elements of \( F_C \) are zero off the diagonal and equal the reciprocal of the radial distance from the Earth’s center to the INS on the diagonal. For an oblate Earth model, the remaining \( F_C \) terms represent the local curvature on the Earth’s surface projected to the INS altitude (see Ref. 8, Sec. 5.3, for closed-form expression).

\( \rho_{ZU} \) is the vertical component of \( \omega_{EN}^N \). The value selected for \( \rho_{ZU} \) depends on the type of N frame utilized, e.g., wander azimuth or free azimuth designed to assure that \( \omega_{EN}^N \) is nonsingular for all Earth locations (see Ref. 8, Sec. 4.6, and Ref. 10, pp. 88–89). \( \mathbf{u}_{ZU} \) is a unit vector upward along the geodetic vertical (the Z axis of the N frame).

Equation (1) uses direction cosine matrix transformed specific force rather than the alternative Ref. 1, Eq. (17), quaternion transformation approach, e.g., for situations where the \( B \) frame attitude is computed in the form of an attitude quaternion. The velocity integration algorithm based on quaternion specific force transformation can be developed by extension of the results presented here.
The digital velocity integration algorithm is formulated directly from Eq. (1) as
\[
v^N_m = v^N_{m-1} + C^N \Delta v^N_{SFm} + \Delta v^N_{G/Coum}
\]
where \(m\) is the digital velocity integration algorithm update rate computer cycle index.

If vertical channel gravity/divergence stabilization is to be incorporated, an additional update operation would be included in Eq. (4) representing the vertical velocity control function (Ref. 8, Sec. 4.4.1, and Ref. 10, pp. 102-103).

Digital algorithms are formulated next for the gravity/Coriolis velocity increment \(\Delta v^N_{G/Coum}\), in Eq. (5) and the integrated transformed specific force increment \(\Delta v^N_{SFm}\) in Eq. (6).

### A. Gravity/Coriolis Velocity Increment

The \(g^N_m\) term in Eq. (5) is a function of position location with very small horizontal components. Because the position varies smoothly over a digital algorithm \(m\) cycle with limited magnitude change (particularly in altitude), \(g^N_m\) in Eq. (5) can be approximated by its average value across the \(m\) cycle. Because the Eq. (5) Coriolis term is small (due to the small size of the angular rates) and because velocity varies smoothly over an \(m\) cycle, the Coriolis contributors can also be approximated by their average value over the \(m\) cycle. The latter rationale forms the basis for the following algorithm for \(\Delta v^N_{G/Coum}\) in Eq. (5) using Eq. (3) for \(\omega^N_{x,y,z}\):

\[
\Delta v^N_{G/Coum} \approx \left\{ g^N_{m-1} - \frac{1}{2} \omega^N_{x,y,z} \times v^N_m + F_C \left( u^N_{Zm} \times v^N_m \times \frac{1}{m} \right) \right\} \tau_m
\]

where \(m - \frac{1}{2}\) designates the parameter value midway between \(t_{m-1}\) and \(t_m\), and \(\tau_m\) is the velocity integration algorithm update period \(t_m - t_{m-1}\).

The \(\omega^N_{x,y,z}\) term in Eq. (7) is evaluated with Eq. (2), and \(g^N_{m-1}\) is calculated from Ref. 1, Eq. (19). Because \(\Delta v^N_{G/Coum}\) is used in Eq. (4) to update \(v^N_m\) from its \(m - 1\) to \(m\) cycle value, \(\omega^N_{x,y,z}\) is not explicitly available for Eq. (7) and must be approximated based on extrapolation from past values. An example is the linear extrapolation algorithm

\[
v^N_m \approx v^N_{m-1} + \frac{1}{j} \left[ v^N_{m-1} - v^N_{m-2} \right]
\]

where

\[
n = \text{computer cycle index for position updates}
\]

\[
j = \text{number of } m \text{ cycles in each } n \text{ cycle}
\]

\[
r = \text{number of } m \text{ cycles since last } n \text{ cycle, i.e., since } t_{m-1}
\]

### B. Integrated Transformed Specific Force Increment

A digital algorithm for integrated transformed specific force increment equation (6) must account for rotation of the local level L frame and the strapdown sensor body B frame during the \(t_{m-1}\) to \(t_m\) computer cycle period. Adopting the same notation used in Ref. 1, Sec. IVA, to describe discrete orientations of the \(L\) and \(B\) frames relative to inertial space \(I\) at computer update time instants, Eq. (6) can be expanded using the Ref. 1, Eq. (3), chain rule as follows:

\[
\Delta v^N_{SFm} = \int_{t_{m-1}}^{t_m} C_1^{L_{(m-1)}} C_2^{L_{(m-1)}} C_3^{B_{(m-1)}} a^N_{SF} \, dt
\]

or, on further expansion,

\[
\Delta v^N_{SFm} = \int_{t_{m-1}}^{t_m} C_1^{L_{(m-1)}} C_2^{L_{(m-1)}} C_3^{B_{(m-1)}} a^N_{SF} \, dt
\]

\[
\Delta v^N_{SFm} = \Delta v^N_{SFm} + \Delta v^N_{SFm}
\]

Equations (11–13) allow for the general case whereby the \(C^N_m\) matrix is updated for \(L\) frame rotation at a cycle rate (index \(n\)) that may differ from (be slower than) the \(C^N_m\) update rate for \(B\) frame rotation (index \(m\)). For example, in the interest of minimizing computer throughput requirements, the software architecture might have the \(n\) cycle \(L\) frame update rate set five times slower than the \(m\) cycle \(B\) frame update rate. Equations (11–13) are also valid, however, if we choose to update \(C^N_m\) at equal rates for \(B\) and \(L\) frame motion, i.e., \(n = m\). Note that, for \(n \neq m\), Eq. (13) requires an \(L\) frame orientation formulation at the \(B\) frame \(m\) cycle update time (for \(t_{m-1}\) in the \(C_3^{L_{(m-1)}}\) matrix). Note also that the form of Eq. (11) is based on the use of \(C_3^{B_{(m-1)}}\) at the preceding \(B\) frame \(m\) cycle, i.e., \(B_{(m-1)}\), in the \(C_3^{L_{(m-1)}}\) matrix. This implies that \(C^N_m\) will be updated for \(B\) frame rotation following the Eq. (11) transformation operation. It remains to define algorithms for \(C_1^{L_{(m-1)}}\) in Eq. (13) to account for local level frame rotation during specific force transformation and for the \(\Delta v^N_{SFm}\) body frame integrated specific force increment term in Eq. (12).

#### 1. Correction for Local Level Frame Rotation
**During Specific Force Transformation**

Because of the slow angular rate of the \(L\) frame relative to inertial space, \(C_1^{L_{(m-1)}}\) in Eq. (13) is very close to the identity matrix \(I\). For many applications, \((C_1^{L_{(m-1)}} - I)\) in Eq. (13) can, therefore, be totally ignored as negligible compared to other acceleration error sources. For high-accuracy applications where \((C_1^{L_{(m-1)}} - I)\) is to be included, a first-order form of the Ref. 1 Eqs. (49) and (50) usually suffices, whereby

\[
C_1^{L_{(m)}} \approx I - (\zeta_{n-1,m} \times )
\]

\[
\zeta_{n-1,m} = \int_{t_{m-1}}^{t_m} \omega^N_L \, dt
\]

We then approximate \(\omega^N_L\) in Eq. (15) using Eq. (3) in Ref. 1, Eq. (13), and the assumption of slowly changing contributors as in Sec. III.A,

\[
\omega^N_L = C_3^N (\omega^N_{x,y,z} + \rho_{ZN-1,m} u^N_{ZN-1,m})
\]

\[
+ F_{C_{e-1,m}} (u^N_{Zm} \times v^N)\]

where the subscript \(n - 1, m\) indicates the value for the parameter midway between times \(t_{m-1}\) and \(t_m\).

Substituting Eq. (16) into Eq. (15) yields

\[
\zeta_{n-1,m} \approx C_3^N \left[ \omega^N_{x,y,z} \times I \left( T_{m-1} + \rho_{ZN-1,m} u^N_{Zm} \times I \right) \right]
\]

\[
+ F_{C_{e-1,m}} (u^N_{Zm} \times \Delta R^N_{1,m})
\]

\[
\Delta R^N_{1,m} = \int_{t_{m-1}}^{t_m} v^N \, dt
\]
The $\omega^N_{f B}$ term in Eq. (17) is evaluated with Eq. (2). As in Sec. III.A, $()_{n-1}$ in Eq. (17) must be approximated based on past value extrapolation; e.g.,

$$()_{n-1} \approx ()_{n-1} + \frac{1}{2}(r/j)[()_{n-1} - ()_{n-2}]$$  \hspace{1cm} (19)

Because Eq. (17) is used to update $\nu^N$ in Eqs. (4), (13), and (14), current values of $\nu^N$ are not available for evaluating $\Delta R^N_{f B}$ in Eq. (18). Hence, past value extrapolation must be employed, such as in Sec. III.A:

$$\Delta R^N_{f B} = \frac{T}{2}(3\nu^N_{m-1} - \nu^N_{m-2}) \quad \text{for} \quad r = 1$$

$$\Delta R^N_{f B} = \frac{T}{2} \left[ 3\nu^N_{m-1} - \nu^N_{m-2} + \sum_{i=m+1}^{m-1} \left( \nu^N_i + \nu^N_{i+1} \right) \right] \quad \text{for} \quad r > 1$$  \hspace{1cm} (20)

2. Body Frame Integrated Specific Force Increment

The $\Delta v^N_{f B}$ integral term in Eqs. (11) and (12) is calculated using a high-speed digital repetition algorithm similar to the type employed in Ref. 1, Eqs. (35) and (36), for attitude updating. The derivation of the algorithm is initially based on first-order approximations for $C^N_{f B}$. The first-order solution is divided into two parts for application of the two-speed algorithm approach: a portion that can be calculated at the $m$ cycle rate which measures the effect of constant $B$ frame angular rate and specific force, and a high-speed portion within the $m$ cycle, which measures dynamic variations in $B$ frame angular rate/specific force. The first-order $m$ cycle portion is then expanded to be analytically exact under constant angular rate/specific force.

Following the development approach in Ref. 1, Sec. IV.A.1, the $C^N_{f B}$ term in the Eq. (12) $\Delta v^N_{f B}$ integrand is expressed as

$$C^N_{f B} = 1 + \frac{\sin(\phi(t))}{\phi(t)}(\phi(t) \times) + \frac{1 - \cos(\phi(t))}{\phi(t)}(\phi(t) \times)^2$$  \hspace{1cm} (21)

where $\phi(t)$ is the rotation vector defining the general orientation of frame $B$ relative to frame $B_{1(m-1)}$ for time $t$ greater than $t_{m-1}$. Reference 1 Eqs. (32) and (33) show that $\phi(t)$ in Eq. (21) can be approximated by

$$\phi(t) \approx \alpha(t)$$  \hspace{1cm} (22)

$$\alpha(t) = \int_{t_{m-1}}^{t} \omega^B_{f B} d\tau$$  \hspace{1cm} (23)

where $r$ is an integration time parameter. A first-order approximation for Eq. (21) that is consistent with Eq. (22) neglects $(\phi(t) \times)^2$ and approximates $\sin(\phi(t)) / \phi(t)$ by unity [assuming that the $m$ cycle rate is selected fast enough to maintain $\phi(t)$ at a reasonably small value, e.g., less than 0.05 rad]. With Eq. (22), Eq. (21) reduces to

$$C^N_{f B} \approx 1 + (\alpha(t) \times)$$  \hspace{1cm} (24)

Substituting Eq. (24) into Eq. (12) then yields to first order

$$\Delta v^N_{f B} = \int_{t_{m-1}}^{t_{m}} \left[ 1 + (\alpha(t) \times) \right] a^N_{f B} dt$$

$$= \int_{t_{m-1}}^{t_{m}} a^N_{f B} dt + \int_{t_{m-1}}^{t_{m}} (\alpha(t) \times) a^N_{f B} dt$$

or, including Eq. (23),

$$\Delta v^N_{f B} = \nu_m + \int_{t_{m-1}}^{t_{m}} (\alpha(t) \times a^N_{f B}) dt$$  \hspace{1cm} (25)

$$\alpha(t) = \int_{t_{m-1}}^{t} \omega^B_{f B} d\tau, \quad \nu(t) = \int_{t_{m-1}}^{t} a^N_{f B} d\tau, \quad \nu_m = \nu(t_m)$$

Equations (26) define a method for calculating $\Delta v^N_{f B}$ in Eq. (11). It is instructive to analyze these equations under constant $B$ frame angular rate $\omega^B_{f B}$ and specific force $a^N_{f B}$, for which

$$\alpha(t) = (t - t_{m-1}) \omega^B_{f B}, \quad \nu(t) = (t - t_{m-1}) a^N_{f B}$$  \hspace{1cm} (27)

$$\omega^B_{f B}, a^N_{f B} = \text{const}$$

Substituting $\alpha(t)$ from Eq. (27) into the Eq. (26) expression yields for constant $B$ frame angular rate and specific force

$$\Delta v^N_{f B} = \nu_m + \int_{t_{m-1}}^{t_{m}} (t - t_{m-1}) \omega^B_{f B} a^N_{f B} dt$$

$$= \nu_m + (\omega^B_{f B}, a^N_{f B}) \int_{t_{m-1}}^{t_{m}} (t - t_{m-1}) dt$$

$$= \nu_m + (\omega^B_{f B}, a^N_{f B}) \frac{1}{2} (t_m - t_{m-1})^2$$

$$= \nu_m + \frac{1}{2} (\omega^B_{f B} (t_m - t_{m-1}) \times (a^N_{f B} (t_m - t_{m-1})$$  \hspace{1cm} (28)

or, with Eqs. (26) and (27) for constant $B$ frame angular rate and specific force,

$$\Delta v^N_{f B} = \nu_m + \frac{1}{2} \nu_m \times \nu_m$$

$$\alpha(t) = \int_{t_{m-1}}^{t} \omega^B_{f B} d\tau, \quad \alpha_m = \alpha(t_m)$$

$$\nu(t) = \int_{t_{m-1}}^{t} a^N_{f B} d\tau, \quad \nu_m = \nu(t_m)$$

Comparing Eq. (26) for the general case with Eq. (29) for the constant angular rate/specific force condition, we see that the difference is the replacement of the integral term with $\frac{1}{2} \nu_m \times \nu_m$.

For situations where constant angular rate/specific force is a reasonable approximation over the $t_{m-1}$ to $t_m$ time interval, Eq. (29) is preferred over Eq. (26) because the integral term (and its attendant high-speed algorithm) is replaced by $\frac{1}{2} \nu_m \times \nu_m$, which is evaluated once each $m$ cycle.

A fundamental limitation in Eq. (26) or Eq. (29) is the first-order approximation that underlies their development, i.e., Eq. (24) for $C^N_{f B}$ that was used in the Eq. (12) $\Delta v^N_{f B}$ expression. It would be desirable if the Eq. (24) approximation could be applied only to the high-frequency content of $C^N_{f B}$ with the low-frequency content retaining the full Eq. (21) form. Such an algorithm can be synthesized by first noting that

$$\frac{d}{dt} (\alpha(t) \times \nu(t)) = (\alpha(t) \times \nu(t)) + (\alpha(t) \times \nu(t))$$

$$= \frac{d}{dt} (\alpha(t) \times \nu(t)) + \nu(t) \times \alpha(t)$$

$$= \frac{d}{dt} (\alpha(t) \times \nu(t)) + \frac{1}{2} (\alpha(t) \times \nu(t))$$

with $\alpha(t)$ and $\nu(t)$ as defined in Eq. (26). Upon rearrangement, Eq. (30) becomes

$$\alpha(t) \times \nu(t) = \frac{d}{dt} (\alpha(t) \times \nu(t)) + \nu(t) \times \alpha(t)$$  \hspace{1cm} (31)

Trivially,

$$\alpha(t) \times \nu(t) = \frac{1}{2} \alpha(t) \times \nu(t) + \frac{1}{2} \alpha(t) \times \nu(t)$$  \hspace{1cm} (32)

We now substitute Eq. (31) for one of the terms on the right in Eq. (32) to obtain

$$\alpha(t) \times \nu(t) = \frac{1}{2} \frac{d}{dt} (\alpha(t) \times \nu(t)) + \frac{1}{2} (\alpha(t) \times \nu(t) + \nu(t) \times \alpha(t))$$  \hspace{1cm} (33)

From Eq. (26) we know that

$$\alpha(t) = \omega^B_{f B}, \quad \nu(t) = a^N_{f B}$$  \hspace{1cm} (34)
whereby Eq. (33) assumes the form

\[
\alpha(t) \times \Delta a_{SF} = \frac{d}{dt} \left( \alpha(t) \times v(t) \right) + \frac{1}{2} \left( \alpha(t) \times a_{SF} + v(t) \times \omega_{IB} \right) \tag{35}
\]

Equation (35) is an alternate for the integrand in the Eq. (26) expression. Substitution of Eq. (35) for the integrand then yields the following equivalent form:

\[
\Delta V_{SFm}^{\text{(an-1)}} = v_m + \frac{1}{2} \left( \alpha_m \times v_m \right) + \frac{1}{2} \int_{t_{m-1}}^{t_m} \left( \alpha(t) \times a_{SF} + v(t) \times \omega_{IB} \right) dt \tag{36}
\]

If we now compare \(\Delta V_{SFm}^{\text{(an-1)}}\) in Eqs. (36) and (29) under constant angular rate/specific force conditions, we see that they are equivalent except for the integral term in Eq. (36). It is easily verified by substitution of Eq. (27) that the integral term in Eq. (36) vanishes for constant \(B\) frame angular rate/specific force. We conclude that the integral term in Eq. (36) represents the integrated contribution of the high-frequency content in the Eq. (12) \(\Delta V_{SFm}^{\text{(an-1)}}\) integrand; the remaining terms, i.e., \(v_m + \frac{1}{2} \left( \alpha_m \times v_m \right)\), represent the low-frequency content.

The integral term in Eq. (36), denoted as sculling, measures the rectification of combined dynamic angular rate/specific force into a net constant contribution to \(\Delta V_{SFm}^{\text{(an-1)}}\). The rectification is a maximum under classical sculling motion defined as sinusoidal angular rate/specific force in which the angular rate about one \(B\) frame axis is at the same frequency and in phase with the specific force along another \(B\) frame axis (with rectified constant specific force then produced along the average third axis direction). This is the same principle used by mariners to propel a boat in the forward direction using a single oar operated with an undulating motion (also denoted as sculling, the original use of the term). Note that the \(\Delta V_{SFm}^{\text{(an-1)}}\) integral term in Eq. (26) has also been denoted as sculling even though it contains large contributions under constant angular rate/specific force, i.e., nonsculling conditions. The \(\frac{1}{2} \left( \alpha_m \times v_m \right)\) term in Eq. (36) is identified here as velocity rotation compensation. The velocity rotation correction has been adopted to denote that this rotation compensation term feeds the velocity rate equation (in contrast with a position rotation compensation term to be discussed in Sec. IV that feeds the position rate equation). With these definitions, a comparison between Eqs. (26) and (36) identifies the integral term in Eq. (26) as representing the composite of sculling and velocity rotation compensation effects. Using the latter terminology, Eq. (36) is rewritten as

\[
\Delta V_{SFm}^{\text{(an-1)}} = v_m + \Delta V_{velc} + \Delta V_{scull} \tag{37}
\]

\[
\Delta V_{scull}(t) = \frac{1}{2} \int_{t_{m-1}}^{t_m} \left( \alpha(t) \times a_{SF} + v(t) \times \omega_{IB} \right) dt
\]

\[
\Delta V_{velc} = \Delta V_{scull}(t_m) \tag{38}
\]

\[
\alpha(t) = \int_{t_{m-1}}^{t_m} \omega_{IB} dt, \quad \alpha_m = \alpha(t_m)
\]

\[
v(t) = \int_{t_{m-1}}^{t_m} a_{SF} dt, \quad v_m = v(t_m)
\]

\[
\Delta V_{velc} = \frac{1}{2} \left( \alpha_m \times v_m \right) \tag{39}
\]

where \(\Delta V_{velc}\) is the velocity rotation compensation term and \(\Delta V_{scull}\) is the sculling term. Alternatively, beginning from the Eq. (26) version,

\[
\Delta V_{SFm}^{\text{(an-1)}} = v_m + \Delta V_{rot/scull} \tag{40}
\]

with \(\alpha(t)\) and \(v_m\) from sculling Eq. (38) and where \(\Delta V_{rot/scull}\) is the composite sculling and velocity rotation compensation term.

Equations (37–39) are completely equivalent to Eqs. (40) and (41); both equation sets exhibit only first-order accuracy. However, Eq. (37) is now in a form that enables us to substitute an expanded expression for the Eq. (39) velocity rotation compensation term that makes Eq. (37) exact under constant rate/specific force conditions. This is an important extension because general motion is typically dominated by low-frequency angular rate and specific force components that may have large amplitudes under extreme maneuvers (where second-order algorithm errors may not be negligible). The extension to exactness is not possible for Eqs. (40) and (41) because the rotation compensation effect is imbedded within the integral, which includes the first-order sculling term. The following subsections derive an exact \(\Delta V_{rots}\) velocity rotation compensation algorithm for Eq. (37) in addition to digital integration algorithms for the Eq. (38) integral terms. Using the same procedure, a digital integration algorithm can also be developed for \(\Delta V_{rot/scull}\) in Eqs. (40) and (41), as shown in Ref. 8, Sec. 7.2.2.2.2.

**Exact velocity rotation compensation.** The exact velocity rotation compensation algorithm is defined as the algorithm that, when substituted for \(\Delta V_{rots}\) in Eq. (37), provides an exact solution for \(\Delta V_{SFm}^{\text{B(an-1)}}\) in Eq. (12) under constant \(B\) frame angular rate/specific force conditions. The exact velocity rotation compensation algorithm is derived from Eq. (12) using Eq. (21) for \(C_{SF}^{\text{SF}}\) under constant angular rate/specific force. We first consider the more general condition where only the direction of the angular rate vector is constant, i.e., a nonconing environment in which the angular rate vector is not rotating. From Eq. (23), for a nonconing angular rate condition,

\[
\alpha(t) = \alpha(t) u_m, \quad \alpha(t) = \int_{t_{m-1}}^{t_m} \omega dt, \quad \frac{\alpha(t)}{\alpha_m} = u_m \tag{42}
\]

where \(\omega\) is the magnitude of \(\omega_{IB}\) and \(u_m\) is a unit vector along \(\omega_{IB}\), which is considered constant in the \(B\) frame.

As discussed in Ref. 1, Sec. IV.A.1, for the case where \(\omega_{IB}\) is not rotating, \(\phi(t)\) is equal to \(\alpha(t)\) (the integral of \(\omega_{IB}\)). Under this restriction, Eq. (21) with Eq. (42) for \(\phi(t)\) substituted in Eq. (12) gives for the nonconing angular rate condition

\[
\Delta V_{SFm}^{\text{B(an-1)}} = \int_{t_{m-1}}^{t_m} \left[ 1 + \sin \alpha(t) \right] a_{SF} dt + \left( 1 - \cos \alpha(t) \right) u_m \times \int_{t_{m-1}}^{t_m} \sin \alpha(t) dt \tag{43}
\]

For nonconing angular rate and constant \(B\) frame specific force, Eq. (43) can be expanded to

\[
\Delta V_{SFm}^{\text{B(an-1)}} = \int_{t_{m-1}}^{t_m} a_{SF} dt + \left( u_m \times a_{SF} \right) \int_{t_{m-1}}^{t_m} \sin \alpha(t) dt
\]

\[
+ \left[ u_m \times \left( u_m \times a_{SF} \right) \right] \int_{t_{m-1}}^{t_m} \left( 1 - \cos \alpha(t) \right) dt \tag{44}
\]

Section III.B.2 nomenclature is now applied with the nonconing rate/constant specific force assumption and appropriate Eq. (42) relationships,

\[
v_m = \int_{t_{m-1}}^{t_m} a_{SF} dt = a_{SF} (t_m - t_{m-1}) = a_{SF} T_m, \quad a_{SF} = \frac{v_m}{T_m}
\]

\[
\frac{\alpha(t)}{\alpha_m} = u_m, \quad u_m = \frac{\alpha_m}{\alpha_m}
\]

\[
\Delta V_{rot/scull} = \int_{t_{m-1}}^{t_m} \left( \alpha(t) \times a_{SF} \right) dt
\]
and \( \alpha_m = \alpha(t_m) \) is the magnitude of \( \alpha(t_m) \). Substituting Eqs. (45) into Eq. (44) then yields for nonconcing angular rate and constant specific force

\[
\Delta v_{\text{SF}} = v_m + \frac{\alpha_m \times \alpha_m \times v_m}{\alpha_m} \int_{t_m}^{t} \sin \alpha(t) \, dt + \frac{\alpha_m \times (\alpha_m \times v_m)}{\alpha_m} \int_{t_m}^{t} (1 - \cos \alpha(t)) \, dt
\]

(46)

To evaluate the integral terms in Eq. (46), we now adopt the constant angular rate condition whereby \( \omega \) in Eq. (42) is constant. Then,

\[
\alpha(t) = \omega(t - t_m), \quad \omega = \text{const}
\]

(47)

Applying Eq. (47) in Eq. (46) with Eq. (45) for \( \alpha_m \) allows the integral terms to be evaluated for constant \( B \) frame angular rate as

\[
\Delta v_{\text{SF}} = v_m + \left( 1 - \cos \alpha_m \right) \alpha_m \times v_m
\]

(48)

Substitution in Eq. (46) then yields the desired form for the exact \( \Delta v_{\text{SF}} \) solution under constant \( B \) frame angular rate and specific force

\[
\Delta v_{\text{SF}} = v_m + \left( 1 - \cos \alpha_m \right) \alpha_m \times v_m + \frac{\alpha_m \times (\alpha_m \times v_m)}{\alpha_m}
\]

(49)

Equation (49) constitutes an exact solution for \( \Delta v_{\text{SF}} \) under constant angular rate/specific force. We are now in a position to compare Eq. (49) with Eq. (37) under the same conditions to identify the exact velocity rotation compensation term. Under constant rate/specific force conditions, the sculling term in Eq. (37) vanishes (see discussion in Sec. II.B.2), and \( \Delta v_{\text{SF}} \) is given by

\[
\Delta v_{\text{SF}} = v_m + \Delta v_{\text{rot}}
\]

(50)

If we compare Eqs. (49) and (50) it should be clear from its definition that the exact velocity rotation compensation term \( \Delta v_{\text{rot}} \) is

\[
\Delta v_{\text{rot}} = \left( 1 - \cos \alpha_m \right) \alpha_m \times v_m + \frac{\alpha_m \times (\alpha_m \times v_m)}{\alpha_m}
\]

(51)

The trigonometric coefficients in Eq. (51) can be calculated from the Taylor series formulates

\[
\frac{1 - \cos \alpha_m}{\alpha_m} = \frac{1}{2!} \frac{\alpha_m^2}{4!} + \frac{\alpha_m^4}{6!} + \cdots
\]

(52)

\[
\frac{1}{\alpha_m} \left( 1 - \sin \alpha_m \right) = \frac{1}{3!} \frac{\alpha_m^2}{5!} + \frac{\alpha_m^4}{7!} + \cdots
\]

Equation (51) with Eqs. (52) constitute an alternative algorithm for the \( \Delta v_{\text{rot}} \) velocity rotation compensation term in Eq. (39) that will generate an exact solution for \( \Delta v_{\text{SF}} \) in Eq. (37) under constant \( B \) frame angular rate/specific force conditions. In contrast, the \( \Delta v_{\text{rot}} \) algorithm in Eq. (39) is accurate to only first order. Note that, to first order in \( \alpha_m \), Eq. (51) with Eq. (52) reduces to the Eq. (39) \( \Delta v_{\text{rot}} \) form (as it should).

**Integrated specific force and sculling increments.** In this subsection we develop algorithms for calculating \( v_m \) and \( \Delta v_{\text{SF}} \), integral terms in Eq. (37) and (38) [the \( \alpha_m \) term for these equations is provided from the attitude algorithm in Ref. 1, Eqs. (41)]. A similar procedure can be used to develop an algorithm for \( \Delta v_{\text{rot/sculling}} \) in Eqs. (40) and (41). Following the identical procedure used in Ref. 1, Sec. IV.A.1, for the coning algorithm, we develop the \( \Delta v_{\text{sculling}} \), sculling algorithm by considering \( \Delta v_{\text{sculling}} \) to be the value at \( t = t_m \) of the general function \( \Delta v_{\text{sculling}} \) (as in Eq. (38)). Let us consider the Eq. (38) \( \Delta v_{\text{sculling}} \) integration as being divided into portions up to and after a general time \( t_{m-1} \) within the \( t_{m-1} \) to \( t_m \) interval so that

\[
\Delta v_{\text{sculling}} = \Delta v_{\text{sculling}} + \delta v_{\text{sculling}}
\]

(53)

\[
\delta v_{\text{sculling}}(t) = \frac{1}{2} \int_{t_{m-1}}^{t} \left( \alpha(t) \times a_{SF}^B + v(t) \times \omega_{\text{r}}^B \right) \, dt
\]

We now define the next \( l \) cycle time point \( t_l \) within the \( t_{m-1} \) to \( t_m \) interval so that Eqs. (53) at \( t_l \) with \( \alpha(t) \) and \( v(t) \) from Eq. (38), including initial conditions, become

\[
\alpha(t) = \alpha_{l-1} + \Delta \alpha(t)
\]

\[
\Delta \alpha(t) = \int_{t_{m-1}}^{t} \omega_{\text{r}}^B \, dt, \quad \Delta \alpha_{l} = \Delta \alpha(t_l)
\]

(54)

\[
\alpha_l = \alpha_{l-1} + \Delta \alpha_l, \quad \alpha_m = \alpha(t_l) = \alpha_{l-1} + \Delta \alpha_{l-1}
\]

(55)

\[
\alpha_l = \alpha_{l-1} + \Delta \alpha_{l-1}, \quad \nu_l = \nu_{l-1} + \Delta \nu_l
\]

(56)

\[
\Delta v_{\text{sculling}} = \Delta v_{\text{sculling}} + \delta v_{\text{sculling}}(t_l)
\]

\[
\delta v_{\text{sculling}} = \delta v_{\text{sculling}}(t_l)
\]

where \( l \) is the high-speed computer cycle index. Equations (54)–(56) constitute the proof of a digital recursive algorithm at the \( l \) computer cycle rate for calculating the \( \Delta v_{\text{sculling}} \), sculling term and \( v_m \) as a summation of changes in \( \Delta v_{\text{sculling}} \) and \( v_m \) over the \( t_{m-1} \) to \( t_m \) interval. It remains to determine a digital equivalent for the \( \delta v_{\text{sculling}} \) integral term in Eq. (56). We begin by substitution of \( \alpha(t) \) and the definitions for \( \Delta \alpha_l \) and \( \Delta \nu_l \) from Eq. (54) into \( \delta v_{\text{sculling}} \):

\[
\delta v_{\text{sculling}} = \frac{1}{2} \left( \nu_{l-1} \times \nu_l + \nu_{l-1} \times \nu_l \right) + \frac{1}{2} \int_{t_{m-1}}^{t_l} \left( \Delta \alpha(t) \times a_{SF}^B + \Delta \nu(t) \times \omega_{\text{r}}^B \right) \, dt
\]

Development of a digital algorithm for the integral term in sculling Eq. (57) is based on an assumed form for the \( B \) frame angular rate/specific force history during the \( t_{l-1} \) to \( t_l \) time interval. Unlike the coning algorithm, very little published work exists for selecting angular rate/specific force time histories for application to sculling algorithm design. In principle, the approaches used for the coning algorithm can also be applied for sculling, including optimization for sculling-type motion (see discussion in Ref. 1, Sec. IV.A.1). For this paper, we provide an example based on general linearly changing angular rate/specific force over the \( t_{l-1} \) to \( t_l \) time interval:

\[
\omega_{\text{r}}^B \approx A + B(t - t_{l-1}), \quad a_{SF}^B \approx C + D(t - t_{l-1})
\]

(58)

where \( A, B, C, \) and \( D \) are constant vectors.

An algorithm for the integral term in Eq. (57) can be developed by first substituting Eq. (58) for \( a_{SF}^B \) and \( \omega_{\text{r}}^B \) in Eq. (57) and then calculating the Eq. (57) integral term analytically over the \( t_{l-1} \) to \( t_l \) time interval. The intermediate result is an equation for the Eq. (57)
where the results of the latter procedure (detailed in Ref. 8, Sec. 7.2.2.2.2) show that for the Eq. (58) linearly ramping model, the algorithm equivalent to Eqs. (54-57) is given by

\[ \Delta \alpha_\ell = \text{integrated angular rate sensor outputs from Ref. 1, Eqs. (46)} \]

\[ \Delta v_l = \int_{t_{l-1}}^{t_l} dv \]

\[ \delta v_{\text{scul}} = \frac{1}{2} \left[ (\alpha_{t-1} + C_{\text{scul}}) - \Delta \alpha_\ell \right] \times \Delta v_l \]

\[ \Delta v_{\text{scul}} = \delta v_{\text{scul}} \]

\[ \Delta v_{\text{scul}} = 0 \text{ at } t = t_{m-1} \]

where

\[ \Delta v_l = \text{summation of integrated specific force output increments from accelerometers} \]

\[ dv = \text{analytical representation of pulse output from strapdown accelerometers, } a_P^B dt \]

Equation (61) for \( \Delta v_{\text{scul}} \) has been classified as a second-order algorithm because it includes current and past \( l \) cycle \( \Delta \alpha_\ell, \Delta v_\ell \) products. The \( l, l-1 \) cycle \( \Delta \alpha_\ell, \Delta v_\ell \) product terms in \( \delta v_{\text{scul}}, i.e., the \{ 1 \}, \) terms, stem from the approximation of linearly ramping angular rate and specific force in the \( t_{l-2} \) to \( t_l \) time interval. If the angular rate and specific force terms were approximated as parabolically varying functions of time, a third-order algorithm would result, containing \( l, l-1, l-2 \) cycle \( \Delta \alpha_\ell, \Delta v_\ell \) products. If the angular rate and specific force were approximated as constants over \( t_{l-1} \) to \( t_l \), the \{ 1 \} terms in Eq. (61) would vanish, resulting in a first-order algorithm for \( \Delta v_{\text{scul}} \). Finally, if angular rate and specific force are slowly varying, we can approximate \( \Delta v_{\text{scul}} \) as being equal to zero. Alternatively (and more accurately), we can set the \( l \) cycle rate equal to the \( m \) cycle rate, which equates \( \Delta v_{\text{scul}} \) to \( \delta v_{\text{scul}} \) in Eq. (61) calculated once at time \( t_m \) [and noting from the initial condition definitions in Eq. (60) and Ref. 1, Eqs. (46), that \( \alpha_{t-1} \) and \( \nu_{t-1} \) would be zero]. Note that setting the \( l \) and \( m \) rates equal can also be achieved by increasing the \( m \) rate to match the \( l \) rate. The result would be a single high-speed, higher-order algorithm with a simpler software architecture than the two-speed approach but requiring more throughput. Continuing advances in the speed of modern-day computers may make this the preferred approach for the future.

IV. Position Update Algorithms

In this section we develop digital integration algorithms for calculating position relative to the Earth in the form of altitude \( h \) above the Earth’s surface and the \( C_E^N \) direction cosine matrix defining the angular orientation between the local level \( N \) frame and the Earth-fixed \( E \) frame (from which latitude/longitude can be extracted). Two algorithm forms are developed: a typical form based on trapezoidal integration of velocity and a high-resolution form that accounts for dynamic attitude and velocity changes within the position update period. The high-resolution algorithm is modeled after the Sec. III (two-speed velocity update) approach.

Both the typical and high-resolution forms can be represented by the continuous differential equation form of Ref. 1, Eqs. (21) and (22), repeated here as

\[ \dot{h} = u_N^N \cdot v_N^N \]

\[ C_E^N = C_N^E \left( \omega_{EN}^N \times \right) \]

where \( h \) is altitude above the Earth’s surface. The typical and high-resolution forms derive from a general updating formulation for \( h \) and \( C_E^N \). The following sections formulate the general position updating process and then derive computational approaches for typical and high-resolution position updating.

A. Position Updating in General

The general altitude \( h \) updating algorithm is formulated as the integral of Eq. (62) over a position update cycle \( \tau \):

\[ h_\tau = h_{\tau-1} + \Delta h_\tau \]

Allowing for the higher-speed digital computation loop, i.e., the \( m \) loop for altitude and velocity integration, Eq. (65) can be written as

\[ \Delta h_\tau = u_{ZN}^N \cdot \sum_{m=1}^{\tau} hN_m \]

\[ \Delta R_m^N = \int_{t_{m-1}}^{t_m} v_N^N dt \]

If vertical channel gravity/divergence stabilization is to be incorporated, an additional operation would be included in Eq. (64) representing the altitude control function (see Ref. 8, Sec. 4.4.1, and Ref. 10, pp. 102-103).

The general updating algorithm for the \( C_E^N \) direction cosine matrix is designed to achieve the same numerical result at the update times as would the formal continuous integration of the Eq. (63) \( C_E^N \) expression at the same time instant. The algorithm is developed by envisioning the local level navigation \( N \) frame orientation history in the digital updating world (produced in Eq. (63) by \( \omega_{EN}^N \)) as being constructed of successive discrete orientations relative to the Earth \( (E \) frame) at each update time instant. The general updating algorithm for \( C_E^N \) is then constructed as follows using the Ref. 1, Eq. (3), direction cosine matrix product chain rule:

\[ C_{E_N(E)} = C_{E_{N(a-1)}} \cdot C_{E_{N(a)}} \]

where

\[ N_{E(a)} = \text{discrete orientation of the } N \text{ frame in rotating Earth frame space } (E) \text{ at computer update time } t_{a} \]

\[ C_{E_{N(a-1)}} = C_N^E \text{ relating the } N \text{ frame at time } t_{a-1} \text{ to the } E \text{ frame} \]

\[ C_{E_{N(a)}} = C_N^E \text{ relating the } N \text{ frame at time } t_{a} \text{ to the } E \text{ frame} \]

\[ C_{N_{E(a-1)}} = \text{direction cosine matrix that accounts for } N \text{ frame rotation relative to the Earth } (E) \text{ from its orientation at time } t_{a-1} \text{ to its orientation at time } t_{a} \]

The \( C_{E_{N(a)}} \) matrix in Eq. (68) is defined formally as

\[ C_{E_{N(a-1)}} = I + \int_{t_{a-1}}^{t_a} C_{E_{N(a-1)}(t)} dt \]
Following the same development procedure as for \( C_{\xi E}^{B(m-1)} \) in Ref. 1, Sec. IV.A.1, the \( C_{N E}^{(n-1)} \) matrix can also be expressed in terms of the rotation vector defining the frame \( N_{E(n-1)} \) attitude relative to frame \( N_{E(n-1)} \). Applying Ref. 1, Eq. (4), with Taylor series expansion for the coefficient terms obtains

\[
C_{N E}^{(n-1)} = I + \frac{\sin \xi_n}{\xi_n} (\xi_n \times) + \frac{(1 - \cos \xi_n)}{\xi_n^2} (\xi_n \times) \times (\xi_n \times) \times \ldots
\]

(70)

where \( \xi_n \) is the rotation vector defining the frame \( N_{E(n-1)} \) attitude at time \( t_n \), relative to the frame \( N_{E(n-1)} \) attitude at time \( t_{n-1} \).

The angular rate of the \( N \) frame relative to the Earth \( \omega_{EN} \) is small and typically no larger than one or two Earth rates. As such, because the \( t_{n-1} \) to \( t_n \) update cycle is relatively short, \( \xi_n \) will be very small in magnitude. Because \( \omega_{EN} \) is small and slowly changing over a typical \( t_{n-1} \) to \( t_n \) update cycle (due to small changes in velocity and position over this time period), the \( N \) frame rate vector \( \omega_{EN} \) can be approximated as nonrotating. The result is that \( \xi_n \), for Eq. (70) can be calculated as the integral of the simplified form of the Ref. 1, Eq. (10), rotation vector rate expression whereby the cross-product terms are neglected:

\[
\xi_n \approx \int_{t_{n-1}}^{t_n} \omega_{EN} \, dt
\]

(71)

A discrete digital algorithm for the Eq. (71) \( \xi_n \) integral can be constructed by first approximating Eq. (3) for \( \omega_{EN} \) as

\[
\omega_{EN} \approx \rho_{ZN} \omega_{ZN} + \frac{1}{T_m} \left[ F_{C_{-}} + \left( u_{ZN}^N \times v^N \right) \right]
\]

(72)

where \( (h_{n-1/2}) \) is the value for \( (h) \) midway between times \( t_{n-1} \) and \( t_n \). Using Eq. (72) in Eq. (71) and applying the Eq. (67) definition then obtains

\[
\xi_n \approx \rho_{ZN} \omega_{ZN} + u_{ZN}^N T_n + \frac{1}{T_m} \left( u_{ZN}^N \times \Delta \Omega_{RN}^N \right)
\]

(73)

where \( T_n \) is the computer cycle update period \( t_n - t_{n-1} \). The \( (h_{n-1/2}) \) terms in Eq. (73) are all functions of position, which has not yet been updated. Hence, to calculate the \( (h_{n-1/2}) \) terms, an approximate extrapolation formula must be used based on previously computed values for the \( (h) \) parameters. For example, a linear extrapolation formula using the last two computed values for \( (h) \) would be

\[
(h_{n-1} + h_{n-2}) = \frac{1}{2} (h_{n-1} + h_{n-2}) = \frac{1}{2} (h_{n-1} - h_{n-2})
\]

(74)

The method for calculating the \( \Delta \Omega_{RN}^N \) term for Eqs. (66) and (73) from the Eq. (67) integral depends on whether typical trapezoidal integration is used for position updating or whether a more precise high-resolution integration approach is to be applied. Both are described in the following sections.

B. Typical Position Updating

Applying typical trapezoidal integration for the \( h \) and \( C_{\xi} \) updating process would utilize Eqs. (64), (66), (68), (70), (73), and (74) with a trapezoidal integration algorithm in Eq. (67) for \( \Delta \Omega_{RN}^N \).

\[
\Delta \Omega_{RN}^N \approx \int_{t_{n-1}}^{t_n} \left[ (v^N + v_{m-1}^N) \right] T_m
\]

(75)

C. High-Resolution Position Updating

The high-resolution approach for implementing the \( h \) and \( C_{\xi} \) updating process utilizes Eqs. (64), (66), (68), (70), (73), and (74) with a high-speed digital integration algorithm in Eq. (67) for \( \Delta \Omega_{RN}^N \). The digital algorithm for \( \Delta \Omega_{RN}^N \) is developed by first expanding the Eq. (67) \( v^N \) integrand. Using the expression for \( v^N \) in Eq. (4) with

Eq. (6), \( v^N \) can be defined as a continuous time function at a general point in time since the last \( t_{n-1} \) update:

\[
v^N(t) = v_{m-1}^N + C_{L}^N \Delta \Omega_{SF}^N(t) + \Delta v_{SF}^N(t - t_{m-1}) T_m
\]

(76)

\[
\Delta v_{SF}^N(t) = \int_{t_{m-1}}^{t} C_L^N a_{BF}^N dt
\]

Equations (76) are based on the assumption that gravity/Coriolis term \( \Delta v_{SF}^N(t) \) can be approximated as the integral of a constant over \( t_{m-1} \) to \( t \). With Eq. (76), \( \Delta \Omega_{SF}^N \) from Eq. (67) is given by

\[
\Delta \Omega_{SF}^N = \left( \frac{V_N^N + \frac{1}{2} \Delta v_{SF}^N(t_{m-1}) \Delta t_{m-1}}{T_m} \right)
\]

(77)

\[
\Delta \Omega_{SF}^N = \int_{t_{m-1}}^{t} \Delta \Omega_{SF}^N dt
\]

where \( \Delta \Omega_{SF}^N \) is the \( L \) frame coordinate portion of \( \Delta \Omega_{RN}^N \) produced by specific force.

Equations (11), (13), and (36) show that \( \Delta v_{SF}^N(t) \) in Eq. (77) can be approximated to first order (in body rotation angle) by

\[
\Delta v_{SF}^N = \left( C_L^N \Delta \Omega_{SF}^N(t) + \frac{1}{2} \frac{C_L^N}{\Lambda} \Delta \Omega_{SF}^N(t) \right)
\]

(78)

In Eq. (78), the \( \Lambda \) notation in subscripts and superscripts, used for clarity in Eqs. (11) and (13), has been dropped for simplicity. In addition, \( C_L^N(t_{m-1}), -1 \) and \( \Delta \Omega_{SF}^N(t) \) in the first part of the \( \Delta v_{SF}^N \) expression have been approximated to be linearly ramping in time over \( t_{m-1} \) to \( t \). Note also, as in Sec. III.B.1, that \( C_L^N(t_{m-1}) \) terms in Eq. (78) can be approximated by the identity matrix for all but very high-precision applications. Based on Eq. (78) and including
where Eq. (79) has the following equivalent forms:

\[
\Delta R_{\text{SF}m}^{\theta} = \frac{1}{3} \left[ (\zeta_{n-1,m} - \zeta_{n-1,m-1}) \times \Delta R_{\text{SF}m}^{L(n-1)} \right] T_m
\]

\[
+ C_{L(n-1)}^{L(n-1)} \Delta R_{\text{SF}m}^{L(n-1)}
\]

\[
\Delta R_{\text{SF}m}^{\psi} = \int_{t_{n-1}}^{t_n} \Delta R_{\text{SF}m}^{\psi(n-1)}(t) \, dt
\]

\[
= \int_{t_{n-1}}^{t_n} \left[ v(t) + \frac{1}{2} (\alpha(t) \times v(t)) + \Delta v_{\text{scrl}}(t) \right] \, dt
\]

with \( \Delta v_{\text{scrl}}(t) \), \( \alpha(t) \), and \( v(t) \) from sculling Eq. (38).

Following a similar development path as used in Sec. III.B.2 for the body frame integrated specific force increment, the \( \int \frac{1}{2} (\alpha(t) \times v(t)) \, dt \) term in the Eq. (79) \( \Delta R_{\text{SF}m}^{\theta} \) expression can be revised into a nonintegral term plus an integral term that vanishes under constant angular rate/specific force, both being of first-order accuracy. The nonintegral term will then be extended into a more accurate form that is exact under constant angular rate/specific force conditions. We begin by using classical integration by parts substitution as in Sec. III.B.2 leading to Eq. (35) to show that the \( \int \frac{1}{2} (\alpha(t) \times v(t)) \, dt \) term in Eq. (79) has the following equivalent forms:

\[
R_0 = \int_{t_{n-1}}^{t_n} \frac{1}{2} (\alpha(t) \times v(t)) \, dt
\]

\[
r_1 = \int_{t_{n-1}}^{t_n} \frac{1}{2} (\alpha(t) \times v(t)) \, dt
\]

\[
r_2 = \int_{t_{n-1}}^{t_n} \frac{1}{2} (\alpha(t) \times v(t)) \, dt
\]

\[
= \frac{1}{2} \left( S_m \times v_m \right) - \frac{1}{2} \left( S_m(t) \times a_{\text{SF}}^B \right) dt
\]

\[
= \frac{1}{2} \left( S_m \times v_m \right) + \frac{1}{2} \int_{t_{n-1}}^{t_n} \left( S_m(t) \times \omega_{FB}^m \right) dt
\]

\[
S_m(t) = \int_{t_{n-1}}^{t} \alpha(t) \, dt, \quad S_m(t) = \int_{t_{n-1}}^{t} v(t) \, dt
\]

\[
\alpha_m = \alpha(t_m), \quad v_m = v(t_m)
\]

\[
S_m = S_m(t_m) \quad S_m = S_m(t_m)
\]

where \( S_m \) and \( S_m \) are time integrals of \( \alpha \) and \( v \).

Because \( r_0 \) and \( r_1 \) are analytically equivalent to the original integral form \( r_0 \), we can write

\[
\int_{t_{n-1}}^{t_n} \frac{1}{2} (\alpha(t) \times v(t)) \, dt = \frac{1}{3} (r_0 + r_1 + r_2)
\]

Substituting for \( r_0 \), \( r_1 \), and \( r_2 \) from Eq. (80) into Eq. (81) and combining terms then yields

\[
\int_{t_{n-1}}^{t_n} \frac{1}{2} (\alpha(t) \times v(t)) \, dt = \frac{1}{6} \left( S_m \times v_m \times \alpha_m \times S_m \right)
\]

\[
- \frac{1}{6} \int_{t_{n-1}}^{t_n} \left[ S_m(t) \times a_{\text{SF}}^B - S_m(t) \times \omega_{FB}^m - \alpha(t) \times v(t) \right] dt
\]

We now substitute Eq. (82) with the Eq. (80) definitions into Eqs. (77) and (79) to obtain the desired form for calculating \( \Delta R_{\text{SF}m}^{\psi} \):

\[
\Delta R_{\text{SF}m}^{\psi} = \int_{t_{n-1}}^{t_n} \Delta R_{\text{SF}m}^{\psi(n-1)}(t) \, dt
\]

\[
\Delta R_{\text{SF}m}^{\psi} = \frac{1}{6} \left[ (\zeta_{n-1,m} - \zeta_{n-1,m-1}) \times \Delta R_{\text{SF}m}^{L(n-1)} \right] T_m
\]

\[
+ C_{L(n-1)}^{L(n-1)} \Delta R_{\text{SF}m}^{L(n-1)}
\]

\[
\Delta R_{\text{SF}m}^{\theta} = S_m \times \Delta R_{\text{scrlm}}^{\theta} + \Delta R_{\text{cxmlm}}^{\theta}
\]

\[
\Delta R_{\text{SF}m}^{\psi} = S_m \times \Delta R_{\text{scrlm}}^{\psi} + \Delta R_{\text{cxmlm}}^{\psi}
\]

\[
\Delta R_{\text{cxmlm}}^{\theta} = \frac{1}{6} \int_{t_{n-1}}^{t_n} \left[ \Delta v_{\text{scrl}}(t) - S_m(t) \times a_{\text{SF}}^B 
\right.
\]

\[
+ S_m(t) \times \omega_{FB}^m + \alpha(t) \times v(t) \] dt

(85)

with \( \Delta v_{\text{scrl}}(t) \), \( \alpha(t) \), and \( v(t) \) from sculling equation (38) and

\[
S_m(t) = \int_{t_{n-1}}^{t} \alpha(t) \, dt, \quad S_m(t) = S_m(t_m)
\]

(86)

where \( \Delta R_{\text{scrlm}} \) is position rotation compensation analogous to the velocity rotation compensation term in Eqs. (37) and (39), and \( \Delta R_{\text{cxmlm}}^{\theta} \) is the scrolling term analogous to the scrolling term in Eqs. (37) and (38). The term scrolling was coined by the writer merely to have a name for the term and also to have one that sounds like sculling but for position integration (change in the position vector \( R \) stressing the \( R \) sound). The complex mathematical derivations and associated algorithms that accompany scrolling may be a more appropriate reason for the name.

A key characteristic of Eq. (84) is that the \( \Delta R_{\text{scrlm}} \) scrolling term from Eq. (85) is identically zero under constant body axis angular rate and specific force conditions. This can be readily verified from Eq. (85) by substituting a constant angular rate and specific force vector for the \( \omega_{FB}^m \) and \( a_{\text{SF}}^B \) terms and carrying out the indicated operations analytically. As such, \( \Delta R_{\text{scrlm}} \) will only produce an output under the presence of dynamic body axis angular rate/specific force components. This is an important characteristic because, for most real dynamic environments, the magnitude of high-frequency angular rate/specific force is small so that first-order approximations accurately apply (first order in integrated body angular rate/specific force over the \( t_{n-1} \) to \( t_n \) time interval). We conclude that the analytical form for \( \Delta R_{\text{scrlm}} \) will also yield a reasonably accurate solution under situations where the low-frequency body angular rate and specific force components are large.

The Eq. (86) first-order version of position rotation compensation \( \Delta R_{\text{scrlm}} \) can have noticeable second-order error under extreme maneuvers. The form of Eq. (84) that has \( \Delta R_{\text{scrlm}} \) separate from other terms allows us to expand \( \Delta R_{\text{scrlm}} \) to a more accurate form that is exact under constant angular rate/specific force (as in the first subsection of Sec. III.B.2 for the velocity rotation compensation term). The following sections develop algorithms for the exact position rotation compensation term in Eq. (84) and for the scrolling and other integral terms in Eq. (85).

1. Exact Position Rotation Compensation

An improved accuracy version of \( \Delta R_{\text{scrlm}} \) for Eq. (84) is developed by specifying the solution to be exact under constant body angular rate/specific force but to first order, to also equal \( \Delta R_{\text{cxmlm}}^{\theta} \) in Eq. (86) under general angular rate/specific force conditions. The derivation begins by returning to the basic definition for \( \Delta R_{\text{SF}m}^{\theta} \) in Eq. (79):

\[
\Delta R_{\text{SF}m}^{\theta} = \int_{t_{n-1}}^{t_n} \Delta R_{\text{SF}m}^{\theta(n-1)}(t) \, dt
\]

As with Eqs. (42) and (44), we now use the exact definition for \( \Delta v_{\text{SF}m}^{R(n-1)}(t) \) in Eq. (87) based on constant \( B \) frame specific force and nonconing angular rate

\[
\Delta R_{\text{SF}m}^{\psi} = \int_{t_{n-1}}^{t_n} \int_{t_{n-1}}^{t} a_{\text{SF}}^B \, d\tau \, dt
\]

\[
+ \left( a_m \times a_{\text{SF}}^B \right) \int_{t_{n-1}}^{t} \int_{t_{n-1}}^{t} \sin(\tau) \, d\tau \, dt
\]

\[
+ \left[ a_m \times \left( a_m \times a_{\text{SF}}^B \right) \right] \int_{t_{n-1}}^{t} \int_{t_{n-1}}^{t} \left( 1 - \cos(\tau) \right) \, d\tau \, dt
\]
Finally, we compare Eq. (96) and its first-order version with Eq. (97) to deduce the sought-after exact position rotation compensation algorithm. Including trigonometric expansion formulas, the result is

$$\Delta R_{\text{rot}} = \frac{1}{2} \left( S_{\alpha m} \times v_m + \alpha_m \times S_{\alpha m} \right)$$

(97)

We also note that by applying Taylor series expansion to the trigonometric terms [as shown subsequently in Eq. (99)] Eq. (96) to first order is given by

$$\Delta R_{\text{rot}} = \frac{1}{2} \left( S_{\alpha m} \times v_m + \alpha_m \times S_{\alpha m} \right)$$

(98)

Finally, we compare Eq. (96) and its first-order version with Eq. (97) to deduce the sought-after exact position rotation compensation algorithm. Including trigonometric expansion formulas, the result is

$$\Delta R_{\text{rot}} = \frac{1}{2} \left( S_{\alpha m} \times v_m + \alpha_m \times S_{\alpha m} \right)$$

(99)

Equations (99) can be utilized in Eq. (84) in place of $\Delta R_{\text{rot}}$ from Eq. (84) to obtain the equivalent higher-order equation for $\Delta R_{\text{rot}}$, that is exact under constant body angular rate/force conditions.

2. **Scrolling and Other Integral Term Increments**

The computer algorithms used to implement the integration operations in Eq. (85) are executed at high computer repetition rate, i.e., the sculling and force condition is equivalently treated. The $\omega(t), v_m, \alpha(t)$, and $\alpha_m$ integral terms in Eq. (85) are provided by Eqs. (54) and (55). The remaining integral terms in Eq. (85) can be rewritten to reflect the high-speed computing cycle as follows:

$$S_{\omega}(t) = S_{\omega_{m-1}} + \Delta S_{\omega}(t)$$

$$\Delta S_{\omega}(t) = \int_{t_{m-1}}^{t} \omega_m(t) \, dt$$

$$S_{\alpha}(t) = S_{\alpha_{m-1}} + \Delta S_{\alpha}(t)$$

$$\Delta S_{\alpha}(t) = \int_{t_{m-1}}^{t} \alpha_m(t) \, dt$$

$$S(t) = S_{v_{m-1}} + \Delta S_{v}(t)$$

$$\Delta S_{v}(t) = \int_{t_{m-1}}^{t} v_m(t) \, dt$$

$$S_{\alpha m}(t) = S_{\alpha m_{m-1}} + \Delta S_{\alpha m}(t)$$

$$\Delta S_{\alpha m}(t) = \int_{t_{m-1}}^{t} \alpha_m(t) \, dt$$

$$S_{v_m}(t) = S_{v_{m-1}} + \Delta S_{v_m}(t)$$

$$\Delta S_{v_m}(t) = \int_{t_{m-1}}^{t} v_m(t) \, dt$$

$$S_{\alpha m}(t) = S_{\alpha m_{m-1}} + \Delta S_{\alpha m}(t)$$

$$\Delta S_{\alpha m}(t) = \int_{t_{m-1}}^{t} \alpha_m(t) \, dt$$
\[ \Delta R_{\text{scrl}} = \Delta R_{\text{scrl}}^{t=0} + \delta R_{\text{scrl}}, \quad \Delta R_{\text{scrl}} = \Delta R_{\text{scrl}}(t = t_{m}) \]

\[ \delta R_{\text{scrl}} = \frac{1}{6} \int_{\tau_{t-1}}^{\tau_{t}} \left[ \frac{5}{6} \Delta \nu_{\text{scrl}}(t) - S_{u}(t) \times a_{\text{scrl}}^{p}(t) \right] dt \]

\[ + S_{v}(t) \times a_{\text{scrl}}^{p}(t) + (\pi(t) - v_{t}) \right] dt \]

\[ \Delta \nu_{\text{scrl}}(t) = \Delta \nu_{\text{scrl}}^{t=0} + \delta \nu_{\text{scrl}}(t) \]

\[ \Delta R_{\text{scrl}} = \Delta R_{\text{scrl}}^{t=0} + \delta R_{\text{scrl}}^{t=0} + \delta R_{\text{scrl}}^{t=0} + \delta R_{\text{scrl}}^{t=0} \]

Equations (105) can be classified as a second-order algorithm for \( \delta R_{\text{scrl}} \) because they include current and past cycle \( \Delta \alpha_{c}, \Delta v_{t} \) products. If the angular rate/specific force profile was approximated as constant over two successive \( l \) cycles, the \((\Delta \alpha_{t} - \Delta \alpha_{t-1})\) and \((\Delta v_{t} - \Delta v_{t-1})\) terms in Eq. (105) would vanish, resulting in a first-order \( \delta R_{\text{scrl}} \) algorithm. Under conditions where the angular rate and specific force can be approximated as constant, i.e., slowly varying over an \( m \) cycle, \( \Delta R_{\text{scrl}} \) in Eq. (105) is approximately zero and the \( \delta R_{\text{scrl}}, \delta R_{\text{scrl}}^{t=0}, \delta R_{\text{scrl}}^{t=0} \) calculations in Eq. (105) can be deleted. Alternatively, (and more accurately), for slowly varying angular rate and specific force, one \( l \) cycle of Eq. (105) can be executed each \( m \) cycle, noting from the initial condition definitions that \( \alpha_{t-1} \), \( v_{t-1} \), \( S_{u_{t-1}} \), and \( S_{v_{t-1}} \) are zero. As noted in the second subsection of Sec. III.B.2, setting the \( l \) and \( m \) rates equal can also be achieved by increasing the \( m \) rate to match the \( l \) rate. This result would be a single high-speed, higher-order algorithm with a simpler software architecture than the two-speed approach but requiring more throughput. Continuing advances in the speed of modern-day computers may make this the preferred approach for the future.

V. Velocity/Position Integration Algorithm Summary

Table 1 is a summary of the algorithms described for the speed-down inertial navigation velocity/position integration function listed in the order that they would be executed in the navigation computer. Note in Table 1 that the normal speed attitude calculation follows the normal speed position calculation, in contrast to Table 1 of Ref. 1, which calculates attitude before position. Having the attitude follow the position calculation allows the high-resolution \( \Delta R_{v}^{p} \) from Eq. (77) to be used in Ref. 1, Eq. (53), rather than the less-accurate Ref. 1, Eq. (56), trapezoidal algorithm form of \( \Delta R_{v}^{p} \).

VI. Algorithm and Execution Rate Selection

Section VI of Part I (Ref. 1) discusses the general process of algorithm selection for a given application with required execution rates to achieve specified accuracy goals. A principal part of this process involves estimating the algorithm error under anticipated angular rate/specific force maneuvers/vibrations compared with specified error budget requirements. Evaluation of candidate algorithm error characteristics is generally performed using computerized time-domain simulators that exercise the algorithms, in particular groupings, at their selected repetition rates. The simulators generate speed-down inertial sensor angular rate/specific force profiles for algorithm test input together with known navigation parameter solutions for algorithm output comparison, e.g., Ref. 8, Secs. 11.2.1-11.2.4.

For the two-speed velocity/position updating approach described, the repetition rate for the moderate speed (\( m \) cycle) algorithm would typically be selected based on maximum angular rate/specific force considerations to minimize power series truncation error in the moderate- and high-speed algorithms. The repetition rate for the high-speed (\( l \) cycle) algorithms would typically be selected based on the anticipated speed-down inertial sensor assembly vibration environment to accurately account for vibration induced sculling/sculling effects.

For the velocity algorithms, simplified analytical error models can also be used to predict high-speed sculling algorithm error under selected sculling rates/amplitudes as a function of algorithm repetition rate (Refs. 5-7 and 8, Chap. 10). The sculling rates/amplitudes must be derived either from empirical data or, more commonly, from analytical models of the sensor assembly mount imbalance and its response to external input vibration at particular frequencies (Ref. 8, Chap. 10). Frequency-domain simulators can be used to evaluate high-speed sculling algorithm error under specified input vibration power spectral density profiles and sensor assembly mount imbalance as a function of algorithm repetition rate (Ref. 8, Chap. 10). For example, the sculling algorithm described by Eqs. (59-61) can be shown by such simulators to have an error of 0.044 \( \mu \)g when operated at a 2-Hz repetition rate under exposure to 7.6 \( \mu \)g rms wideband random linear input vibration (flat 0.04 \( g^{2}/\text{Hz} \) density from 20 to 1000 Hz, then decreasing logarithmically to 0.01 \( g^{2}/\text{Hz} \) at 2000 Hz). The linear vibration generates a multiaxis 3.6 \( g \) rms specific force/angular oscillation of the sensor assembly with a corresponding rectified sculling acceleration of 1300 \( m/s^{2} \).
The capabilities of modern-day computers and INS software technology make it reasonable to specify that the navigation algorithm error be no greater than 5% of the equivalent error produced by the INS inertial sensors (whose cost increases dramatically with accuracy demands). For an INS with a 40-μg accelerometer bias accuracy requirement (typical for an aircraft INS having 2–3 fps 1σ velocity accuracy), the 0.044-μg sculling algorithm error is almost two orders of magnitude within the 5% allowance, providing a wide design margin for the algorithm 2-kHz repetition rate selection. For this case, a 1-kHz sculling algorithm rate would probably be more appropriate; however, 2 kHz might still be utilized for compatibility with the 2-kHz rate selected for the coning algorithm in Part 1 (Ref. 1) under the same conditions.

In the case of the positioning algorithms, the typical form presented in Sec. IV.B is usually adequate for almost all applications (to date). For the exceptional cases where very high-resolution position updating is required, the time interval for the accuracy requirement is usually restricted to brief periods during the application mission profile. Moreover, for some of these applications, postprocessing is acceptable using data recorded during the high-resolution time interval; hence, the complexity of the high-resolution algorithms would not be a real-time computer throughput issue. For example, for synthetic aperture radar (SAR) motion compensation, high-resolution position data are required for only brief intervals, e.g., 5–10 s, during SAR data acquisition, which may then be subsequently processed for SAR image formation. We also note that, in high-resolution applications, the Earth-referenced position of the INS chassis mount is usually the required output, which equals the sum of Earth-referenced inertial sensor assembly position (calculated by the inertial navigation algorithms) plus vibrationspecific forces.

### Table 1 Summary of strapdown INS velocity/position computation algorithms

<table>
<thead>
<tr>
<th>Algorithm function</th>
<th>Input</th>
<th>Output</th>
<th>Equation number</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>High-speed calculations</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrated B frame angular rate increments</td>
<td>$\alpha_i$, $\alpha_{i+m}$</td>
<td>$\Delta v_i$, $v_i, v_{i+m}$</td>
<td>Ref. 1, Table 1</td>
</tr>
<tr>
<td>Integrated B frame specific force increments</td>
<td>$\Delta \alpha_i$, $\alpha_{i+m}$</td>
<td>$\Delta v_i$, $v_i, v_{i+m}$</td>
<td>(55) or (60)</td>
</tr>
<tr>
<td>Sculling increment</td>
<td>$\Delta v_i$, $v_i, v_{i+m}$</td>
<td>$\Delta v_{sculling}, v_{sculling}$</td>
<td>(56) or (61)</td>
</tr>
<tr>
<td>Doubly integrated B frame angular rate and specific force increments (for high-resolution position algorithm)</td>
<td>$\Delta \alpha_i$, $\alpha_{i+m}$</td>
<td>$\Delta S_{i+1} - S_i$, $S_i$, $S_{i+m}$</td>
<td>(100) or (104)</td>
</tr>
<tr>
<td>Sculling increment (for high-resolution position algorithm)</td>
<td>$\Delta v_i$, $v_i, v_{i+m}$</td>
<td>$\Delta R_{sculling}$</td>
<td>(101) or (105)</td>
</tr>
<tr>
<td><strong>Normal-speed calculations for Earth-related parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N$ frame plumb-bob gravity components</td>
<td>$C_{E}^{N}$, $h$</td>
<td>$g_{N}^{E}$</td>
<td>Ref. 1, Eq. (19)</td>
</tr>
<tr>
<td>$N$ frame Earth rate components</td>
<td>$C_{E}^{N}$, $h$</td>
<td>$\omega_{E}^{N}$</td>
<td>(2)</td>
</tr>
<tr>
<td>Vertical transport rate component</td>
<td>$C_{E}^{N}$, $h$</td>
<td>$\beta_{ZN}$</td>
<td>Ref. 8, Sec. 4.6</td>
</tr>
<tr>
<td>Curvature matrix</td>
<td>$C_{E}^{N}$, $h$</td>
<td>$F_{C}$</td>
<td>Ref. 8, Sec. 5.1.3</td>
</tr>
<tr>
<td><strong>Normal-speed velocity calculations</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B$ frame velocity rotation compensation (exact formulation)</td>
<td>$\alpha_i$, $v_i$</td>
<td>$\Delta v_{rot}$</td>
<td>(51), (52)</td>
</tr>
<tr>
<td>$B$ frame velocity rotation compensation</td>
<td>$\alpha_i$, $v_i$</td>
<td>$\Delta v_{rot}$</td>
<td>(39)</td>
</tr>
<tr>
<td>$B$ frame integrated specific force increment</td>
<td>$v_{i+m}$, $\Delta v_{rot}$, $\Delta v_{rot}$</td>
<td>$\Delta S_{i+1} - S_i$, $S_i$, $S_{i+m}$</td>
<td>(37)</td>
</tr>
<tr>
<td>$L$ frame integrated specific force increment</td>
<td>$\Delta S_{i+1} - S_i$, $S_i$, $S_{i+m}$</td>
<td>$\Delta v_{rot}$</td>
<td>(11)</td>
</tr>
<tr>
<td>L frame rotation vector (cycle $n - 1$ to $m$)</td>
<td>$\omega_{E}^{N}$, $\beta_{ZN}$, $F_{C}$, $v_{i}$</td>
<td>$\xi_{i}$, $\eta_{i}$, $\zeta_{i}$</td>
<td>(17), (19), (20)</td>
</tr>
<tr>
<td>L frame rotation matrix (first-order form)</td>
<td>$L_{i+1} - L_i$, $L_{i}$</td>
<td>$C_{L}^{N}<em>{i+1}$, $C</em>{L}^{N}_{i}$</td>
<td>(14)</td>
</tr>
<tr>
<td>L frame rotation compensation</td>
<td>$\Delta v_{rot}$, $\Delta v_{rot}$, $\Delta v_{rot}$</td>
<td>$\Delta S_{i+1} - S_i$, $S_i$, $S_{i+m}$</td>
<td>(13)</td>
</tr>
<tr>
<td>Integrated Coriolis acceleration and plumb-bob gravity increment</td>
<td>$\omega_{E}^{N}$, $\beta_{ZN}$, $F_{C}$, $v_{i}$</td>
<td>$\Delta v_{rot}$</td>
<td>(7), (8), (9)</td>
</tr>
<tr>
<td>$N$ frame velocity update</td>
<td>$\Delta v_{rot}$</td>
<td>$\Delta v_{rot}$</td>
<td>(4)</td>
</tr>
<tr>
<td><strong>Normal-speed position calculations</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position rotation compensation (high-resolution position algorithm, exact form)</td>
<td>$\alpha_i$, $S_{i}$, $S_{i+m}$</td>
<td>$\Delta R_{rot}$</td>
<td>(99)</td>
</tr>
<tr>
<td>Position rotation compensation (high-resolution position algorithm, first-order accuracy form)</td>
<td>$\alpha_i$, $S_{i}$, $S_{i+m}$</td>
<td>$\Delta R_{rot}$</td>
<td>(86)</td>
</tr>
<tr>
<td>Body frame position increment due to specific force (high-resolution position algorithm)</td>
<td>$S_{i+m} - \Delta R_{rot}$</td>
<td>$\Delta R_{rot}$</td>
<td>(84)</td>
</tr>
<tr>
<td>$N$ frame position increment (high-resolution position algorithm)</td>
<td>$\Delta S_{rot}$, $\Delta S_{rot}$, $v_{i}$</td>
<td>$\Delta R_{rot}$</td>
<td>(83)</td>
</tr>
<tr>
<td>$N$ frame position increment (trapezoidal position algorithm)</td>
<td>$\Delta S_{rot}$, $\Delta S_{rot}$, $v_{i}$</td>
<td>$\Delta R_{rot}$</td>
<td>(75)</td>
</tr>
<tr>
<td>Altitude change</td>
<td>$\Delta h_{i}$</td>
<td>$\Delta h_{i}$</td>
<td>(66)</td>
</tr>
<tr>
<td>Position rotation vector</td>
<td>$\beta_{ZN}$, $F_{C}$, $\Delta R_{rot}$</td>
<td>$\xi_{i}$</td>
<td>(73)</td>
</tr>
<tr>
<td>Position rotation change matrix</td>
<td>$\xi_{i}$</td>
<td>$C_{N}^{E}(\xi_{i})$</td>
<td>(70)</td>
</tr>
<tr>
<td>Altitude update</td>
<td>$h_{i+1} - h_{i}$, $h_{i}$</td>
<td>$h_{i}$</td>
<td>(64)</td>
</tr>
<tr>
<td>Position direction cosine matrix update</td>
<td>$C_{N}^{E}(\xi_{i+1})$, $C_{N}^{E}(\xi_{i})$</td>
<td>$C_{E}^{N}(h_{i})$</td>
<td>(68)</td>
</tr>
<tr>
<td><strong>Normal-speed attitude calculations</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attitude direction cosine matrix update</td>
<td>$C_{E}^{N}$</td>
<td>$C_{E}^{N}$</td>
<td>Ref. 1, Table 1</td>
</tr>
</tbody>
</table>
force induced displacement of the sensor assembly relative to the INS chassis/mount (due to compliance of elastomeric isolators that interface the sensor assembly to the INS chassis). The latter displacement can be computed under dynamic maneuvers by quasi-static flexure modeling, i.e., displacement equals average specific force times the square of the sensor assembly/isolator undamped natural frequency, and for appropriate digital filtering of vibration-induced jitter (Ref. 8, Chap. 9). Note that, in principle, the displacement can also be measured directly using specially installed sensing devices.

As an example of the inertial navigation position integration algorithm selection process, let us consider a high-resolution application with an overall INS requirement for position error fluctuations to be significantly less than 1 cm during 5–10 s periods (not unusual for applications where the actual requirement is a function of error frequency content and not clearly known). Allowing design margin for error in the sensor assembly to chassis/mount flexure displacement calculation (described in the preceding paragraph), we budget the INS accuracy specification into a requirement for the position algorithm to have less than 0.01-cm dynamic position error fluctuation during 5–10 s. Let us further assume for this example that the basic position algorithm update rate has been selected to be 50 Hz and that the selected inertial velocity algorithm accuracy is compatible with high-resolution position updating requirements, e.g., includes high-rate sculling. Simplified pencil-and-paper analysis of the typical form equation (75) position algorithm (or other versions) can be used to assess its accuracy at 50 Hz using the high-resolution algorithm to represent the correct truth model. An analytical model for the high-resolution $\Delta R_{\text{typ}}^0$ increment truth model can be derived using Eq. (25) for $\Delta v_{\text{typ}}^0$ in Eq. (79):

$$\Delta R_{\text{typ}}^0 = \int_{t_{n-1}}^{t_n} \Delta v_{\text{typ}}^0(t) \, dt$$

Neglecting the small $L$ frame rotation effect, it can be shown that position updating based on the Eq. (75) typical algorithm is equivalent to Eq. (79) with $\Delta R_{\text{typ}}^0$ replaced by the typical algorithm equivalent $\Delta R_{\text{typ}}^0$, given by

$$\Delta R_{\text{typ}}^0 = \frac{1}{2} \Delta v_{\text{typ}}^0 T_n$$

For the Eq. (58) linearly ramping specific force/angular rate model in Eqs. (106) and (107), the position increments for the truth model $\Delta R_{\text{typ}}^0$, and for the typical algorithm $\Delta R_{\text{typ}}^0$, become

$$\Delta R_{\text{typ}}^0 = \frac{1}{2} C T_m^2 + \frac{1}{2} (D + A \times C) T_m^3 + \frac{1}{12} (A \times D + B \times C) T_m^4 + \frac{1}{24} B \times D T_m^5$$

$$\Delta R_{\text{typ}}^0 = \frac{1}{2} C T_m^2 + \frac{1}{2} (D + A \times C) T_m^3 + \frac{1}{12} (A \times D + B \times C) T_m^4 + \frac{1}{24} B \times D T_m^5$$

Comparing $\Delta R_{\text{typ}}^0$ with the $\Delta R_{\text{typ}}^0$ truth model in Eq. (108) allows the error in $\Delta R_{\text{typ}}^0$ to be assessed for selected maneuver values. Under a constant $C$ specific force maneuver, $\Delta R_{\text{typ}}^0$ equals $\Delta R_{\text{typ}}^0$, and, hence, is error free. For $D = 3$ g/s or for $C = 3$ g with $A = 1$ rad/s, the calculated error in $\Delta R_{\text{typ}}^0$ (using $T_m = 0.02$ s for the 50-Hz update rate) is 0.00196 cm or $0.00196 = 0.098$ cm in 1 s. Compared with the 0.01 cm in 5–10 s requirement, the 0.098 cm in 1 s figure would be considered unacceptable.

Position algorithm assessment under vibration can also be analytically estimated. For example, for the 3.6 g rms sensor assembly vibration (in the preceding sculling example), the associated velocity vibration is 11.2 cm/s rms centered around the sensor assembly 50-Hz mount (which would be accurately measured by the hypothesized velocity algorithm). The aliasing error associated with sampling the vibrating velocity at 50 Hz for the Eq. (75) algorithm can produce a $1.12 \times 0.02 = 0.022$ cm error each position update. If the error is random per update, the total cumulative error in 1 s (50 updates) would be $0.22 \times \sqrt{50} = 1.6$ cm; if the error is systematic, the position error in 1 s would be $0.22 \times 50 = 11.2$ cm. In either case, the algorithm error greatly exceeds the 0.01 cm over 5–10 s requirement.

Based on such analyses, let us assume we have elected to use the Eqs. (83) and (84) high-resolution position algorithm to assure 5–10 s, 0.01-cm high-quality resolution. The next question is which terms in Eq. (84) are to be included. The $S_{\text{typ}}$ term in Eq. (84) is the dominant term for integrating velocity into position and must be included. Under a 3-g constant specific force maneuver, $S_{\text{typ}}$ from Eq. (85) equals 0.59 cm per 50-Hz position update cycle or 29.4 cm in 1 s. The next most important term is the $\Delta R_{\text{rot}}$ rotation compensation term. Using Eq. (86) with Eq. (85) input, the magnitude of $\Delta R_{\text{rot}}$, under a constant 3 g/1 rad/s maneuver is 0.0039 cm per update cycle or 0.20-cm cumulative position change in 1 s. (Note, for a 3-gs linearly ramping specific force, $S_{\text{typ}}$ also equals 0.0039 cm per cycle and sums to 0.20 cm in 1 s.) For the 0.01 cm over 5–10 s requirement, the $\Delta R_{\text{rot}}$ term is, therefore, also needed. The question of whether to include the $\Delta R_{\text{rot}}$ term can be addressed by analyzing the magnitude of $\Delta R_{\text{rot}}$ under dynamic vibration motion using a rearranged version of Eq. (84):

$$\Delta R_{\text{rot}} = \frac{\Delta R_{\text{F}}^0 - S_{\text{typ}} - \Delta R_{\text{rot}}}{\alpha}$$

Consider the 3.6-g rms vibration condition under 1-rad/s constant angular rate. For a 3.6-g rms pure sine wave, i.e., $3.6 \times \sqrt{2} = 5.1$-g amplitude, at the 50-Hz isolator resonance frequency, the magnitudes of $S_{\text{typ}}$ and $\Delta v_{\text{rot}}$ over 0.02 s are, from Eqs. (85), 0.32 cm and 0 cm/s, respectively. For the 1 rad/s rate over 0.02 s, $\alpha = 0.02$ rad. Thus, from Eq. (86), $\Delta R_{\text{rot}}$ is $(0.32 \times 0.02)/6) = 0.0011 \text{ cm}$, which, if systematic, accumulates in 1 s to 0.0011 x 50 = 0.053 cm. If random from cycle to cycle, the error accumulation over 10 s would be 0.0011 x $(50 \times 10) = 0.024$ cm. The true solution $\Delta R_{\text{rot}}$ for this particular case can be demonstrated by analytical integration of Eq. (106) to be $\Delta R_{\text{rot}} = S_{\text{typ}}$. Thus, from Eq. (109) and the latter $\Delta R_{\text{rot}}$ analyses, the cumulative magnitude of $\Delta R_{\text{rot}}$ is 0.053 cm/s (if systematic) and 0.024 cm over 10 s (if random). To meet the accuracy requirement of 0.01 cm over 5–10 s, we conclude that $\Delta R_{\text{rot}}$ will also be required. The final question is which particular terms in the Eq. (105) $\Delta R_{\text{rot}}$ algorithm are needed. The answer can be obtained from similar individual analyses of each term in Eq. (105) to identify which are significant relative to the requirement. A simpler approach is to arbitrarily, but conservatively, use the full Eq. (105) form. The rationale might be that the savings in using a simplified version, e.g., without the second-order terms, is not worth the time and cost for justification, assuming computer throughput is not an issue. The latter approach has additional merit because it totally frees the system designer of concern for INS algorithm error during the development optimization process of the system using the INS.

The position algorithm selection process just described is fairly rudimentary, admittedly conservative, but sufficient if the outcome is the conservative approach of applying the full high-resolution algorithm, particularly if the accuracy requirement cannot be clearly defined. Had the choice been to use the typical algorithm or an alternate version thereof, a more sophisticated process would have been required to assure adequate performance over a more accurate and complete set of defined operating conditions. For example, complex maneuver/vibration profiles can be simulated and input to the trial algorithm, with its accuracy evaluated using the high-resolution algorithm (with the same input) as a reference. In this regard, the high-resolution algorithm can be viewed as a truth model for position algorithm evaluation but available for use if the trial algorithm is inadequate. An assessment of the need to include particular terms in the scrolling portions of the high-resolution algorithm can be made similarly by calculating the magnitude of each term under simulated input vs error allowances. (A term is needed if its magnitude exceeds the allowance.) The latter step can be augmented using analytical models for input conditions, similar to the approach described in the last example.
VII. Concluding Remarks

Reference 1 defined requirements for the strapdown INS integration algorithms in the form of continuous differential equations and developed the attitude integration algorithms. In Part 2, we have presented a comprehensive design process for development of the specific force transformation/velocity integration and position integration algorithms based on the two-speed updating approach described in Part 1 (Ref. 1) for attitude integration: use of an exact moderate-speed algorithm for the basic integration function fed by a high-speed algorithm to measure high-frequency rectification effects. The moderate-speed algorithms are analytically exact under constant angular rate/specific force; the high-speed algorithms account for deviations from constant angular rate/specific force (sculling for the velocity algorithm and scrolling for the position algorithm). Where computer throughput restrictions are not an issue, the two-speed structure can be compressed into a single high-speed format by operating the moderate-speed algorithm at the high-speed rate. A summary of the velocity/position integration algorithms developed herein is provided in Table 1 as a listing in the order they would be executed in the navigation computer. A similar table is provided in Part 1 (Ref. 1) for the attitude integration algorithms.

References

Errata

Strapdown Inertial Navigation Integration Algorithm Design
Part 1: Attitude Algorithms

Paul G. Savage
Strapdown Associates, Inc., Maple Plain, Minnesota 55359


Errata

Strapdown Inertial Navigation Integration Algorithm Design
Part 2: Velocity and Position Algorithms

Paul G. Savage
Strapdown Associates, Inc., Maple Plain, Minnesota 55359

[J. Guidance 21 (2), pp. 208–221 (1998)]

Equation (4) and (44) should be corrected as follows:

Equation (4) should read

$$C_{4i}^L = I + \frac{\sin \phi}{\phi}(\phi \times) + \left(1 - \cos \phi\right)\left(\phi \times\right)^2$$

Equation (44) should read

$$\int_{t-1}^t \left(\Delta \alpha(t) \times \omega^{B}_F\right) dt = \frac{1}{12}\left(\Delta \alpha_{t-1} \times \Delta \alpha_t\right)$$

The correction carries into the expression

$$\Delta v_{SFu}^L \approx \frac{1}{2}\left(c_{L_{1}(t-1)}^L + c_{L_{1}(t-1)}^L\right)\int_{t-1}^t c_{B(t)}^B a_{SFu}^B dt$$

Equations (11) and (12) are unchanged, but Eq. (13) becomes

$$\Delta v_{SFu}^L = \frac{1}{2}\left(c_{L_{1}(t-1)}^L + c_{L_{1}(t-1)}^L\right) \Delta v_{SFu}^L$$

The correction carries into the $$\Delta v_{SFu}^L$$ term in Eqs. (78) as follows:

$$\Delta v_{SFu}^L(t) = \frac{1}{2} \left(c_{L_{1}(t-1)}^L - c_{L_{1}(t-1)}^L\right) \Delta v_{SFu}^L \left((t - t_{m-1})^2 / T_u^2\right)$$

and finally into Eq. (79), which changes the $$\frac{1}{2}$$ term to $$\frac{1}{6}$$ in the expression $$\Delta R_{SFu}^B$$.