Sensors **2011**, *11*, 7993-8017; doi:10.3390/s110807993

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

Field Programmable Gate Array Based Parallel Strapdown Algorithm Design for Strapdown Inertial Navigation Systems

Zong-Tao Li [1], Tie-Jun Wu [2], Can-Long Lin [1,2] and Long-Hua Ma [2,*]

[1] Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China; E-Mails: ztli@iipc.zju.edu.cn (Z.-T.L.); canlonglin@foxmail.com (C.-L.L.)

[2] Institute of Navigation Guidance and Control, Zhejiang University, Hangzhou 310027, China; E-Mail: tjwu@zju.edu.cn (T.-J.W.)

* Author to whom correspondence should be addressed; E-Mail: lhma@iipc.zju.edu.cn; Tel.: +86-571-8795-2841; Fax: +86-571-8795-2841.

Received: 4 July 2011; in revised form: 3 August 2011 / Accepted: 11 August 2011 / Published: 15 August 2011

Abstract: A new generalized optimum strapdown algorithm with coning and sculling compensation is presented, in which the position, velocity and attitude updating operations are carried out based on the single-speed structure in which all computations are executed at a single updating rate that is sufficiently high to accurately account for high frequency angular rate and acceleration rectification effects. Different from existing algorithms, the updating rates of the coning and sculling compensations are unrelated with the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples. When the output sampling rate of inertial sensors remains constant, this algorithm allows increasing the updating rate of the coning and sculling compensation, yet with more numbers of gyro incremental angle and accelerometer incremental velocity in order to improve the accuracy of system. Then, in order to implement the new strapdown algorithm in a single FPGA chip, the parallelization of the algorithm is designed and its computational complexity is analyzed. The performance of the proposed parallel strapdown algorithm is tested on the Xilinx ISE 12.3 software platform and the FPGA device XC6VLX550T hardware platform on the basis of some fighter data. It is shown that this parallel strapdown algorithm on the FPGA platform can greatly decrease the execution time of algorithm to meet the real-time and high precision requirements of system on the high dynamic environment, relative to the existing implemented on the DSP platform.

Keywords: strapdown algorithm; coning and sculling compensation; parallelization design; computation complexity; FPGA

1. Introduction

In a strapdown inertial navigation system (SINS), inertial sensors are rigidly attached to the vehicle, which leads to the system suffering from the highly dynamic vehicle movement environment. In addition, inertial sensors may be subject to high frequency motion as a result of body bending and engine-induced vibration. The strapdown algorithms adopted by most modern SINSs are constructed based on a general two-speed structure by which the position, velocity and attitude (PVA) updating operations are divided into two parts [1,2]: an exact moderate-speed updating routine (e.g., 50 ~ 200 Hz) typically designed to update each PVA based on the maximum angular rate and acceleration of vehicle; and a high-speed updating routine (e.g., 1 ~ 4 KHz for an aircraft INS with the positioning accuracy of less than 1 nmile/h) designed to accurately account for vibration-induced coning and sculling effects based on the anticipated vibration condition of the system. The original intention of the two-speed structure is to overcome the throughput limitations of early computer techniques, but the limitation is rapidly becoming insignificant with the continuous improvement in the performance of modern high-speed computers [3]. On the other hand, along with the fast progress of modern vehicles in ultra-high speed and other maneuvering performances, there exist more and more urgent demands to promote the navigation and control precision of the vehicles in high dynamic motion. It provides the motivation to return to a simpler single-speed structure of the strapdown algorithm in which all computations are executed at a single updating rate that is sufficiently high to accurately account for high frequency angular rate and acceleration rectification effects.

Two key compensation algorithms designed to operate in severe maneuvering and vibratory environments are critical in determining the performance of a SINS, *i.e.*, the coning compensation that works when the vehicle's angular rate vector is rotating, and the sculling compensation that takes effect when the vehicle's angular rate or specific force acceleration vector is rotating, or when the ratio of the angular rate to specific force is not constant. Thus in order to improve the navigation accuracy of the system, particularly in the environments where the angular rate vector or the specific force vector of the vehicle is large, several algorithms have been developed for the coning and sculling compensation. A substantial number of integration algorithms have been designed for coning compensation to improve the attitude accuracy without sacrificing computer throughput [4-11]. Analogous to the coning compensation algorithm adopted in attitude updating, a number of sculling compensation algorithms have also been designed for velocity updating, and the equivalence between coning and sculling compensation algorithms is discussed in [12,13]. A detailed statement of the coning and sculling compensation algorithms is given in [1,3,14-17]. Most algorithms for the coning and sculling compensations are based on truncated Taylor series expansion approximations for the angular rate of vehicle over updating cycles [2,3,6,7,9-11,18]. The accuracy of the coning and sculling compensation algorithms is determined by the updating rate of the coning and sculling compensations and the order of the truncated Taylor series expansion for the angular rate and specific force. Generally, in order to improve the accuracy of these algorithms, the updating rate must be increased to keep track of vehicle angular and linear motions more accurately. Among these existing algorithms, however, when the sampling rates of inertial sensors remain constant, and the number of the gyro incremental angle samples for coning compensation and the number of the accelerometer incremental velocity samples for sculling compensation are selected, the updating rates of these algorithms are also determined. The increase of the updating rates results in the decrease of the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples for the coning and sculling compensation (namely, the decrease of order of the coning and sculling compensation algorithms), which in turn reduces the accuracy of the algorithms.

Furthermore, in recent SINS applications the strapdown algorithm with coning and sculling compensations is commonly implemented in a Digital Signal Processor (DSP) platform supplemented with Field Programmable Gate Array (FPGA) for data acquisition and noise filtering. Due to the serial execution mode of the DSP, however, it cannot support an updating rate fast enough for a high-order algorithm. In order to tackle the conflict between the computation complexity of the high-order algorithm and the updating speed of the algorithm implemented on a DSP platform, Xie [19] proposed a strapdown algorithm architecture on dual DSPs and FPGA which in essence works in a parallel computation mode. Jew [20] presented a framework for designing inertial navigation systems on a single-chip FPGA, in which the strapdown algorithm is implemented by the PowerPC hardcore of the FPGA. Although to some extent these methods improved the performance of the strapdown algorithms, they all work in a serial mode, thus it did not make full use of the parallel computation characteristics of the FPGA. Some other researchers [21,22] suggested using a single-chip FPGA to implement multi-processing cores and parallel computing, but there is no any implementation scheme discussed in detail.

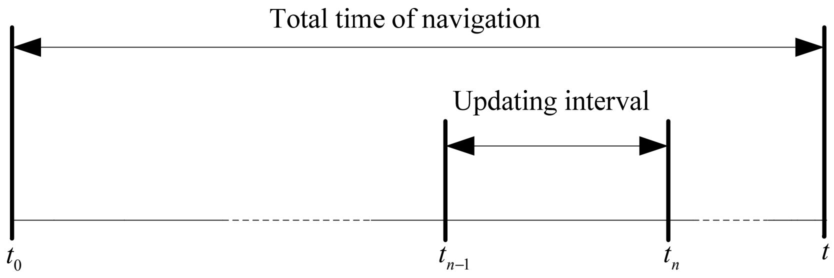
In this paper, a new generalized optimum strapdown algorithm with coning and sculling compensations is presented in Section 2, in which the PVA updating operations are carried out based on the single-speed structure in which all computations are executed at a single updating rate that is sufficiently high to accurately account for high frequency angular rate and acceleration rectification effects. Different from existing algorithms, the updating rates of the coning and sculling compensations are unrelated with the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples. Then, in order to implement the new strapdown algorithm in a single chip FPGA, the parallelization of the algorithm is designed in Section 3, and its computational complexity is analyzed in Section 4. In Section 5, the performance of the proposed parallel strapdown algorithm is tested on the software platform of Xilinx ISE 12.3 and the hardware platform of FPGA device XC6VLX550T on the basis of some fighter aircraft data. The contributions of this paper are finally summarized in Section 6.

2. Generalized Optimum Strapdown Algorithm

In order to reduce the computational complexity and decouple the relationship between the updating rates of the coning and sculling compensations and the number of the gyro incremental angle and the accelerometer incremental velocity samples, the strapdown algorithm proposed in this section is constructed on the basis of a single-speed structure, *i.e.*, the PVA are updated in all the intervals

$[t_{n-1}, t_n]$, $n = 1, 2, \cdots$, with the equal length $T_n = t_n - t_{n-1}$ in the whole navigation time $[t_0, t]$, as is shown in Figure 1.

Figure 1. Intervals associated with strapdown algorithm.



According to the chain rule of matrix product, the updating of the attitude matrix $\mathbf{C}_B^N$ is generally constructed as follows [1,2]:

$$\mathbf{C}_{B_n}^{N_n} = \mathbf{C}_{N_{n-1}}^{N_n} \mathbf{C}_{B_{n-1}}^{N_{n-1}} \mathbf{C}_{B_n}^{B_{n-1}} \tag{1}$$

where $\mathbf{C}_{B_{n-1}}^{N_{n-1}}$ and $\mathbf{C}_{B_n}^{N_n}$ are the attitude matrixes relating the B frame to the N frame at time t_{n-1} and at time t_n, respectively; $\mathbf{C}_{B_n}^{B_{n-1}}$ is the direction cosine matrix that accounts for angular motion of the B frame from time t_{n-1} to time t_n; $\mathbf{C}_{N_{n-1}}^{N_n}$ is the direction cosine matrix that accounts for the N frame rotation frame from time t_{n-1} to time t_n.

According to the velocity rate equation in the N frame [3], the velocity $\mathbf{v}^N$ at the time t_n can be obtained by integrating the specific forces sensed by the accelerometers, the Coriolis accelerations due to the rotations of the navigation and earth frames and the gravity, namely:

$$\mathbf{v}_n^N = \mathbf{v}_{n-1}^N + \Delta\mathbf{v}_{SF_n}^N + \Delta\mathbf{v}_{G/COR_n}^N \tag{2}$$

where $\mathbf{v}_n^N$ and $\mathbf{v}_{n-1}^N$ are the velocity of the system relative to the E frame at time t_n and t_{n-1}, respectively; $\Delta\mathbf{v}_{SF_n}^N$ and $\Delta\mathbf{v}_{G/COR_n}^N$ are the integrated transformed specific force increment and the gravity-Coriolis velocity increment over the updating interval $[t_{n-1}, t_n]$, respectively, calculated by:

$$\Delta\mathbf{v}_{SF_n}^N = \int_{t_{n-1}}^{t_n} \mathbf{C}_B^N \mathbf{f}^B dt \tag{3a}$$

$$\Delta\mathbf{v}_{G/COR_n}^N = \int_{t_{n-1}}^{t_n} \left[\mathbf{g}_P^N - \left(\boldsymbol{\omega}_{EN}^N + 2\boldsymbol{\omega}_{IE}^N \right) \times \mathbf{v}^N \right] dt \tag{3b}$$

Considering the rotation of the navigation frame and the body frame over the updating interval $[t_{n-1}, t_n]$, $\Delta\mathbf{v}_{SF_n}^N$ in Equation (3a) can be expanded according to the chain rule of matrix product as follows:

$$\Delta\mathbf{v}_{SF_n}^N = \int_{t_{n-1}}^{t_n} \mathbf{C}_B^N \mathbf{f}^B dt = \int_{t_{n-1}}^{t_n} \mathbf{C}_{N_{n-1}}^{N_n} \mathbf{C}_{B_{n-1}}^{N_{n-1}} \mathbf{C}_B^{B_{n-1}} \mathbf{f}^B dt = \mathbf{C}_{N_{n-1}}^{N_n} \mathbf{C}_{B_{n-1}}^{N_{n-1}} \Delta\mathbf{v}_{SF_n}^{B_{n-1}} \tag{4}$$

where:

$$\Delta\mathbf{v}_{SF_n}^{B_{n-1}} = \int_{t_{n-1}}^{t_n} \mathbf{C}_B^{B_{n-1}} \mathbf{f}^B dt \tag{5}$$

Because the variation of the position of the system is small over the updating interval $[t_{n-1}, t_n]$ (for example, when the updating interval length T_n is 0.0005 s, the variation of the position of the 6 Mach

Hypersonic cruise missile is only approximately equal to 1.0 m), $\mathbf{g}_P^N$ in Equation (3b) can be approximately by its mean value over the updating interval [t_{n-1}, t_n]. Similarly, the Coriolis term $\left(\boldsymbol{\omega}_{EN}^N + 2\boldsymbol{\omega}_{IE}^N\right) \times \mathbf{v}^N$ in Equation (3b) can also be approximately by its mean value over the updating interval [t_{n-1}, t_n] in view of the small variations of the angular rates $\boldsymbol{\omega}_{IE}^N$ and $\boldsymbol{\omega}_{EN}^N$ as well as the velocity $\mathbf{v}^N$ over the updating interval. Thus, the gravity-Coriolis velocity increment term $\Delta\mathbf{v}_{G/COR_n}^N$ can be approximated by:

$$\begin{aligned} \Delta\mathbf{v}_{G/COR_n}^N &\approx \left\{ \mathbf{g}_{P_{n-1/2}}^N - \left[2\boldsymbol{\omega}_{IE_{n-1/2}}^N + \boldsymbol{\omega}_{EN_{n-1/2}}^N \right] \times \mathbf{v}_{n-1/2}^N \right\} T_n \\ &= \left\{ \mathbf{g}_{P_{n-1/2}}^N - \left[2\mathbf{C}_{E_{n-1/2}}^N \boldsymbol{\omega}_{IE}^E + \mathbf{F}_{C_{n-1/2}}^N \left(\mathbf{u}_{ZN}^N \times \mathbf{v}_{n-1/2}^N \right) + \rho_{ZN_{n-1/2}} \mathbf{u}_{ZN}^N \right] \times \mathbf{v}_{n-1/2}^N \right\} T_n \end{aligned} \tag{6}$$

where $\mathbf{u}_{ZN}^N$ is the unit vector along the Z-axis of the navigation frame, and $\mathbf{u}_{ZN}^N = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$.

According to the altitude and position matrix rate equation [3], the altitude h and position matrix $\mathbf{C}_E^N$ at the time t_n can be obtained as follows:

$$h_n = h_{n-1} + \Delta h_n \tag{7a}$$

$$\mathbf{C}_{E_n}^N = \mathbf{C}_{N_{E_{n-1}}}^{N_{E_n}} \mathbf{C}_{E_{n-1}}^N \tag{7b}$$

where h_n and h_{n-1} are the altitudes at the time t_n and t_{n-1}, respectively; Δh_n is the altitude change from the time t_{n-1} to the time t_n; $\mathbf{C}_{E_n}^N$ and $\mathbf{C}_{E_{n-1}}^N$ are the position matrix at the time t_n and t_{n-1}, respectively; $\mathbf{C}_{N_{E_{n-1}}}^{N_{E_n}}$ is the direction cosine matrix that accounts for the navigation frame rotation relative to the Earth frame from the time t_{n-1} to the time t_n; and:

$$\Delta h_n = \int_{t_{n-1}}^{t_n} \mathbf{u}_{ZN}^N \mathbf{v}^N dt = \mathbf{u}_{ZN}^N \Delta\mathbf{R}_n^N \tag{8}$$

Similar to the attitude matrix updating, $\mathbf{C}_{N_{E_n}}^{N_{E_{n-1}}}$ in Equation (7b) can also be approximated in terms of a rotation vector as follows (accurate to second order):

$$\mathbf{C}_{N_{E_{n-1}}}^{N_{E_n}} \approx \mathbf{I} - (\xi_n \times) + \frac{1}{2}(\xi_n \times)(\xi_n \times) \tag{9}$$

where ξ_n is the rotation vector defining the navigation frame rotation relative to the earth frame from the time t_{n-1} to the time t_n; and ζ_n can be approximately expressed as follows:

$$\xi_n \approx \int_{t_{n-1}}^{t_n} \boldsymbol{\omega}_{EN}^N dt = \mathbf{F}_{C_{n-1/2}}^N \left(\mathbf{u}_{ZN}^N \times \Delta\mathbf{R}_n^N \right) + \rho_{ZN_{n-1/2}} \mathbf{u}_{ZN}^N T_n \tag{10}$$

Note that $\Delta\mathbf{R}_n^N$ should be calculated first to obtain the altitude h and the position matrix $\mathbf{C}_E^N$. Considering that the change of the velocity is small over the updating interval [t_{n-1}, t_n], $\Delta\mathbf{R}_n^N$ can be computed based on a trapezoidal integration algorithm as follows:

$$\Delta\mathbf{R}_n^N \approx \frac{1}{2}\left(\mathbf{v}_n^N + \mathbf{v}_{n-1}^N \right) T_n \tag{11}$$

2.1. Body Frame Rotation Update

The direction cosine matrix $\mathbf{C}_{B_n}^{B_{n-1}}$ in Equation (1) is used to update the attitude matrix $\mathbf{C}_B^N$ which accounts for the angular rate $\boldsymbol{\omega}_{IB}^B$ of the B frame relative to the inertial space. According to the relationship between rotation vector and direction cosine matrix, $\mathbf{C}_{B_n}^{B_{n-1}}$ can be expressed as follows:

$$\mathbf{C}_{B_n}^{B_{n-1}} = \mathbf{I} + \frac{\sin\Phi_n}{\Phi_n}(\boldsymbol{\Phi}_n \times) + \frac{1-\cos\Phi_n}{\Phi_n^2}(\boldsymbol{\Phi}_n \times)(\boldsymbol{\Phi}_n \times) \tag{12}$$

where $\boldsymbol{\Phi}_n$ is the rotation vector that accounts for angular motion of the body frame from time t_{n-1} to time t_n; Φ_n is the magnitude of $\boldsymbol{\Phi}_n$ In practice, $\boldsymbol{\Phi}_n$ can be obtained by the following rotation vector rate equation [4]:

$$\dot{\boldsymbol{\Phi}} \approx \boldsymbol{\omega}_{IB}^B + \frac{1}{2}\boldsymbol{\Phi} \times \boldsymbol{\omega}_{IB}^B + \frac{1}{12}\boldsymbol{\Phi} \times \left(\boldsymbol{\Phi} \times \boldsymbol{\omega}_{IB}^B\right) \tag{13}$$

where $\boldsymbol{\omega}_{IB}^B$ represents the angular rate of the B frame. The last two terms in Equation (13) are non-commutative, thus have to be calculated and compensated based on the gyro incremental angles in order to improve the computation accuracy. The triple-cross-product term in Equation (13) is usually quite small and can be neglected [4]. Then under the second order accuracy, the rotation vector $\boldsymbol{\Phi}_n$ in Equation (12) can be approximated by the integral of Equation (13) from t_{n-1} to t_n, *i.e.*,

$$\boldsymbol{\Phi}_n = \int_{t_{n-1}}^{t_n} \left[\boldsymbol{\omega}_{IB}^B + \frac{1}{2}\left(\boldsymbol{\alpha}(t) \times \boldsymbol{\omega}_{IB}^B\right)\right] dt = \boldsymbol{\alpha}_n + \boldsymbol{\beta}_n \tag{14}$$

where $\boldsymbol{\beta}_n$ is defined as the coning compensation from t_{n-1} to t_n, and:

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}(t_n), \quad \boldsymbol{\alpha}(t) = \int_{t_{n-1}}^{t} \boldsymbol{\omega}_{IB}^B d\tau \tag{15a}$$

$$\boldsymbol{\beta}_n = \boldsymbol{\beta}(t_n), \quad \boldsymbol{\beta}(t) = \frac{1}{2}\int_{t_{n-1}}^{t} \left(\boldsymbol{\alpha}(\tau) \times \boldsymbol{\omega}_{IB}^B\right) d\tau \tag{15b}$$

For a SINS, the coning motion is the worst working condition which will induce serious attitude errors [5-7,18]. In other words, if in the case of coning movement the attitude errors are made minimal by a certain algorithm, the errors in the other cases will also be minimal by the same algorithm. So in order to develop the new strapdown algorithm, it is assumed that the vehicle is undergoing a pure coning movement, defined by the following angular rate:

$$\boldsymbol{\omega}_{IB}^B(t) = \left[a\Omega\cos(\Omega t) \quad b\Omega\sin(\Omega t) \quad 0\right]^T \tag{16}$$

where $\boldsymbol{\Omega}$ is the frequency associated with the coning motion; a and b are the amplitudes of the coning motion.

According to Equations (15a), (15b) and (16), the coning compensation term $\boldsymbol{\beta}_n$ has the following form:

$$\boldsymbol{\beta}_n = \left[0 \quad 0 \quad \frac{ab}{2}(\Omega T_n - \sin\Omega T_n)\right]^T \tag{17}$$

Equation (17) shows an interesting property that the coning compensation is constant over all updating cycles, regardless of the absolute time at which the updating cycle begins. It only depends on the duration of the updating cycle.

According to the concept of distance between the cross products [6,11], the cross products with equal distance behave exactly the same in a pure coning environment defined by Equation (16). The coning compensation that uses the concept will have the simplest form, the optimal accuracy and the minimum computational throughput. Taking the advantage of this property, a generalized optimum algorithm for the integral in Equation (15b) consists of the sum of all possible cross products of the gyro incremental angle samples, making up the computation over the updating interval of rotation vector, such as [9]:

$$\hat{\boldsymbol{\beta}}_n = \sum_{i=1}^{N-1} k_i \boldsymbol{\alpha}_{n-i} \times \boldsymbol{\alpha}_n \tag{18}$$

where N is the number of gyro incremental angle samples for the calculation of the coning compensation term; $\boldsymbol{\alpha}_{n-i}$ (1, 2,…, $N-1$) is the gyro incremental angle sample in the ($n-i$)-*th* updating cycle; k_i (1, 2,…, $N-1$) is the constant coefficients for the cross product of $\boldsymbol{\alpha}_{n-i}$ and $\boldsymbol{\alpha}_n$.

Substituting Equations (15a) and (16) into Equation (18), and expanding each terms using Taylor series, the coning compensation term $\hat{\boldsymbol{\beta}}_n$ over the updating interval [t_{n-1}, t_n] is obtained as:

$$\hat{\boldsymbol{\beta}}_n = \begin{bmatrix} 0 & 0 & ab\sum_{i=1}^{\infty}(-1)^{i+1}\sum_{j=1}^{N-1} A_{ij}K_j\lambda^{2i+1} \end{bmatrix}^T \tag{19}$$

where $\lambda = \Omega T_n$; A_{ij} is a constant defined by:

$$A_{ij} = \frac{(j+1)^{2i+1} - 2\bullet j^{2i+1} + (j-1)^{2i+1}}{(2i+1)!}, i = 1,2,\cdots,\infty, j = 1,2,\cdots,N-1 \tag{20}$$

In order to derive k_i (1, 2,…, $N-1$) in Equation (18), expanding Equation (17) by using Taylor series yields:

$$\boldsymbol{\beta}_n = \begin{bmatrix} 0 & 0 & ab\sum_{i=1}^{\infty}(-1)^{i+1} C_i\lambda^{2i+1} \end{bmatrix}^T \tag{21}$$

where C_i is a constant defined by:

$$C_i = \frac{1}{(2i+1)!\times 2} \tag{22}$$

Combining Equation (19) with Equation (21), the following simultaneous equations for constant coefficients K_i, $i = 1, 2,…, N-1$), can be obtained:

$$\sum_{j=1}^{N-1} A_{ij}K_j = C_i, i = 1,2,\cdots,N-1 \tag{23}$$

In a matrix form, Equation (23) is equivalent to:

$$\left[A_{ij}\right]_{(N-1)\times(N-1)} \cdot \left[K_j\right]_{(N-1)\times 1} = \left[C_i\right]_{(N-1)\times 1} \tag{24}$$

According to Equation (24), coefficients K_i can be solved as follows:

$$\left[K_j\right]_{(N-1)\times 1} = \left[A_{ij}\right]_{(N-1)\times(N-1)}^{-1} \cdot \left[C_i\right]_{(N-1)\times 1} \tag{25}$$

where $\left[x_i\right]_{m\times 1}$ and $\left[x_{ij}\right]_{m\times n}$ are m-dimensional column vector and the m-by-n matrix, respectively.

Note that different from other existing algorithms, the updating rate of the proposed optimal coning compensation algorithm is independent of the number of gyro incremental angle samples in the calculation of the coning compensation. Thus, this algorithm allows the updating speed to be increased, at the same time increasing the number of gyro incremental angle samples, in order to improve the attitude accuracy of the system.

2.2. Navigation Frame Rotation Update

The direction cosine matrix $\mathbf{C}_{N_{n-1}}^{N_n}$ in Equation (1) is used to update the attitude matrix $\mathbf{C}_B^N$ which accounts for the angular rate $\boldsymbol{\omega}_{IN}^N$ of the N frame relative to the inertial space. Similar to the computation of $\mathbf{C}_{B_n}^{B_{n-1}}$, according to the relationship between rotation vector and direction cosine matrix, $\mathbf{C}_{N_{n-1}}^{N_n}$ can also be expressed in a second order form as follows:

$$\left[K_j\right]_{(N-1)\times 1} = \left[A_{ij}\right]_{(N-1)\times(N-1)}^{-1} \cdot \left[C_i\right]_{(N-1)\times 1} \tag{25}$$

$$\mathbf{C}_{N_{n-1}}^{N_n} = \mathbf{I} - \frac{\sin\zeta_n}{\zeta_n}(\boldsymbol{\zeta}_n\times) + \frac{1-\cos\zeta_n}{\zeta_n^2}(\boldsymbol{\zeta}_n\times)(\boldsymbol{\zeta}_n\times) \approx \mathbf{I} - (\boldsymbol{\zeta}_n\times) + \frac{1}{2}(\boldsymbol{\zeta}_n\times)(\boldsymbol{\zeta}_n\times) \tag{26}$$

where $\boldsymbol{\zeta}_n$ is the rotation vector that accounts for the angular motion of the N frame from time $t_{n\text{-}1}$ to time t_n; ζ_n is the magnitude of $\boldsymbol{\zeta}_n$.

Because the updating interval length T_n is short (generally equal to 0.0005 s–0.01 s), the angular rate $\boldsymbol{\omega}_{IN}^N$ is small and changes slowly over the updating interval [$t_{n\text{-}1}$, t_n] (due to the small changes in velocity and position over this updating cycle). Then according to the rotation vector rate equation, $\boldsymbol{\zeta}_n$ can be approximated as follows:

$$\boldsymbol{\zeta}_n \approx \int_{t_{n-1}}^{t_n} \boldsymbol{\omega}_{IN}^N dt = \int_{t_{n-1}}^{t_n} \left(\mathbf{C}_E^N \boldsymbol{\omega}_{IE}^E + \boldsymbol{\omega}_{EN}^N\right) dt \approx \mathbf{C}_{E_{n-1/2}}^N \boldsymbol{\omega}_{IE}^E T_n + \mathbf{F}_{C_{n-1/2}}^N \left(\mathbf{u}_{ZN}^N \times \Delta\mathbf{R}_{n-1/2}^N\right) + \rho_{ZN_{n-1/2}} \mathbf{u}_{ZN}^N T_n \tag{27}$$

where $\boldsymbol{\omega}_{IE}^E$ and $\boldsymbol{\omega}_{EN}^N$ are the angular rates of the earth frame relative to the inertial frame and the navigation frame relative to the earth frame, respectively; $\mathbf{C}_B^N$ is the position matrix relating the earth frame with the navigation frame; the subscript n − 1/2 indicates the midpoint of the updating interval [$t_{n\text{-}1}$, t_n].

2.3. Integrated Specific Force Increment Update

Similar to the attitude updating algorithm, the integral term $\Delta\mathbf{v}_{SF_n}^{B_{n-1}}$ in Equation (4) can be formulated based on the first order approximation of $\mathbf{C}_B^{B_{n-1}}$ as follows:

$$\Delta\mathbf{v}_{SF_n}^{B_{n-1}} \approx \int_{t_{n-1}}^{t_n} \left[\mathbf{I} + \left(\boldsymbol{\alpha}(t)\times\right)\right]\mathbf{f}^B dt = \boldsymbol{\upsilon}_n + \int_{t_{n-1}}^{t_n} \left(\boldsymbol{\alpha}(t)\times\mathbf{f}^B\right) dt \tag{28}$$

where:

$$\boldsymbol{\upsilon}_n = \boldsymbol{\upsilon}(t_n), \quad \boldsymbol{\upsilon}(t) = \int_{t_{n-1}}^{t} \mathbf{f}^B d\tau \tag{29}$$

The integrand term $\boldsymbol{\alpha}(t)\times\mathbf{f}^B$ in Equation (28) has the following expression [3]:

$$\boldsymbol{\alpha}(t)\times\mathbf{f}^B = \frac{1}{2}\frac{d}{dt}\left(\boldsymbol{\alpha}(t)\times\boldsymbol{\upsilon}(t)\right) + \frac{1}{2}\left(\boldsymbol{\alpha}(t)\times\mathbf{f}^B + \boldsymbol{\upsilon}(t)\times\boldsymbol{\omega}_{IB}^B\right) \tag{30}$$

$\Delta \mathbf{v}_{SF_n}^{B_{n-1}}$ can also be expressed as follows:

$$\Delta \mathbf{v}_{SF_n}^{B_{n-1}} = \boldsymbol{\upsilon}_n + \frac{1}{2}\left(\boldsymbol{\alpha}_n \times \boldsymbol{\upsilon}_n\right) + \frac{1}{2}\int_{t_{n-1}}^{t_n} \left(\boldsymbol{\alpha}(t) \times \mathbf{f}^B + \boldsymbol{\upsilon}(t) \times \boldsymbol{\omega}_{IB}^B\right) dt = \boldsymbol{\upsilon}_n + \Delta \mathbf{v}_{Rot_n} + \Delta \mathbf{v}_{Scul_n} \tag{31}$$

where $\Delta \mathbf{v}_{Rot_n}$ denotes the velocity rotation compensation term; $\Delta \mathbf{v}_{Scul_n}$ denotes the sculling compensation term; and:

$$\Delta \mathbf{v}_{Rot_n} = \frac{1}{2}\left(\boldsymbol{\alpha}_n \times \boldsymbol{\upsilon}_n\right) \tag{32a}$$

$$\Delta \mathbf{v}_{Scul_n} = \frac{1}{2}\int_{t_{n-1}}^{t_n} \left(\boldsymbol{\alpha}(t) \times \mathbf{f}^B + \boldsymbol{\upsilon}(t) \times \boldsymbol{\omega}_{IB}^B\right) dt \tag{32b}$$

In principle, the approaches used for the coning compensation can also be applied to the sculling compensation. Similar to the optimal generalized coning compensation algorithm in Equation (18), a generalized sculling compensation algorithm that has the simplest form, the optimal accuracy and the minimum computational throughput takes the following form:

$$\Delta \hat{\mathbf{v}}_{Scul_n} = \left[\sum_{i=1}^{N-1} L_i \boldsymbol{\alpha}_{n-i}\right] \times \boldsymbol{\upsilon}_n + \left[\sum_{i=1}^{N-1} L_i \boldsymbol{\upsilon}_{n-i}\right] \times \boldsymbol{\alpha}_n \tag{33}$$

where N is the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples for the calculation of the sculling compensation term; $\boldsymbol{\alpha}_{n-i}$ $(1, 2,\ldots, N-1)$ and $\boldsymbol{\upsilon}_{n-i}$ $(1, 2,\ldots, N-1)$ are the gyro incremental angle and accelerometer incremental velocity in the $(n-i)$*-th* updating cycle; L_i $(1, 2,\ldots, N-1)$ is the constant coefficients for the cross product of $\boldsymbol{\alpha}_{n-i}$ with $\boldsymbol{\upsilon}_n$ and $\boldsymbol{\upsilon}_{n-i}$ with $\boldsymbol{\alpha}_n$.

Considering the equivalency between the coning compensation and sculling compensation [12,23], similar to Equation (25), the coefficients L_i can also be calculated as follows:

$$\left[L_j\right]_{(N-1)\times 1} = \left[A_{ij}\right]_{(N-1)\times(N-1)}^{-1} \cdot \left[C_i\right]_{(N-1)\times 1} \tag{34}$$

where A_{ij} and C_i can be calculated according to Equations (20) and (22), respectively.

Note that different from other existing algorithms, the updating rate of the proposed optimal sculling compensation algorithm is also independent of the number of gyro incremental angle samples and accelerometer incremental velocity samples. Thus, this algorithm allows the updating speed to be increased, at the same time increasing the number of gyro incremental angle samples and accelerometer incremental velocity samples, in order to improve the accuracy of the related velocity updating algorithm.

2.4. Related Parameters Extrapolation Update

Because the gravity anomaly and the vertical deviation over the earth surface resulting from mass irregular and shape asymmetric distributions are generally small (the maximum value of the gravity anomaly is only tens of mgal; and the maximum value of the vertical deviation is only tens of arcs), the following approximate model of the gravity is used for most SINS applications:

$$g(L,h) = g_0(L)\frac{R_0^2}{\left(h+R_0\right)^2} \approx g_0(L)\left(1-\frac{2h}{R_0}\right) \tag{35}$$

where $R_0 = \sqrt{R_M R_N}$, and g_0 is approximated by the following formula based on the WGS-84 data [24-27]:

$$g_0(L) \approx 9.7803253359 \frac{\left(1 + 0.001931853 \sin^2 L\right)}{\sqrt{1 - 0.006694380 \sin^2 L}} \tag{36}$$

Then the gravity $\mathbf{g}_P^N$ in the navigation frame can be expressed as follows:

$$\mathbf{g}_P^N = \begin{bmatrix} 0 & 0 & -g(L,h) \end{bmatrix}^T \tag{37}$$

For the conventional ellipsoidal earth surface model and the E-N-U navigation frame [1,3], $\boldsymbol{\omega}_{IE}^E$, $\boldsymbol{\omega}_{EN}^N$ and $\mathbf{C}_E^N$ can also be expressed in the following form:

$$\boldsymbol{\omega}_{IE}^E = \begin{bmatrix} 0 & 0 & \omega_{IE} \end{bmatrix}^T \tag{38a}$$

$$\boldsymbol{\omega}_{EN}^N = \mathbf{F}_C^N \left(\mathbf{u}_{ZN}^N \times \mathbf{v}^N\right) + \rho_{ZN} \mathbf{u}_{ZN}^N \tag{38b}$$

$$\begin{aligned} \mathbf{C}_E^N &= \begin{bmatrix} -\cos\alpha\sin\lambda - \sin\alpha\sin L\cos\lambda & \cos\alpha\cos\lambda - \sin\alpha\sin L\sin\lambda & \sin\alpha\cos L \\ \sin\alpha\sin\lambda - \cos\alpha\sin L\cos\lambda & -\sin\alpha\cos\lambda - \cos\alpha\sin L\sin\lambda & \cos\alpha\cos L \\ \cos L\cos\lambda & \cos L\sin\lambda & \sin L \end{bmatrix} \\ &= \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \end{aligned} \tag{38c}$$

with:

$$\mathbf{F}_C^N = \begin{bmatrix} \left(\frac{1}{R_N + h} - \frac{1}{R_M + h}\right)\sin\alpha\cos\alpha & -\left(\frac{\sin^2\alpha}{R_N + h} + \frac{\cos^2\alpha}{R_M + h}\right) & 0 \\ \left(\frac{\cos^2\alpha}{R_N + h} + \frac{\sin^2\alpha}{R_M + h}\right) & \left(-\frac{1}{R_N + h} + \frac{1}{R_M + h}\right)\sin\alpha\cos\alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{39a}$$

$$\begin{aligned} R_M &= \frac{R_e(1-e)^2}{\left[1 - (2-e)^2 \sin^2 L\right]^{\frac{3}{2}}} \approx R_e\left(1 + 2e - 3e\sin^2 L\right) \\ R_N &= \frac{R_e}{\left[1 - (2-e)e\sin^2 L\right]^{\frac{1}{2}}} \approx R_e\left(1 + e\sin^2 L\right) \end{aligned} \tag{39b}$$

$$\sin^2 L = \left(C_{33}\right)^2, \sin\alpha\cos\alpha = \frac{C_{13}C_{23}}{1 - \left(C_{33}\right)^2}, \sin^2\alpha = \frac{\left(C_{13}\right)^2}{1 - \left(C_{33}\right)^2}, \cos^2\alpha = \frac{\left(C_{23}\right)^2}{1 - \left(C_{33}\right)^2} \tag{39c}$$

where $\mathbf{F}_C^N$ is the curvature matrix in the navigation frame that is a function of position over the earth; ρ_{ZN} is the Z-axis component of $\boldsymbol{\omega}_{EN}^N$; $\mathbf{v}^N$ is the velocity vector of the vehicle relative to the earth projected on the navigation frame; R_M and R_N are the radii of curvature at the earth surface in meridian and in prime vertical, respectively; h is the height; α is the wander angle; L is the latitude; R_e is the equatorial radius of earth; e is the flattening of earth; C_{ij} is the i-row and j-column component of $\mathbf{C}_E^N$.

By the definition of the navigation frame, the orientation of the X axis and the Y axis around the Z axis is somewhat arbitrary. So ρ_{ZN} in Equation (38b) depends on the selection of the axes orientation of the navigation frame. The navigation frame is generally selected as a wander azimuth navigation frame for most SINSs [3]. In this case, ρ_{ZN} is given by:

$$\rho_{ZN} = 0 \tag{40}$$

Because the related parameters ( ) (namely, $\mathbf{g}_P^N$, $\mathbf{F}_C^N$, ρ_{ZN}, $\mathbf{v}^N$, $\Delta\mathbf{R}^N$ and $\mathbf{C}_E^N$) in Equations (6), (10) and (27) are all the functions of position or velocity, and the values of these parameters at the current cycle are not available, thus the $(\ )_{n-1/2}$ terms can be approximately calculated based on the linear extrapolation formula in the following form:

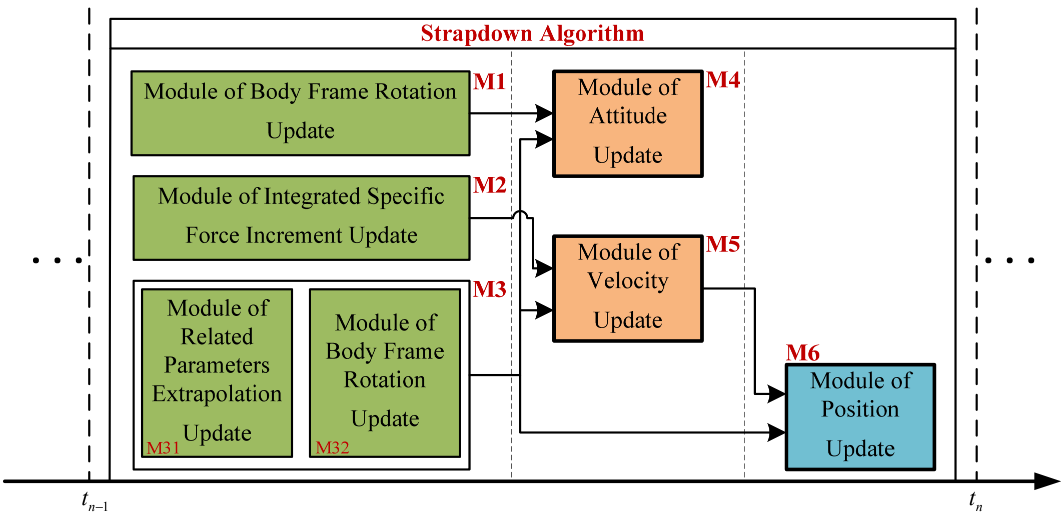
$$(\ )_{n-1/2} \approx (\ )_{n-1} + \frac{1}{2}\left[(\ )_{n-1} - (\ )_{n-2}\right] = \frac{3}{2}(\ )_{n-1} - \frac{1}{2}(\ )_{n-2} \tag{41}$$

3. Strapdown Algorithm Parallelization

Although the sampling rate of inertial sensors can be up to 2 kHz or even higher, taking into account the complexity of the employed strapdown algorithm and the ability of the current processors, the updating rate of the strapdown algorithm in a serial mode is limited, usually only 200–500 Hz when implemented in a DSP. An effective way to break through the limitation of commonly used navigation computers is to implement the strapdown algorithm on a purely parallel computing platform such as FPGA, and execute the calculations in the algorithm "as concurrently as possible" to make full use of the capability of the parallel computing platform.

The strapdown algorithm proposed in Section 2 can be divided into six modules doing the following calculations severally: the body frame rotation update (M1), the integrated specific force increment update (M2), the related parameters extrapolation update and the navigation frame rotation update (M3), the attitude update (M4), the velocity update (M5) and the position update (M6), as shown in Figure 2. Among them, M1, M2 and M3 can be executed first in a parallel mode; M4 and M5 have to be executed afterwards but also in a parallel mode; finally, M6 is executed.

Figure 2. Functional-block diagram of parallel strapdown algorithm.



3.1. Module of Body Frame Rotation Update (M1)

The calculations involved in the module M1 are described by Equations (12), (14) and (18), as shown in Figure 3. Once the number of the gyro incremental angle samples N is selected, the corresponding coefficients K_i of the coning compensation can firstly be determined offline by Equations (20), (22) and (25) and stored in the memory of the navigation computer for online use; secondly, the coning compensation $\boldsymbol{\beta}_n$ and rotation vector $\boldsymbol{\Phi}_n$ can be successively computed according to Equations (14) and (18), respectively; finally, the direction cosine matrix $\mathbf{C}_{B_n}^{B_{n-1}}$ that accounts for the angular motion of the B frame is obtained by Equation (12). Note that since the accuracy of the coning compensation directly determines the attitude accuracy of the system, particularly in high dynamic conditions, the coning compensation is generally designed to accurately account for the vibration induced coning effects by selecting the appropriate number of the gyro incremental angle samples.

Figure 3. Diagram of module M1.

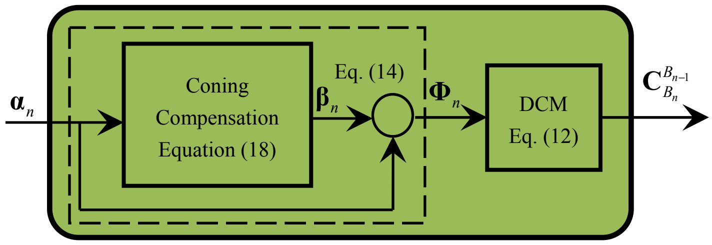


Figure 3 shows that Equations (11), (14) and (18) can only be executed in a serial mode. From Equations (12), (14) and (18), it is shown that M1 contains the following operations of related minimum calculation particles: cross-product of vectors, product of a skew symmetric matrix with itself, addition of matrixes or vectors and calculation of sine or cosine function. The operations within these minimum calculation particles can be further concurrently processed, which will be discussed in Section 3.2.

3.2. Module of Integrated Specific Force Increment Update (M2)

The module M2 carries out the calculations defined by Equations (31), (32a) and (33), as shown in Figure 4. Similar to the coning compensation, once the number of the gyro incremental angle and accelerometer incremental velocity samples N is selected, the corresponding coefficients L_i of the sculling compensation can firstly be determined offline by Equations (20), (22) and (34) and stored in the memory of the navigation computer for online use; secondly, the sculling compensation $\Delta\mathbf{v}_{Scul_n}$ and the velocity rotation compensation $\Delta\,\mathbf{v}_{Rot_n}$ can be successively computed according to Equations (32a) and (33); finally, the integrated specific force increment $\Delta\mathbf{v}_{SF_n}^{B_{n-1}}$ that accounts for the linear motion of the B frame is calculated by Equation (31). Note that since the accuracy of the sculling compensation directly determines the velocity accuracy of the system, particularly in high dynamic conditions, the sculling compensation is generally designed to accurately account for the vibration induced sculling effects by selecting the appropriate number of the gyro incremental angle and accelerometer incremental velocity samples.

Figure 4. Diagram of module M2.

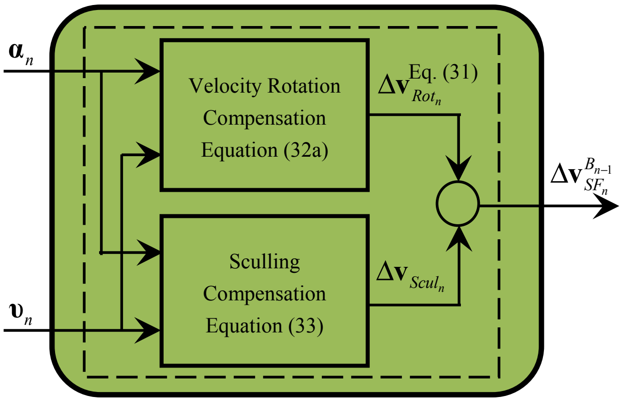


Figure 4 shows that the calculations of the velocity rotation compensation and the sculling compensation can be executed in a parallel mode according to Equations (32a) and (33). From Equations (31), (32a) and (33), it is shown that M2 contains the following operations of related minimum calculation particles: cross-product of vectors and addition of vectors. Similar to the operations within these minimum calculation particles in M1, the operations within these minimum calculation particles can be further processed concurrently, which will be discussed in Section 3.2.

3.3. Module of Related Parameters Extrapolation and Navigation Frame Rotation Update (M3)

The module M3 can be further divided into two serial-process modules: the module of related parameters extrapolation update (M31) and the module of navigation frame rotation update (M32), as shown in Figure 5. The computation tasks completed in M31 and M32 are described by Equations (26), (27) and (35–41) respectively.

Figure 5. Diagram of module M3.

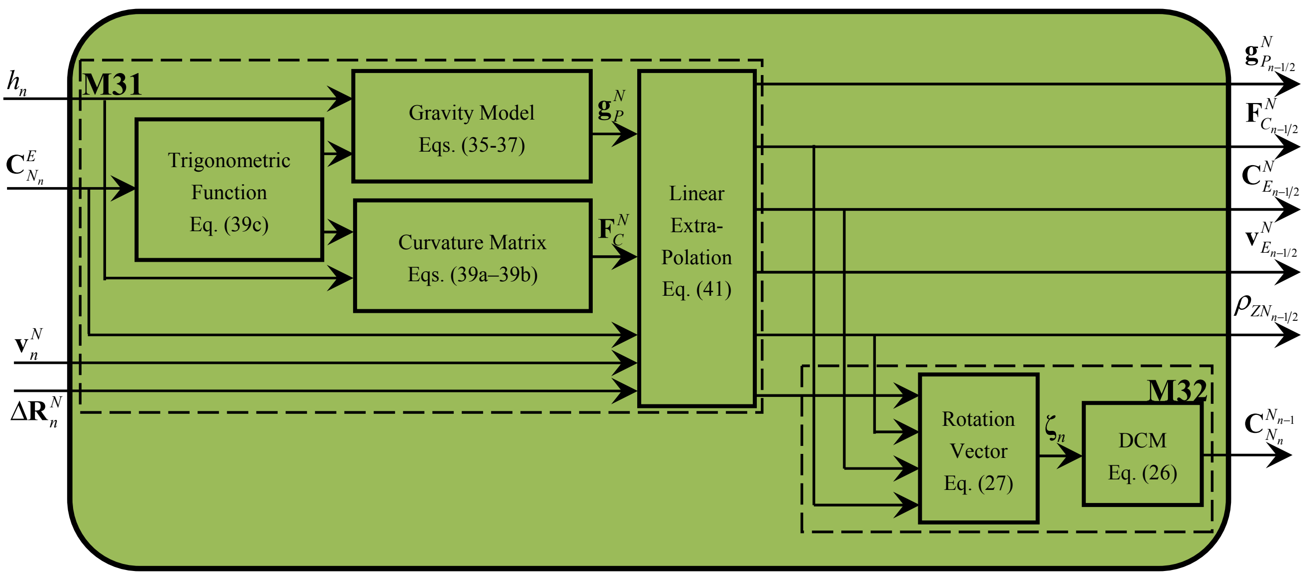
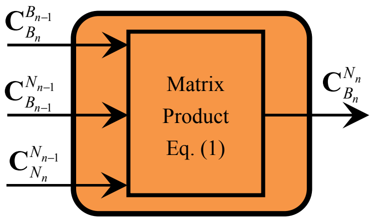


Figure 5 shows that in M31, the calculation of the gravity and the curvature matrix can be firstly executed in a parallel mode according to Equations (35–39), and then the related parameters ($\mathbf{g}_{P_{n-1/2}}^{N}$, $\mathbf{v}_{E_{n-1/2}}^{N}$, $\mathbf{F}_{C_{n-1/2}}^{N}$, $\rho_{ZN_{n-1/2}}$, $\Delta\mathbf{R}_{n-1/2}^{N}$ and $\mathbf{C}_{E_{n-1/2}}^{N}$) can also be calculated in a parallel mode according to Equation (41); in M32, Equations (26) and (27) can only be executed in a serial mode. From Equations (26), (27) and Equations (35–41), it is shown that M31 and M32 contain the following operations of related minimum calculation particles: cross-product of vectors, product of a skew symmetric matrix with its own and addition of matrixes or vectors. Similar to the operations within these minimum calculation particles in module M1, the operations within these minimum calculation particles can be further processed concurrently, which will be discussed in Section 3.2.

3.4. Module of Attitude Update (M4)

The module M4 is used to calculate Equation (1), as shown in Figure 6.

Figure 6. Diagram of module M4.



From Equation (1), it is shown that M4 only contains the following operation of related minimum calculation particles: product of matrixes. Similar to the operations within these minimum calculation particles in module M1, the product operation of matrixes can be further processed concurrently, which will be discussed in Section 3.2.

3.5. Module of Velocity Update (M5)

The calculations implemented in the module M5 are defined by Equations (2), (4) and (6), as shown in Figure 7.

Figure 7. Diagram of module M5.

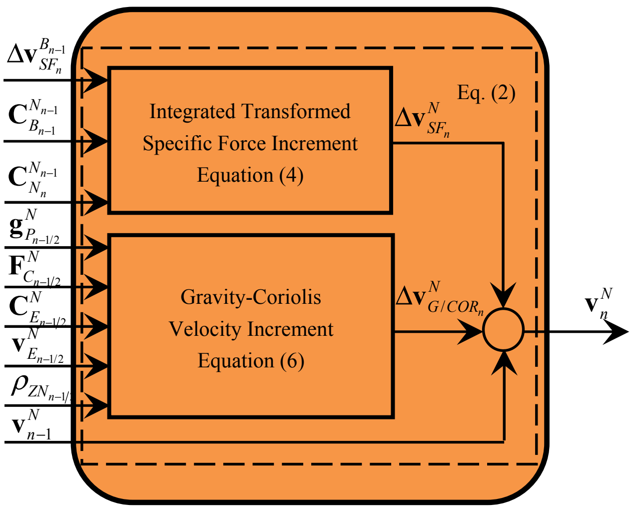


Figure 7 shows that the calculation of the integrated transformed specific force increment and the gravity-Coriolis velocity increment can be executed in a parallel mode according to Equations (4) and (6). From Equations (2), (4) and (6), it is shown that M5 contains the following operations of related minimum calculation particles: cross-product of vectors, product of a skew symmetric matrix with its own, product of matrixes or a matrix with a vector and addition of vectors. Similar to the operations within these minimum calculation particles in module M1, the operations within these minimum calculation particles can be further parallelized, which will be discussed in Section 3.2.

3.6. Module of Position Update (M6)

In the module M6, the computations defined by Equations (7–11) are carried out, as shown in Figure 8.

Figure 8. Diagram of module M6.

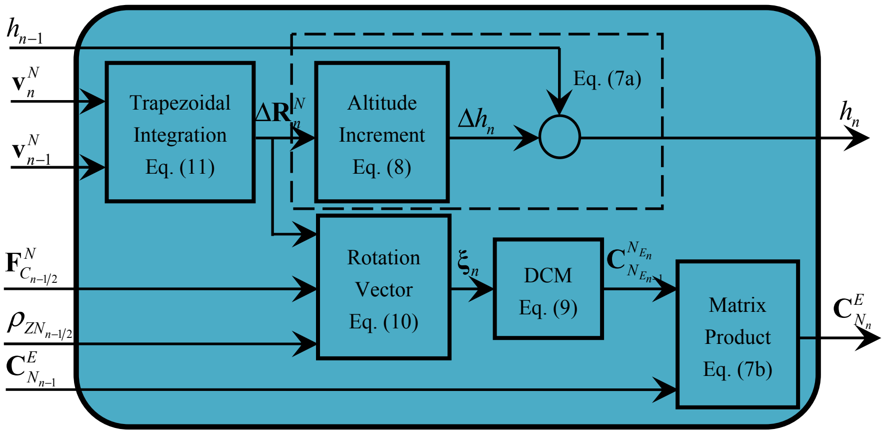


Figure 8 shows that after the calculation of $\Delta\mathbf{R}_n^N$, the calculation of the altitude h and the position matrix $\mathbf{C}_E^N$ can be executed in a parallel mode according to Equations (7–10), in which Equations (7b), (9) and (10) can only be executed in a serial mode, and Equations (7a) and (8) can also only be executed in a serial mode. From Equations (7–11), it is shown that M6 contains the following operations of related minimum calculation particles: cross-product of vectors, product of a matrix with a vector and addition of vectors. Similar to the operations within these minimum calculation particles in module M1, the operations within these minimum calculation particles can be further parallelized, which will be discussed in Section 3.2.

4. Computation Complexity Analysis

From Figure 2, it can be shown that the execution time of the parallel strapdown algorithm on FPGA can be as follows:

$$T_{Nav} = \max\{T_{M1}, T_{M2}, T_{M3}\} + \max\{T_{M4}, T_{M5}\} + T_{M6} \tag{42}$$

with:

$$T_{M3} = T_{M31} + T_{M32} \tag{43}$$

where, T_{M1}, T_{M2}, T_{M3}, T_{M4}, T_{M5} and T_{M6} are the execution time required by the modules M1, M2, M3, M4, M5 and M6, respectively; T_{M31} and T_{M32} are the execution time required by the modules M31 and M32, respectively.

In order to make the updating cycle of the strapdown algorithm shortest, the maximum parallelism degree is usually used as a performance index to optimize the calculation particles involved in the modules M1–M6.

Assume that the execution time of addition (subtraction), multiplication, division, trigonometric and square root operation are defined as T_A, T_M, T_D, T_T and T_S, respectively. The summation of 3-dimensional vectors, for instance by:

$$\mathbf{V}_{VA} = \mathbf{V}_1 + \mathbf{V}_2 \tag{44}$$

contains three addition operations which can be executed in a parallel mode. Thus the execution time of the addition operation for two 3-dimensional vectors is

$$T_{VA} = T_A \tag{45}$$

The addition of two 3-by-3 matrixes, for instance by:

$$\mathbf{C}_{MA} = \mathbf{C}_1 + \mathbf{C}_2 \tag{46}$$

contains nine addition operations which can also be executed in a parallel mode. Thus the execution time of the addition operation for two 3-by-3 matrixes is

$$T_{MA} = T_A \tag{47}$$

The cross-product of two 3-dimensional vectors, expressed for instance by:

$$\mathbf{V}_{VCP} = \mathbf{V}_1 \otimes \mathbf{V}_2 = \begin{bmatrix} 0 & -V_{1Z} & V_{1Y} \\ V_{1Z} & 0 & -V_{1X} \\ -V_{1Y} & V_{1X} & 0 \end{bmatrix} \begin{bmatrix} V_{2X} \\ V_{2Y} \\ V_{2Z} \end{bmatrix} = \begin{bmatrix} V_{1Y}V_{2Y} - V_{1Z}V_{2X} \\ V_{1Z}V_{2X} - V_{1X}V_{2Z} \\ V_{1X}V_{2Y} - V_{1Y}V_{2X} \end{bmatrix} \tag{48}$$

contains six multiplication operations and three subtraction operations. All the multiplication operations or the subtraction operations can be executed in a parallel mode, but the subtraction operations must be executed after the multiplication operations. Thus the execution time of the cross-product operation for two 3-dimensional vectors is:

$$T_{VCP} = T_M + T_A \tag{49}$$

The product of two 3-by-3 matrixes, for instance by:

$$\mathbf{C}_C^A = \mathbf{C}_B^A \mathbf{C}_C^B \tag{50}$$

contains 27 multiplication operations and 18 addition operations. The multiplication operations can be executed in a parallel mode, but the addition operations must be executed twice in a parallel mode after the multiplication operations. Thus the execution time of the product operation for two 3-by-3 matrixes is:

$$T_{MP} = T_M + 2T_A \tag{51}$$

The product of a 3-by-3 matrix with a 3-dimensional vector, which can be defined for instance by:

$$\mathbf{V}^A = \mathbf{C}_B^A \mathbf{V}^B \tag{52}$$

contains nine multiplication operations and 6 addition operations. The multiplication operations can be executed in a parallel mode, but the addition operations must be executed twice in a parallel mode after the multiplication operations. Thus the execution time of the product operation for a 3-by-3 matrix with a 3-dimensional vector is:

$$T_{MVP} = T_M + 2T_A \tag{53}$$

The product of a skew symmetric matrix with itself defined, for instance by:

$$\begin{aligned}\mathbf{C} &= (\mathbf{V}_1 \otimes)(\mathbf{V}_1 \otimes) \\ &= \begin{bmatrix} 0 & -V_{1Z} & V_{1Y} \\ V_{1Z} & 0 & -V_{1X} \\ -V_{1Y} & V_{1X} & 0 \end{bmatrix} \begin{bmatrix} 0 & -V_{1Z} & V_{1Y} \\ V_{1Z} & 0 & -V_{1X} \\ -V_{1Y} & V_{1X} & 0 \end{bmatrix} = \begin{bmatrix} -V_{1Z}^2 - V_{1Y}^2 & V_{1X}V_{1Y} & V_{1X}V_{1Z} \\ V_{1X}V_{1Y} & -V_{1Z}^2 - V_{1X}^2 & V_{1Y}V_{1Z} \\ V_{1X}V_{1Z} & V_{1Y}V_{1Z} & -V_{1Y}^2 - V_{1X}^2 \end{bmatrix}\end{aligned} \tag{54}$$

contains six multiplication operations and three addition operations. The multiplication operations or the addition operations can be executed in a parallel mode, but the addition operations must be executed after the multiplication operations. Thus the execution time of the product operation for a skew symmetric matrix with itself is:

$$T_{VCP2} = T_M + T_A \tag{55}$$

Based on the aforementioned execution time analysis of the basic computational operations involved in the modules M1–M6, we can evaluate the computational complexity of each module.

4.1. Analysis of Module M1

In the calculation of the coning compensation term $\boldsymbol{\beta}_n$ defined by Equation (18), the $N - 1$ vectors cross-product operation can be executed in a parallel mode, and the summation of $N - 1$ vectors can also be successively executed in a parallel mode $[\log_2(N-1)]$ times. Thus the execution time of the coning compensation is:

$$T_{Con} = T_{VCP} + T_M + \lceil \log_2(N-1) \rceil T_{VA} = 2T_M + \left[\lceil \log_2(N-1) \rceil + 1 \right] T_A \tag{56}$$

And in the calculation of the direction cosine matrix $\mathbf{C}_{B_n}^{B_{n-1}}$ according to Equation (12), the calculation of Φ_n^2, Φ_n and $(\boldsymbol{\Phi}_n \times)(\boldsymbol{\Phi}_n \times)$ can be executed in a parallel mode. Thus the execution time of the matrix $\mathbf{C}_{B_n}^{B_{n-1}}$ calculation is:

$$T_{Rot2DCM} = T_M + 2T_A + T_S + T_T + T_A + T_D + T_M + 2T_A = 2T_M + 4T_A + T_D + T_S + T_T \tag{57}$$

Based on Equations (56) and (57) and refer to Figure 3, the execution time of module M1 can be obtained as follows:

$$T_{M1} = T_{Con} + T_{VA} + T_{Rot2DCM} = 4T_M + \left[\lceil \log_2(N-1) \rceil + 6 \right] T_A + T_D + T_S + T_T \tag{58}$$

4.2. Analysis of Module M2

Figure 5 shows that the calculation of the velocity rotation compensation and the sculling compensation can be executed in a parallel mode. According to Equation (48), the execution time of the velocity rotation compensation is:

$$T_{Rot} = T_{VCP} + T_M = 2T_M + T_A \quad (59)$$

Similar to the calculation of the coning compensation term $\boldsymbol{\beta}_n$, in the calculation of the sculling compensation term $\Delta\mathbf{v}_{Scul_n}$ defined by Equation (33), the cross-product operations of the $2(N-1)$ vectors can be executed in a parallel mode, and the summation of $N-1$ vectors can also be successively executed in a parallel mode $[\log_2(N-1)]$ times. Thus the execution time of the coning compensation is:

$$T_{scul} = T_{VCP} + T_M + \lceil \log_2(N-1) \rceil T_{VA} = 2T_M + \left[\lceil \log_2(N-1) \rceil + 1\right] T_A \quad (60)$$

Based on Equations (59) and (60) and referring to Figure 4, the execution time of module M2 can be obtained as follows:

$$T_{M2} = \max\{T_{Rot}, T_{scul}\} + 2T_{VA} = 2T_M + \left[\lceil \log_2(N-1) \rceil + 3\right] T_A \quad (61)$$

4.3. Analysis of Module M3

Figure 6 shows that the calculation of the gravity and the curvature matrix can be executed in a parallel mode. According to Equations (35–37), the execution time of the gravity calculation is:

$$T_G = \max\{T_{G1}, T_{G2}, T_{G3}\} + 2T_M + T_D \quad (62)$$

where T_{G1}, T_{G2} and T_{G3} are the execution time of the calculations for the terms $\left(1 + 0.001931853\sin^2 L\right)$, $\sqrt{1 - 0.006694380\sin^2 L}$ and $\left(1 - \frac{2h}{R_0}\right)$, respectively; and:

$$T_{G1} = T_M + T_A \quad (63a)$$

$$T_{G2} = T_M + T_A + T_S \quad (63b)$$

$$T_{G3} = T_M + T_A + T_D \quad (63c)$$

According to Equation (39), the execution time of the curvature matrix calculation is:

$$T_F = \max\{T_{F1}, T_{F2}\} + T_A + T_D + 2T_M + T_A \quad (64)$$

where T_{F1} and T_{F1} are the execution time of the calculations for R_M and R_N, respectively; and:

$$T_{F1} = 3T_M + 2T_A \quad (65a)$$

$$T_{F2} = 2T_M + T_A \quad (65b)$$

Thus the execution time of module M31 can be obtained as follows:

$$T_{M31} = T_M + T_A + T_M + T_D + \max\{T_G, T_F\} + T_M + T_A \quad (66)$$

According to Equations (26) and (27), the execution time of module M32 can be estimated by:

$$T_{M32} = \left(T_M + T_{MVP} + T_A\right) + \left(T_{VCP2} + T_M + T_A\right) = 4T_M + 5T_A \quad (67)$$

Based on Equations (66) and (67), and refer to Figure 6, the total execution time of module M3 is:

$$T_{M3} = T_{M31} + T_{M32} \quad (68)$$

4.4. Analysis of Module M4

According to Equation (1), the execution time of module M4 is as follows:

$$T_{M4} = 2T_{MP} = 2T_M + 4T_A \tag{69}$$

4.5. Analysis of Module M5

Figure 8 shows that the calculation of the integrated transformed specific force increment and the gravity-Coriolis velocity increment can be executed in a parallel mode. According to Equation (4), the execution time of the integrated transformed specific force increment calculation is:

$$T_{SF} = T_{MP} + T_{MVP} = 2T_M + 4T_A \tag{70}$$

According to Equation (6), the execution time of the gravity-Coriolis velocity increment calculation is:

$$T_{G-C} = T_{MVP} + T_M + T_A + T_{VCP} + T_A + T_M = 4T_M + 5T_A \tag{71}$$

Based on Equations (70) and (71), and refer to Figure 8, the execution time of module M5 can be obtained as follows:

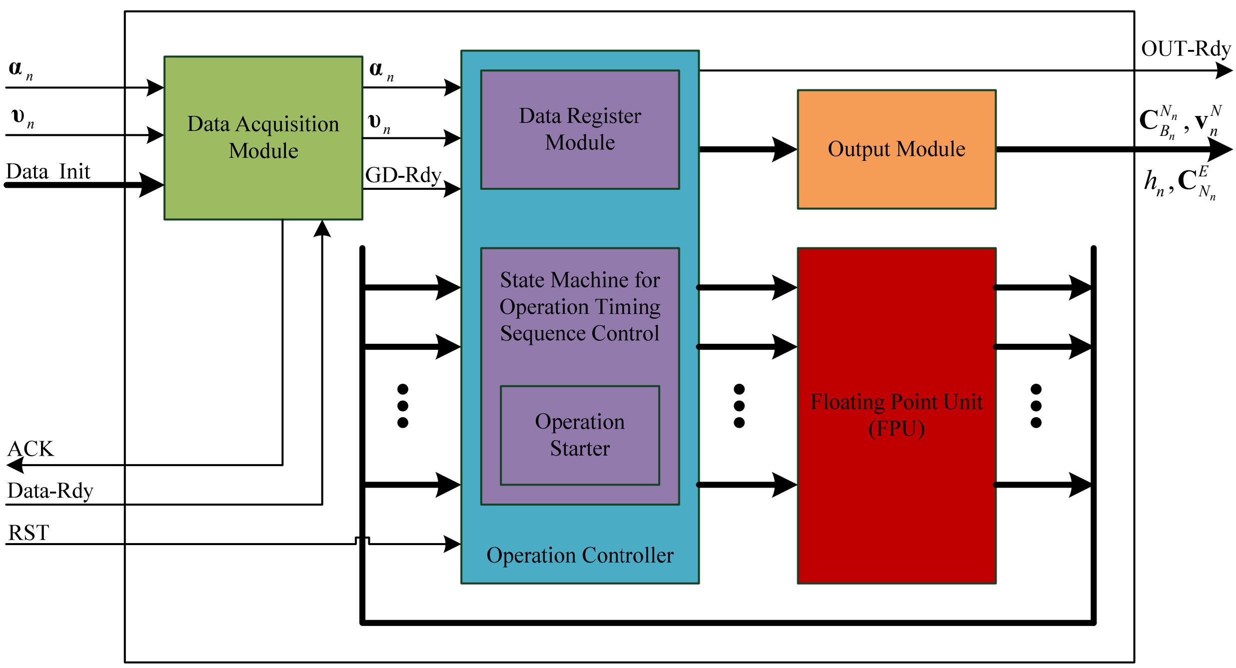
$$T_{M5} = \max\{T_{SF}, T_{G-C}\} + 2T_A = 4T_M + 7T_A \tag{72}$$

4.6. Analysis of Module M6

Figure 9 shows that the calculation of the altitude matrix and the position matrix can be executed in a parallel mode. According to Equations (7a), (8) and (11), the execution time of the altitude calculation is:

$$T_{Alt} = T_{VA} + 2T_M + T_A = 2T_M + 2T_A \tag{73}$$

Figure 9. Implementation of parallel strapdown algorithm base on FPGA.



According to Equations (7b) and Equations (9–11), the execution time of the position matrix calculation is:

$$T_{PosDCM} = T_{VA} + 2T_M + T_M + T_A + T_{VCP2} + T_M + T_A + T_{MP} = 6T_M + 6T_A \tag{74}$$

Based on Equations (73) and (74) and refer to Figure 9, the execution time of module M6 can be obtained as follows:

$$T_{M6} = \max\{T_{Alt}, T_{PosDCM}\} = 6T_M + 6T_A \tag{75}$$

The execution times of the floating-point operations T_A, T_M, T_D, T_T and T_S are slightly different when implemented on different FPGA platforms. But compared with T_m, T_A and T_D, T_T and T_S are generally larger and has the following approximate relationship with T_M:

$$T_D \approx 4T_M, \quad T_T \approx 4T_M, \quad T_S \approx 4T_M, \quad T_A \approx 2T_M \tag{76}$$

And the number of the gyro incremental angle samples and the accelerometer incremental velocity samples used in the calculation of the coning compensation and the sculling compensation is generally less than four [2], namely, N $\leq$ 4.Then according to Equations (42), (58), (61), (68), (69), (72) and (75), the length of the execution time of the parallelized strapdown algorithm can be estimated as follows:

$$\begin{aligned} T_{Nav} &= \max\{T_{M1}, T_{M2}, T_{M3}\} + \max\{T_{M4}, T_{M5}\} + T_{M6} \\ &= T_{M3} + T_{M5} + T_{M6} = 22T_M + 24T_A + 2T_D \approx 78T_M \end{aligned} \tag{77}$$

In contrast with the parallel implementation of the strapdown algorithm proposed in Section 3, the execution time of the original strapdown algorithm in a serial mode given in Section 2 is:

$$T'_{Nav} = T'_{M1} + T'_{M2} + T'_{M3} + T'_{M4} + T'_{M5} + T'_{M6} \tag{78}$$

with:

$$T'_{M1} = (7N + 11)T_M + (4N + 16)T_A + 2T_D + T_S + 2T_T \tag{79a}$$

$$T'_{M2} = (14N - 5)T_M + (8N - 1)T_A \tag{79b}$$

$$T'_{M3} = 88T_M + 57T_A + 7T_D + T_S \tag{79c}$$

$$T'_{M4} = 54T_M + 36T_A \tag{79d}$$

$$T'_{M5} = 43T_M + 35T_A \tag{79e}$$

$$T'_{M6} = 49T_M + 34T_A \tag{79f}$$

Thus according to Equations (76–79), the ratio of the execution time of the parallelized strapdown algorithm to the execution time of the original strapdown algorithm can be estimated as follows:

$$\eta = \frac{T_{Nav}}{T'_{Nav}} = \frac{22T_M + 24T_A + 2T_D}{(240 + 21N)T_M + (177 + 12N)T_A + 9T_D + 2T_S + 2T_T} \approx 9.44\% \tag{80}$$

where N is assumed to be 4. Equation (80) indicate that the parallelization design of the new optimum strapdown algorithm can significantly increase the updating rate of the algorithm, thus providing an important foundation to improve the accuracy of SINS working in high dynamic environments.

5. Implementation and Simulation of Parallel Strapdown Algorithm on FPGA

The parallel strapdown algorithm proposed in Section 3 has been implemented on a FPGA platform in the structure shown in Figure 9. The data acquisition module receives the input signal (the gyro incremental angle $\boldsymbol{\alpha}_n$, the accelerometer incremental velocity $\boldsymbol{\upsilon}_n$ and the initial alignment data *Data_Init*) and writes the data to the data register module in the operation controller, then notifies the operation controller to start the strapdown calculations through the signal *GD_Rdy*. The operation controller sends the data stored in the data register module in a parallel mode to the input registers of the floating point unit (FPU), and starts the FPU by the operation starter. The operation results ($\mathbf{C}_{B_n}^{N_n}$, $\mathbf{v}_n^N$, h_n and $\mathbf{C}_{N_n}^{E}$) are then exported through the output module. The handshake signals *ACK* and *Data_Rdy* are used for the communication between the data acquisition module and the external module such as the initial alignment or noise filtering of inertial sensor output samples that is beyond the scope of this paper; and the operation controller can be reset by the signal *RST*. The state machine in the operation controller is used to control the execution of operations in an appropriate time sequence.

All floating-point operations are carried out in the FPU which is composed of five arithmetic sub-units executing the operations for addition, multiplication, division, square root calculation and trigonometric calculation, respectively. Among them, the adder unit contains *k* floating-point adders; the multiplier unit contains *l* floating-point adders; the divider unit contains *m* floating-point adders; the square root arithmetic unit contains *n* floating-point adders; where different values of *k*, *l*, *m* and *n* can be selected according to the hardware resources of the selected FPGA platform.

The parallel strapdown algorithm has been simulated on the Xilinx ISE 12.3 software platform and the hardware platform of the FPGA device XC6VLX550T. The floating-point adder/subtractor, multiplier and other floating-point operations in FPU are constructed by the Xilinx IP core.

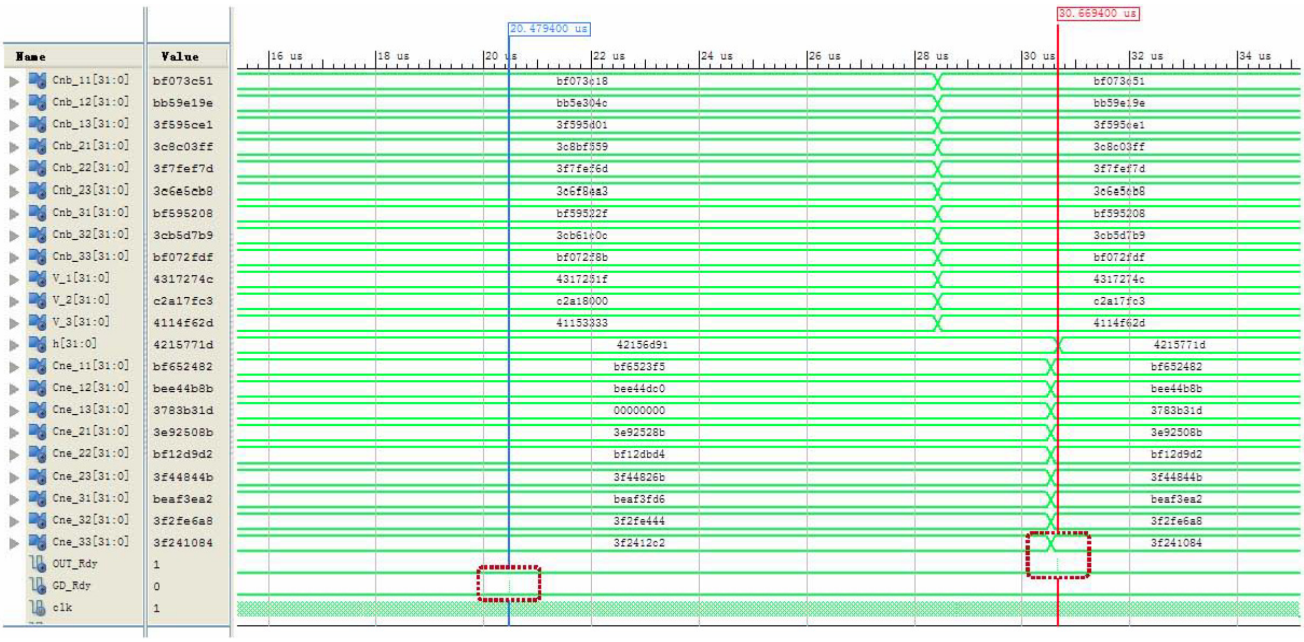
In the simulation, both the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples for the calculation of the coning compensation and the sculling compensation are set to two (namely, $N = 2$), then, according to Equations (20), (22), (25) and (34), the coning compensation term and the sculling compensation term defined in Equations (18) and (33) can be expressed as follows:

$$\hat{\boldsymbol{\beta}}_n = \frac{1}{12}\boldsymbol{\alpha}_{n-1} \times \boldsymbol{\alpha}_n \tag{81a}$$

$$\Delta\hat{\mathbf{v}}_{Scul_n} = \frac{1}{12}\boldsymbol{\alpha}_{n-1} \times \boldsymbol{\upsilon}_n + \frac{1}{12}\boldsymbol{\upsilon}_{n-1} \times \boldsymbol{\alpha}_n \tag{81b}$$

To demonstrate the performance of the proposed parallel strapdown algorithm, the simulation results in a typical updating interval are shown in Figure 10 as the behavioral simulation waveform graph yielded by Xilinx ISE 12.3, and listed in Table 1 where all the data are accurate to four decimal places. The updating interval length T_n in the simulation is 1.0e−3 s, and the clock frequency of the FPGA is set to 10 ns.

Figure 10. Behavioral simulation waveform graph of parallel strapdown algorithm.



The same simulation scenarios are calculated on a MATLAB R2007a platform with the strapdown algorithm in Section 2, the results are consistent with those shown in Table 1.

Table 1. (**a**) Simulation results—gyro and accelerometer inputs; (**b**) Simulation results—attitude, velocity and position outputs.

Sample time	**Gyro incremental angle sample**			**Accelerometer incremental velocity sample**		
	x axis (°)	**y axis (°)**	**z axis (°)**	**x axis (m/s)**	**y axis (m/s)**	**z axis (m/s)**
t_{n-1}	7.3154e−5	−4.0469e−6	7.0390e−6	−1.4102e−4	3.3985e−4	9.9255e−3
t_n	7.1058e−5	3.9166e−6	7.8809e−6	−2.2112e−4	1.4560e−4	9.5916e−3

(a)

Update time	**Attitude**			**Velocity**			**Position**		
	roll angle (°)	**pitch angle (°)**	**yaw angle (°)**	**x axis (m/s)**	**y axis (m/s)**	**z axis (ms)**	**latitude (°)**	**longitude (°)**	**altitude (m)**
t_{n-1}	−0.1943	1.2738	121.8839	151.1450	−80.7500	9.3250	39.8598	116.4813	37.3570
t_n	−58.1158	1.2720	−0.1906	151.1533	−80.7497	9.3103	39.8598	116.4813	37.3663

(b)

Figure 10 shows that the signal *GD_Rdy* has a pulse output at time 20.47 us, and the signal *OUT_Rdy* has a pulse output at time 30.66 us. This means that the start time and end time of the parallel strapdown algorithm on the FPGA platform are 20.47 us and 30.66 us, respectively. Then the execution time of this parallel strapdown algorithm is only 10.19 µs, when the clock frequency is selected as 10 ns. But the execution time of a strapdown algorithm on a DSP platform is generally in milliseconds. Thus the execution speed of parallel strapdown algorithm on the FPGA platform is much faster than the conventional algorithm on a DSP platform.

The resource utilization of the parallel strapdown algorithm on the hardware platform of the FPGA device XC6VLX550T is shown in Table 2 where Slice Registers, Slice LUTs and DSP48Es are the registers, the look-up tables and the multipliers based on the intellectual property (IP) hard-core of the Xilinx FPGA, respectively.

Table 2. Resource utilization of parallel strapdown algorithm.

Resource Type	Slice Registers	Slice LUTs	DSP8Es
Usage Amount	24,837 (3%)	26,074 (7%)	105 (12%)
Available Amount	687,360	343,680	864

6. Conclusions

In this paper, a new generalized optimum strapdown algorithm with the coning and sculling compensation is presented, in which the PVA updating operations are carried out based on the single-speed structure in which all computations are executed at a single updating rate that is sufficiently high to accurately account for high frequency angular rate and acceleration rectification effects. Different from existing algorithms, the updating rates of the coning and sculling compensations are unrelated with the number of the gyro incremental angle samples and the number of the accelerometer incremental velocity samples. When the output sampling rate of inertial sensors remains constant, this algorithm allows increasing the updating rate of the coning and sculling compensation, yet with more numbers of gyro incremental angle and accelerometer incremental velocity in order to improve the accuracy of system. Then, in order to implement the new strapdown algorithm in a single chip FPGA, the parallelization of the algorithm is designed and its computational complexity is analyzed. The performance of the proposed parallel strapdown algorithm is tested on the software platform of Xilinx ISE 12.3 and the FPGA device XC6VLX550T hardware platform on the basis of some fighter data. It is shown that this parallel strapdown algorithm on the FPGA platform can greatly decrease the execution time of algorithm to meet the real-time and high precision requirements of system on the high dynamic environment, relative to the existing implemented on the DSP platform.

Acknowledgments

The authors would like to gratefully acknowledge the financial support from the National Nature Science Foundation of China under grant No. 61070003 and Zhejiang Provincial Natural Science Foundation of China under grant No. R1090052.

References

1. Savage, P.G. Strapdown inertial navigation integration algorithm design. Part 1: Attitude algorithms. *J. Guid. Control Dyn.* **1998**, *21*, 19-28.
2. Titterton, D.H.; Weston, J.L. *Strapdown Inertial Navigation Technology*, 2nd ed.; AIAA: Reston, VA, USA, 2004.
3. Savage, P.G. *Strapdown Analytics*, 2nd ed.; Strapdown Associates, Inc.: Maple Plain, MN, USA, 2007.
4. Bortz, J.E. A new mathematical formulation for strapdown inertial navigation. *IEEE Trans. Aerosp. Electron. Syst.* **1971**, *7*, 61-66.
5. Miller, R.B. A new strapdown attitude algorithm. *J. Guid. Control Dyn.* **1983**, *6*, 287-291.

6. Lee, J.G.; Yoon, Y.J. Extension of strapdown attitude algorithm for high-frequency base motion. *J. Guid. Control Dyn.* **1990**, *13*, 738-743.
7. Jiang, Y.F.; Lin, Y.P. Improved strapdown coning algorithms. *IEEE Trans. Aerosp. Electron. Syst.* **1992**, *28*, 484-490.
8. Musoff, H.; Murphy, J.H. A study of recent strapdown navigation attitude algorithms. In *Proceedings of AIAA Guidance, Navigation and Control Conference*, Monterey, CA, USA, 9–11 August 1993; pp. 1717-1723.
9. Ignagni, M. Efficient class of optimized coning compensation algorithms. *J. Guid. Control Dyn.* **1996**, *19*, 424-429.
10. Park, C.G.; Kim, K.J.; Chung, D.; Lee, J.G. Generalized coning compensation algorithm for strapdown system. In *Proceedings of AIAA Guidance, Navigation and Control Conference*, San Diego, CA, USA, 1996.
11. Park, C.G.; Kim, K.J. Formalized approach to obtaining optimal coefficients for coning algorithms. *J. Guid. Control Dyn.* **1999**, *22*, 165-168.
12. Ignagni, M.B. Duality of optimal strapdown sculling and coning compensation algorithms. *Navigation* **1998**, *45*, 85-95.
13. Roscoe, K.M. Equivalency between strapdown inertial navigation coning and sculling integrals/algorithms. *J. Guid. Control Dyn.* **2001**, *24*, 201-205.
14. Savage, P.G. Strapdown inertial navigation integration algorithm design. Part 2: Velocity and position algorithms. *J. Guid. Control Dyn.* **1998**, *21*, 208-221.
15. Savage, P.G. A unified mathematical framework for strapdown algorithm design. *J. Guid. Control Dyn.* **2006**, *29*, 237-249.
16. Savage, P.G. *Computational Elements for Strapdown Systems*; Available online: http://ftp.rta.nato.int/Public/PubFullText/RTO/EN/RTO-EN-SET-064/EN-SET-064-03.pdf (accessed on 4 July 2011).
17. Savage, P.G. Coning algorithm design by explicit frequency shaping. *J. Guid. Control Dyn.* **2010**, *33*, 1123-1132.
18. Musoff, H.; Murphy, J.H. Study of strapdown navigation algorithms. *J. Guid. Control Dyn.* **1995**, *18*, 287-290.
19. Xie, X.; Guo, M.-F.; Zhou, B. Design of system for high performance navigation computer system base on dual DSPs & FPGA. *Control Autom.* **2009**, *25*, 1-2.
20. Jew, M.; El-Osery, A.; Bruder, S. Implementation of an FPGA-based aided IMU on a low-cost autonomous outdoor robot. In *Proceedings of IEEE/ION Position Location and Navigation Symposium (PLANS 2010)*, Indian Wells, CA, USA, 4–6 May 2010; pp. 1043-1051.
21. Jia, S.W.; Feng, D.-Z.; Wang, M.-G. The reserch on algorithm for SINS based on FPGA. *J. Proj. Rocket. Missiles Guid.* **2006**, *26*, 953-955.
22. Yang, F.; Hao, Y.-P.; Miao, L. Based on FPGA by constructed multi-processor navigation system design. *J. Proj. Rocket. Missiles Guid.* **2007**, *27*, 126-128.
23. Geng, Y.R.; Wang, Y.Z.; Cui, Z.X. Optimal design for sculling algorithms of SINS. *J. Beijing Univ. Aeronaut. Astronaut.* **2001**, *27*, 694-697.

24. Grejner-Brzezinska, D.A.; Yi, Y.; Toth, C.; Anderson, R.; Davenport, J.; Kopcha, D.; Salman, R. Enhanced gravity compensation for improved inertial navigation accuracy. In *Proceedings of 16th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS/GNSS 2003)*, Portland, OR, USA, 9–12 September 2003.
25. Kwon, J.H.; Jekeli, C. Gravity requirements for compensation of ultra-precise inertial navigation. *J. Navig.* **2005**, *58*, 479-492.
26. Akeila, E.; Salcic, Z.; Swain, A. Direct gravity estimation and compensation in strapdown INS applications. In *Proceedings of 3rd International Conference on Sensing Technology*, Tainan, Taiwan, 30 November–3 December 2008.
27. Groves, P.D. *Principles of GNSS, Intertial, and Multisensor Integrated Navigation Systems*; Artech House: Norwood, MA, USA, 2008.