BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

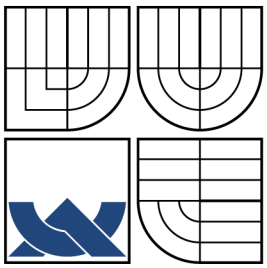
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

OPTIMAL STATE ESTIMATION OF A NAVIGATION MODEL SYSTEM

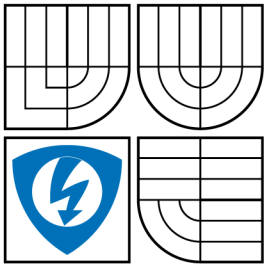
MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE

Bc. MILAN PAPEŽ



BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ



FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

OPTIMAL STATE ESTIMATION OF A NAVIGATION MODEL SYSTEM

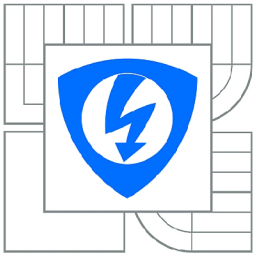
OPTIMÁLNÍ ODHAD STAVU MODELU NAVIGAČNÍHO SYSTÉMU

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE
Bc. MILAN PAPEŽ

SUPERVISOR
VEDOUCÍ PRÁCE
Ing. JAKUB DOKOUPIL, Ph.D.

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Milan Papež	***ID:***	120609
Ročník:	2	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

Optimální odhad stavu modelu navigačního systému

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je porovnat vlastnosti implementací rozdílných forem Kalmanova filtru s ohledem na kvalitu odhadovaných parametrů vystupujících v úlohách navigace. Systém generující odhad stavu bude vyhodnocovat informace z datové fůze inerciálních snímačů a signály z modulu s technologií GPS. Student formuluje optimalizační problém, jehož řešením bude rekurzivní aktualizace střední hodnoty parametrů a jejich kovarianční matice za předpokladu, že pozorování je zatíženo šumem jehož funkce hustoty pravděpodobnosti má normální rozdělení, implementuje jednotlivé způsoby filtrace kovarianční matice založené na rozkladu této matice. Studentem bude vyhodnocen vliv nepřesností a absence určitostí v popisu matematického modelu a vliv výpočetní nepřesnosti na pevné řádové čárce s použitím prostředí MATLAB/Simulink.

DOPORUČENÁ LITERATURA:

[1] SIMON, D. Optimal State Estimation. New Jersey: Wiley-Interscience, 2006. 526 stran. ISBN 0 471-70858-5
[2] VERHEAGEN, M., VERDULT, V. Filtering and System Identification: a least squares approach. Cambridge: Cambridge University Press, 2007. 405 stran. ISBN 0-521-87512-9

Termín zadání:	11.2.2013	***Termín odevzdání:***	20.5.2013

Vedoucí práce: Ing. Jakub Dokoupil, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

ABSTRACT

This thesis presents an investigation of the possibility of using the fixed-point arithmetic in the inertial navigation systems, which use the local level navigation frame mechanization equations. Two square root filtering methods, the Potter's square root Kalman filter and UD factorized Kalman filter, are compared with respect to the conventional Kalman filter and its Joseph's stabilized form. The effect of rounding errors to the Kalman filter optimality and the covariance matrix or its factors conditioning is evaluated for a various lengths of the fractional part of the fixed-point computational word. Main contribution of this research lies in an evaluation of the minimal fixed-point arithmetic word length for the Phi-angle error model with noise statistics which correspond to the tactical grade inertial measurements units.

KEYWORDS

Inertial navigation, multisensor data fusion, Kalman filtering, square root filtering, fixed-point arithmetic, floating-point arithmetic, Phi-angle error model, 15-state loosely coupled integration approach

ABSTRAKT

Tato diplomová práce se zabývá vyšetřováním možnosti použití aritmetiky pracující v pevné řadové čárce u inerciálních navigačních systémů, které využívají navigační rovnice vyjádřené v lokální navigační soustavě. Dva typy odmocninových filtrů, Potterův odmocninový Kalmanův filtr a UD faktorizovaný Kalmanův filtr, jsou porovnány vzhledem ke konvenčnímu Kalmanově filtru a jeho Josephově stabilizované formě. Je zde vyhodnocen vliv zaokrouhlovacích chyb na optimalitu Kalmanova filtru a na podmíněnost jeho kovarianční matice resp. jejich faktorů. Hlavní přínos této práce spočívá ve vyhodnocení minimální délky výpočetního slova aritmetiky pracující v pevné řadové čárce pro Phi-angle chybový model s uvažovanými statistikami šumu, které odpovídají kvalitě taktických inerciálně měřicích jednotek.

KLÍČOVÁ SLOVA

Inerciální navigace, datová fůze, Kalmanova filtrace, odmocninová filtrace, aritmetika s pevnou řadovou čárkou, aritmetika s plovoucí řadovou čárkou, Phi-angle chybový model, 15-stavový slabě spjatý přístup k integraci navigačního algoritmu

PAPEŽ, Milan *Optimal state estimation of a navigation model system*: master's thesis. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Control and Instrumentation, 2013. 108 p. Supervised by Ing. Jakub Dokoupil, Ph.D.

DECLARATION

I declare that I have written my master's thesis on the theme of "Optimal state estimation of a navigation model system" independently, under the guidance of the master's thesis supervisor and using the technical literature and other sources of information which are all quoted in the thesis and detailed in the list of literature at the end of the thesis.

As the author of the master's thesis I furthermore declare that, as regards the creation of this master's thesis, I have not infringed any copyright. In particular, I have not unlawfully encroached on anyone's personal and/or ownership rights and I am fully aware of the consequences in the case of breaking Regulation § 11 and the following of the Copyright Act No 121/2000 Sb., and of the rights related to intellectual property right and changes in some Acts (Intellectual Property Act) and formulated in later regulations, inclusive of the possible consequences resulting from the provisions of Criminal Act No 40/2009 Sb., Section 2, Head VI, Part 4.

Brno ....

(author's signature)

ACKNOWLEDGEMENT

I would like to thank my supervisor Ing. Jakub Dokoupil, Ph.D. for his guidance, support, encouragement and pation throughout elaborating this work. Last but not least, I am very gratefull to my family, especially my parents for their support.

Brno ....
(author's signature)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Thesis Motivation

A high-performance inertial navigation system (INS) is today one of the most critical part of each aircraft. Especially in a long-range and long-term flight is needed an accurate, robust and high-rate inertial navigation. As a counterpart for these requirements, there is a cost and energy efficiency. Most of the modern Digital Signal Processors (DSP), which use the floating-point arithmetic, are incompatible with these requirements when we assume e.g. long duration unmanned aircraft missions, where a need for the energy efficiency can be crucial. A way how to reduce the cost and increase the energy efficiency lies in a consideration if we are able to use simpler processors or Field-Programmable Gate Arrays (FPGA). These commonly uses the fixed-point arithmetic for performing mathematical operations. Hence, a question of correct and efficient implementation with an attention to a numerical issues becomes one of the most important.

Most of the inertial navigation systems are based on the conventional Kalman filter. Theoretically it is not possible for the Kalman filter to become numerically unstable, but from a practical point of view, especially if we use a short length of the computational word, there is a chance, that the filter become unstable. The task related to the numerical difficulties connected with the conventional Kalman filter was investigated many times over past years. A special attention was paid to a spacecraft navigation and orbit determination problems as can be seen for example in [1][2][3]. The methods which have an ability to deal with numerical deficiencies of the Kalman filter are generally called as the square root filtering methods. Although these methods appear in the era of first cosmic flights, thus when computers had very limited computational power, they are still an inspiration for newly invented last squares methods as can be seen for example in [4]. A motivation for their use is not only the fixed-point implementation, but their ability to make an estimation algorithm more robust, even if the floating-point arithmetic is used. Another reason for their use is an effort to make some system economically advantageous, thus when we tray to have a system with no unnecessary level of redundancy.

1.2 Thesis Objectives

The main objective of this thesis is to compare different numerical implementations of the Kalman filter with an attention to a quality of the estimated state variables as they are represented in an inertial navigation system which uses an external aiding by the Global Navigation Satellite System (GNSS). It is an aim of this thesis to make

such a comparison for the inertial navigation system algorithm expressed in the local navigation frame with using the Phi-angle error model in the so-called Loosely Coupled navigation integration approach. We will try to evaluate how various word lengths of the fixed-point arithmetic can affect a precision of the navigation system's output. Our next goal is to show how different numerical implementations of the Kalman filter can lead to obtaining better results and make the navigation algorithm more robust. Finally we try to answer a question if it is possible to use a length of the fixed-point computational word which is less or similar to the length of the double precision floating-point arithmetic, because if it is, then the navigation system can be implemented beneficially.

1.3 Thesis Organization

Chapter 1 - it is right here, thus only for a summary, some introductory remarks, thesis objectives and description of the thesis.

Chapter 2 - presents an introduction to the today's most commonly used arithmetic systems, although it is a very known topic, we describe some notions, only for completeness.

Chapter 3 - deals with a stochastic dynamical system model which is suitable through this work. Further, the conventional Kalman filter and its numerically more stable implementations are presented.

Chapter 4 - presents basic principles of the inertial navigation i.e. the coordinate frames, inertial navigation system mechanization algorithm and Phi-angle error model. A description of the inertial sensors inaccuracies and how to use them for the inertial measurement unit data generation, is proposed.

Chapter 5 - an experimental part of the thesis which presents how the numerical round-off errors can affect a trajectory estimated by the inertial navigation system. For this purpose is evaluated a criteria function for various lengths of the fractional part of the fixed-point computational word. Further, an influence to the covariance matrix or their square root factors conditioning is shown. Finally, a divergence caused by the rounding is described.

Chapter 6 - summarizes results achieved through this thesis and states possible extensions and final conclusions.

2 FINITE WORD LENGTH ARITHMETIC

2.1 Introduction

All computing machines are able to compute only with a finite precision, thus they are able to compute only in a certain subset of the real numbers. Assuming a computer which computes with an infinite precision is not realistic because it will takes probably an infinite time for obtaining a result. Hence, it is suitable to have a finite precision of computing. All numbers represented in the computer are spread out by a finite interval, which represents the machine resolution, hence a numerical round-off arises due to an execution of mathematical operations. These round-off errors can significantly affect some types of mathematical algorithms. The objective of this chapter is a brief description of commonly used arithmetic systems represented in the today's computers and embedded systems.

2.2 Fixed-Point Arithmetic

Let's assume that we have an N-bit binary word for interpreting a real value, then using the two's complement representation, the fixed-point number can be expressed as figure 2.1 shows. There can be seen that the bit word is divided into three parts, these are 1-bit for expressing a sing (S), a-bits for the integer part (I) and b-bits for the fractional part (F). The red dot represents the fixed binary point.

Fig. 2.1: The binary representation of the two's complement fixed-point number

Using this representation, a real world value can be approximated as a number from the following range [5]

$$-2^{N-1-b} \leq v \leq +2^{N-1-b} - 2^{-b} \tag{2.1}$$

where a concrete value can be obtained as

$$v = 2^{-b}\left[-2^{N-1}x_{N-1} + \sum_{i=0}^{N-2} 2^i x_i\right] \tag{2.2}$$

the term x_i is considered as the i-th bit of the binary word. It is clear now, that the machine resolution, commonly called as the machine epsilon, is of the value 2^{-b}. Assuming $b = 0$, the previous representation becomes signed integer as can be seen from 2.1 and 2.2.

2.3 Floating-Point Arithmetic

The floating-point binary representation is depicted in figure 2.2. There we can see, that the binary word is divided into three parts, these are 1-bit for the sign (S) expression, w-bits for the exponent (E) and t-bits for the trailing significant field (T) or mantissa. The bit d_0, which is decreased from the p-bit field, represents the first bit in the mantissa. It is the bit before the binary point and is implicitly included in the exponent part.

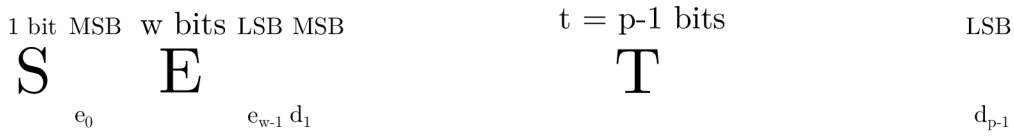


Fig. 2.2: The floating-point binary representation according to the std. IEEE 754

The floating-point arithmetic is standardized by the Institute of Electrical and Electronics Engineers (IEEE) as the std. IEEE 754 [6]. This standard recommends the parameters of the binary interchange format as it is shown in Table 2.1.

Format	**Sign**	**Exponent (w)**	**Mantisa (t)**
binary32 (Single)	1	8	23
binary64 (Double)	1	11	52
binary128 (Quadruple)	1	15	112

Tab. 2.1: The floating-point number parameters according to the std. IEEE 754

A value of the floating-point number can be obtained according to the expression 2.3, where the first two rows represent the normal and subnormal numbers. The remaining rows express some special values. These are the signed zero, positive or negative infinity and not-a-number.

$$v = \begin{cases} (-1)^S 2^{(E-bias)} \left(1 + \sum_{i=1}^{p-1} d_i 2^{-i}\right) & 1 \leq E \leq 2^w - 2 \\ (-1)^S 2^{emin} \left(0 + \sum_{i=1}^{p-1} d_i 2^{-i}\right) & E = 0, T \neq 0 \\ (-1)^S (+0) & E = 0, T = 0 \\ (-1)^S (+\infty) & E = 2^w - 1, T = 0 \\ NaN & E = 2^w - 1, T \neq 0 \end{cases} \tag{2.3}$$

The values of the *bias* and *emin* are represented as $bias = emax = 2^{w-1} - 1$ and $emin = 1 - emax = 2 - 2^{w-1}$ respectively.

2.4 Floating-Point vs. Fixed-Point Comparison

Although the floating-point arithmetic is evolutionary far away in a comparison to the fixed-point arithmetic, there are still some considerable differences, which can be useful when we try to implement numerical algorithms efficiently. Of course, the dynamic range of the floating-point is much more greater than the fixed-point, but its speed is slower and a chip integration area is greater. Both of these bottlenecks are caused due to complex algorithms for the arithmetical operations. As these algorithms are simpler for the fixed-point numbers a greater speed and smaller chip integration area can be achieved. Especially the chip integration area significantly affects the power consumption, operating cost and cost of the device itself. There is no doubt about the fact, that an algorithm implementation in the fixed-point arithmetic takes more time in a comparison to a floating-point implementation.

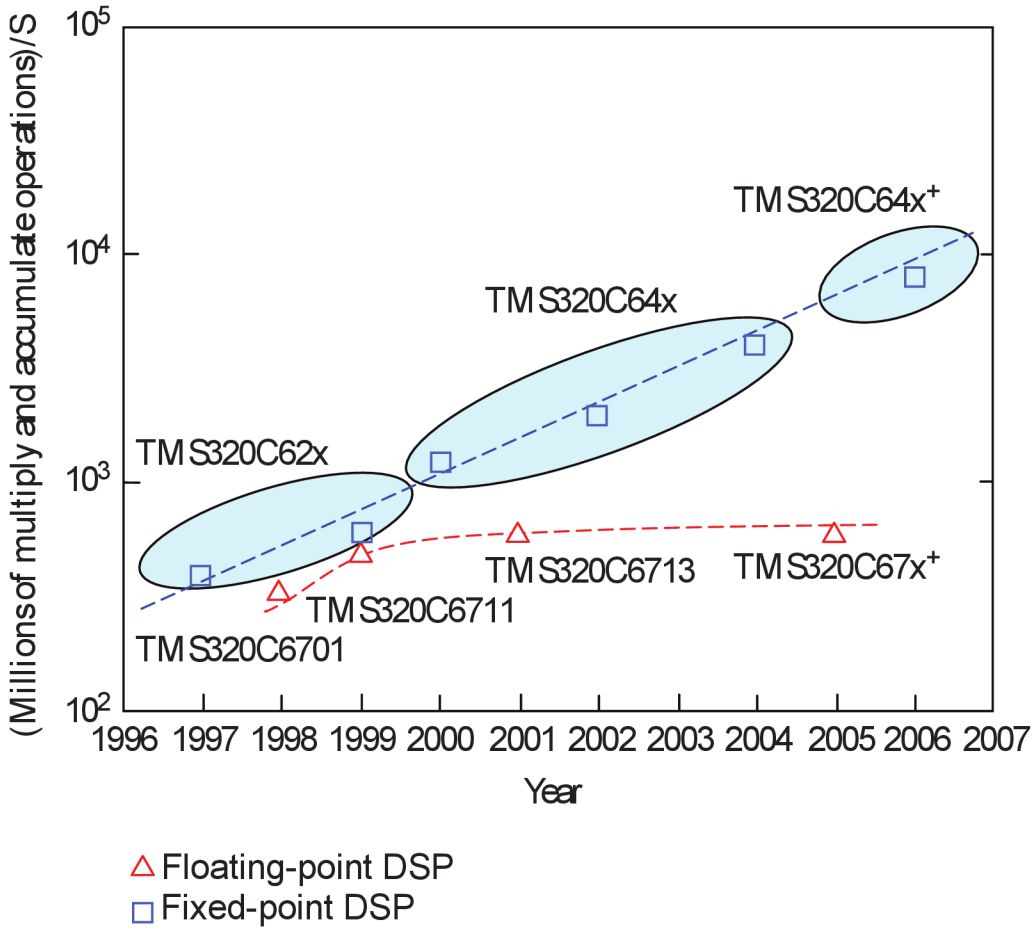


Fig. 2.3: A comparison of the MACs per second for the floating-point and fixed-point DSPs. (Reprinted from [7] and corrected.)

A comparison of the number of multiply and accumulate operations per second for the floating-point and fixed-point DSPs is depicted in figure 2.3. There we can

see that the trend for the fixed-point DSPs grows more rapidly in a comparison to the floating-point DSPs. This difference can be significant for the inertial navigation systems which employ the Kalman filtering algorithm with a high state space dimension. Another important benefit for the fixed-point arithmetic is its suitability for the systolic array implementation of the algorithms for matrix operations. All previously mentioned advantages and disadvantages are summarized in table 2.2.

	Fixed-Point	**Floating-Point**
Dynamic Range	-	+
Development Time	-	+
Operational Complexity	+	-
Arithmetic Operations Speed	+	-
Chip Integration Area	+	-
Power Consumption	+	-
Device Cost	+	-
Operating Cost	+	-
Systolic Array Algorithms	+	-

Tab. 2.2: Fixed-point vs. floating-point arithmetic comparison

It is important to note, that the table 2.2 can not be considered dogmatically. All properties need to be determined individually for all devices since they strongly dependent on a level of optimisation used during their development.

3 KALMAN FILTERING

3.1 Introduction

This chapter deals about the Kalman filter and its numerically more stable implementations. In the first section we look at a system which is needed for the purposes of this text as a whole. Next section is concerned with a derivation of the conventional Kalman filter for the system presented in the first section. Third part is focused on a derivation of some numerically robust Kalman filters.

3.2 System Model

Let us assume that we have a stochastic linear time-variant dynamic system given by the following equations

$$\dot{x}_t = F_t x_t + G_t w_t \tag{3.1}$$

$$z_t = H_t x_t + v_t \tag{3.2}$$

where first of these equations represents the system state model and the second represents the measurement or observation model. The n-dimensional state vector x_t is related from the one time step to the other by the system transition matrix F_t which is of the dimension $n \times n$. The system measurement is expressed by the l-dimensional vector z_t and is considered as a linear transformation from the state space to the measurement space and computed by the observation matrix H_t of the dimension $l \times n$. The term w_t is considered as a random variable, especially as a random vector of dimension m, which is assumed as the Gaussian or normally distributed with zero mean and known covariance and which represents the process noise. The term v_t, which represents a measurement noise, is similarly the Gaussian random vector with zero mean and known covariance and is of the dimension l. The matrix G_t is the process noise distribution matrix and is of the dimension $n \times m$.

At a beginning we consider only the deterministic part of the equation 3.1 as follows

$$\dot{x}_t = F_t x_t \tag{3.3}$$

If we assume that we have a set of n linearly independent vectors which together form the fundamental matrix of 3.3 given as

$$X_t = \{x_{1,t}\ x_{2,t}\ ...\ x_{n,t}\} \tag{3.4}$$

where $x_{i,t} = [x_{1i}\ x_{2i}\ ...\ x_{ni}]^T$ and if we substitute this matrix into 3.3, then the following matrix differential equation is obtained

$$\dot{X}_t = F_t X_t \tag{3.5}$$

Post-multiplying the previous equation by $X_{t_0}^{-1}$ yields

$$\begin{aligned} (X_t \dot{X}_{t_0}^{-1}) &= F_t(X_t X_{t_0}^{-1}) && (3.6) \\ \dot{\Phi}_{t,t_0} &= F_t \Phi_{t,t_0} && (3.7) \end{aligned}$$

where the term $\Phi_{t,t_0} = X_t X_{t_0}^{-1}$ is the normalised fundamental matrix. Again post-multiply, but now with an initial condition vector x_{t_0} yields

$$(\dot{\Phi_{t,t_0}} x_{t_0}) = F_t(\Phi_{t,t_0} x_{t_0}) \tag{3.8}$$

which gives us the general expression for the solution of the equation 3.3 as follows

$$x_t = \Phi_{t,t_0} x_{t_0} \tag{3.9}$$

Now we include the stochastic part of the equation 3.1 and we will deal with it as with the deterministic input. This assumption is not, from a point of view of the stochastic system theory, correct and rigorous, but it leads to the same results. Let's start with an assumed solution given as follows

$$x_t = \Phi_{t,t_0} c_t \tag{3.10}$$

where c_t is a vector which need to be expressed. Taking the time derivative of this equation yields

$$\begin{aligned} \dot{x}_t &= \dot{\Phi}_{t,t_0} c_t + \Phi_{t,t_0} \dot{c}_t && (3.11) \\ &= F_t x_t + G_t w_t && (3.12) \end{aligned}$$

Comparing the second terms of the equations 3.11 and 3.12 gives

$$\Phi_{t,t_0} \dot{c}_t = G_t w_t \tag{3.13}$$

Now pre-multiplying by $\Phi_{t_0,t}$ and integrating this expression yields

$$c_t = c_{t_0} + \int_{t_0}^{t} \Phi_{\tau,t_0}^{-1} G_\tau w_\tau d\tau \tag{3.14}$$

Substituting this into 3.10 and replacing c_{t_0} as x_{t_0} gives us the final solution of the equation 3.1 as follows

$$x_t = \Phi_{t,t_0} x_{t_0} + \int_{t_0}^{t} \Phi_{t,\tau} G_\tau w_\tau d\tau \tag{3.15}$$

where the matrix Φ_{t,t_0} is, in general, computed as

$$\Phi_{t,t_0} = e^{\int_{t_0}^{t} F_\tau d\tau} \tag{3.16}$$

However, this expression can be used only if the following condition holds for all t and τ[8]

$$F_t \int_{t_0}^{t} F_\tau d\tau = \int_{t_0}^{t} F_\tau d\tau F_t \tag{3.17}$$

Because this is a true only for some special cases the elements of matrix F_t will be considered as constant values between the time t_0 and t. Now we can use simplified expression for computing Φ_{t,t_0} as follows

$$\Phi_{t,t_0} = e^{F_{t_0}(t-t_0)} \tag{3.18}$$

where the time index t_0 in the matrix F represents that all entries are computed at the beginning of the time interval $[t_0, t]$. The matrix exponential $e^{F_{t_0}(t-t_0)}$ can be computed by Taylor series expansion given as follows

$$\Phi_{t,t_0} = \sum_{i=0}^{\infty} \frac{1}{i!} F_{t_0}^i \Delta t^i \tag{3.19}$$

where Δt is equal to $(t - t_0)$. There are several ways how to compute the matrix exponential, but using the first two or three terms of this expansion is commonly sufficient. Using this approximation is advantageous especially in such cases, where the complete analytical expression of the matrix exponential is too complex.

Now we can formulate discrete time solution simply as

$$x_{k+1} = \Phi_{k+1,k} x_k + \int_{k}^{k+1} \Phi_{k+1,\tau} G_\tau w_\tau d\tau \tag{3.20}$$

which can be used for expressing the discrete-time equivalent of the system given by the equations 3.1 and 3.2 as follows

$$x_{k+1} = \Phi_{k+1,k} x_k + w_k \tag{3.21}$$

$$z_k = H_k x_k + v_k \tag{3.22}$$

where w_k is equal to the integral on the right-hand side of equation 3.20.

Finally we need to express discrete-time covariance matrices of this system. The process noise covariance matrix is given as

$$Q_k = E\left\{w_k w_k^T\right\} \tag{3.23}$$

$$= \int_{k}^{k+1} \Phi_{k+1,\tau} G_\tau Q_\tau G_\tau^T \Phi_{k+1,\tau}^T d\tau \tag{3.24}$$

If we assume that the matrices in 3.24 are constant between the times k and $k+1$ then we can use the following approximation

$$Q_k = \Phi_{k+1,k} G_k Q G_k^T \Phi_{k+1,k}^T \Delta t \tag{3.25}$$

The measurement noise covariance matrix can be expressed as [9]

$$R_k = E\left\{v_{(t)} v_{(t)}^T\right\} \frac{1}{\Delta t} = \frac{R}{\Delta t} \tag{3.26}$$

Where Q and R are the continuous covariances of the process and measurement noise respectively.

3.3 Conventional Kalman Filter

This section presents a Kalman filter derivation or rather some inductive proof type of derivation, because we do not derive the Kalman filter from the general perspective of the Bayesian estimation theory, but we start from an a priori knowledge about the equation of linear mean square estimate of the random variable x conditioned on the random variable z, thus we tray to estimate the system state 3.21 from an information about the measurements 3.22, as they are related to this state.

At a beginning we need to describe the system noise statistics more closely as follows

$$E\{w_k\} = 0 \tag{3.27}$$

$$E\{v_k\} = 0 \tag{3.28}$$

$$\begin{aligned} E\left\{w_k w_j^T\right\} &= Q_k \delta_{kj} \quad k = j \\ E\left\{w_k w_j^T\right\} &= 0 \qquad\quad k \neq j \end{aligned} \tag{3.29}$$

$$\begin{aligned} E\left\{v_k v_j^T\right\} &= R_j \delta_{kj} \quad k = j \\ E\left\{v_k v_j^T\right\} &= 0 \qquad\quad k \neq j \end{aligned} \tag{3.30}$$

$$E\left\{v_k w_j^T\right\} = 0 \quad \forall k, j \tag{3.31}$$

where we point out that the both random vectors w_k and v_k are zero mean with covariances given by Q_k and R_k respectively and that they are not correlated in the discrete times k and j as it is denoted by the Kronecker delta function δ_{kj}. We assume that the random variables are Gaussian, hence it can be proved from 3.29 that the joint probability density function $p(w_0, w_1, \ldots, w_N)$ can be rewritten as $p(w_0, w_1, \ldots, w_N) = p(w_0)p(w_1)\ldots p(w_N)$, which indicates that the random variable w_k represents the white noise. The same assumptions hold for the measurement noise v_k. The last from the previous expressions 3.31 indicates that the process and measurement noises are not correlated between each other.

As it is stated in the first paragraph of this section, the proof starts with the equation of the linear mean square estimate as follows

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k\left(z_k - H_k \widehat{x}_{k|k-1}\right) \tag{3.32}$$

where the first time subscript in the state vector $\widehat{x}_{k|k}$ represents the discrete time of computing or current system time step and the second subscript represents the measurement time, thus its value k indicates that the estimate is based on the measurement history up to and including time k. We call this estimate as an a posteriori

estimate since it is calculated after an incorporation of the last measurement. Similarly, we have an a priori estimate $\widehat{x}_{k|k-1}$ conditioned on the measurement history up to and including time $k-1$, thus the estimate computed before the measurement k is taken. The Kalman gain is generally of the form $K = P_{xz}P_{zz}^{-1}$. We do not assume this form at this moment because we will tray to show, only through a simple algebraic approach, how to obtain concrete form of the Kalman gain for the system given by 3.21 and 3.22. For this purpose we need to define a criteria function as follows

$$J = tr\ E\left\{\left(x_k - \widehat{x}_{k|k}\right)\left(x_k - \widehat{x}_{k|k}\right)^T\right\} \tag{3.33}$$

thus we tray to minimize a sum of squares of the differences between a true value of the state vector x_k entries and their estimated counterparts from $\widehat{x}_{k|k}$, which directly gives the trace of the a posteriori covariance matrix. However, we can try to minimize the a posteriori covariance matrix without the trace considerations. This is not a limitation, because if the diagonal entries of the a posteriori covariance are minimized, then the off diagonal entries are minimized too as they represent the variances. Hence, the task is to minimize the following a posteriori covariance matrix

$$\begin{aligned} P_{k|k} &= E\left\{\widetilde{x}_{k|k}\widetilde{x}_{k|k}^T\right\} \\ &= E\left\{\left(x_k - \widehat{x}_{k|k}\right)\left(x_k - \widehat{x}_{k|k}\right)^T\right\} \end{aligned} \tag{3.34}$$

So we need to express 3.34 in terms of the a posteriori error as follows

$$\begin{aligned} \widetilde{x}_{k|k} &= x_k - \widehat{x}_{k|k} \\ &= \Phi_{k,k-1}x_{k-1} + w_k - \widehat{x}_{k|k-1} - K_k\left(z_k - H_k\widehat{x}_{k|k-1}\right) \\ &= \Phi_{k,k-1}\widetilde{x}_{k-1|k-1} + w_k - K_k\left(H_k x_k + v_k - H_k\Phi_{k,k-1}x_{k-1|k-1}\right) \\ &= \Phi_{k,k-1}\widetilde{x}_{k-1|k-1} + w_k - K_k\left(H_k\Phi_{k,k-1}x_{k-1} - H_k\Phi_{k,k-1}x_{k-1|k-1} + H_k w_k + v_k\right) \\ &= \Phi_{k,k-1}\left(I - K_k H_k\right)\widetilde{x}_{k-1|k-1} + \left(I - K_k H_k\right)w_k - K_k v_k \end{aligned} \tag{3.35}$$

now substituting this equation into 3.34 yields

$$\begin{aligned} P_{k|k} &= E\left\{\widetilde{x}_{k|k}\widetilde{x}_{k|k}^T\right\} \\ &= \Phi_{k,k-1}\left(I - K_k H_k\right)E\left\{\widetilde{x}_{k-1|k-1}\widetilde{x}_{k-1|k-1}^T\right\}\left(I - K_k H_k\right)^T\Phi_{k,k-1}^T + \\ &+ \left(I - K_k H_k\right)E\left\{w_k w_k^T\right\}\left(I - K_k H_k\right)^T + K_k E\left\{v_k v_k^T\right\}K_k^T \\ &= \left(I - K_k H_k\right)P_{k|k-1}\left(I - K_k H_k\right)^T + K_k R K_k^T \end{aligned} \tag{3.36}$$

where we use the fact, that the expectation operator $E\{.\}$ is linear, which leads to an appearance of the relations 3.29, 3.30, 3.31. Further, we assume that the a posteriori estimation error $\widetilde{x}_{k|k}$ is not correlated with the process and measurement noise. The last row of 3.36 is commonly called as the Joseph's stabilized form of the a posteriori covariance matrix. This equation will be needed in the subsequent

section about numerically more stable Kalman filter implementations. At this point we need to perform a minimization of this equation. This task can be solved by the completing-the-square method as follows

$$\begin{aligned} P_{k|k} &= P_{k|k-1} - P_{k|k-1}H_k^T K_k^T - K_k H_k P_{k|k-1} + K H_k P_{k|k-1} H_k^T K_k^T + K_k R K_k^T \\ &= P_{k|k-1} - P_{k|k-1}H_k^T K_k^T - K_k H_k P_{k|k-1} + K_k Y_k K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} + \left(K_k - P_{k|k-1}H_k^T Y_k^{-1}\right) Y_k \left(K_k - P_{k|k-1}H_k^T Y_k^{-1}\right)^T \end{aligned} \tag{3.37}$$

where we introduce the following matrix

$$Y_k = H_k P_{k|k-1} H_k^T + R_k \tag{3.38}$$

which represents the so-called innovation covariance matrix. Now it is obvious that if we choose the Kalman gain as follows

$$K_k = P_{k|k-1}H_k^T Y_k^{-1} \tag{3.39}$$

than the a posteriori covariance matrix will be minimized. Now substituting this gain into 3.37 gives the minimal form of the a posteriori covariance matrix as follows

$$P_{k|k} = (I - K_k H_k)\, P_{k|k-1} \tag{3.40}$$

Now we need to express the a priori covariance matrix which is given as follows

$$\begin{aligned} P_{k|k-1} &= E\left\{\widetilde{x}_{k|k-1}\widetilde{x}_{k|k-1}^T\right\} \\ &= E\left\{\left(x_k - \widehat{x}_{k|k-1}\right)\left(x_k - \widehat{x}_{k|k-1}\right)^T\right\} \end{aligned} \tag{3.41}$$

Similarly, for this purpose we need to express the a priori error as follows

$$\begin{aligned} \widetilde{x}_{k|k-1} &= x_k - \widehat{x}_{k|k-1} \\ &= \Phi_{k,k-1}x_{k-1} + w_k - \Phi_{k,k-1}x_{k-1|k-1} \\ &= \Phi_{k,k-1}\widetilde{x}_{k-1|k-1} + w_k \end{aligned} \tag{3.42}$$

where we use the a priori estimate given as

$$\widehat{x}_{k|k-1} = \Phi_{k,k-1}\widehat{x}_{k-1|k-1} \tag{3.43}$$

which can be obtained by applying the expectation operator to the equation 3.21. Substituting 3.42 into 3.41 yields

$$\begin{aligned} P_{k|k-1} &= E\left\{\widetilde{x}_{k|k-1}\widetilde{x}_{k|k-1}^T\right\} \\ &= E\left\{\left(\Phi_{k,k-1}\widetilde{x}_{k-1|k-1} + w_k\right)\left(\Phi_{k,k-1}\widetilde{x}_{k-1|k-1} + w_k\right)^T\right\} \\ &= E\left\{\Phi_{k,k-1}\widetilde{x}_{k-1|k-1}\widetilde{x}_{k-1|k-1}^T\Phi_{k,k-1}^T + \Phi_{k,k-1}\widetilde{x}_{k-1|k-1}w_k^T\right. \\ &+ \left. w_k\widetilde{x}_{k-1|k-1}^T\Phi_{k,k-1}^T + w_k w_k^T\right\} \\ &= \Phi_{k,k-1}E\left\{\widetilde{x}_{k-1|k-1}\widetilde{x}_{k-1|k-1}^T\right\}\Phi_{k,k-1}^T + E\left\{w_k w_k^T\right\} \\ &= \Phi_{k,k-1}P_{k-1|k-1}\Phi_{k,k-1}^T + Q_k \end{aligned} \tag{3.44}$$

where we use the linearity of the expectation operator as in the previous case, so the assumption 3.29 can appear, and next we use the fact that the a priori estimation error is not correlated with the process noise.

Now we have everything what we need to summarize the equations for the conventional Kalman filter as follows

$$\begin{aligned}
\widehat{x}_{k|k-1} &= \Phi_{k,k-1}\widehat{x}_{k-1|k-1} && (3.45)\\
P_{k|k-1} &= \Phi_{k,k-1}P_{k-1|k-1}\Phi_{k,k-1}^T + Q_k && (3.46)\\
K_k &= P_{k|k-1}H_k^T\left(H_kP_{k|k-1}H_k^T + R_k\right)^{-1} && (3.47)\\
\widehat{x}_{k|k} &= \widehat{x}_{k|k-1} + K_k\left(z_k - H_k\widehat{x}_{k|k-1}\right) && (3.48)\\
P_{k|k} &= \left(I - K_kH_k\right)P_{k|k-1} && (3.49)
\end{aligned}$$

The initial conditions for the state vector and covariance matrix are $\widehat{x}_{0|-1}$ and $P_{0|-1}$ respectively, thus we need to set an initial estimate of the state with a corresponding degree of uncertainty.

3.4 Numerically Robust Kalman Filter Forms

The problem of numerical stability of the conventional Kalman filter is given mainly due to the form of equation 3.49, thus the form of the a posteriori covariance matrix. There we can see that for maintaining its positive definiteness the product of the Kalman gain and the observation matrix need to be less than the identity matrix. If we assume that we have a computer which has an infinite length of the computational word, then we can compute this product without any loss of the precision, thus this product is computed without any round-off error and is really less than the identity matrix. However, this is not realistic, so we need to consider the effect of rounding errors which arise due to a finite length of the computational word. These rounding errors are more significant as the machine epsilon, the value of least significant bit of the fractional part of the computational word, is greater. As the number of bits of the fractional part is smaller and numbers in the system model are expressed in very different units (the model is badly conditioned) a probability that the a posteriori covariance matrix become negative definite, indefinite or non-symmetric is greater. This can lead to a temporary divergence or to the suboptimal performance of the algorithm in a better case. In a worst case it can cause an absolute failure of the algorithm.

Several modifications of the conventional Kalman filter for avoiding these numerical problems were invented. All of these are based on some kind of the matrix factorization methods which can improve numerical properties of the covariance

matrix computation. Especially the Square Root and UD factorized Kalman filters are the most commonly used implementations. It is important to note, that all these methods are algebraically equivalent, but not numerically, which is very advantageous.

For the end of this opening we need to note that these numerical difficulties are not only the problem of the Kalman filter, but other leas-squares algorithms too. Generally, it is the problem of all least-squares algorithms, which uses the Riccati equation of the form given by equation 3.49.

The organization of this section is as follows. At the beginning we will look at the Joseph Stabilized Form of the Riccati equation. Next the Square Root filter with using the Potter's square-root measurement update step and the Modified Gram-Schmidt algorithm for the time update step will be shown. Finally we will look at the Bierman's UD factorized measurement update step together with the Modified Weighted Gram-Schmidt algorithm for the time update step. In the all subsequent sections the measurement time index will be expressed as the superscript $(-)$ and $(+)$, thus the a priori and a posteriori covariance matrices will be marked as P_k^- and P_k^+ respectively.

3.4.1 Joseph Stabilized Form

Through the derivation of the conventional Kalman filter in the previous section (the last row of equation 3.36) the form of the a posteriori covariance matrix, which is commonly known as the Joseph's stabilized form, was obtained as follows

$$P_k^+ = (I - K_k H_k)\, P_k^- \,(I - K_k H_k)^T + K_k R_k K_k^T \tag{3.50}$$

This expression is mathematically equivalent to the equation 3.49, however the numerical properties are different since we have a various number of arithmetic operations for each of these expressions. If we assume the previously mentioned rounding error which can make the result of the parentheses in 3.49 negative definite (theoretically impossible as we know from the Kalman filter derivation that the value of the criteria function after minimization is positive and non-zero), then the quadratic form of the first term of the equation 3.50 makes the resulting covariance matrix positive definite, even in a case of the negative definite parentheses. This assumption holds if the process noise covariance matrix R_k is positive definite, because if it is, then the result of the quadratic form of the second term of 3.50 is positive definite too. If we consider the second term of equation 3.50, then can be seen, that this term increases, each time step, the uncertainty of the estimated state. Hence, this term represents the linear absolute forgetting factor.

3.4.2 Square Root Filter

The idea of the square root filtering is based on the factorization of positive semi-definite matrix to the product of an upper and lower triangular matrices as follows

$$P = SS^T \tag{3.51}$$

where the factor S is generally non-unique, thus we can find numerous expressions for matrix S. As the diagonal elements of the covariance matrix represent variances, one can expect, that the diagonal entries of the factor S now represents the standard deviations. This is not true if the product S is of the Cholesky type. Therefore, from the Cholesky factorization can be simply derived an expression for obtaining the standard deviations. Now it is clear that if we will propagate through time the factor S only, instead of the covariance matrix as a whole, then we can be certain about the covariance matrix positive definiteness, even for the negative definite factor S. From this point, we can feel that the precision of the covariance matrix computation is now two times better than in the case of the conventional Kalman filter covariance matrix. This fact can be indicated by the condition number, which can be considered as some kind of a measure of the linear dependence of the matrix rows. The condition number κ of the matrix P can be computed as follows

$$\kappa(P) = \frac{\sigma_{max}(P)}{\sigma_{min}(P)} \tag{3.52}$$

where $\sigma_{max}(.)$ and $\sigma_{min}(.)$ are the maximum and minimum singular values of a matrix (.) respectively. The vector which contains the singular values of P can be expressed as

$$\sigma(P) = \sqrt{\lambda(PP^T)} \tag{3.53}$$

where $\lambda(.)$ represents eigenvalue vector. The above expression simply means that the singular values of the matrix are the absolute values of its eigenvalues. However, this statement holds for the symmetric matrices only. If the condition number approaches to a large value, then the matrix P is more ill-conditioned. On the other hand the best conditioning can be achieved when it is one. The relation between the condition number of P and S can be expressed as [1]

$$\kappa(P) = \kappa(SS^T) = (\kappa(S))^2 = \frac{\sigma^2_{max}(S)}{\sigma^2_{min}(S)} = \frac{\sigma_{max}(P)}{\sigma_{min}(P)} \tag{3.54}$$

which implies

$$\kappa(S) = \sqrt{\kappa(P)} \tag{3.55}$$

Now it is clear that we have a two times greater precision for the covariance matrix computation and we can state that we are able to use the single precision arithmetic instead of the double precision arithmetic, but this is only a theoretical statement, practically it is strongly dependent on the examined system model.

Time Update

Consider now that we have the a posteriori covariance matrix given as the product of its Cholesky factors as it is expressed by the following equation

$$P_k^+ = S_k^+ (S_k^+)^T \tag{3.56}$$

If we now substitute equation 3.56 into 3.46, then we can obtain the following relation for the a priory covariance matrix

$$\begin{aligned} P_k^- &= \Phi_k S_{k-1}^+ (S_{k-1}^+)^T \Phi_k^T + Q_k \\ &= \begin{bmatrix} \Phi_k S_{k-1}^+ & Q_k^{1/2} \end{bmatrix} \begin{bmatrix} \Phi_k S_{k-1}^+ & Q_k^{1/2} \end{bmatrix}^T \\ &= WW^T \end{aligned} \tag{3.57}$$

where W now represents a matrix of dimension $n \times (n+m)$ which is not equal to the triangular matrix S_k^-. The factor S_k^- can be obtained if we rewrite equation 3.57 as follows

$$\begin{aligned} P_k^- &= WW^T \\ &= \left(WV^T\right)\left(VW^T\right) \\ &= S_k^- (S_k^-)^T \end{aligned} \tag{3.58}$$

Where V is an orthogonal matrix of dimension $(n+m) \times (n+m)$ which represents a linear transformation between the following matrices

$$\begin{bmatrix} (S_k^-)^T \\ 0 \end{bmatrix} = V \begin{bmatrix} (S_{k-1}^+)^T \Phi_k^T \\ Q_k^{T/2} \end{bmatrix} \tag{3.59}$$

The task about finding the matrix V can be approached in several ways e.g. we can employ Householder transformation, Givens rotations, classical Gram-Schmidt algorithm or Modified Gram-Schmidt algorithm. The greatest numerical robustness from these algorithms has the MGS. So this algorithm, applied to the rows of the matrix W, is given for $j = n, n-1, ..., 1$ and $i = 1, 2, ..., j-1$ as follows[10]

$$\beta_j = \left\| w_j^{(n-j)} \right\|_2^2 \tag{3.60}$$

$$s_{jj}^- = \beta_j^{1/2} \tag{3.61}$$

$$v_j = w_j^{(n-j)} / s_{jj}^- \tag{3.62}$$

$$s_{ij}^- = \langle w_i, v_j \rangle \tag{3.63}$$

$$w_i^{(n-j+1)} = w_i^{(n-j)} - s_{ij}^- v_j \tag{3.64}$$

where $\|\cdot\|_2$ represents the Euclidean norm and $\langle \cdot, \cdot \rangle$ is the dot product.

Measurement Update

Similarly as in the time-update step we start the derivation with a consideration that we have the a priory covariance matrix expressed as the product of the Cholesky factors as follows

$$P_k^- = S_k^-(S_k^-)^T \tag{3.65}$$

Substituting this product into the equation 3.49 leads to the a posteriori covariance matrix written as

$$\begin{aligned} P_k^+ &= S_k^-(S_k^-)^T - S_k^-(S_k^-)^T H_k^T \left(H_k S_k^-(S_k^-)^T H_k^T + R_k\right)^{-1} H_k S_k^-(S_k^-)^T \\ &= S_k^- \left(I - (S_k^-)^T H_k^T \left(H_k S_k^-(S_k^-)^T H_k^T + R_k\right)^{-1} H_k S_k^-\right)(S_k^-)^T \\ &= S_k^- \left(I - v\left(v^T v + r_k\right)^{-1} v^T\right)(S_k^-)^T \\ &= S_k^- \left(I - \alpha^{-1} v v^T\right)(S_k^-)^T \\ &= S_k^- W W^T (S_k^-)^T \end{aligned} \tag{3.66}$$

where we place the following substitutions

$$v = (S_k^-)^T H_k^T \tag{3.67}$$

$$\alpha = v^T v + r_k \tag{3.68}$$

$$W W^T = I - \alpha^{-1} v v^T \tag{3.69}$$

It is important to note that we change in the third row of equation 3.66 the matrix R_k for the scalar r_k. This simplification means that we will be able to compute with the scalar measurements only, which is not restrictive, because we can process the measurement-update sequentially, thus one diagonal entry of R_k with corresponding entry from z_k at a given time. This is very advantageous because we do not need to compute an inverse of the innovation matrix, which substantially increase the computational time of the Kalman filter. The main disadvantage in computing the inverse of the innovation matrix lies in the fact that this process makes the resulting covariance matrix numerically badly conditioned in such a case, where the length of the computational word is short. Hence, the sequential measurement-update brings more numerical robustness into the Kalman filter computation. However, this approach requires the diagonal matrix R_k. The diagonalization procedure can be performed before an implementation of the algorithm, but only for cases, where R_k is time-invariant.

The problem of factoring the a posteriori covariance matrix is now given by searching for an expression of the factor W. This can be done by rewriting the equation 3.69 as follows

$$W W^T = \left(I - \gamma \alpha^{-1} v v^T\right)\left(I - \gamma \alpha^{-1} v v^T\right)^T \tag{3.70}$$

where we introduce a variable γ, which need to be found. For this purpose we make the following treatment

$$\begin{aligned} I - \alpha^{-1} v v^T &= I - 2\gamma\alpha^{-1} v v^T + \gamma^2 \alpha^{-2} v v^T v v^T \\ 0 &= \gamma^2 \alpha^{-2} \beta v v^T + 2\gamma\alpha^{-1} v v^T + \alpha^{-1} v v^T \\ 0 &= \left(\gamma^2 \alpha^{-1} \beta + 2\gamma + 1\right) \alpha^{-1} v v^T \\ 0 &= \gamma^2 \alpha^{-1} \beta + 2\gamma + 1 \end{aligned} \tag{3.71}$$

where we can obtain, after some arrangements, the following two solutions

$$\gamma_{1,2} = \frac{1}{1 \mp \sqrt{r_k \alpha}} \tag{3.72}$$

Only the positive expression is important for us, because if $r_k \alpha \to 1$ then $\gamma \to \infty$. Rewriting the expression 3.66 as follows

$$S_k^+ (S_k^+)^T = S_k^- W W^T (S_k^-)^T \tag{3.73}$$

yields the final term for updating the factor of the a posteriori covariance matrix

$$\begin{aligned} S_k^+ &= S_k^- W \\ &= S_k^- \left(I - \gamma\alpha^{-1} v v^T\right) \\ &= S_k^- - \gamma\alpha^{-1} S_k^- v v^T \\ &= S_k^- - \gamma K_k v^T \end{aligned} \tag{3.74}$$

where the Kalman gain K_k is expressed as

$$K_k = \alpha^{-1} S_k^- v \tag{3.75}$$

The Potter's Square Root measurement-update can be summarized as follows

$$v = (S_k^-)^T H_k^T \tag{3.76}$$

$$\alpha = v^T v + r_k \tag{3.77}$$

$$K_k = \alpha^{-1} S_k^- v \tag{3.78}$$

$$x_k^+ = x_k^- + K_k \left(z_k - H_k x_k^-\right) \tag{3.79}$$

$$\gamma = \frac{1}{1 + \sqrt{r_k \alpha}} \tag{3.80}$$

$$S_k^+ = S_k^- - \gamma K_k v^T \tag{3.81}$$

If one need to initialize the factor S_0^-, then the Cholesky decomposition need to be used. The estimated state vector is initialized as same as in the case of the conventional Kalman filter.

3.4.3 UD Filter

The main disadvantage of the Square Root filter is the need for the square root computation, which brings more computational burden and some difficulties from an implementation point of view. These drawbacks can be avoided by introducing the UD factorization of any positive semi-definite matrix as follows

$$P = UDU^T \tag{3.82}$$

where U is an upper triangular matrix with unit diagonal entries and D is a diagonal matrix. Both of these are assumed as square matrices of the same dimension. Now one can ask how we obtain a two times greater precision with this expression. The answer is expressed by the following relation

$$P = (U\sqrt{D})(U\sqrt{D})^T = SS^T \tag{3.83}$$

which gives the relation for the condition number as

$$\kappa(P) = \kappa((U\sqrt{D})(U\sqrt{D})^T) = (\kappa(U\sqrt{D}))^2 = \frac{\sigma^2_{max}(U\sqrt{D})}{\sigma^2_{min}(U\sqrt{D})} = \frac{\sigma_{max}(P)}{\sigma_{min}(P)} \tag{3.84}$$

where the terms $\sigma_{max}(.)$ and $\sigma_{min}(.)$ represent the maximum and minimum singular values of a matrix (.) respectively. Hence, the condition number of product $U\sqrt{D}$ is

$$\kappa(U\sqrt{D}) = \sqrt{\kappa(P)} \tag{3.85}$$

Now finding an expressions for propagating the a priori and a posteriori covariance matrices in the terms of 3.82 leads to an improved version of the previously expressed Square Root filter, however without the square root computation.

Time Update

Similarly as with the Square Root filter we firstly make an assumption that we have the a posteriori covariance matrix expressed in the factorized terms, but now we consider the terms of the UD factorization as follows

$$P_k^+ = U_k^+ D_k^+ (U_k^+)^T \tag{3.86}$$

where U_k^+ is the upper-triangular matrix of the dimension $n \times n$ with unit diagonal entries. The matrix D_k^+ is diagonal and of the dimension $n \times n$. Again we need to note that we need to employ the Cholesky UD factorization for obtaining the standard deviations because the diagonal entries of factor D_k^+ do not directly interpret the variances. Substituting now equation 3.86 into 3.46 yields

$$\begin{aligned} P_k^- &= \Phi_k U_{k-1}^+ D_{k-1}^+ (U_{k-1}^+)^T \Phi_k^T + Q_k \\ &= \begin{bmatrix} \Phi_k U_{k-1}^+ & I \end{bmatrix} \begin{bmatrix} D_{k-1}^+ & 0 \\ 0 & Q_k \end{bmatrix} \begin{bmatrix} \Phi_k U_{k-1}^+ & I \end{bmatrix}^T \\ &= W \bar{D} W^T \end{aligned} \tag{3.87}$$

where the block-diagonal matrix $\bar{D}$ is of dimension $(n+m) \times (n+m)$. The matrix W is not interchangeable for the factor U_k^- as before and is of the dimension $n \times (n+m)$. Incorporating again the orthogonal transformation matrix V, which is now of dimension $n \times (n+m)$, leads to another expression for the a priori covariance matrix as follows

$$\begin{aligned} P_k^- &= \left(U_k^- V\right) \bar{D} \left(U_k^- V\right)^T \\ &= U_k^- \left(V \bar{D} V^T\right) (U_k^-)^T \\ &= U_k^- D_k^- (U_k^-)^T \end{aligned} \tag{3.88}$$

which yields

$$U_k^- V = \left[\begin{array}{cc} \Phi_k U_{k-1}^+ & I \end{array} \right] \tag{3.89}$$

where we need to find the matrix V as before.

In the second row of the equation 3.88 can be seen that $D_k^- = V \bar{D} V^T$, thus the diagonal entries d_{jj}^- are expressed as $v_j \bar{D} v_j^T$ for $j = 1, 2, ..., n$ (where we assume v_j as a row vector of the matrix V), which is nothing else than the weighted dot product. Assuming this type of dot product the following Weighted Modified Gram-Schmidt algorithm, applied to the rows of W, can be defined for $j = n, n-1, ..., 1$ and $i = 1, 2, ..., j-1$ as follows

$$d_{jj}^- = \left\| w_j^{(n-j)} \right\|_{\bar{D}}^2 \tag{3.90}$$

$$u_{ij}^- = \left\langle w_i, w_j \right\rangle / d_{jj}^- \tag{3.91}$$

$$w_i^{(n-j+1)} = w_i^{(n-j)} - u_{ij}^- w_j \tag{3.92}$$

where $\|\cdot\|_{\bar{D}}$ represents the Euclidean norm weighted by the matrix $\bar{D}$ and $\langle \cdot, \cdot \rangle$ is the standard dot product.

Measurement Update

If we assume the following expression for the a priori covariance matrix

$$P_k^- = U_k^- D_k^- (U_k^-)^T \tag{3.93}$$

where U_k^- is the upper-triangular matrix and D_k^- is the diagonal matrix, both of the dimension $n \times n$, then the a posteriori covariance matrix 3.49 can be rewritten as

$$\begin{aligned} P_k^+ &= U_k^- D_k^- (U_k^-)^T - U_k^- D_k^- (U_k^-)^T H_k^T \left(H_k U_k^- D_k^- (U_k^-)^T H_k^T + R_k\right)^{-1} H_k U_k^- D_k^- (U_k^-)^T \\ &= U_k^- \left(D_k^- - D_k^- (U_k^-)^T H_k^T \left(H_k U_k^- D_k^- (U_k^-)^T H_k^T + R_k\right)^{-1} H_k U_k^- D_k^-\right) (U_k^-)^T \\ &= U_k^- \left(D_k^- - D_k^- f \left(f^T D_k^- f + r_k\right)^{-1} f^T D_k^-\right) (U_k^-)^T \\ &= U_k^- \left(D_k^- - \alpha^{-1} v v^T\right) (U_k^-)^T \\ &= U_k^- \left(\bar{U} \bar{D} \bar{U}^T\right) (U_k^-)^T \\ &= \left(U_k^- \bar{U}\right) \bar{D} \left(U_k^- \bar{U}\right)^T \end{aligned} \tag{3.94}$$

where the following terms were introduced

$$f = (U_k^-)^T H_k^T \tag{3.95}$$

$$v = D_k^- f \tag{3.96}$$

$$\alpha = f^T v + r_k \tag{3.97}$$

The problem of factorizing the matrix expression 3.94 is now given as

$$\bar{U}\bar{D}\bar{U}^T = D_k^- - \alpha^{-1} v v^T \tag{3.98}$$

Obtaining the factors $\bar{U}$ and $\bar{D}$ leads to a general recursion of the UD measurement-update

$$\begin{array}{rcl} U_k^+ & = & U_k^- \bar{U} \\ D_k^+ & = & \bar{D} \end{array} \tag{3.99}$$

Similarly as with the Square Root Filter are there several approaches how to obtain factors of the equation 3.98. Bierman was presented in [12] how to obtain a solution for updating the terms U_k^+ and D_k^+ directly as follows

$$f = (U^-)^T H^T \tag{3.100}$$

$$v = D^- f \tag{3.101}$$

$$\alpha_j = \alpha_{j-1} + f_j v_j \tag{3.102}$$

$$d_j^+ = d_j^- \frac{\alpha_{j-1}}{\alpha_j} \tag{3.103}$$

$$b_j = v_j \tag{3.104}$$

$$p_j = -\frac{f_j}{\alpha_{j-1}} \tag{3.105}$$

$$U_{ij}^+ = U_{ij}^- + b_i p_j \tag{3.106}$$

$$b_i := b_i + U_{ij}^+ v_j \tag{3.107}$$

$$K = \frac{b}{\alpha_n} \tag{3.108}$$

where $j = 1, 2...., n$ and $i = 1, 2, ..., j-1$. It is crucial to set $\alpha_0 = r$ before we go through the first iteration. If we have l-dimensional observation vector, then the equations 3.100-3.108 need to be performed l-times. An initialization of the factors U_0^- and D_0^- need to be performed by the Cholesky UDU^T decomposition.

4 AIDED INERTIAL NAVIGATION

4.1 Introduction

It is clear form the title that an aided inertial navigation system is based on using inertial sensors, thus on using some accelerometers and gyroscopes. Therefore, as the accelerometer measure an acceleration, we need to perform a time integration of its output for obtaining a velocity. As we have the velocity, we need to perform another time integration for obtaining a position. Similarly an output of the gyroscope need to be integrated for obtaining an attitude. The task related to integrating these outputs is difficult because all measurements are obtained with some degree of uncertainty. This uncertainty is represented by a measurement noise, additional error, scale factor error, non-linearity etc. All these effects are integrated together with useful information. Therefore, integrating the sensor's outputs becomes a very critical part of the navigation process, because as more as uncertain the output of the sensors is, the navigation system's error grows more quickly. A way how to reduce this error lies in a modelling of the previously mentioned effects. If we are able to estimate values of these effects, then we can use them for correcting the outputs of the inertial sensors. Unfortunately, assuming this corrections, it is not possible to cover it in all, and although the navigation system's error grows more slightly than before, it is still an unbounded grow. If we need to make the navigation system's error bounded, then an external aiding, with bounded error, need to be incorporated. One of the most popular aiding source is the Global Navigation Satellite System (GNSS). Although the GNSS measurements are erroneous, these errors are not grow over time. The measurements of the GNSS and the inertial sensors are then fused together, which makes the inertial system's error bounded. The obvious question can appear, so why we do not use the GNSS only? The answer is, the GNSS measurements can be significantly noisy and the sample rate is low. However, a combination of the GNSS and the inertial sensor's measurements leads to a high-rate and more accurate navigation system's results.

There are several approaches how to fuse data from the GNSS and the inertial sensors. These approaches are divided according to a degree of coupling between the GNSS and the inertial navigation system into four categories, which are named as the Loosely Coupled Integration, Tightly Coupled Integration, Deep Integration and Ultra Deep Integration. This is a general division and all others can be considered as a combination of these approaches or something between them. All these are very good described in [13].

The organization of this chapter is as follows. For support our next effort, we first describe a number of coordinate frames, which detailed knowledge is crucial for

understanding the navigation algorithm. Next we focus our attention to a derivation of the kinematic equations of motion. From these equations we can next derive a trajectory (sensor data) generator. Last part of this chapter is focused to a derivation of navigation error equations with their corresponding error state space model. These sections are based on the author's studies of [15] [16] [20] [21].

4.2 Coordinate Frames and Earth Model

For describing a movement of target in the vicinity of the Earth is suitable to use a number of coordinate frames. If we do so, the mathematical model of the inertial navigation becomes, in some sense, modular. This modularity is a suitable property as it can be used for defining a number of inertial navigation algorithms with a possibility of extensions. This extension can be for example the so-called wander azimuth coordinate frame, which can be used for avoiding the singularity as it arises in the local navigation frame due to the vehicle's movement through the poles of the Earth.

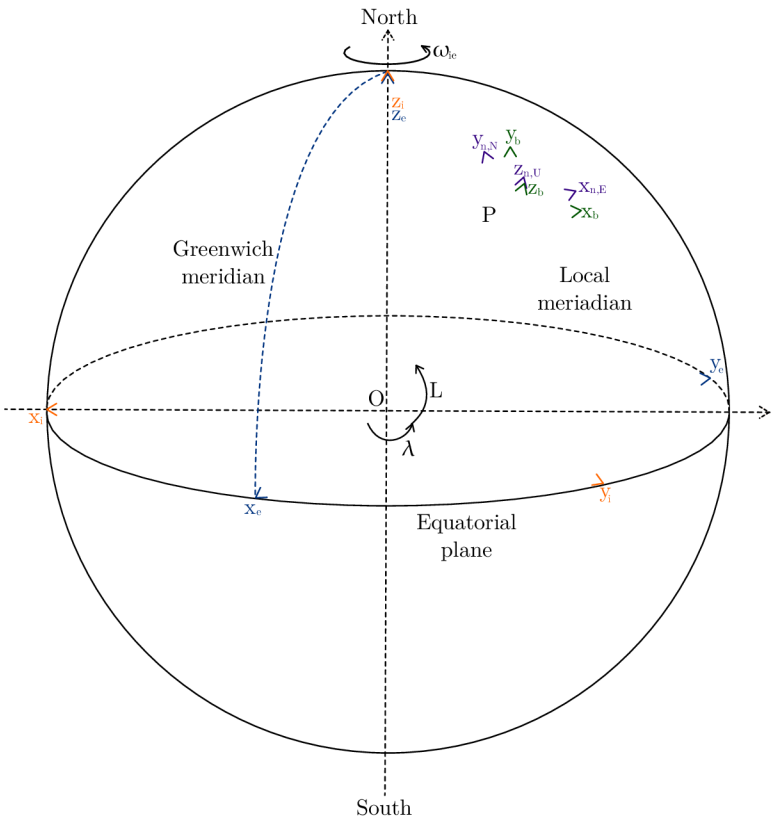


Fig. 4.1: Relations between the Earth Centered Inertial (ECI) frame, Earth Centered Earth Fixed (ECEF) frame, local navigation frame and body frame.

4.2.1 Coordinate Frames Description

Earth Centered Inertial (ECI) frame (i-frame) - an orthogonal set of axes denoted by triad $[x_i, y_i, z_i]$ as depicted in figure 4.1. This orthogonal frame has its origin at the centre of the Earth and is nonrotating and nonaccelerating with respect to the rest of the Universe. This assumption is not considered with an acceleration due to the rotation of the Earth around the Sun and the Galaxy rotation because it is not measurable. The inertial sensors measure a motion just with respect to the inertial frame.

Earth Centered Earth Fixed (ECEF) frame (e-frame) - similarly it is a set of orthogonal axes, which are denoted by triad $[x_e, y_e, z_e]$ as depicted in figure 4.1. An origin of this frame is placed at the centre of the Earth and its axes are fixed to the Earth, hence they rotate with respect to the inertial frame. The axis x_e intersects the Greenwich meridian or the conventional meridian where zero degree of the longitude is defined. An angular rate of the ECEF frame with respect to the ECI frame is denoted as ω_{ie}.

Local navigation frame (n-frame) - an orthogonal axes set with an origin at the point P as depicted in figure 4.1. The axes are denoted by triad $[x_n, y_n, z_n]$. More commonly used notation, especially between navigation engineers is $[E, N, U]$, which are the abbreviations of east, north and up. In some cases we can find a definition by triad $[N, E, D]$ (down), which involve a different orientation of the axes. For the purposes of this work the first option will be considered. The origin of the n-frame is placed at the curve which is defined by a local meridian plane. The axis $y_{n,N}$ is always pointed to the north pole of the Earth. For that reason, there is a possibility to gain an infinite speed when e.g. an aircraft flight over the north or south pole. This lack can by avoided by a definition of the so-called wander azimuth frame, as was written in the introductory of this section, but for now we will not consider this option due to simplification.

Body frame (b-frame) - this axes set is defined by triad $[x_b, y_b, z_b]$ and has an origin at the same place as the n-frame i.e. at the point P. The axes are aligned with the roll, pitch and yaw axes of an aircraft. The inertial navigation system is commonly installed in the b-frame. This is not strict, so it can be installed elsewhere, but this include another transformation needed for computing the navigation solution, which involves higher computational burden.

4.2.2 Earth Model

The Earth model defines a number of important constants and variables, which are necessary for running the navigation algorithm. The Earth itself is a very complex shape in a detail. Modelling all aspects of this shape is very difficult, therefore its surface is approximated by an ellipsoid, whose radius is defined by the mean sea level. The necessary constants, which are used through this text are defined in the table 4.1.

R_e	e	ω_{ie}
6378140[m]	0.00335281066475[-]	7.2921151467e-5[rad/s]

Tab. 4.1: Definition of the Earth model constants. R_e is the main radius of the Earth, e is the eccentricity of the Earth and ω_{ie} is the angular rate of the Earth.

A two important radii are needed for computing the time change of the latitude and longitude. First one R_N is the radius of curvature, known as the meridian radius for the north-south motion.

$$R_N(L) = \frac{R_e(1-e^2)}{(1-e^2\sin^2 L)^{3/2}} \tag{4.1}$$

And the second one R_E is the transverse radius of curvature for the east-west motion, written as

$$R_E(L) = \frac{R_e}{(1-e^2\sin^2 L)^{1/2}} \tag{4.2}$$

Both of this change its values when the object varies its latitude as moves in the vicinity of the Earth.

An angular rate of change of the latitude is obtained from figure 4.1 as the fraction of the translational velocity in the north direction and the sum of Earth's meridian radius with a height above the Earth's surface as follows

$$\dot{L} = \frac{v^n_{en,N}}{R_N(L)+h} \tag{4.3}$$

Similarly, the time change of longitude is obtained from figure 4.1 as the fraction of the east translational velocity and sum of the transverse radius of the Earth and the height as

$$\dot{\lambda} = \frac{v^n_{en,E}}{(R_E(L)+h)\cos L} \tag{4.4}$$

where the sum in the denominator changes its value with a change of the cosine of latitude. This is obvious directly from the geometry of figure 4.1 and the Pythagorean theorem.

4.2.3 Coordinate Transformations

The coordinate transformation matrices are useful when we need to transform a vector expressed in some frame to a vector expressed in another frame. Main two transformation matrices used through this text are C_e^n direction cosine matrix, which transform a vector expressed in the e-frame (ECEF) to a vector expressed in the n-frame, and C_n^b, which transform vectors between the n-frame and the b-frame.

First of these is obtained by rotating the n-frame from its starting position by the longitude angle λ around the z-axis and next rotating by the latitude angle L around the y-axis. At last, the first matrix in the expression 4.5 is used for changing the axes order from [E,U,N] to [E,N,U].

$$\begin{aligned} C_e^n &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} cL & 0 & sL \\ 0 & 1 & 0 \\ -sL & 0 & cL \end{bmatrix} \begin{bmatrix} c\lambda & s\lambda & 0 \\ -s\lambda & c\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} -s\lambda & c\lambda & 0 \\ -sLc\lambda & -sLs\lambda & cL \\ cLc\lambda & cLs\lambda & sL \end{bmatrix} \end{aligned} \tag{4.5}$$

Second of these can by obtained in the similar way as before. We start with a rotation of the b-frame around its z-axis by the yaw angle, next we need a rotation around the x-axis by the pitch angle and the last rotation is around the y-axis by the roll angle as follows

$$\begin{aligned} C_n^b &= C_3 C_2 C_1 \\ &= \begin{bmatrix} c\gamma & 0 & -s\gamma \\ 0 & 1 & 0 \\ s\gamma & 0 & c\gamma \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta & s\theta \\ 0 & -s\theta & c\theta \end{bmatrix} \begin{bmatrix} c\psi & s\psi & 0 \\ -s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} c\gamma c\phi - s\gamma s\theta s\psi & -c\theta s\psi & s\gamma c\psi + c\gamma s\theta s\psi \\ c\gamma s\phi + s\gamma s\theta c\psi & c\theta c\psi & s\gamma s\psi - c\gamma s\theta c\psi \\ -s\gamma c\theta & s\theta & c\gamma c\theta \end{bmatrix} \end{aligned} \tag{4.6}$$

where the indexes in the terms of right-hand side of 4.6 denote the order of rotations. Now, if we look at this matrix, then we can see, that the all angles can be computed back from the entries, which are expressed only by the two (e.g. $c\theta c\phi$) or less terms (i.e. $s\theta$).

4.3 Navigation System Equations

This section deals with a derivation of inertial navigation system mechanization equations. Firstly we start with a velocity equation, which describe a movement

of a target expressed in the local navigation frame. Next is described an attitude equation of the object's body frame which is related to the navigation frame. Further, a position equation of the target, expressed in the geodetic coordinates i.e. in the terms of latitude, longitude and altitude, is presented. As was mentioned in the introductory of this chapter, the sensor's measurement comprises errors, so the final part of this section briefly describes equations which are commonly used for their modelling. At this point is important to note, that the time indexes in the all following subsection, are omitted due to a simplification of the notation.

4.3.1 Velocity Equation

Let's start a derivation of the velocity time propagation equation with the following relation

$$v_{en}^{n} = C_{e}^{n}\dot{r}_{en}^{e} \tag{4.7}$$

which describes the transformation of the e-frame expressed velocity to the n-frame expressed velocity. Taking the time derivative of this equation yields

$$\dot{v}_{en}^{n} = C_{e}^{n}\ddot{r}_{en}^{e} + \dot{C}_{e}^{n}\dot{r}_{en}^{e} \tag{4.8}$$

where the r_{en}^{e} term is defined as

$$r_{en}^{e} = C_{i}^{e}r_{en}^{i} \tag{4.9}$$

and its first time derivative as

$$\begin{aligned} \dot{r}_{en}^{e} &= C_{i}^{e}\dot{r}_{en}^{i} + \dot{C}_{i}^{e}r_{en}^{i} \\ &= C_{i}^{e}\left(\dot{r}_{en}^{i} - \Omega_{ie}^{i}r_{en}^{i}\right) \end{aligned} \tag{4.10}$$

Previous relation uses the time propagation equation for the transformation matrix C_{i}^{e} as follows

$$\dot{C}_{i}^{e} = -C_{i}^{e}\Omega_{ie}^{i} \tag{4.11}$$

where Ω_{ie}^{i} is the skew-symmetric matrix of the e-frame to i-frame angular rate expressed in the i-frame. The second time derivative of 4.9 gives

$$\begin{aligned} \ddot{r}_{en}^{e} &= \dot{C}_{i}^{e}\left(\dot{r}_{en}^{i} - \Omega_{ie}^{i}r_{en}^{i}\right) + C_{i}^{e}\left(\ddot{r}_{en}^{i} - \dot{\Omega}_{ie}^{i}r_{en}^{i} - \Omega_{ie}^{i}\dot{r}_{en}^{i}\right) \\ &= -C_{i}^{e}\Omega_{ie}^{i}\left(\dot{r}_{en}^{i} - \Omega_{ie}^{i}r_{en}^{i}\right) + C_{i}^{e}\left(\ddot{r}_{en}^{i} - \Omega_{ie}^{i}\dot{r}_{en}^{i}\right) \\ &= C_{i}^{e}\left(\ddot{r}_{en}^{i} - 2\Omega_{ie}^{i}\dot{r}_{en}^{i} + \Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i}\right) \end{aligned} \tag{4.12}$$

substituting both of these time derivatives into the 4.8 yields

$$\begin{aligned} \dot{v}_{en}^{n} &= C_{e}^{n}C_{i}^{e}\left(\ddot{r}_{en}^{i} - 2\Omega_{ie}^{i}\dot{r}_{en}^{i} + \Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i}\right) - \Omega_{en}^{n}C_{e}^{n}\dot{r}_{en}^{e} \\ &= C_{i}^{n}\left(\ddot{r}_{en}^{i} - 2\Omega_{ie}^{i}C_{n}^{i}\dot{r}_{en}^{n} - 2\Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i} + \Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i}\right) - \Omega_{en}^{n}\dot{r}_{en}^{n} \\ &= C_{i}^{n}\left(\ddot{r}_{en}^{i} - 2\Omega_{ie}^{i}C_{n}^{i}\dot{r}_{en}^{n} - \Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i}\right) - \Omega_{en}^{n}\dot{r}_{en}^{n} \\ &= C_{i}^{n}\left(\ddot{r}_{en}^{i} - \left(\Omega_{in}^{i} + 2\Omega_{ie}^{i}\right)C_{n}^{i}\dot{r}_{en}^{n} - \Omega_{ie}^{i}\Omega_{ie}^{i}r_{en}^{i}\right) \end{aligned} \tag{4.13}$$

where in the first row we use a time propagation equation of the transformation matrix C_e^n as follows

$$\dot{C}_e^n = -\Omega_{en}^n C_e^n \tag{4.14}$$

and the expression for the time derivative of r_{en}^i as

$$\begin{aligned} \dot{r}_{en}^i &= \dot{C}_e^i r_{en}^e + C_e^i \dot{r}_{en}^e \\ &= \Omega_{ie}^i r_{en}^i + C_e^i \dot{r}_{en}^e \end{aligned} \tag{4.15}$$

which can be obtained from the equation 4.9. The skew-symmetric matrix Ω_{en}^n in 4.14 is represented by the n-frame to e-frame angular velocity expressed in the n-frame.

The specific force, measured by the accelerometer, is obtained as a combination of an inertial acceleration, gravity and the Earth's centripetal acceleration

$$f_{en}^n = \ddot{r}_{en}^n - g_{en}^n - \Omega_{ie}^n \Omega_{ie}^n r_{en}^n \tag{4.16}$$

Expressing the first right-hand side term of 4.16 in the inertial frame yields

$$\begin{aligned} C_i^n \ddot{r}_{en}^i &= f_{en}^n + g_{en}^n + C_i^n \Omega_{ie}^n \Omega_{ie}^n C_n^i r_{en}^n \\ \ddot{r}_{en}^i &= C_n^i \left(f_{en}^n + g_{en}^n \right) + \Omega_{ie}^i \Omega_{ie}^i r_{en}^i \end{aligned} \tag{4.17}$$

then by substituting this equation into 4.13, the final relation for the velocity time propagation can be written as:

$$\dot{v}_{en}^n = C_b^n f_{ib}^b - \left(\Omega_{en}^n + 2\Omega_{ie}^n \right) v_{en}^n + g_{en}^n \tag{4.18}$$

Now we need to perform a derivation of the angular rate terms used in the previous equations. For expressing the term ω_{en}^n we need firstly introduce a temporary angular rate of the n-frame with respect to the e-frame expressed in the n-frame as

$$\omega_{en'}^{n'} = \begin{bmatrix} cL & 0 & sL \\ 0 & 1 & 0 \\ -sL & 0 & cL \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\lambda} \end{bmatrix} + \begin{bmatrix} 0 \\ -\dot{L} \\ 0 \end{bmatrix} = \begin{bmatrix} sL\dot{\lambda} \\ -\dot{L} \\ cL\dot{\lambda} \end{bmatrix} \tag{4.19}$$

the order of scalars in this vector is organized as $[U, E, N]$, so for change this arrangement into $[E, N, U]$, the other transformation need to be included as follows

$$\omega_{en}^n = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \omega_{en'}^{n'} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} sL\dot{\lambda} \\ -\dot{L} \\ cL\dot{\lambda} \end{bmatrix} = \begin{bmatrix} -\dot{L} \\ cL\dot{\lambda} \\ sL\dot{\lambda} \end{bmatrix} = \begin{bmatrix} -\frac{v_{en,N}^n}{R_N(L)+h} \\ \frac{v_{en,E}^n}{R_E(L)+h} \\ \frac{v_{en,E}^n \tan L}{R_E(L)+h} \end{bmatrix} \tag{4.20}$$

The angular rate of the e-frame with respect to the i-frame expressed in the n-frame can be obtained as

$$\omega_{ie}^n = C_e^n \begin{bmatrix} 0 \\ 0 \\ \omega_{ie} \end{bmatrix} = \omega_{ie} \begin{bmatrix} 0 \\ cL \\ sL \end{bmatrix} \tag{4.21}$$

where the transformation matrix C_e^n is defined by equation 4.5.

At this point we have everything what we need for expressing the discrete-time version of equation 4.18 as follows

$$v^n_{en,t_0+\Delta t} = v^n_{en,t_0} + \int_{t_0}^{t_0+\Delta t} \left(f^n_{ib,t} - \left(\Omega^n_{en,t} + 2\Omega^n_{ie,t} \right) v^n_{en,t} + g^n_{en,t} \right) dt \tag{4.22}$$

where we introduce the time indexes into 4.18. If we assume that the integrand in the equation 4.22 is constant between the time steps then the following approximation can be made

$$v^n_{en,k+1} = v^n_{en,k} + \left(f^n_{ib,k} - \left(\Omega^n_{en,k} + 2\Omega^n_{ie,k} \right) v^n_{en,k} + g^n_{en,k} \right) \Delta t \tag{4.23}$$

where Δt represents the sample time period.

4.3.2 Attitude Equation

The time propagation equation of the attitude, for small angle deviations between the time steps, can be expressed as

$$\dot{C}^n_b = C^n_b \Omega^b_{nb} \tag{4.24}$$

where Ω^b_{nb} is the skew-symmetric matrix of the b-frame to n-frame angular rate expressed in the b-frame. If we assume, that the angular rate of the gyroscope is measured as a sum of the angular rate of the b-frame with respect to the n-frame and an angular rate of the n-frame with respect to the i-frame as follows

$$\omega^b_{ib} = \omega^b_{nb} + \omega^b_{in} \tag{4.25}$$

then we can express the skew-symmetric matrix Ω^b_{nb} as

$$\begin{aligned} \Omega^b_{nb} &= \Omega^b_{ib} - \Omega^b_{in} \\ &= \Omega^b_{ib} - C^b_n \Omega^n_{in} C^n_b \end{aligned} \tag{4.26}$$

The angular rate of the n-frame with respect to the i-frame is defined as a sum of the angular rate of the e-frame with respect to the i-frame, thus as the Earth's angular rate, and the angular rate of the n-frame with respect to the e-frame. So the last equation can by rewritten as

$$\Omega^b_{nb} = \Omega^b_{ib} - C^b_n \left(\Omega^n_{ie} + \Omega^n_{en} \right) C^n_b \tag{4.27}$$

Now the discrete-time equivalent of equation 4.24 can be obtained as follows

$$C^n_{b,t_0+\Delta t} = C^n_{b,t_0} e^{\int_{t_0}^{t_0+\Delta t} \Omega^b_{nb} dt} = C^n_{b,t_0} e^{(\alpha^b_{nb}\times)} = C^n_{b,t_0} \sum_{i=0}^{\infty} \frac{(\alpha^b_{nb}\times)}{i!} = C^n_{b,t_0} C^{b,t_0}_{b,t_0+\Delta t} \tag{4.28}$$

where the Taylor series expansion for the term $e^{(\alpha^n_{nb}\times)}$ can be rewritten as (inspired by [15] p.138)

$$\begin{aligned} e^{(\alpha^b_{nb}\times)} &= \textstyle\sum_{i=0}^{\infty} \frac{(\alpha^b_{nb}\times)}{i!} \\ &= I_{3\times3} + (\alpha^b_{nb}\times) + \frac{(\alpha^b_{nb}\times)^2}{2} + \frac{(\alpha^b_{nb}\times)^3}{6} + \frac{(\alpha^b_{nb}\times)^4}{24} + \dots \\ &= I_{3\times3} + \left(1 - \frac{\left\|\alpha^b_{nb}\right\|^2}{6} + \dots\right)(\alpha^b_{nb}\times) + \left(\frac{1}{2} - \frac{\left\|\alpha^b_{nb}\right\|^2}{24} + \dots\right)(\alpha^b_{nb}\times)^2 \\ &= I_{3\times3} + \frac{1}{\left\|\alpha^b_{nb}\right\|}\left(\textstyle\sum_{i=0}^{\infty}(-1)^i \frac{\left\|\alpha^b_{nb}\right\|^{2i+1}}{(2i+1)!}\right)(\alpha^b_{nb}\times) + \\ &+ \frac{1}{\left\|\alpha^b_{nb}\right\|^2}\left(1 - \textstyle\sum_{i=0}^{\infty}(-1)^i \frac{\left\|\alpha^b_{nb}\right\|^{2i}}{(2i)!}\right)(\alpha^b_{nb}\times)^2 \\ &= I_{3\times3} + \frac{\sin\left\|\alpha^b_{nb}\right\|}{\left\|\alpha^b_{nb}\right\|}(\alpha^b_{nb}\times) + \frac{1-\cos\left\|\alpha^b_{nb}\right\|}{\left\|\alpha^b_{nb}\right\|^2}(\alpha^b_{nb}\times)^2 \end{aligned} \tag{4.29}$$

where we use $(\alpha^b_{nb}\times)^3 = -\left\|\alpha^b_{nb}\right\|^2(\alpha^b_{nb}\times)$ and $(\alpha^b_{nb}\times)^4 = -\left\|\alpha^b_{nb}\right\|^2(\alpha^b_{nb}\times)^2$. The notation $\|.\|$ express the Euclidean norm. Finally, the relation for the discrete-time attitude DCM update is as follows

$$C^n_{b,k+1} = C^n_{b,k} C^{b,k}_{b,k+1} \tag{4.30}$$

where $C^{b,k}_{b,k+1}$ is given by the last row of the equation 4.29.

4.3.3 Position Equation

The time propagation of the position equation, expressed in the n-frame, can by computed as

$$\dot{r}^n_{en} = v^n_{en} - \omega^n_{en} \times r^n_{en} \tag{4.31}$$

but because we need to obtain the position in the geodetic coordinates the resulting equation is obtained by using the coordinate transformation matrix as follows

$$\dot{r}_n = T v^n_{en} \tag{4.32}$$

where the notation r_n now represents the position of the n-frame's origin expressed in geodetic coordinates. Therefore we involve the transformation matrix T as follows

$$T = \begin{bmatrix} 0 & \frac{1}{R_N(L)+h} & 0 \\ \frac{1}{(R_E(L)+h)\cos(L)} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.33}$$

This transformation matrix can be obtained directly from the geometry of figure 4.1 in the similar way as the time derivatives of the latitude and longitude expressed by the equations 4.3 and 4.4 respectively.

The discrete-time version of the position can be obtained simply as the time integration of the equation 4.32 as follows

$$L_{t_0+\Delta t} = L_{t_0} + \int_{t_0}^{t_0+\Delta t} \frac{v^n_{en,N,t}}{R_N(L_t)+h_t} dt \tag{4.34}$$

$$\lambda_{t_0+\Delta t} = \lambda_{t_0} + \int_{t_0}^{t_0+\Delta t} \frac{v^n_{en,E,t}}{(R_E(L_t)+h_t)\cos L_t} dt \tag{4.35}$$

$$h_{t_0+\Delta t} = h_{t_0} + \int_{t_0}^{t_0+\Delta t} v^n_{en,U,t} dt \tag{4.36}$$

again assuming the constant integrand between time steps in the equations 4.34 - 4.36, the final form of the discrete-time equivalent of the equation 4.32 can be written as

$$L_{k+1} = L_k + \frac{v^n_{en,N,k}}{R_N(L_k)+h_k}\Delta t \tag{4.37}$$

$$\lambda_{k+1} = \lambda_k + \frac{v^n_{en,E,k}}{(R_E(L_k)+h_k)\cos L_k}\Delta t \tag{4.38}$$

$$h_{k+1} = h_k + v^n_{en,U,k}\Delta t \tag{4.39}$$

where Δt represents the sample time period as in the case of velocity equation.

4.3.4 Sensor Errors

The accelerometer and gyroscope errors can be divided, according to [15], into four categories. These are a fixed contribution, temperature-dependent variation, run-to-run variation and an in-run variation. First two of these can be calibrated in a laboratory, so their influence can be compensated systematically inside the navigation processor. On the other hand, the last two, need to be estimated before their use for the sensor's output correction, because these are not possible to compensate during the laboratory calibration since they are represented by random processes. Hence, they are modelled in the navigation system estimation algorithm. An approach how to model these two error sources follows.

The run-to-run variation acts as the static component of the output error and is different after each turn-on, but constant during an operation period. This error part is commonly called as the fixed bias or turn-on bias. The in-run variation represents the variable component of the output error as it varies through time. This error source is called as the in-run bias of drift. Both biases, the turn-on and in-run, can be modelled as the random walk plus random constant process. Another way how to model these biases lies in an employment of the first order Gauss-Markov process. More about both of these stochastic system models can be found for example in [17]. There is considered the first option, so the models of the accelerometer and

gyroscope biases are expressed as follows

$$\dot{b}_a = \nu_{au} \tag{4.40}$$

$$\dot{b}_g = \nu_{gu} \tag{4.41}$$

where ν_{au} and ν_{gu} are Gaussian random variables.

Assuming these models, the following accelerometer measurement vector can be defined as

$$\tilde{f}^b_{ib} = (I_3 + M_a) f^b_{ib} + b_a + \nu_{av} \tag{4.42}$$

where $\tilde{f}^b_{ib}$ express the measured value of the specific force, f^b_{ib} is the true value of the specific force, b_a express the previously mentioned accelerometers' biases and ν_{av} is a zero mean uncorrelated Gaussian white noise with known covariances. For the gyroscope measurement vector we have the following expression

$$\tilde{\omega}^b_{ib} = (I_3 + M_g)\omega^b_{ib} + b_g + \nu_{gv} \tag{4.43}$$

where we have similarly the $\tilde{\omega}^b_{ib}$, which represents the gyroscope measurement of the body angular rate, ω^b_{ib} is the corresponding true value, b_g express the gyroscopes' biases and ν_{gv} is again a zero mean uncorrelated Gaussian white noise.

The gyroscopes' and accelerometers' scale factor error terms are noted by M_g and M_a respectively. These scale factor error terms are not considered in the subsequent navigation system error model, but are used in the following trajectory generator description, so they are introduced only due to a more general expression.

The estimated value of the measurements 4.42 and 4.43, thus the measurements compensated by the biases and scale factors, can be written as follows

$$\hat{f}^b_{ib} = (I_3 - \hat{M}_a)(\tilde{f}^b_{ib} - \hat{b_a}) \tag{4.44}$$

$$\hat{\omega}^b_{ib} = (I_3 - \hat{M}_g)(\tilde{\omega}^b_{ib} - \hat{b_g}) \tag{4.45}$$

where all variables has the similar meaning as before. The only difference is given by the hat which means that the variables represent the estimates.

4.4 Sensor Data Generator

The inertial sensor data generator is created for the purpose of generating the gyroscope and accelerometer data from an a priory knowledge about a reference trajectory.

An equation for the gyroscope data generator can be obtained by combining 4.43 and 4.25 as follows

$$\tilde{\omega}^b_{ib} = (I + M_g)\left(C^b_n\left(\omega^n_{ie} + \omega^n_{en}\right) + \omega^b_{nb}\right) + b_g + \nu_{gv} \tag{4.46}$$

Computation of the terms ω_{en}^n and ω_{ie}^n (equations 4.20 and 4.21 respectively) is based on the navigation solution from the previous time step. The term ω_{nb}^b need to be specified a priory from the data about the required trajectory, thus we use

$$\begin{aligned}\begin{bmatrix} \omega_{nb,x}^b \\ \omega_{nb,y}^b \\ \omega_{nb,z}^b \end{bmatrix} &= \begin{bmatrix} \dot{\gamma} \\ 0 \\ 0 \end{bmatrix} + C_3 \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + C_3 C_2 \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \\ &= \begin{bmatrix} \dot{\gamma} + s\phi\dot{\theta} - c\phi s\theta\dot{\psi} \\ c\psi\dot{\theta} + s\psi s\theta\dot{\psi} \\ c\theta\dot{\psi} \end{bmatrix}\end{aligned} \tag{4.47}$$

which can be derived from transformation matrix 4.6. All what we need now for generating angular rates from the gyroscopes, is a definition of the roll, pitch and yaw angle expressions and their first time derivatives. A concrete form of these expressions is defined by the desired trajectory.

Similarly, combining 4.42 and 4.18 the following expression for the specific force generation, expressed in the body frame, can be derived as

$$\tilde{f}_{ib}^b = (I + M_a)\, C_n^b \left(a + (2\omega_{ie}^n + \omega_{en}^n) \times v_{en}^n - g\right) + b_a + \nu_{av} \tag{4.48}$$

Computation of the terms ω_{en}^n and ω_{ie}^n is solved in the same way as before, thus they are directly computed from the equations 4.20 and 4.21 respectively. What need to be specified now, is the desired acceleration vector a. This term need to be defined with relation to the previous angular rate equations which as they are given by ω_{nb}^b. The term v_{en}^n is computed as the time integration of the specified acceleration a.

Circle Trajectory

The circle trajectory with a constant altitude can be simulated by using the following equations [18]

$$\begin{aligned} a_{E,k} &= -\tfrac{2\pi v_g \cos\psi_k}{T_{circle}} \\ a_{N,k} &= -\tfrac{2\pi v_g \sin\psi_k}{T_{circle}} \\ a_{U,k} &= 0 \end{aligned} \tag{4.49}$$

where all these represent an entries of the vector $a = [a_{E,k}\ a_{N,k}\ a_{U,k}]$ and where v_g is the ground speed which is defined as a constant value

$$v_g = constant$$

Further, the term T_{circle} express the circulation period. Now a definition of the yaw angle increment and its rate is

$$\begin{aligned} \Delta\psi &= mod\left(\tfrac{2\pi\Delta t}{T_{circle}}, 2\pi\right) \\ \dot{\Delta\psi} &= \tfrac{2\pi}{T_{circle}} \end{aligned} \tag{4.50}$$

the roll and pitch angle increments and the corresponding rates are set to zeros.

The translational velocity, position and the angular increments are integrated each simulation step according to the next equations

$$v^n_{en,k+1} = v^n_{en,k} + a_k \Delta t \tag{4.51}$$

$$\begin{aligned} L_{k+1} &= L_k + \frac{v^n_{en,N,k}}{R_N(L_k)+h_k}\Delta t \\ \lambda_{k+1} &= \lambda_k + \frac{v^n_{en,E,k}}{(R_E(L_k)+h_k)\cos L_k}\Delta t \\ h_{k+1} &= h_k + v^n_{en,E,k}\Delta t \end{aligned} \tag{4.52}$$

$$\begin{aligned} \gamma_{k+1} &= \gamma_k + \Delta\gamma_k \\ \theta_{k+1} &= \theta_k + \Delta\theta_k \\ \psi_{k+1} &= \psi_k + \Delta\psi_k \end{aligned} \tag{4.53}$$

$$\begin{aligned} \dot{\gamma_{k+1}} &= \dot{\gamma_k} + \Delta\dot{\gamma_k} \\ \dot{\theta_{k+1}} &= \dot{\theta_k} + \Delta\dot{\theta_k} \\ \dot{\psi_{k+1}} &= \dot{\psi_k} + \Delta\dot{\psi_k} \end{aligned} \tag{4.54}$$

Other trajectories can be derived in the similar manner. That can be a static state, straight line, serpentine shape or their combinations.

4.5 Navigation Error Equations

The main purpose of error equations is to obtain an error correction which can be used to correct the inertial navigation solution as we tray to prevent from the divergence caused due to the inertial sensors inaccuracies. Another reason why we need to derive an error model is because the navigation equations are non-linear and the direct linearisation leads to too complex equations. However, the direct linearisation techniques are sometimes used too, as has been shown for example in [19]. Generally it is possible to meet two types of the navigation error equations through the literature. These are denoted after Greek letters Ψ and ϕ, where each of these represent a vector of attitude errors. Although the Phi-angle error model is derived in the local navigation frame at the true navigation position and the Psi-angle error model in the computer frame at the computed navigation position, it can be shown, that they are algebraically equivalent as was proved in [20]. The difference between these frames makes the Psi-angle error model computational less demanding against to the Phi-angle error model. There are some modifications, which appear in the literature over past few years, e.g. [22] modifies the basic version of the Phi-angle and Psi-angle error models. This modification avoids an explicit occurrence of the specific force in the error model state space transition matrix. Further,[23]

deals with large heading uncertainty error models, as this uncertainty can appear during the coarse alignment phase, and generalizes the basic Phi-error and Psi-error models and their modified versions from [22]. In [24] are investigated properties of the second order error models. However, through this work, we will use only the basic form of the Phi-angle error model, thus the first order approximation.

Since the error model can be derived by the first order linear perturbation analysis the following perturbation terms need to be stated

$$\hat{v}_{en}^{n} = v_{en}^{n} + \delta v_{en}^{n} \tag{4.55}$$

$$\hat{f}_{ib}^{n} = \left(I - (\phi\times)\right) f_{ib}^{n} + \delta f_{ib}^{n} \tag{4.56}$$

$$\hat{g}_{ib}^{n} = g_{ib}^{n} + \delta g_{ib}^{n} \tag{4.57}$$

$$\hat{\omega}_{en}^{n} = \omega_{en}^{n} + \delta\omega_{en}^{n} \tag{4.58}$$

$$\hat{\omega}_{ie}^{n} = \omega_{ie}^{n} + \delta\omega_{ie}^{n} \tag{4.59}$$

$$\hat{C}_{b}^{n} = \left(I - (\phi\times)\right) C_{b}^{n} \tag{4.60}$$

$$\hat{C}_{e}^{n} = \left(I - (\Theta\times)\right) C_{e}^{n} \tag{4.61}$$

All these represent, that the computed value, denoted by the hat, is a sum of the true value and the error term, denoted by δ. The direction cosine errors are expressed in the similar manner, but the error is given as $-(\phi\times)C_{b}^{n}$.

4.5.1 Velocity Error Equation

Let's begin with a derivation of the velocity error equation. The velocity error of a moving target can be expressed by the equation 4.55. If we simply subtract the true inertial navigation velocity mechanization equation 4.18 from its computed equivalent

$$\dot{\hat{v}}_{en}^{n} = \hat{f}_{ib}^{n} - \left(\hat{\omega}_{en}^{n} + 2\hat{\omega}_{ie}^{n}\right) \times \hat{v}_{en}^{n} + \hat{g}_{ib}^{n} \tag{4.62}$$

together with substituting the equations 4.55 - 4.59 into 4.62, then the following velocity error equation can be obtained

$$\begin{aligned} \dot{\delta v}_{en}^{n} &= \left(I - (\phi\times)\right) f_{ib}^{n} + \delta f_{ib}^{n} - \left(\omega_{en}^{n} + \delta\omega_{en}^{n} + 2\left(\omega_{ie}^{n} + \delta\omega_{ie}^{n}\right)\right) \times \left(v_{en}^{n} + \delta v_{en}^{n}\right) \\ &+ g_{ib}^{n} + \delta g_{ib}^{n} - f_{ib}^{n} + \left(\omega_{en}^{n} + 2\omega_{ie}^{n}\right) \times v_{en}^{n} - g_{ib}^{n} \\ &= -\left(\phi\times\right) f_{ib}^{n} + \delta f_{ib}^{n} + \delta g_{ib}^{n} - \left(\delta\omega_{en}^{n} + 2\delta\omega_{ie}^{n}\right) \times v_{en}^{n} - \left(\omega_{en}^{n} + 2\omega_{ie}^{n}\right) \times \delta v_{en}^{n} \end{aligned} \tag{4.63}$$

4.5.2 Attitude Error Equation

The attitude error is represented by the second term in the parentheses of the equation 4.60, thus as the skew-symmetric matrix $(\phi\times)$. From this equation we can write the b-frame to n-frame transformation error matrix as follows

$$\begin{aligned} \delta C_b^n &= \hat{C}_b^n - C_b^n \\ &= -(\phi\times)C_b^n \end{aligned} \tag{4.64}$$

The time derivative of the first row of 4.64 yields

$$\begin{aligned} \delta\dot{C}_b^n &= \dot{\hat{C}}_b^n - \dot{C}_b^n \\ &= -\hat{\Omega}_{bn}^n \hat{C}_b^n + \Omega_{bn}^n C_b^n \\ &= -\hat{\Omega}_{bn}^n \left(I - (\phi\times)\right) C_b^n + \Omega_{bn}^n C_b^n \\ &= \left(-\hat{\Omega}_{bn}^n + \hat{\Omega}_{bn}^n(\phi\times) + \Omega_{bn}^n\right) C_b^n \\ &\approx \left(-\hat{\Omega}_{bn}^n + \Omega_{bn}^n(\phi\times) + \Omega_{bn}^n\right) C_b^n \end{aligned} \tag{4.65}$$

where the equations 4.60 and 4.24 are used. In the last row of 4.65 the second term is replaced by the true value of the body to navigation angular rate skew-symmetric matrix. This adaptation does not have any significant effect, because the values are nearly similar. Let's taking the time derivative of the second row of 4.64 as follows

$$\begin{aligned} \delta\dot{C}_b^n &= -(\dot{\phi\times})C_b^n - (\phi\times)\dot{C}_b^n \\ &= \left(-(\dot{\phi\times}) + (\phi\times)\Omega_{bn}^n\right) C_b^n \end{aligned} \tag{4.66}$$

where the equation 4.24 is used again. Now comparing 4.65 and 4.66 yields

$$-\hat{\Omega}_{bn}^n + \Omega_{bn}^n(\phi\times) + \Omega_{bn}^n = -(\dot{\phi\times}) + (\phi\times)\Omega_{bn}^n \tag{4.67}$$

where we need to express the first term of the right-hand side as follows

$$(\dot{\phi\times}) = (\phi\times)\Omega_{bn}^n - \Omega_{bn}^n(\phi\times) + \hat{\Omega}_{bn}^n - \Omega_{bn}^n \tag{4.68}$$

or in a vector form

$$\dot{\phi} = -\Omega_{bn}^n\phi + \hat{\omega}_{bn}^n - \omega_{bn}^n \tag{4.69}$$

Now we need to express the last two terms of this equation. The first one can be simply expanded as

$$\omega_{bn}^n = \omega_{in}^n - C_b^n\omega_{ib}^b \tag{4.70}$$

and the second one as

$$\begin{aligned} \hat{\omega}_{bn}^n &= \hat{\omega}_{in}^n - \hat{C}_b^n\left(\omega_{ib}^b + \delta\omega_{ib}^b\right) \\ &= \hat{\omega}_{in}^n - \left(I - (\phi\times)\right) C_b^n\omega_{ib}^b - \delta\omega_{ib}^n + (\phi\times)C_b^n\delta\omega_{ib}^b \\ &\approx \hat{\omega}_{in}^n - \left(I - (\phi\times)\right) C_b^n\omega_{ib}^b - \delta\omega_{ib}^n \end{aligned} \tag{4.71}$$

where the last term of the second row is neglected, because the multiplication of two error terms is nearly zero. Now substituting 4.70 and 4.71 back into the equation 4.69 yields

$$\begin{aligned} \dot{\phi} &= -\omega_{bn}^n \times \phi + (\phi\times) C_b^n \omega_{ib}^b + \hat{\omega}_{in}^n - \omega_{in}^n - \delta\omega_{ib}^n \\ &= (\phi\times)\omega_{bn}^n + (\phi\times)\omega_{ib}^n + \delta\omega_{ie}^n + \delta\omega_{en}^n - \delta\omega_{ib}^n \\ &= -\Omega_{in}^n \phi + \delta\omega_{ie}^n + \delta\omega_{en}^n - \delta\omega_{ib}^n \end{aligned} \tag{4.72}$$

which is the final expression of the Phi-angle attitude error.

4.5.3 Position Error Equation

The previously presented equations for the velocity and attitude errors need to be expressed by using the equations 4.55 - 4.60. This is due to the fact, that the direct partial derivative of the velocity 4.18 and attitude 4.24 equations is difficult since the appearance of the direction cosine matrix in both of these equations. On the other hand, the position equation 4.32 do not need this approach, thus we can write the corresponding error equation directly as follows

$$\delta\dot{r}_n = \left.\frac{\partial \dot{r}_n}{\partial r_n}\right|_{\hat{r}_n, \hat{v}_{en}^n} \delta r_n + \left.\frac{\partial \dot{r}_n}{\partial v_{en}^n}\right|_{\hat{r}_n} \delta v_{en}^n \tag{4.73}$$

Another way how to express the position error lies in the equation 4.31, or more precisely, in its equivalent before the time derivation $r_{en}^n = C_e^n r_{en}^e$. It is obvious that this equation contains the direction cosine matrix, thus we need to use similar approach as with the attitude error equation. For this purpose one need to employ 4.61. However, the equation 4.73 is sufficient for our purposes.

4.5.4 State Space Representation of Navigation Error Model

Now we need to rearrange all the previously presented error equations into a state space representation which is appropriate for the Kalman filter. Let's start with the general form of the error state space representation which need to be found as follows

$$\delta\dot{x}_t = F_t \delta x_t + G_t \nu_t \tag{4.74}$$

$$\delta z_t = H_t \delta x_t + v_t \tag{4.75}$$

These equations are very similar to the state space model presented by 3.1 and 3.2, the only difference is, that the state and measurement vector represent deviations.

State Space Error Equations

The state equation 4.74 can be divided into the following two parts

$$\begin{bmatrix} \delta\dot{x}_{ins,t} \\ \delta\dot{x}_{aug,t} \end{bmatrix} = \begin{bmatrix} F_{ins\times ins,t} & F_{ins\times aug,t} \\ F_{aug\times ins,t} & F_{aug\times aug,t} \end{bmatrix} \begin{bmatrix} \delta x_{ins,t} \\ \delta x_{aug,t} \end{bmatrix} + + \begin{bmatrix} G_{ins\times ins,t} & G_{ins\times aug,t} \\ G_{aug\times ins,t} & G_{aug\times aug,t} \end{bmatrix} \begin{bmatrix} \nu_{v,t} \\ \nu_{u,t} \end{bmatrix} \tag{4.76}$$

The first part, denoted by the index ins, comprises the error terms of the position, velocity and attitude as

$$\begin{aligned} \delta x_{ins,t} &= \begin{bmatrix} \delta r_{n,t} & \delta v^n_{en,t} & \phi_t \end{bmatrix}^T \\ &= \begin{bmatrix} \delta L_t & \delta\lambda_t & \delta h_t & \delta v^n_{en,E,t} & \delta v^n_{en,N,t} & \delta v^n_{en,U,t} & \phi_{x,t} & \phi_{y,t} & \phi_{z,t} \end{bmatrix}^T \end{aligned} \tag{4.77}$$

where $\delta r_{n,t}$ is the position error vector, $\delta v^n_{en,t}$ is the velocity error vector and ϕ_t is the attitude error vector.

The second part, denoted by the index aug, represents an augmentation, which comprises the inertial sensor error increments. These correspond to the biases, scale factor error terms, non-orthogonality etc. Through this work, we will consider only the biases as follows

$$\begin{aligned} \delta x_{aug,t} &= \begin{bmatrix} \delta b_{a,t} & \delta b_{g,t} \end{bmatrix}^T \\ &= \begin{bmatrix} \delta b_{a,x,t} & \delta b_{a,y,t} & \delta b_{a,z,t} & \delta b_{g,x,t} & \delta b_{g,y,t} & \delta b_{g,z,t} \end{bmatrix}^T \end{aligned} \tag{4.78}$$

where $\delta b_{a,t}$ is the accelerometer bias vector, which consists of the three scalars, one for each axis, denoted by the subscripts x,y,z. The term $\delta b_{g,t}$ express the gyroscope bias vector, again composed of three scalars, which are denoted similarly as before.

As we have described the error state space vector the transition and noise distribution sub-matrices need to be found. A way, how to obtain the transition matrix $F_{ins\times ins,t}$, lies in the equations 4.73, 4.63 and 4.72. Since we need to express the position error given by 4.73 the partial derivatives need to be evaluated firstly as follows

$$\left.\frac{\partial\dot{r}_n}{\partial r_n}\right|_{\hat{r}_n,\hat{v}^n_{en}} \delta r_n = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \frac{\hat{v}^n_{en,E}\sin\hat{L}}{(R_E(\hat{L})\hat{h})\cos^2\hat{L}}\delta L - \begin{bmatrix} \frac{\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})^2}\delta h \\ \frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})^2\cos\hat{L}}\delta h \\ 0 \end{bmatrix} \tag{4.79}$$

$$\left.\frac{\partial\dot{r}_n}{\partial v^n_{en}}\right|_{\hat{v}^n_{en}} \delta v^n_{en} = \begin{bmatrix} \frac{\delta v^n_{en,N}}{R_N(\hat{L})+\hat{h}} \\ \frac{\delta v^n_{en,E}}{(R_E(\hat{L})+\hat{h})\cos\hat{L}} \\ \delta v^n_{en,U} \end{bmatrix} \tag{4.80}$$

where we neglect the derivatives of the orders higher than first and those, which are derivatives of the inner functions. Further, the equation 4.63 need to expressed in the terms of $\delta\omega^n_{en}$ and $\delta\omega^n_{ie}$. Both of these can be obtained from partial derivatives according to the navigation system state vector (not the error state vector) similarly as before

$$\delta\omega^n_{en} = \begin{bmatrix} -\frac{\delta v^n_{en,N}}{R_N(\hat{L})+\hat{h}} \\ \frac{\delta v^n_{en,E}}{R_E(\hat{L})+\hat{h}} \\ \frac{\delta v^n_{en,E}\tan\hat{L}}{R_E(\hat{L})+\hat{h}} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \frac{\delta v^n_{en,E}}{\left(R_E(\hat{L})+\hat{h}\right)\cos^2\hat{L}}\delta L + \begin{bmatrix} \frac{v^n_{en,N}}{\left(R_N(\hat{L})+\hat{h}\right)^2} \\ -\frac{v^n_{en,E}}{\left(R_E(\hat{L})+\hat{h}\right)^2} \\ -\frac{v^n_{en,E}\tan\hat{L}}{\left(R_E(\hat{L})+\hat{h}\right)^2} \end{bmatrix} \delta h \quad (4.81)$$

$$\delta\omega^n_{ie} = \omega_{ie} \begin{bmatrix} 0 \\ -\sin\hat{L} \\ \cos\hat{L} \end{bmatrix} \delta L \quad (4.82)$$

The gravity error term δg^n_{ib} can be computed as $[0\ 0\ -\frac{\partial g}{\partial h}]$ where $\frac{\partial g}{\partial h}$ is [16]

$$\begin{aligned} \frac{\partial g}{\partial h} &= \frac{\partial}{\partial h} g_0 \left(\frac{R_e}{R_e+h}\right)^2 \\ &= -\frac{2g}{R_e+h} \end{aligned} \quad (4.83)$$

Now we need to put all these terms together and make the resulting matrix $F_{ins\times ins,t}$. For this purpose the following block representation is introduced

$$F_{ins\times ins,t} = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \quad (4.84)$$

substituting now 4.79, 4.80 into 4.73 and 4.81, 4.82 into 4.63 and 4.72 gives, after some rearrangements, the following sub-matrices

$$F_{11} = \begin{bmatrix} 0 & 0 & -\frac{\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})^2} \\ \frac{\hat{v}^n_{en,E}\sin\hat{L}}{(R_E(\hat{L})+\hat{h})\cos^2\hat{L}} & 0 & -\frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})^2\cos\hat{L}} \\ 0 & 0 & 0 \end{bmatrix} \quad (4.85)$$

$$F_{12} = \begin{bmatrix} 0 & \frac{1}{R_N(\hat{L})+\hat{h}} & 0 \\ \frac{1}{(R_E(\hat{L})+\hat{h})\cos\hat{L}} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.86)$$

$$F_{13} = O_{3\times 3} \quad (4.87)$$

$$F_{21} = \begin{bmatrix} 2\omega_{ie}(\hat{v}^n_{en,U}\sin\hat{L}+\hat{v}^n_{en,N}\cos\hat{L})+\frac{\hat{v}^n_{en,E}\hat{v}^n_{en,N}}{(R_E(\hat{L})+\hat{h})\cos^2\hat{L}} & 0 & \frac{\hat{v}^n_{en,U}\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})^2} - \frac{\hat{v}^n_{en,E}\hat{v}^n_{en,N}\tan\hat{L}}{(R_E(\hat{L})+\hat{h})^2} \\ -2\omega_{ie}\hat{v}^n_{en,E}\cos\hat{L} - \frac{(\hat{v}^n_{en,E})^2}{(R_E(\hat{L})+\hat{h})\cos^2\hat{L}} & 0 & \frac{\hat{v}^n_{en,U}\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})^2} + \frac{\hat{v}^n_{en,E}\tan\hat{L}}{(R_E(\hat{L})+\hat{h})^2} \\ -2\omega_{ie}\hat{v}^n_{en,E}\sin\hat{L} & 0 & -\frac{(\hat{v}^n_{en,N})^2}{(R_N(\hat{L})+\hat{h})^2} - \frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})^2} + \frac{2g}{R_e+\hat{h}} \end{bmatrix} \quad (4.88)$$

$$F_{22} = \begin{bmatrix} -\frac{\hat{v}^n_{en,U}}{R_E(\hat{L})+\hat{h}} + \frac{\hat{v}^n_{en,N}\tan\hat{L}}{R_E(\hat{L})+\hat{h}} & \frac{\hat{v}^n_{en,E}\tan\hat{L}}{R_E(\hat{L})+\hat{h}} + 2\omega_{ie}\sin\hat{L} & -\frac{\hat{v}^n_{en,E}}{R_E(\hat{L})+\hat{h}} - 2\omega_{ie}\cos\hat{L} \\ -\frac{2\hat{v}^n_{en,E}\tan\hat{L}}{R_E(\hat{L})+\hat{h}} - 2\omega_{ie}\sin\hat{L} & -\frac{\hat{v}^n_{en,U}}{R_N(\hat{L})+\hat{h}} & -\frac{\hat{v}^n_{en,N}}{R_N(\hat{L})+\hat{h}} \\ \frac{2\hat{v}^n_{en,E}}{R_E(\hat{L})+\hat{h}} + 2\omega_{ie}\cos\hat{L} & \frac{2\hat{v}^n_{en,N}}{R_N(\hat{L})+\hat{h}} & 0 \end{bmatrix} \tag{4.89}$$

$$F_{23} = \begin{bmatrix} 0 & -f^n_{ib,U} & f^n_{ib,N} \\ f^n_{ib,U} & 0 & -f^n_{ib,E} \\ -f^n_{ib,N} & f^n_{ib,E} & 0 \end{bmatrix} \tag{4.90}$$

$$F_{31} = \begin{bmatrix} 0 & 0 & \frac{\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})^2} \\ -\omega_{ie}\sin\hat{L} & 0 & -\frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})^2} \\ \omega_{ie}\cos\hat{L} + \frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})\cos^2\hat{L}} & 0 & -\frac{\hat{v}^n_{en,E}\tan\hat{L}}{(R_E(\hat{L})+\hat{h})^2} \end{bmatrix} \tag{4.91}$$

$$F_{32} = \begin{bmatrix} 0 & -\frac{1}{R_N(\hat{L})+\hat{h}} & 0 \\ \frac{1}{R_E(\hat{L})+\hat{h}} & 0 & 0 \\ \frac{\tan\hat{L}}{R_N(\hat{L})+\hat{h}} & 0 & 0 \end{bmatrix} \tag{4.92}$$

$$F_{33} = \begin{bmatrix} 0 & \omega_{ie}\sin\hat{L} + \frac{\hat{v}^n_{en,E}\tan\hat{L}}{(R_E(\hat{L})+\hat{h})} & -\omega_{ie}\cos\hat{L} - \frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})} \\ -\omega_{ie}\sin\hat{L} - \frac{\hat{v}^n_{en,E}\tan\hat{L}}{(R_E(\hat{L})+\hat{h})} & 0 & -\frac{\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})} \\ \omega_{ie}\cos\hat{L} + \frac{\hat{v}^n_{en,E}}{(R_E(\hat{L})+\hat{h})} & \frac{\hat{v}^n_{en,N}}{(R_N(\hat{L})+\hat{h})} & 0 \end{bmatrix} \tag{4.93}$$

At this point it is obvious that the terms δf^n_{ib} and $\delta\omega^n_{ib}$, from the equations 4.63 and 4.72 respectively, are not used in the previous $F_{ins\times ins,t}$ matrix derivation. So their use becomes important through a derivation of the matrices $F_{ins\times aug,t}$, $G_{ins\times ins,t}$ and $G_{ins\times aug,t}$. Before expressing these matrices the relation for the terms δf^n_{ib} and $\delta\omega^n_{ib}$ need to be found. Let's start with the first of these, thus with the true specific force error term as follows

$$\delta f^n_{ib} = \hat{f}^n_{ib} - f^n_{ib} \tag{4.94}$$

Now by substituting the equations 4.44 and 4.42 into 4.94 (the term f^n_{ib} need to be expressed and scale factor error matrices M_a and $\hat{M}_a$ are set to zero) the following relation can be obtained

$$\begin{aligned} \delta f^n_{ib} &= C^n_b\left(b_a - \hat{b}_a + \nu_{av}\right) \\ &= C^n_b\left(\delta b_a + \nu_{av}\right) \end{aligned} \tag{4.95}$$

where the δb_a is the accelerometers' bias error state variable. Similarly, the error term $\delta\omega^n_{ib}$ can be defined as follows

$$\delta\omega^n_{ib} = \hat{\omega}^n_{ib} - \omega^n_{ib} \tag{4.96}$$

Again, substitution of the equations 4.45 and 4.43 into 4.82 yields

$$\begin{aligned} \delta\omega_{ib}^{n} &= C_b^n \left(b_g - \hat{b}_g + \nu_{gv} \right) \\ &= C_b^n \left(\delta b_g + \nu_{gv} \right) \end{aligned} \tag{4.97}$$

Using 4.95 and 4.97 the previously mentioned matrices $F_{ins\times aug,t}$, $G_{ins\times ins,t}$ and $G_{ins\times aug,t}$ are written as follows

$$F_{ins\times aug,t} = G_{ins\times ins,t} = \begin{bmatrix} O_{3\times 3} & O_{3\times 3} \\ C_b^n & O_{3\times 3} \\ O_{3\times 3} & -C_b^n \end{bmatrix} \tag{4.98}$$

$$G_{ins\times aug,t} = O_{9\times 6} \tag{4.99}$$

For expressing the matrices of augmented part of the error state vector 4.78 the derivatives of δb_a and δb_g need to be expressed first. Starting with δb_a gives the following expression

$$\begin{aligned} \delta\dot{b}_a &= \dot{b}_a - \dot{\hat{b}}_a \\ &= \dot{b}_a - E\left\{\dot{b}_a\right\} \\ &= \nu_{au} \end{aligned} \tag{4.100}$$

It is obvious from 4.95 that the time derivative of δb_a is given as a difference of the time derivatives of true bias and its estimate. The term in the second row of 4.95 expresses that the estimated value of the bias is given as the mean of the true bias. Hence, it is clear that the time derivative of this value is zero and the resulting value of the time derivative of δb_a is given only by the noise term ν_{au}. Similar conclusion holds for the time derivative of the gyroscope bias error as follows

$$\begin{aligned} \delta\dot{b}_g &= \dot{b}_g - \dot{\hat{b}}_g \\ &= \dot{b}_g - E\left\{\dot{b}_g\right\} \\ &= \nu_{gu} \end{aligned} \tag{4.101}$$

Now we can express the matrices of the augmented part of the error model as

$$F_{aug\times ins,t} = O_{6\times 9} \tag{4.102}$$

$$F_{aug\times aug,t} = G_{aug\times ins,t} = O_{6\times 6} \tag{4.103}$$

$$G_{aug\times aug,t} = I_{6\times 6} \tag{4.104}$$

Observation Error Equations

The observation error vector of the equation 4.75 can be expressed as a difference between the GNSS observation vector $\tilde{z}_t$ and the vector which comprises the resulting solution of the navigation system's position r_{en}^n and velocity v_{en}^n denoted here as x_{rv}

$$\delta z_t = \tilde{z}_t - x_{rv,t} \tag{4.105}$$

where the vector $\tilde{z}_t$ is given as

$$\begin{aligned} \tilde{z}_t &= \begin{bmatrix} \tilde{r}^n_{en,t} & \tilde{v}^n_{en,t} \end{bmatrix} \\ &= \begin{bmatrix} \tilde{r}^n_{en,E,t} & \tilde{v}^n_{en,N,t} & \tilde{v}^n_{en,U,t} & \tilde{v}^n_{en,E,t} & \tilde{v}^n_{en,N,t} & \tilde{v}^n_{en,U,t} \end{bmatrix} \end{aligned} \tag{4.106}$$

and the vector $x_{rv,t}$ as

$$\begin{aligned} x_{rv,t} &= \begin{bmatrix} r^n_{en,t} & v^n_{en,t} \end{bmatrix} \\ &= \begin{bmatrix} r^n_{en,E,t} & r^n_{en,N,t} & r^n_{en,U,t} & v^n_{en,E,t} & v^n_{en,N,t} & v^n_{en,U,t} \end{bmatrix} \end{aligned} \tag{4.107}$$

The observation matrix H_t is defined as follows

$$H_t = \begin{bmatrix} T_t^{-1} & O_{3\times3} & O_{3\times3} & O_{3\times3} & O_{3\times3} \\ O_{3\times3} & I_{3\times3} & O_{3\times3} & O_{3\times3} & O_{3\times3} \end{bmatrix} \tag{4.108}$$

where the matrix T_t^{-1} is given as the inverse of matrix 4.33

$$T_t^{-1} = \begin{bmatrix} 0 & (R_E(L) + h)\cos(L) & 0 \\ R_N(L) + h & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.109}$$

There we can ask why the position part of the vectors $\tilde{z}_t$ and $x_{rv,t}$ is not expressed in the terms of latitude, longitude and height. It is due to numerical difficulties which can arise in the Kalman filter innovation matrix inverse computation. As we have the navigation system position expressed in the terms of latitude, longitude and height, the input position observations $\tilde{r}_n$ and the position as the result of the navigation systems integration r_n, need to be both transformed before their use in the Kalman filter residual.

Here we need to note that the discretization of the system transition matrix in 4.74 is performed with using the first two terms of the 3.19, thus we use the first order approximation. For the discrete version of the process noise covariance matrix is used equation 3.25. The discretization of the inertial navigation system mechanization part was described in the previous sections.

Now we have everything what we need for the Kalman filter implementations. The summary of the inertial navigation algorithm is depicted in figure 4.2, where the measurement time indexes are introduced. It is important to say that as we estimate the error between the estimated and true trajectory we use the so-called indirect Kalman filter. This do not brings any change into the conventional Kalman filter equations 3.45 - 3.49, but we need to consider a little difference in the covariance matrix interpretation and if we want to use the estimated error in the so-called feedback or feed-forward implementation of the indirect Kalman filter. Both of these implementations are described for example in [17].

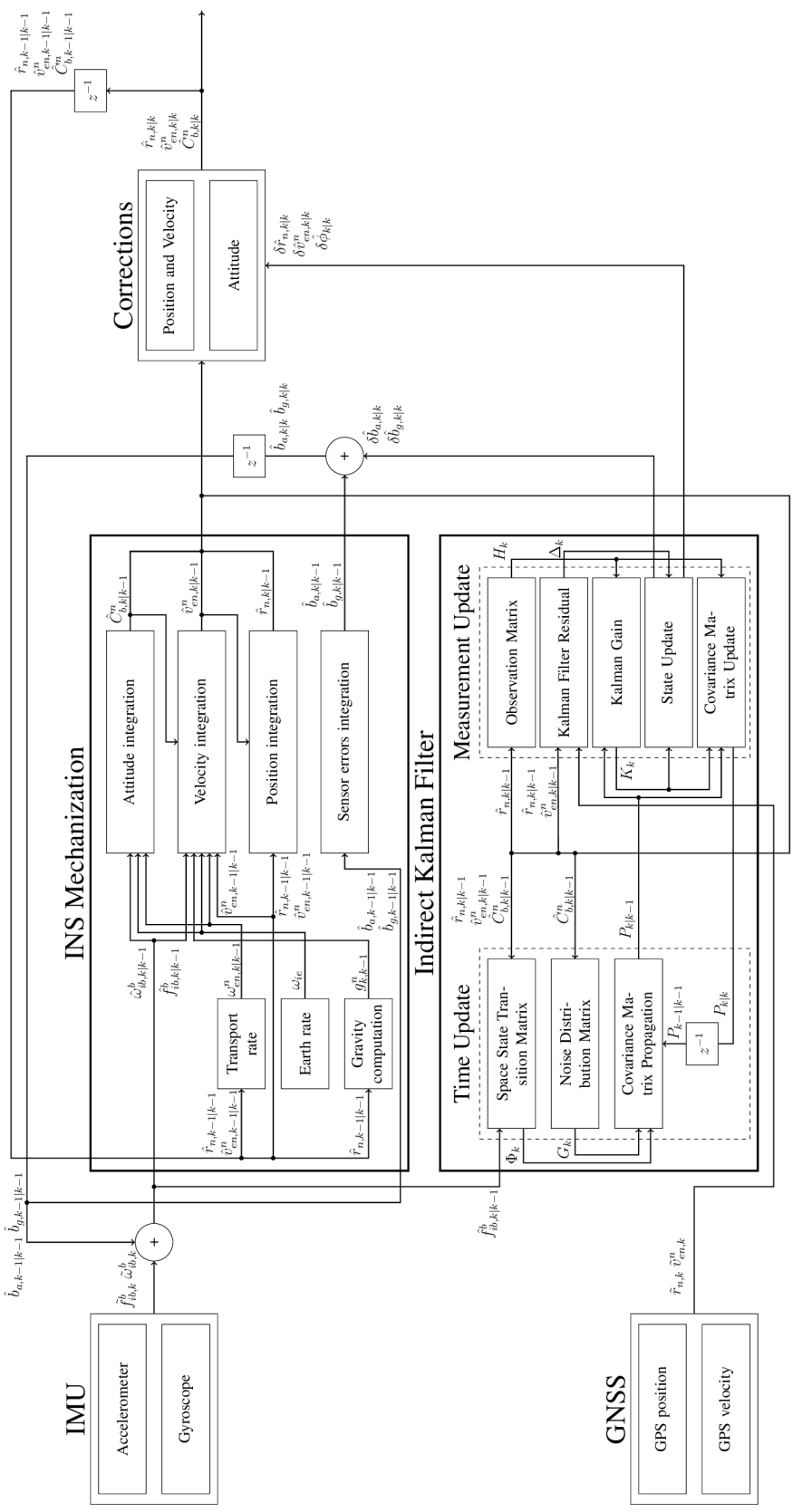


Fig. 4.2: Navigation system algorithm summary

5 EXPERIMENTS AND RESULTS

5.1 Introduction

It is very difficult to obtain some analytical solution, for example in a sense of the sensitivity analysis, which can precisely evaluate all rounding errors in the inertial navigation system or which can describe how are these errors propagated through time. Hence, we need to note, that the following description do not trays to quantify any source of rounding error individually since it is a very difficult and demanding task. It only trays to compare the previously described Kalman filter implementations. Further, we want to describe what the presented results show and where is probably the most significant source of rounding errors from a higher perspective. First part of this chapter presents a description of an experiment and the second part presents a result of experiments where the fractional part of the fixed-point computational word is changed in some interval.

5.2 Experiment Description

Let us assume that we have a vehicle which moves in a circular trajectory with a constant ground speed 10m.s^{-1} and at constant altitude 1000m. An initial value of the latitude, longitude, roll, pitch and yaw angles is set to 0rad. Initial velocity is known from the ground speed. The accelerometers' noise std. deviations are given as $\sigma_a = 9.81 \times 10^{-5}\text{m.s}^{-3/2}$ and $\sigma_{ba} = 6.00 \times 10^{-5}\text{m.s}^{-5/2}$. For the gyroscopes' std. deviations we have $\sigma_g = 2.91 \times 10^{-7}\text{rad.s}^{-1/2}$ and $\sigma_{bg} = 9.20 \times 10^{-7}\text{rad.s}^{-3/2}$. The accelerometers' biases are all three set to -0.03m.s^{-2} and gyroscopes' biases to $2.95\times 10^{-4}\text{rad.s}^{-1}$, where first one of these gyro biases have a negative sign. The GNSS position and velocity measurements have std. deviations $\sigma_r = 10\text{m}$ and $\sigma_v = 4\text{m.s}^{-1}$ respectively. The accelerometers' and gyroscopes' measurements are both sampled every 0.2s and GNSS measurements every 1s. The variances of the initial covariance matrix are set to $1 \times 10^{-6}\text{rad}^2$ for the latitude and longitude, 3.33m^2 for the height, $0.66\text{m}^2\text{.s}^{-2}$ for the east and north velocities, $3.33 \times 10^{-2}\text{m}^2\text{.s}^{-2}$ for the up velocity, $5.8 \times 10^{-6}\text{rad}^2$ for the roll and pitch angles and $2.3 \times 10^{-2}\text{rad}^2$ for the yaw angle. The initial accelerometer-bias variances are all three set to $3.3 \times 10^{-2}\text{m}^2\text{.s}^{-4}$ and the initial gyroscope-bias variances are all three set to $3.3 \times 10^{-4}\text{rad}^2\text{.s}^{-2}$. The off-diagonal elements of the initial covariance matrix and the elements of the initial error state vector are set to zero. Simulation was performed in the MATLAB/Simulink environment with the Fixed-Point Toolbox. The length of the simulation is 1000s. All these experimental conditions hold for the all subsequent sections.

5.3 Different Computational Word Lengths

This section presents simulation results for experiments which tray to evaluate how the optimality of the Kalman filter forms is affected due to changes in the fractional part of the fixed-point computational word, thus we tray to evaluate the effect of rounding errors to the estimated trajectory and compare a performance of the all previously discussed Kalman filter forms.

The integer part of the computational word is set to 24bits (including the sign bit) and is unchanged between the individual experiments. The fractional part is changed in a range from 45bits to 55bits. The integer part of the computational word is chosen with respect to the greatest number in the system, which is one of the Earth's radii summed with the height above the Earth. The 24bits has a sufficient redundancy for the common flight levels. The bottom value of the fractional bit range is chosen with respect to the smallest values which can appear in the navigation system. These are results of the division of the local navigation frame velocities with the sum of the Earth's radius with the height above the Earth, which arises in some elements of the system transition matrix 4.84. This sum is of the second power in some entries, hence one can ask, if the previously mentioned 24bits in the integer part is a sufficient length for expressing a number, which can arises as a result of this second power. Of course, 24bits is not sufficient, but if we take a closer look at the all equations which we need to compute during one iteration (in the navigation system algorithm as a whole), then we can see, that these second powers are expressed only in the denominators of some entries of the transition matrix 4.84. Hence, we can split this fraction into the multiplication of the same two fractions, which avoids the need for a longer integer part of the computational word. It only brings a requirement for a minimal number of bits in the fractional part of the computational word, thus the 45bits. If we decrease this number of bits more, then the previously mentioned multiplication of the two fractions becomes rounded to zero, hence we are right on the edge with this number of bits, so we can expect that as we will approach this length more closely, then the rounding in the transition matrix 4.84 becomes more significant. The other numbers, which are expressed by small values, are the elements of the process noise covariance matrix, which are obtained as the square of the previously mentioned gyroscopes' and accelerometers' std. deviations. The length of 45bits is sufficient for their interpretation.

5.3.1 Criteria Evaluation

We choose the criteria function as a weighted sum of the second powers of the estimated error states as follows

$$J_j = \frac{1}{N} \sum_{i=1}^{N} \delta x_{ij}^2 \tag{5.1}$$

where N is the number of samples in the simulation output dataset and δx_i is the i-th estimated error state vector from this dataset. The subscript j represents that the sum is computed for the j-th entry of the vector δx.

Bars which indicate values of the criteria function for the conventional Kalman filter, its implementation with the stabilized Joseph's version of the discrete Riccati equation, the Square Root filter and UD factorized filter, when the number of bits in the fractional part of the fixed-point computational word is changed, are depicted in figures 5.1 - 5.5. The red line represents a value of the criteria function for the conventional Kalman filter computed in the double precision floating-point arithmetic. We can compare a height of the bars with respect to this line as we expect that the conventional Kalman filter computed in the double precision floating-point arithmetic is a sufficient reference which do not experience any significant rounding errors during the simulation. Hence, we can say that the red line represents an optimal value of the criteria as the process and measurement covariance matrices are set optimally from the previously mentioned std. deviations.

Now let's take a closer look at the figure 5.1 which depicts the values of the criteria function for the latitude (a), longitude (b) and height (c) errors. We can see, that the bars are the same in the range from 55bits down to 51bits for the all Kalman filter implementations. This identity is an expected result as we know that the all implementations are algebraically equivalent. The values of the criteria function for the conventional and Joseph implementations are zero in the interval from 45bits to 48bits, which indicates that the both algorithms fail trough their run in this range. We will describe this situation more precisely in the subsequent section which deals about the estimated trajectories. The criteria function for the Square Root and UD factorized implementations, in the range from 45bits to 49bits, is discrepant. This indicates a suboptimal performance of these implementations. Similar conclusions can be made about the rest of the error state variables depicted in figures 5.2 - 5.5. However, it is obvious that the variables which do not experience any significant movement are least affected.

5.3.2 Covariance Matrix Conditioning

This section take a different look at some lengths of the fractional part of the fixed-point computational word used through the previous evaluation of the criteria function. Thus, we demonstrate, how the covariance matrix conditioning is affected due to rounding errors in the system. Figures 5.6 - 5.9 show the covariance matrix or its factors conditioning for the various lengths of the fractional part of the computational word. Each of these figures depict the conditioning of the conventional covariance matrix $\kappa(P_{conv\_fxp})$, Joseph's form of the covariance matrix $\kappa(P_{jsph\_fxp})$, next the conditioning of the Square Root factor $\kappa(S_{sr\_fxp})$ and UD factor $\kappa(UD^{1/2}_{ud\_fxp})$. These are compared with respect to the conditioning of the conventional Kalman filter covariance matrix $\kappa(P_{conv\_flp})$ computed in the double precision floating-point arithmetic.

In figure 5.6 is used 48bits for the fractional part when the integer part remains set to 24bits. This is the number of bits of the fractional part when the conventional and Joseph's Kalman filters fail as was pointed out through the previous section and as it is depicted in figures 5.1 - 5.5. It can be seen again that the both algorithms fail after approximately 400s. We can see that some values significantly fluctuates from their reference represented by the black line. The peaks show that there is a significant likelihood that the covariance matrix becomes ill-conditioned. On the other hand the conditioning of the Square Root and UD factorized Kalman filter factors do not experience any fluctuations. There we need to remind that the condition number of the Square Root and UD factorized Kalman filter factors is two times smaller in order of magnitude with respect to the conventional and Joseph implementations of the covariance matrix. The resulting trend of the Square Root and UD factors conditioning is significantly smoother, thus we can state that these implementations are numerically more robust. As we increase the number of bits in the fractional part of the computational word, the conventional and Joseph implementations do not fail as it is depicted in figure 5.7, but it is obvious that there is still a possibility, that a divergence can occur. A little bit better situation is depicted in figure 5.8 where the conventional and Joseph implementations are not disrupted significantly, but the trend is not smooth, which still indicates unstability of the covariance matrix computation. Other bit increase in the fractional part indicates that the trend for the conventional and Joseph's implementations becomes completely smooth as it is depicted in figure 5.9. It is clear that it is much more better to use square root filtering methods instead of the conventional and Joseph implementations, even if it brings a little bit more computational burden.

5.3.3 Estimated Trajectories of Conventional Kalman Filter

In this section we take a closer look at the length of the fractional part of the computational word where the conventional and Joseph's implementations diverge. This is 48bits as it is depicted in figures 5.1 - 5.5. A whole navigation period, during this simulation, is depicted in figures 5.10 - 5.24 for the conventional Kalman filter only. This is because the Joseph's version results are nearly identical. Each of these figures depict the navigation system state variable, error state variable estimated by the Kalman filter with corresponding sigma bounds and the variance which serves for computing these sigma bounds. One can ask why we need to depict the variance, it is because we need to show, if the diagonal entries of the covariance matrix, thus the variances, are positive or if they become negative due to rounding errors. All the presented estimates are compared with respect to their equivalents computed in the double precision floating-point arithmetic, where we expect that the effect of rounding errors is negligible.

Now let's focus our attention on the figures 5.10 - 5.12 where the estimates of the latitude, longitude and height are depicted. The part (a) of these figures shows the trajectory estimated by the both implementations in comparison with respect to the reference trajectory (the red line) and the GNSS measurements (the blue line). For example, the estimated height depicted in figure 5.12a shows that the GNSS measurements are successfully filtered for the double precision implementation. Similar conclusion can be made about the latitude and longitude, however due the length of radius of the circle trajectory (≈ 315m) the filtering performance is not visible enough. However, the fixed-point implementation do not goes well. If we look at the estimated error state of the latitude, longitude and height with their corresponding 3-sigma bounds, (these bounds set out the interval in which all the estimated error states should be with the probability 99.73%), then we can see that these estimates diverge. After approximately 400s all the error states and the states of the estimated trajectory fail. From the part (c) of the latitude and longitude figures can be seen, that the diagonal entries of the covariance matrix for the fixed-point implementation becomes negative (even for the Joseph's implementation which is not depicted as was sad in the first paragraph of this section). This is not possible, because as was shown in the Kalman filter derivation, there is still a minimum of the Kalman filter criteria function, which can not be minimized to zero, hence we have only the non-zero and positive variances. However, the variances are negative, so it is obvious that this can be caused only due to the rounding errors since the double precision equivalent of the navigation system implementation have the variances with the positive sign.

If we look at another three figures 5.13 - 5.15, where the estimates for the east,

north and up velocities are depicted in the similar manner as before, then a very good tracking of the reference velocities can be seen. Even the fixed-point implementation goes well for the first 50s. After this time, the estimates of the error sates and their corresponding 3-sigma bounds start to diverge similarly as in the previous case of the position estimates. From the part (c) of the east and north velocity can be seen that the variances become negative after approximately 340s. This negative sing of the variance can be considered as the main reason why the conventional Kalman filter computed in the fixed-point arithmetic with 48bits in the fractional part of the computational word fails. However, one need to consider the estimated variances of the east and north velocities and their corresponding sigma bounds as they start to diverge after 50s (they are significantly different from their reference values represented by the estimates computed in the double precision floating-point arithmetic). From this can be stated, that the rounding in the system, causes a gradual divergence of the velocity estimated error. It is very important to compare that the 340s is less than the $\approx$ 400s when the variances of the other estimated error variables become negative, thus it is obvious that the velocity part of the navigation error model is the most affected one. Now, if we compare the up-direction velocity estimates and the estimates of the height from the previously mentioned figures of the position, then we can state that they are not affected due to rounding since their sigma bounds do not experience any divergence. Of course, they fail too, but this failure is caused due to the divergence of the other estimated states, but not due to a divergence of their variances.

Similar conclusions can be made about 3-sigma bounds corresponding to the roll and pitch angle depicted in figures 5.16 and 5.17 respectively. We can see that they do not indicate any divergence, but the estimates of the reference trajectory of the roll and pitch angles are a little bit different in comparison with respect to their double precision equivalents. This is probably caused mainly due to the transfer of the rounding errors from the velocity error estimates, as they experience the most significant influence of the rounding, into the roll and pith error estimates. If we take a look at the figure 5.18, where the yaw angle estimates are depicted, then we can see that the 3-sigma bounds are affected too, so the slow divergence is apparent.

Figures 5.19 - 5.21 show the estimates for the accelerometers' bias part of the navigation system algorithm. There is not any visible discrepancy of the 1-sigma bounds (68.23%) for the all three components of these biases within the time range from 0s to approximately 400s. However, their reference trajectory estimates, depicted in the parts (a) of the figures 5.19 - 5.21, are affected. Assuming that the bias is in the navigation algorithm modelled only as a constant, which is corrected by the error increments, thus the only mathematical operation at this point is summation, and that these error increments are computed only from the multiplication of the

Kalman gain and the actual innovation, then we can state, that the only possible source of rounding error, can arise in this multiplication.

Similar conclusions can be made about the gyroscopes' biases depicted in figure 5.22 - 5.24. The z-axis bias in figure 5.24a converges very slowly, thus the reference value is reached after 300s, even for the floating-point implementation. This is caused due to weak observability of this channel [25]. The solution lies in an incorporation of the so-called zero update velocity (ZUPT) into the navigation system algorithm.

5.3.4 Estimated Trajectories of UD Factorized Kalman Filter

Here we show and describe results for the UD factorized Kalman filter with the same simulation conditions as in the previous case of the conventional Kalman filter implementation. A comparison of the estimated trajectory, estimated error variables with the corresponding sigma bounds and variances, for the fixed-point implementation of the UD factorized Kalman filter and the floating-point implementation of the conventional Kalman filter, is depicted in figures 5.25 - 5.39. The results of the Square Root Kalman filter are not shown as they are nearly identical with respect to the presented UD factorized Kalman filter.

If we look at figures, where the latitude, longitude and height are depicted, thus at the figures 5.25, 5.26 and 5.27 respectively, then we can see, that the estimation goes well. A closer look shows, that there is a little discrepancy in the estimates of the latitude and longitude error variables and their sigma bounds. This can be observed in figures 5.1 and 5.2 as that part of the bars which is above the red line, thus it is obvious that the length of 48bits in the fractional part of the fixed-point computational word makes the performance of the UD factorized Kalman filter a little bit worse, but the error and trajectory estimates do not diverge over the whole estimation period. It is obvious that the rounding errors make the resulting performance suboptimal, however a divergence do not occur. The figures 5.1-5.5 can make an impression that there is not a significant save in the number of bits between the conventional, Joseph's, UD factorized and Square Root implementations, but one need to consider the results of the covariance matrix and their factors conditioning. As was shown in figures 5.6 - 5.8 the numerical stability of the covariance matrix is significantly threatened, however the numerical stability of the UD and Square Root factors seems to be good. The rest of the depicted variables brings the similar conclusions. It is obvious that the estimation performance is affected only for the variables which do not represent the estimation of the constant value, thus the variables which do not experience any significant movement.

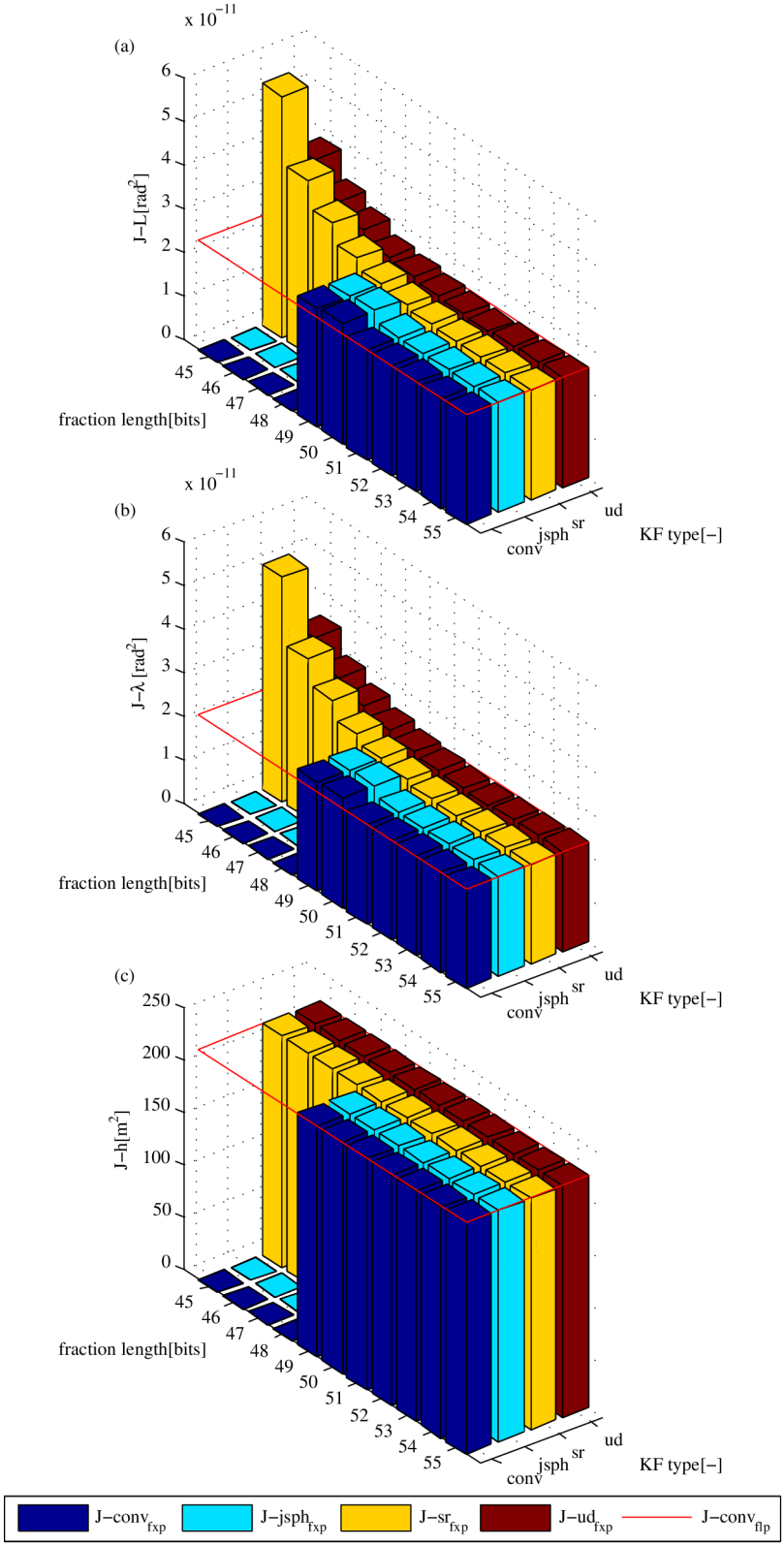


Fig. 5.1: Values of the criteria function J for the latitude (a), longitude (b) and height (c), computed with using the fixed-point arithmetic, which has 24bits in the integer part and various number of bits in the fractional part. All previously mentioned Kalman filter implementations are compared with respect to the conventional Kalman filter computed in the double precision floating-point arithmetic.

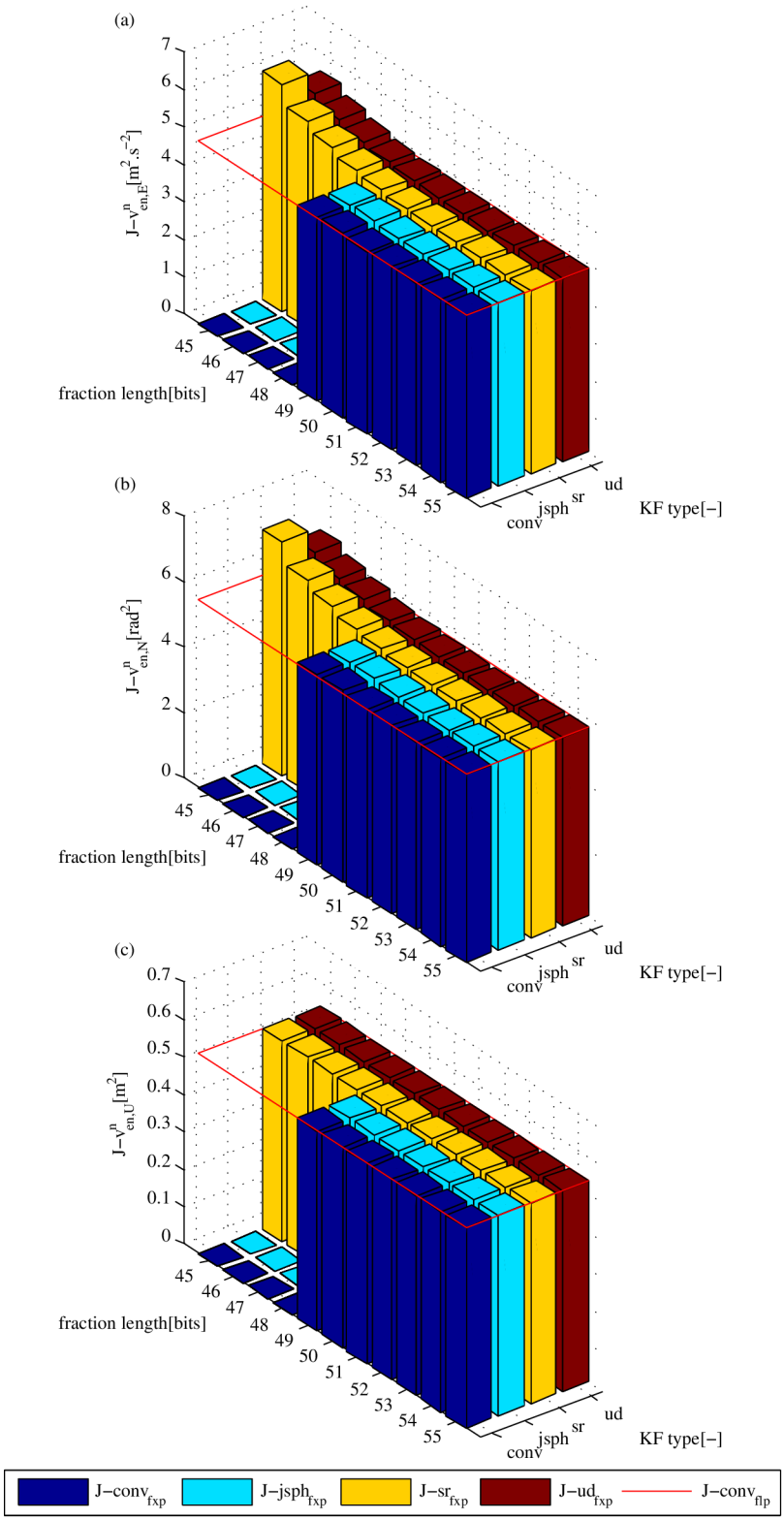


Fig. 5.2: Values of the criteria function J for the east (a), north (b) and up (c) velocities, computed with using the fixed-point arithmetic, which has 24bits in the integer part and various number of bits in the fractional part. All previously mentioned Kalman filter implementations are compared with respect to the conventional Kalman filter computed in the double precision floating-point arithmetic.

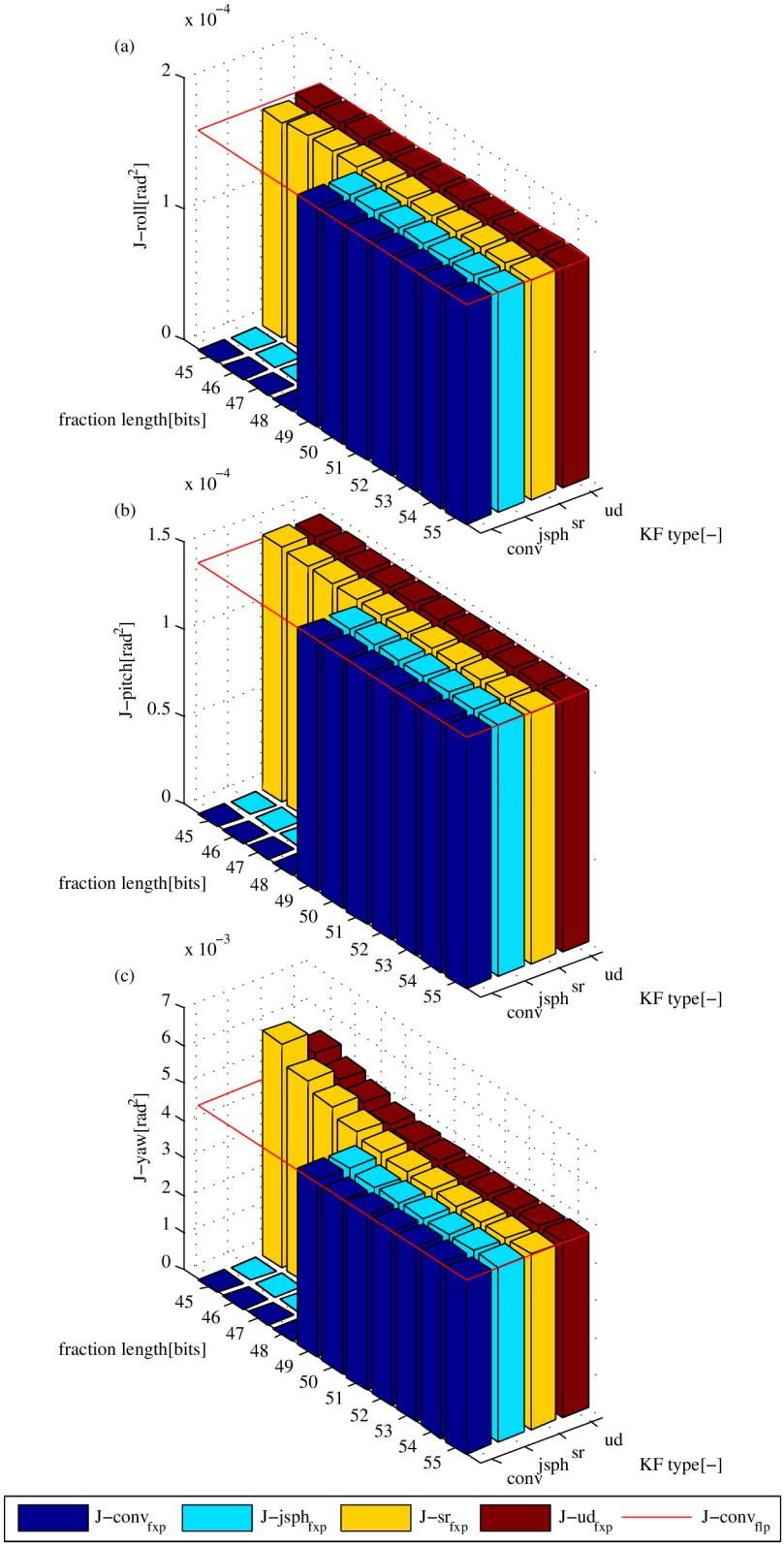


Fig. 5.3: Values of the criteria function J for the roll (a), pitch (b) and yaw (c) angles, computed with using the fixed-point arithmetic, which has 24bits in the integer part and various number of bits in the fractional part. All previously mentioned Kalman filter implementations are compared with respect to the conventional Kalman filter computed in the double precision floating-point arithmetic.

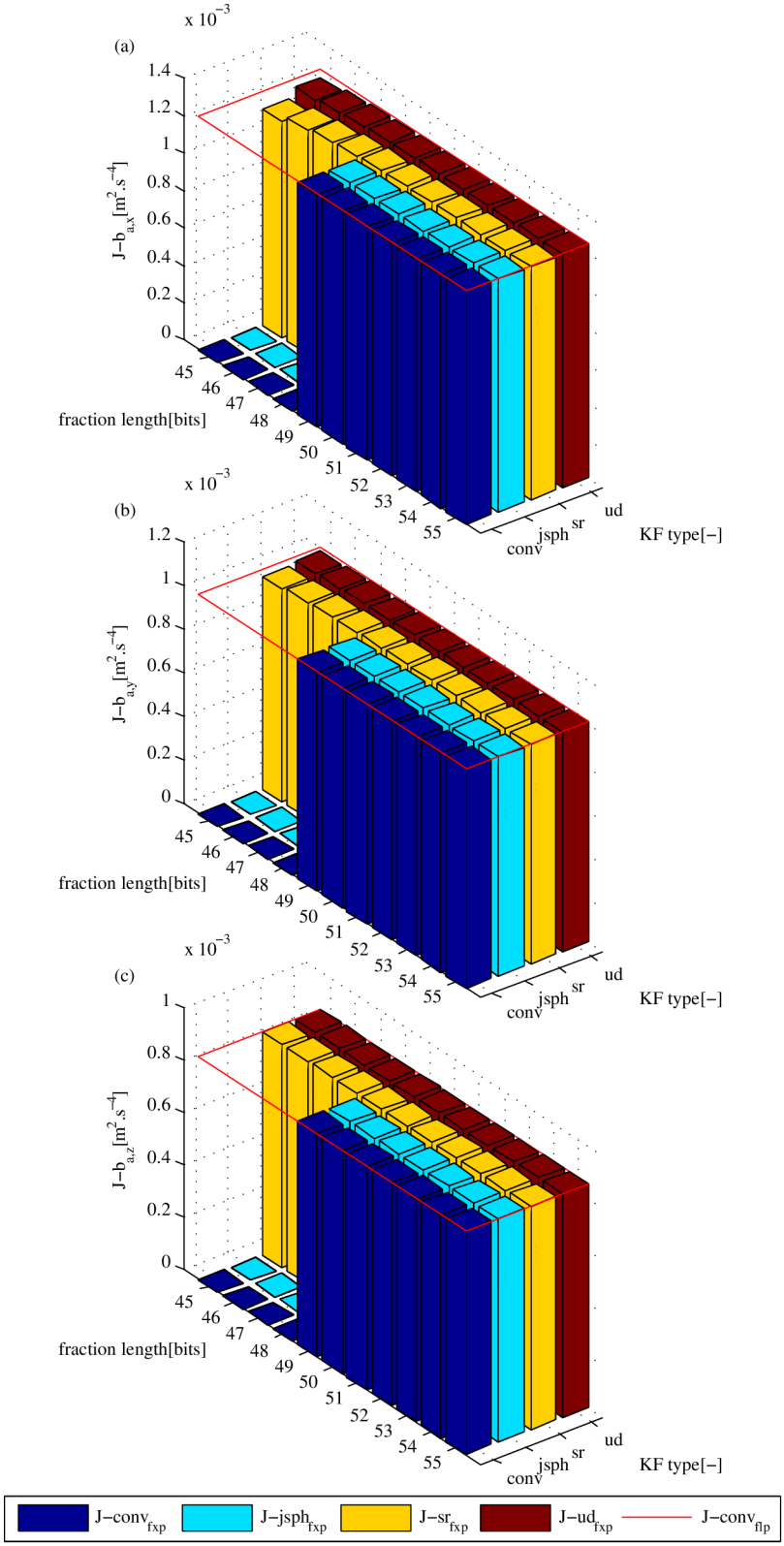


Fig. 5.4: Values of the criteria function J for the accelerometer x (a), y (b) and z (c) axis biases, computed with using the fixed-point arithmetic, which has 24bits in the integer part and various number of bits in the fractional part. All previously mentioned Kalman filter implementations are compared with respect to the conventional Kalman filter computed in the double precision floating-point arithmetic.

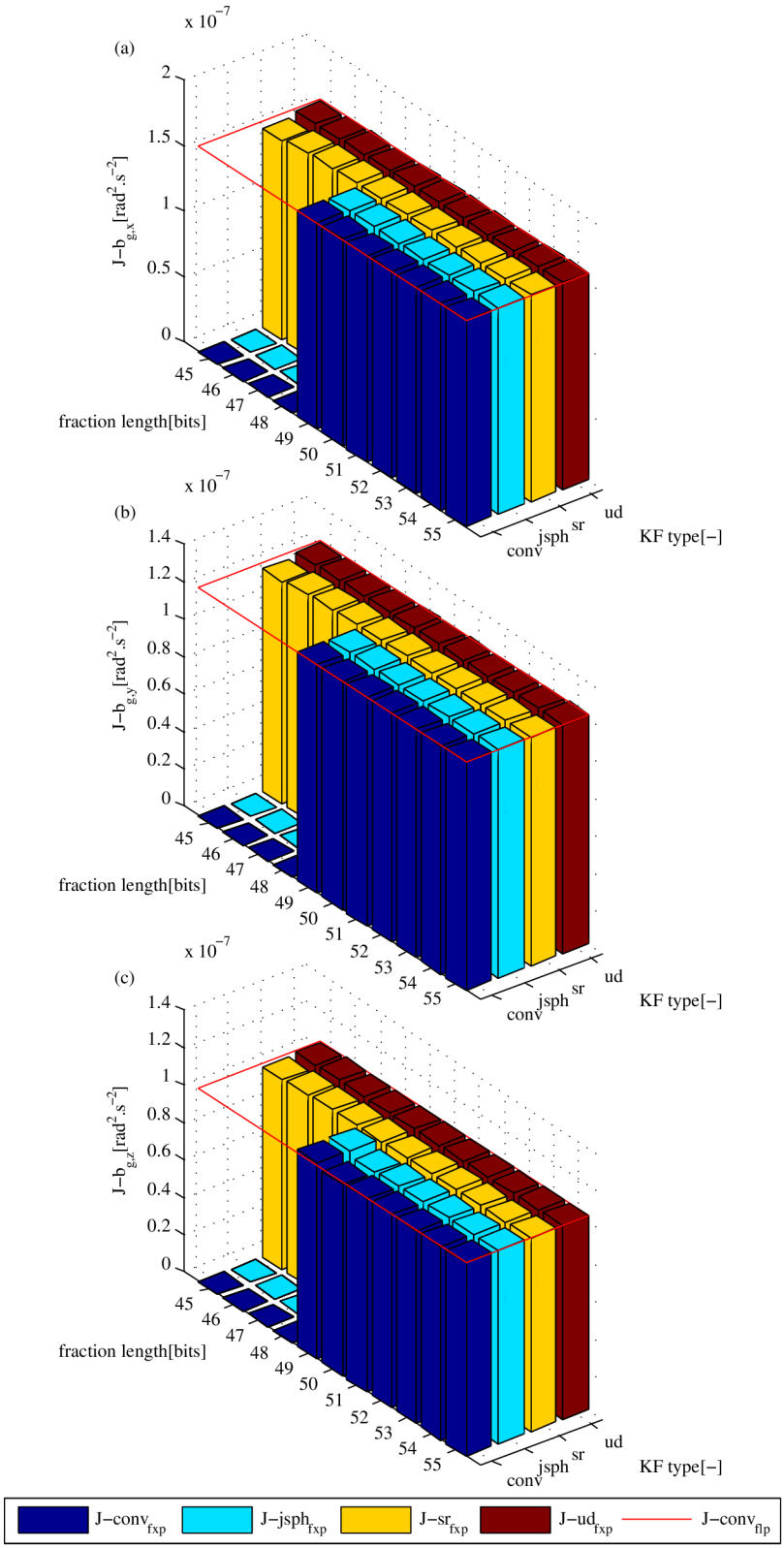


Fig. 5.5: Values of the criteria function J for the gyroscope x (a), y (b) and z (c) axis biases, computed with using the fixed-point arithmetic, which has 24bits in the integer part and various number of bits in the fractional part. All previously mentioned Kalman filter implementations are compared with respect to the conventional Kalman filter computed in the double precision floating-point arithmetic.

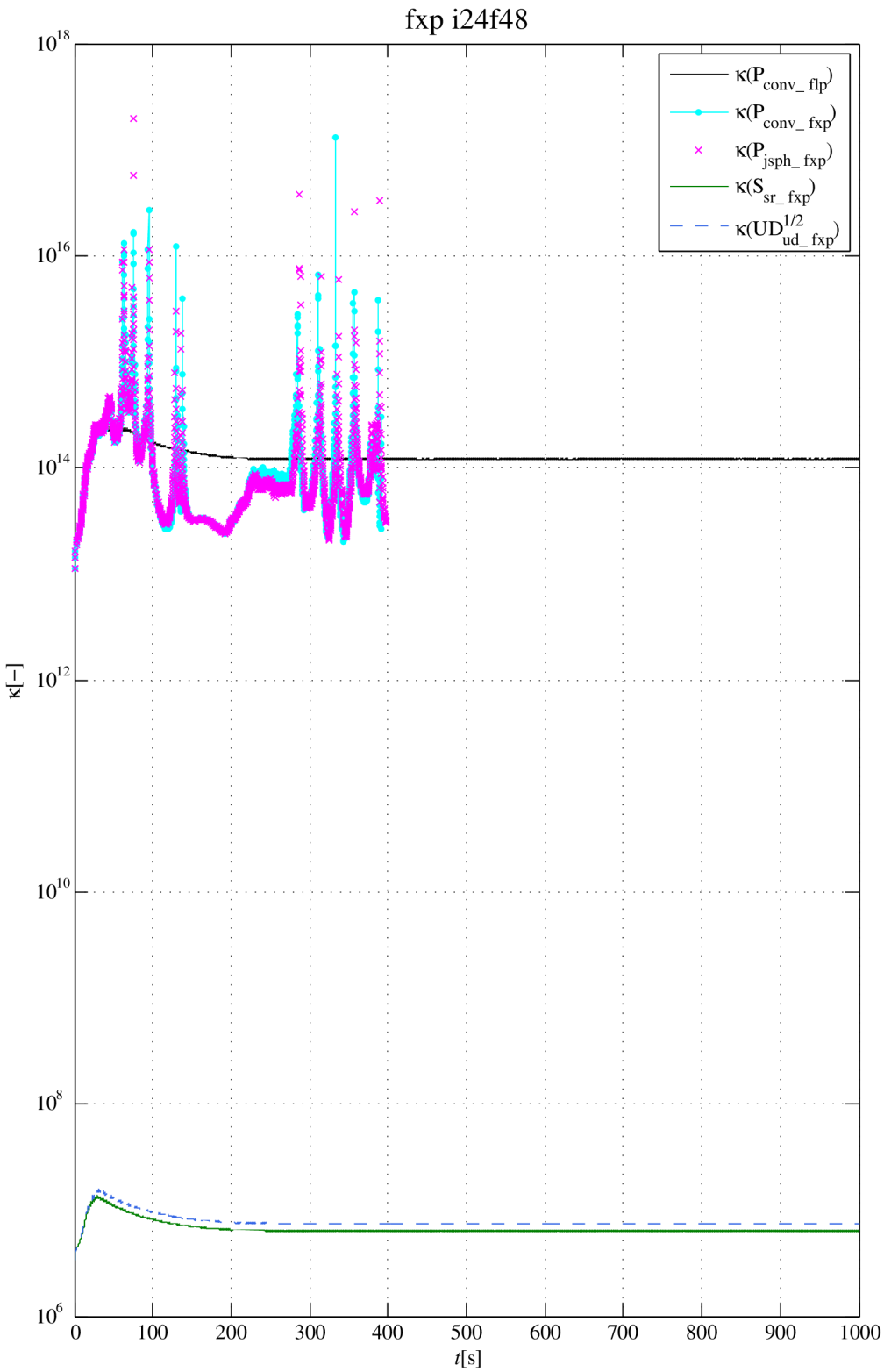


Fig. 5.6: The covariance matrix and its factors condition numbers computed with using the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part. The black line represents the condition number for the conventional Kalman filter computed in the double precision floating-point arithmetic.

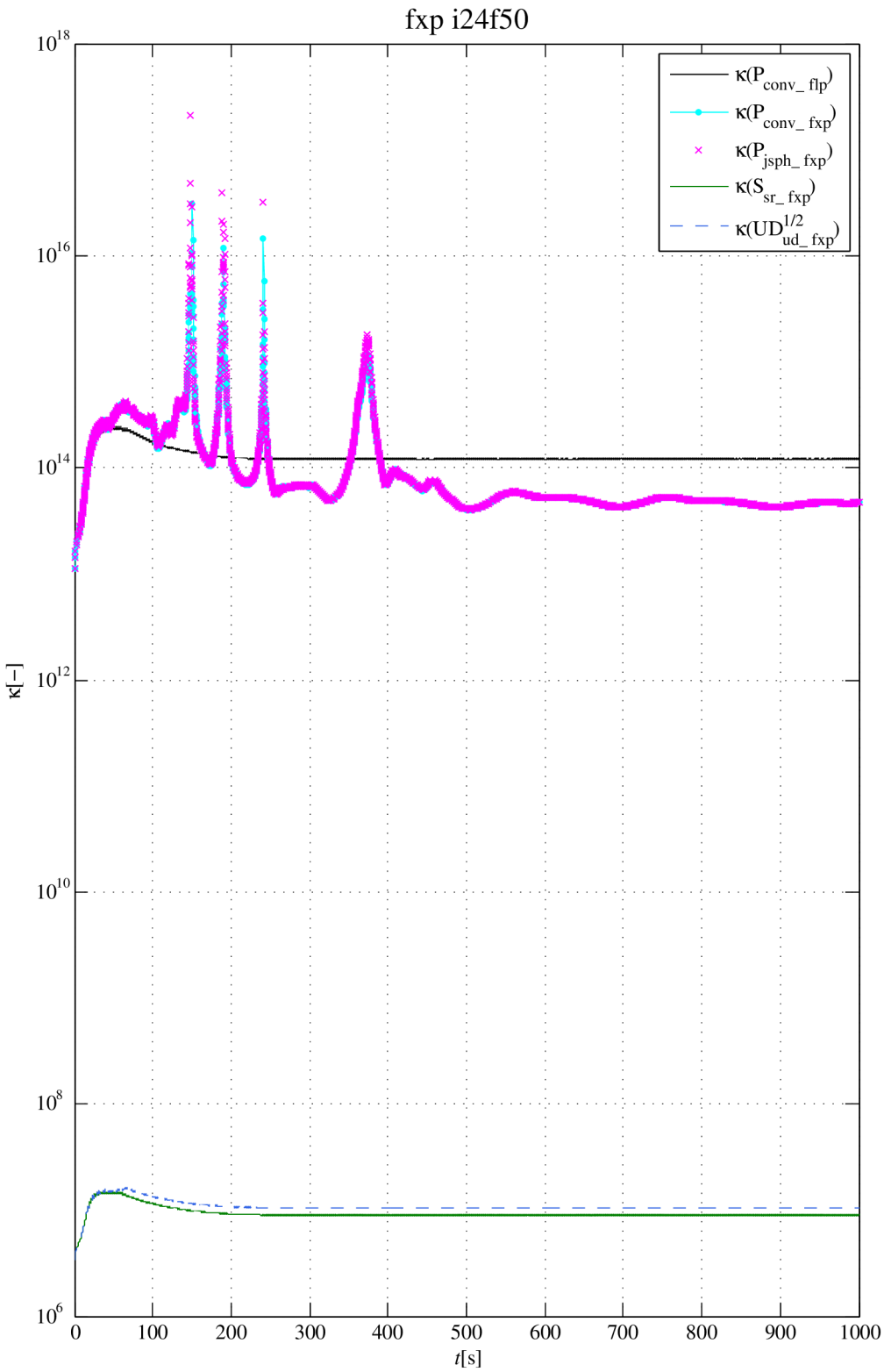


Fig. 5.7: The covariance matrix and its factors condition numbers computed with using the fixed-point arithmetic, which has 24bits in the integer part and 50bits in the fractional part. The black line represents the condition number for the conventional Kalman filter computed in the double precision floating-point arithmetic.

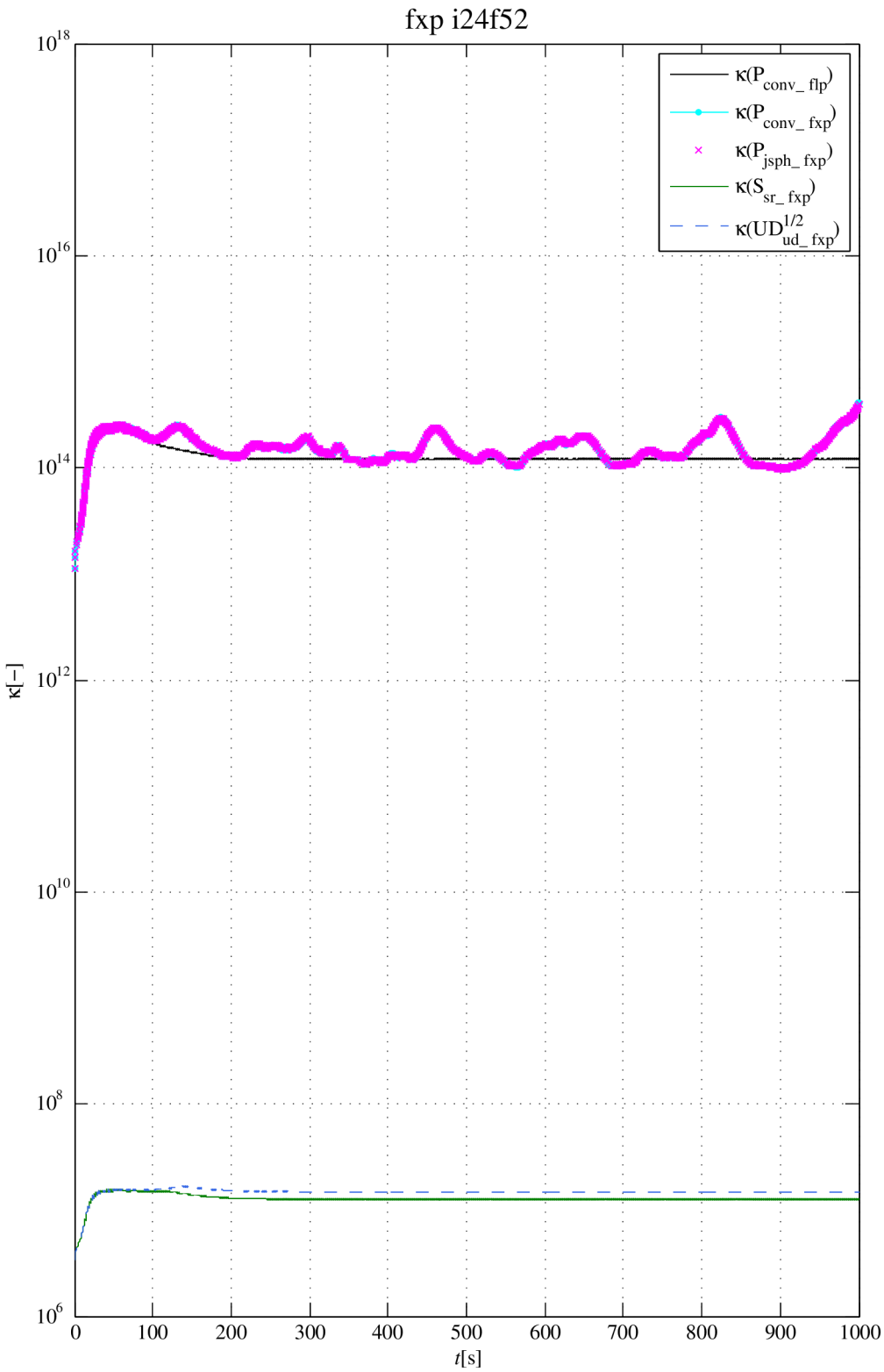


Fig. 5.8: The covariance matrix and its factors condition numbers computed with using the fixed-point arithmetic, which has 24bits in the integer part and 52bits in the fractional part. The black line represents the condition number for the conventional Kalman filter computed in the double precision floating-point arithmetic.

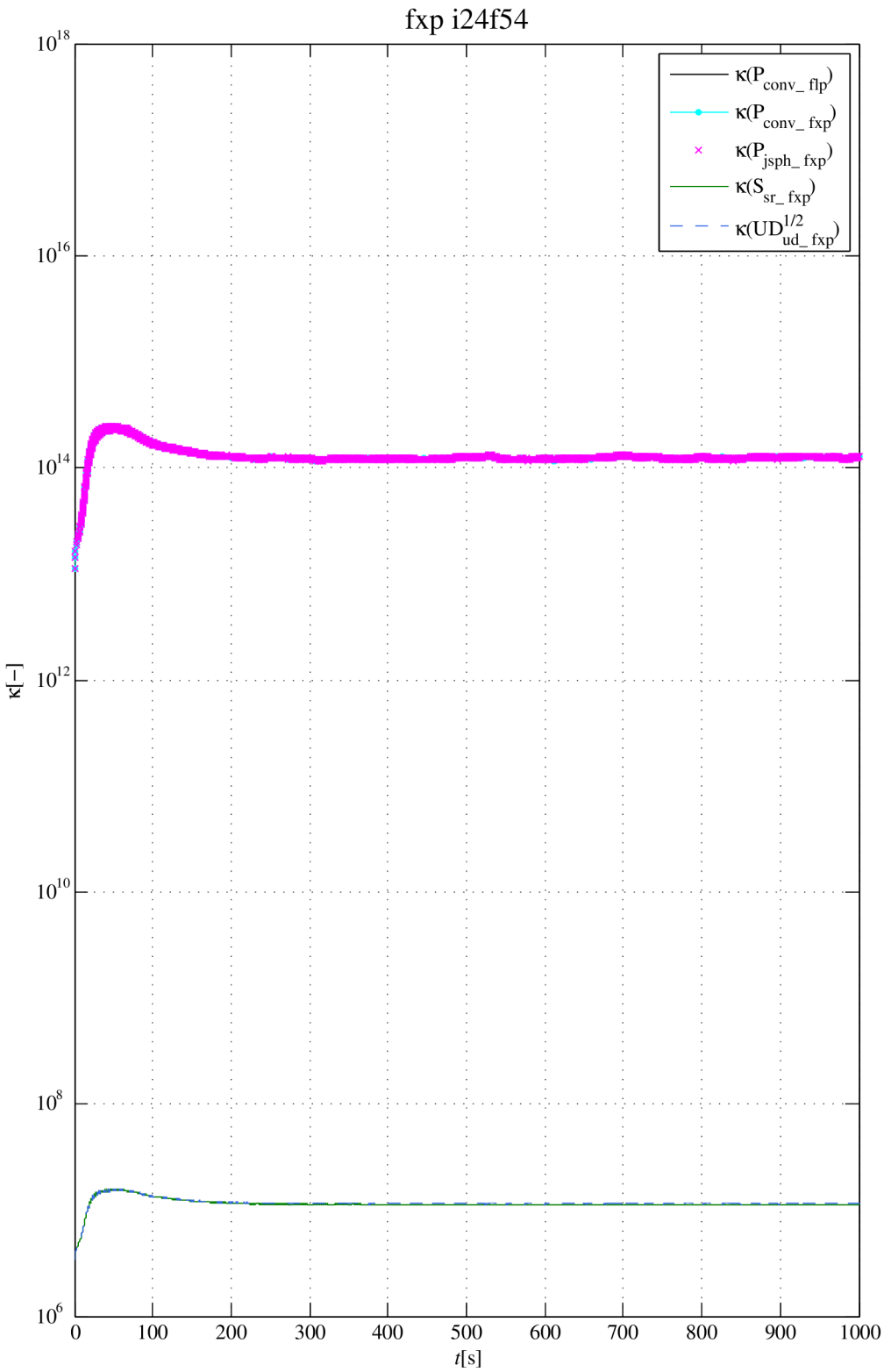


Fig. 5.9: The covariance matrix and its factors condition numbers computed with using the fixed-point arithmetic, which has 24bits in the integer part and 54bits in the fractional part. The black line represents the condition number for the conventional Kalman filter computed in the double precision floating-point arithmetic.

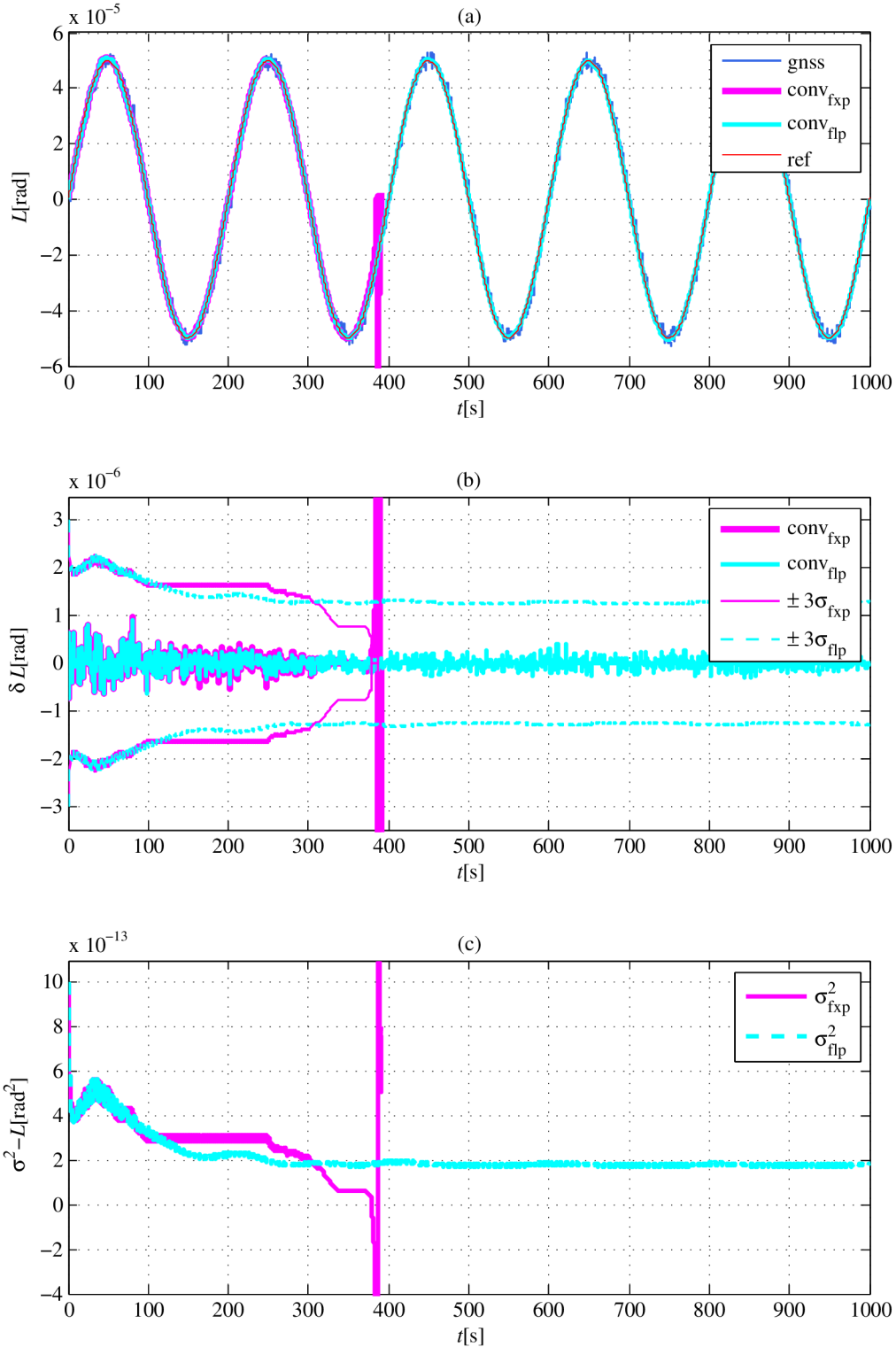


Fig. 5.10: Estimates of the latitude (a), latitude error with corresponding 3-sigma bounds (b) and latitude variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

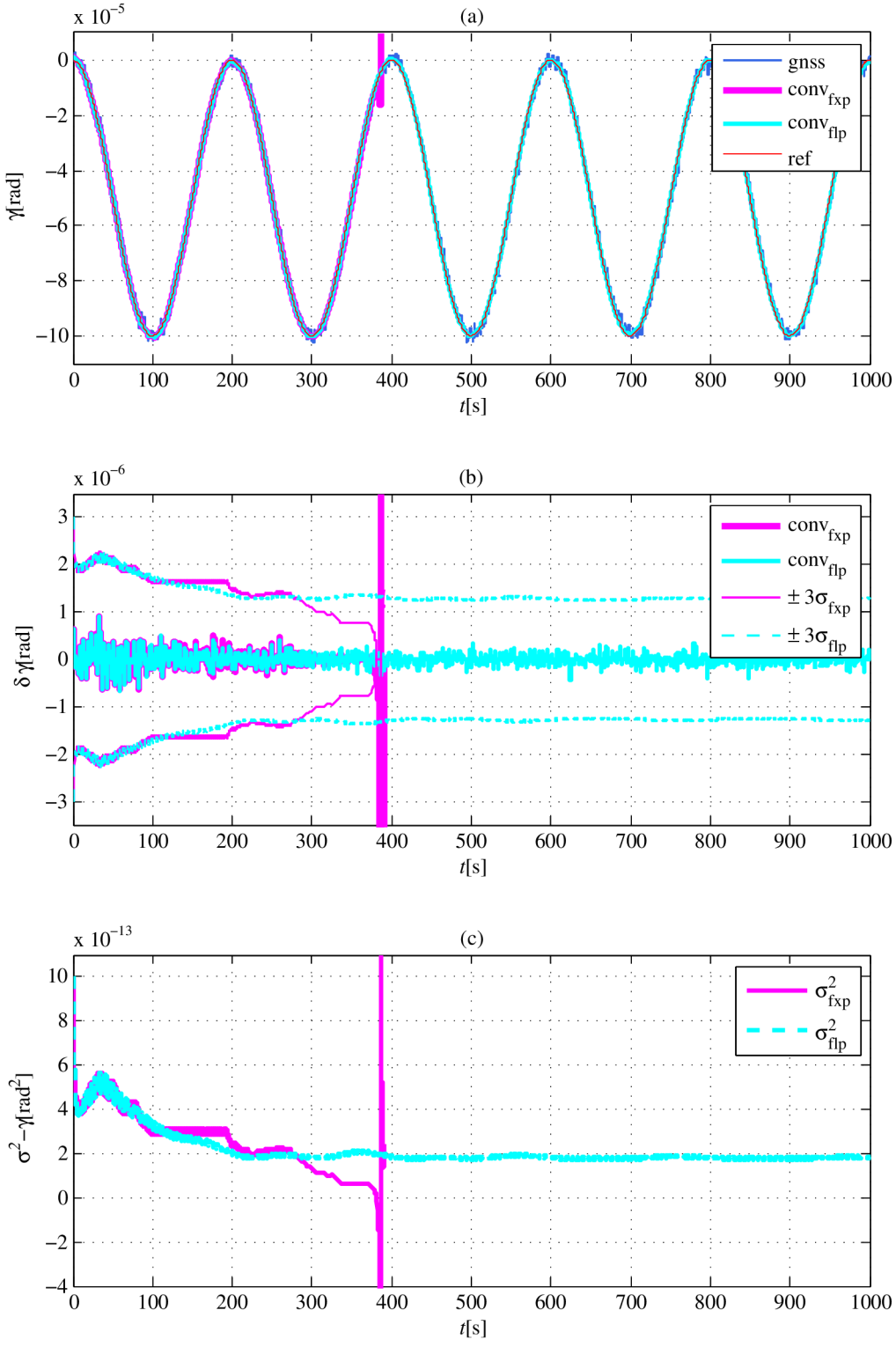


Fig. 5.11: Estimates of the longitude (a), longitude error with corresponding 3-sigma bounds (b) and longitude variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

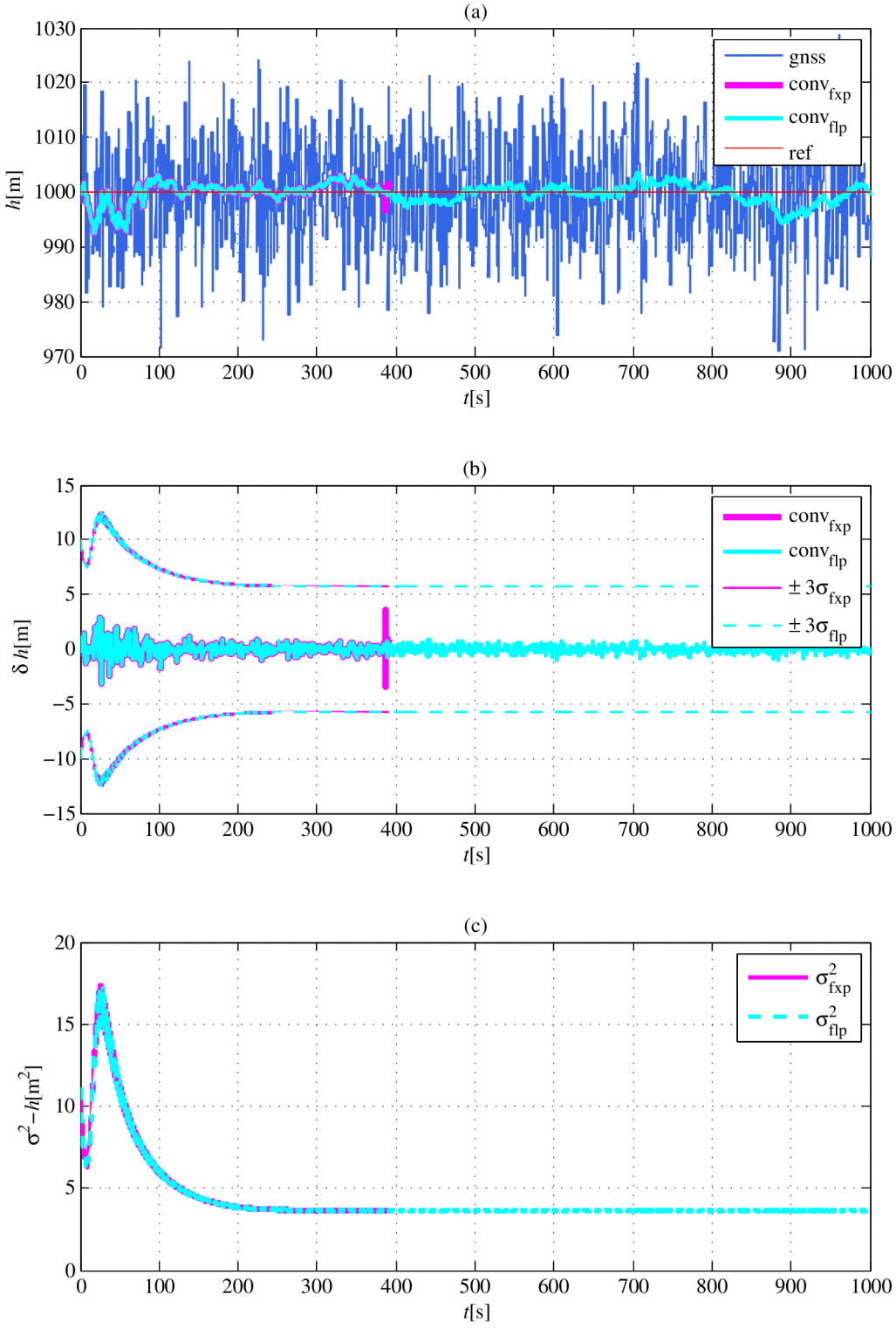


Fig. 5.12: Estimates of the height (a), height error with corresponding 3-sigma bounds (b) and height variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

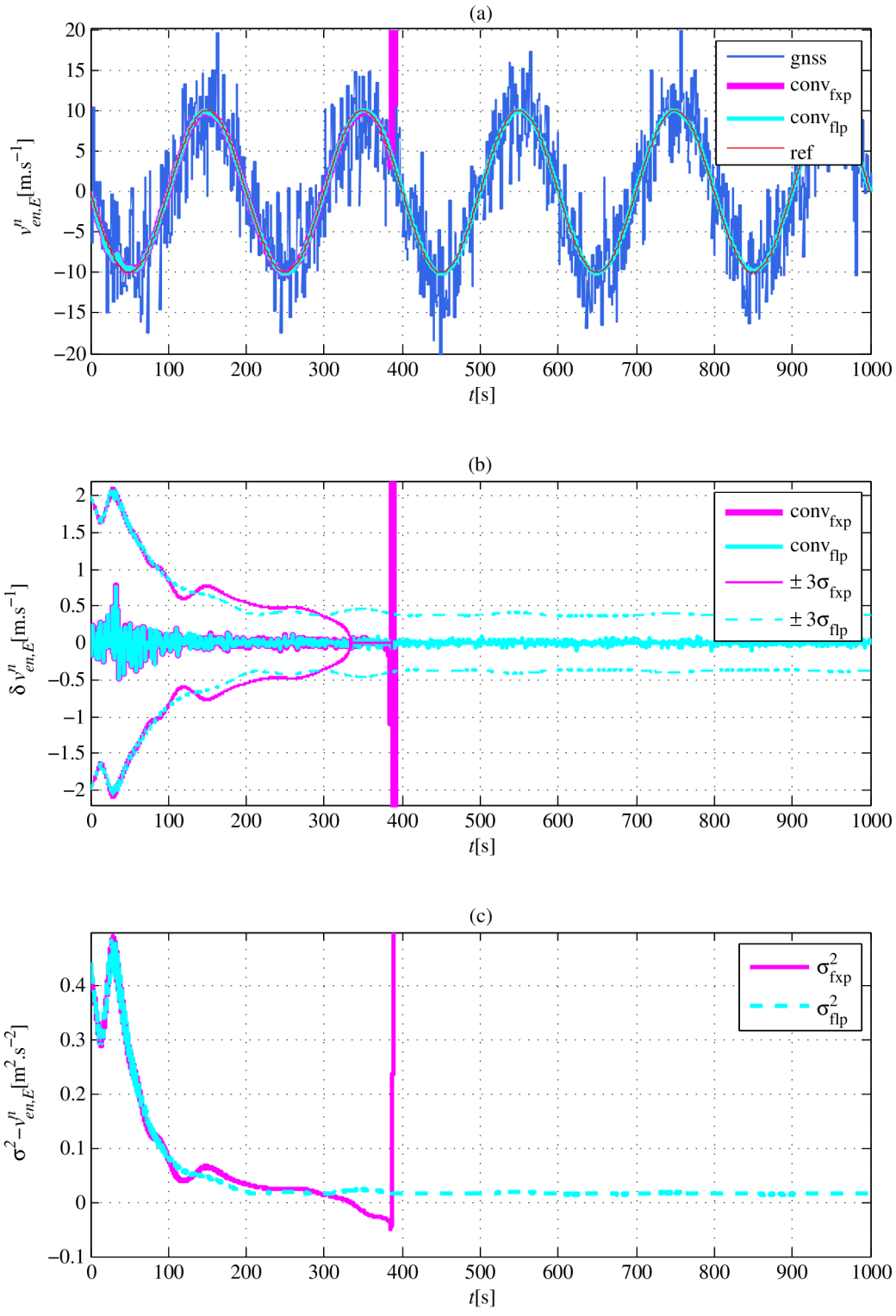


Fig. 5.13: Estimates of the east velocity (a), east velocity error with corresponding 3-sigma bounds (b) and east velocity variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

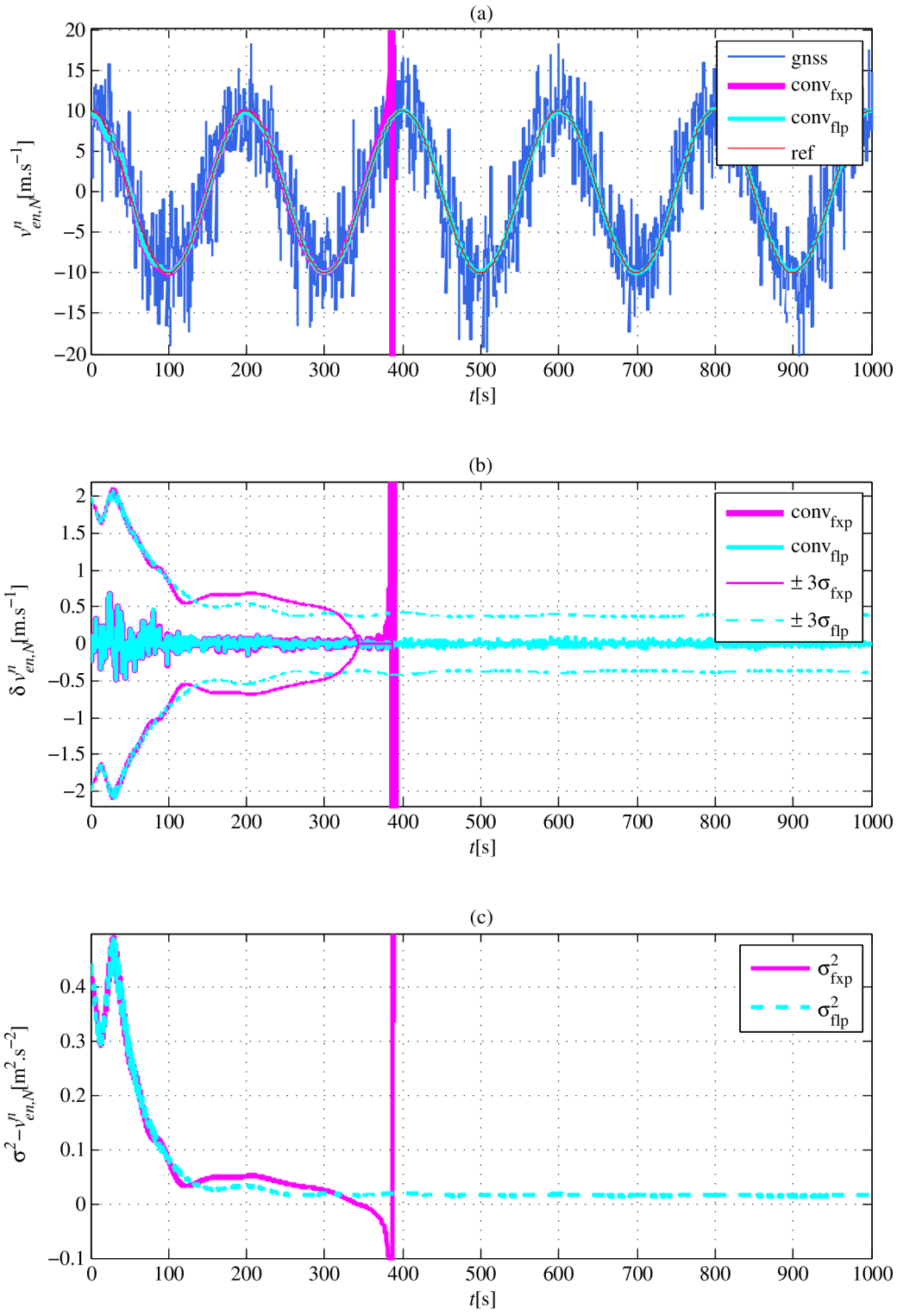


Fig. 5.14: Estimates of the north velocity (a), north velocity error with corresponding 3-sigma bounds (b) and north velocity variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

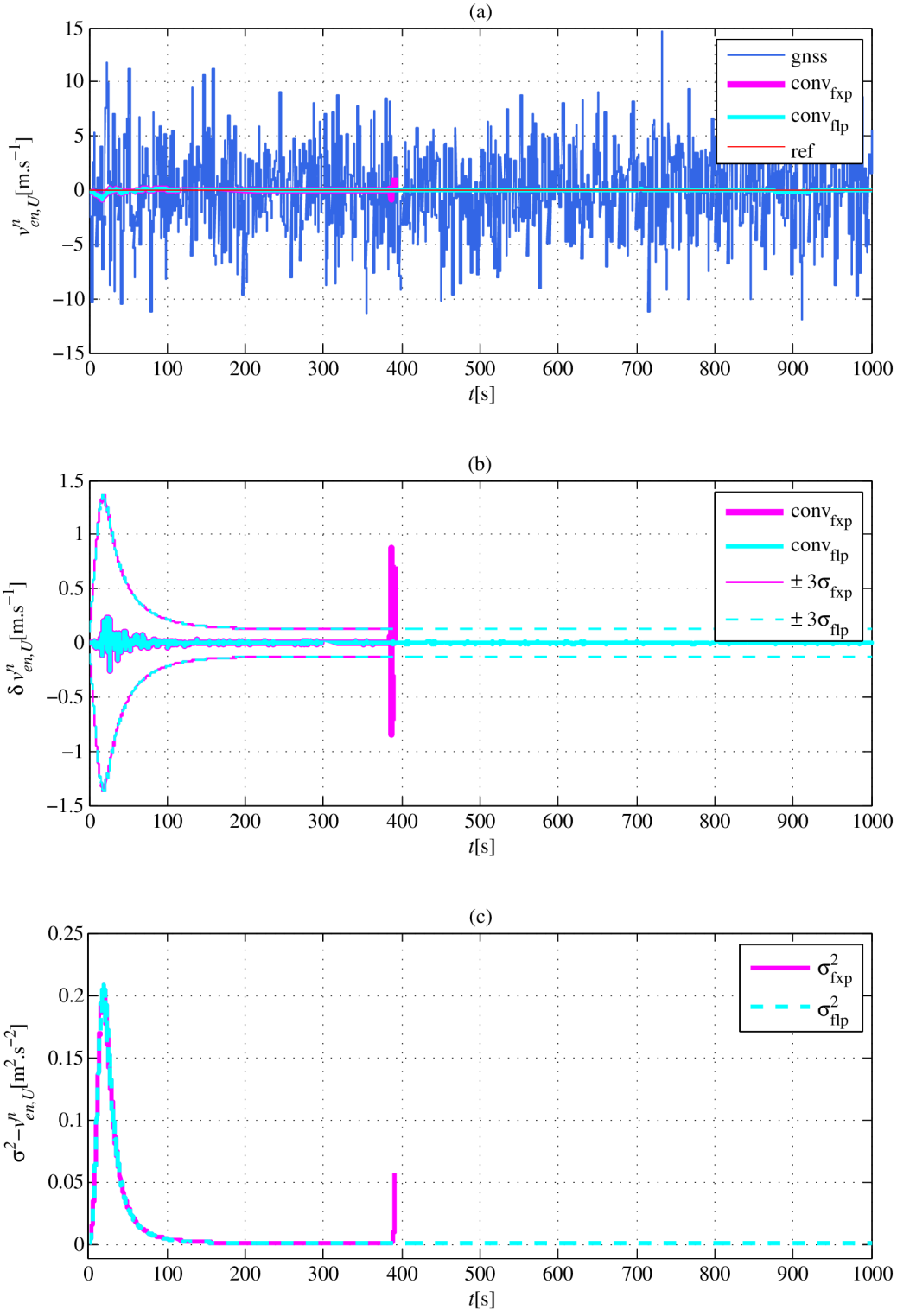


Fig. 5.15: Estimates of the up velocity (a), up velocity error with corresponding 3-sigma bounds (b) and up velocity variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory and blue line the GNSS measurements.

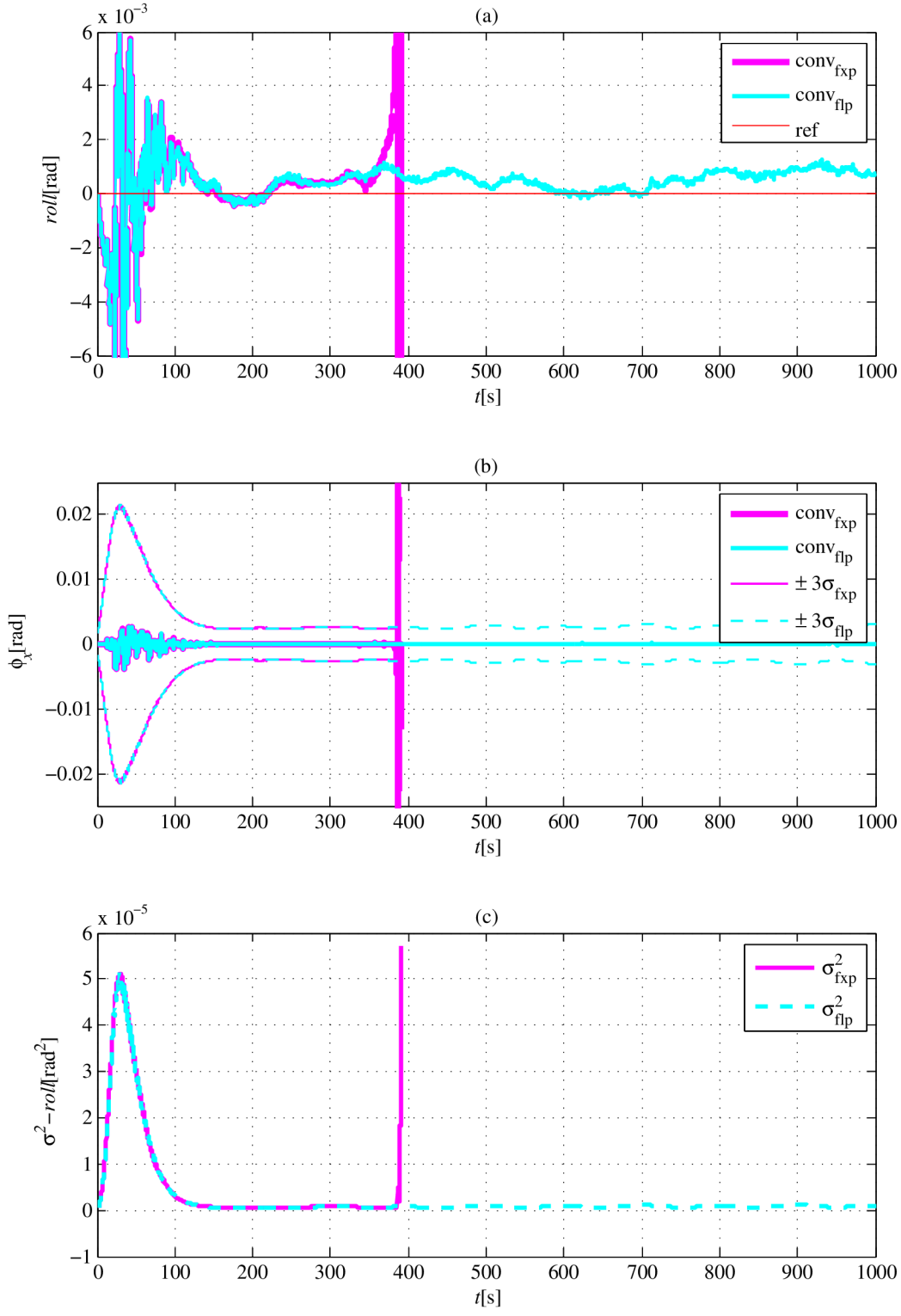


Fig. 5.16: Estimates of the roll (a), roll error with corresponding 3-sigma bounds (b) and roll variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

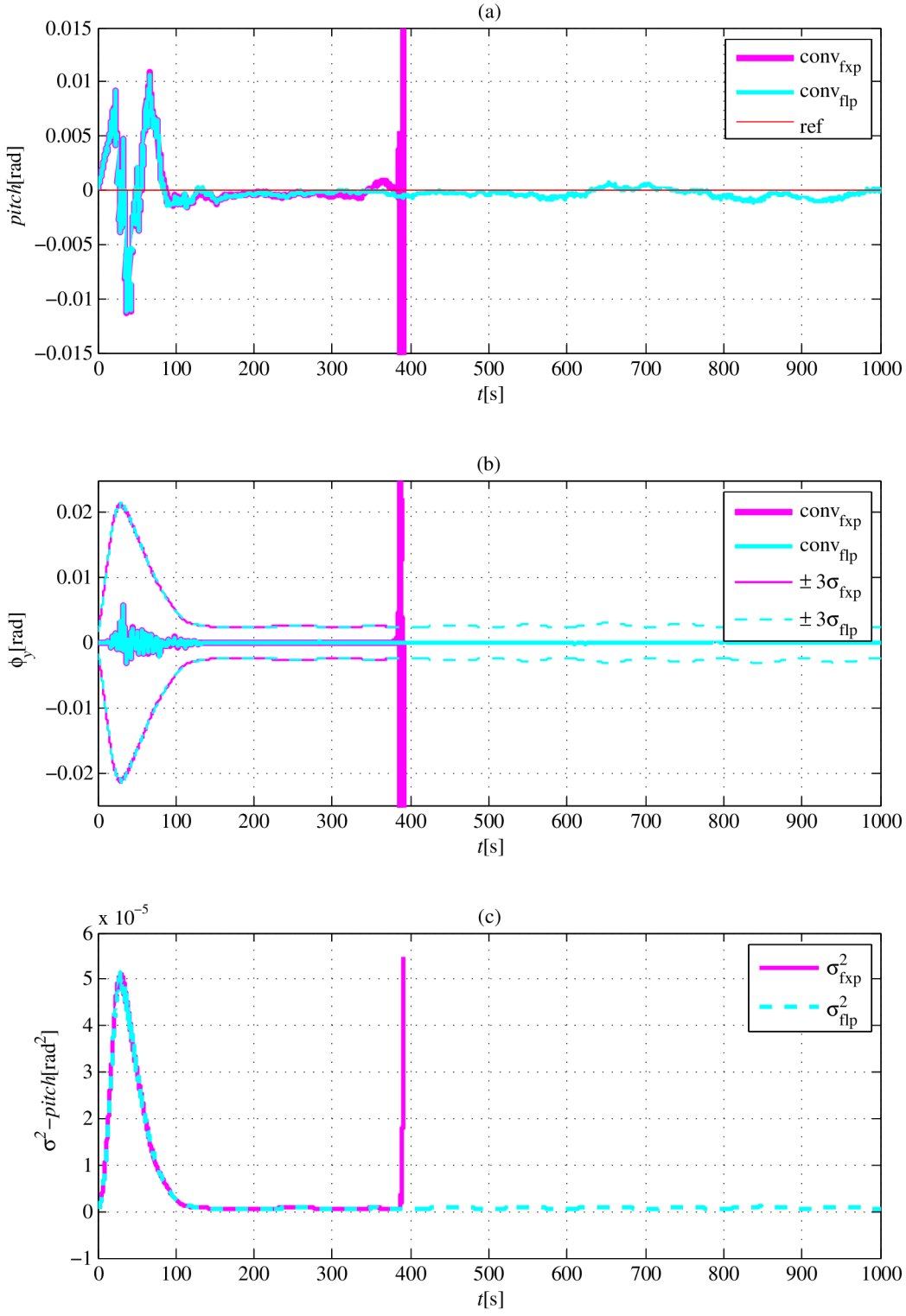


Fig. 5.17: Estimates of the pitch (a), pitch error with corresponding 3-sigma bounds (b) and pitch variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

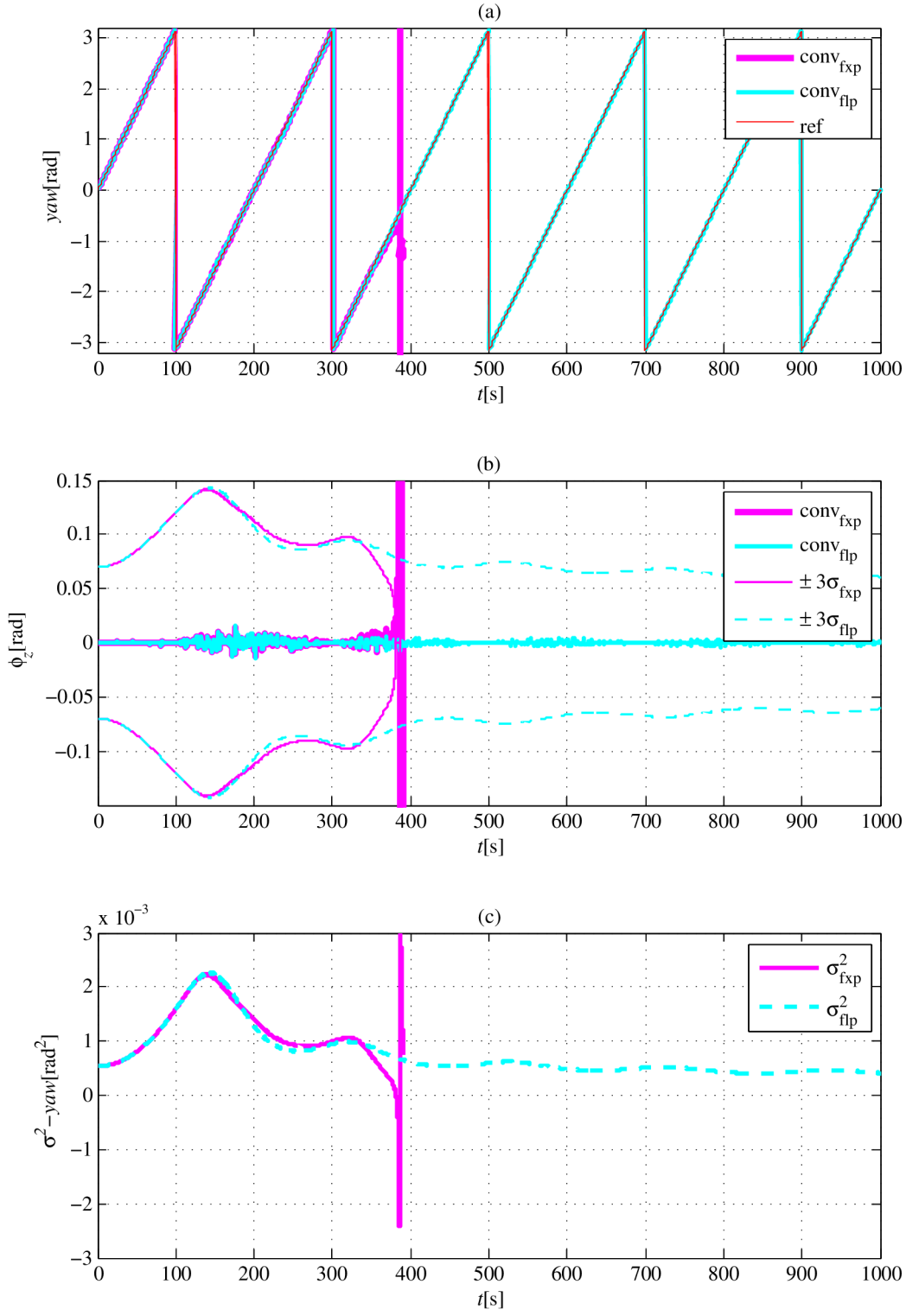


Fig. 5.18: Estimates of the yaw (a), yaw error with corresponding 3-sigma bounds (b) and yaw variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

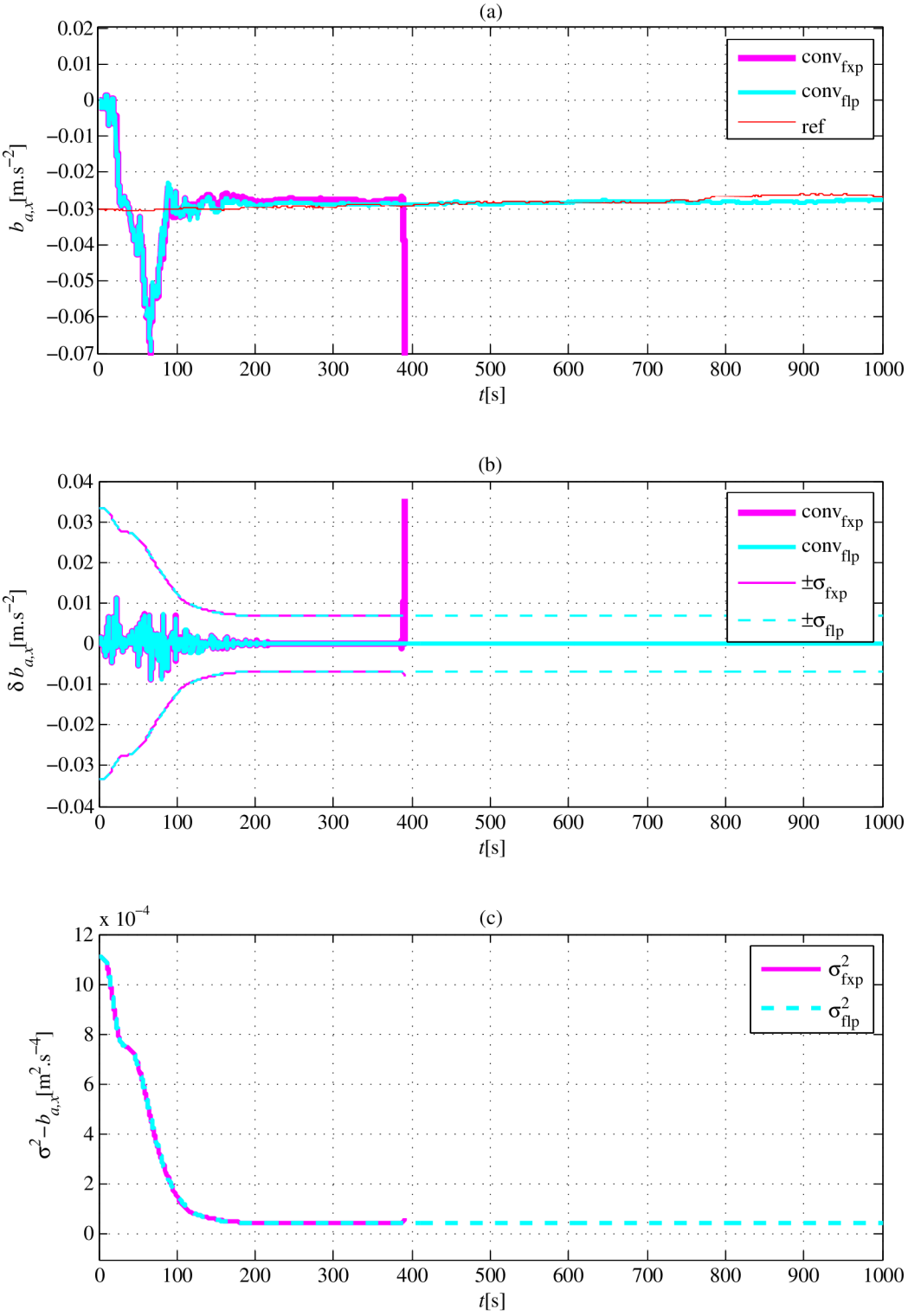


Fig. 5.19: Estimates of the accelerometers' x-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

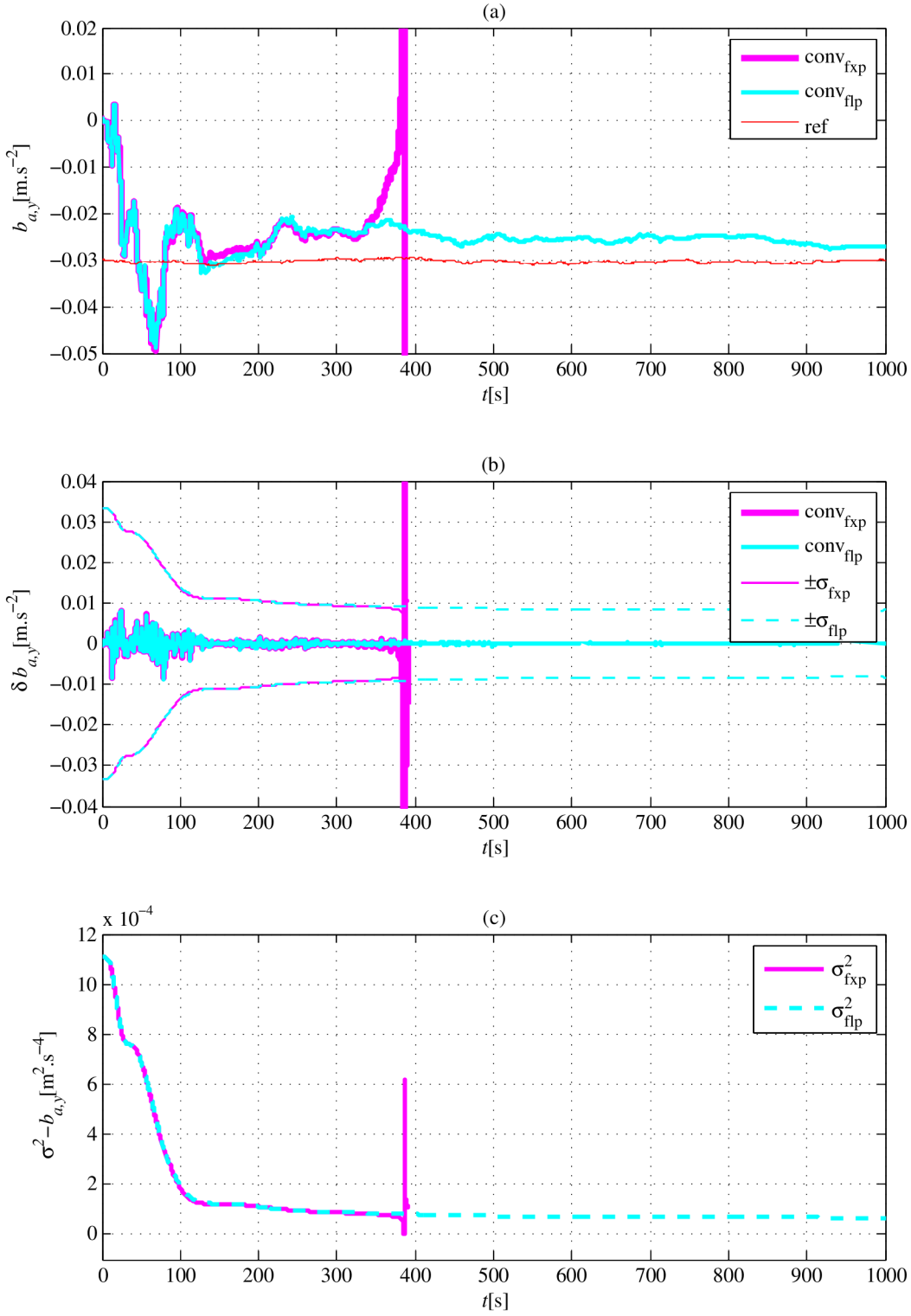


Fig. 5.20: Estimates of the accelerometers' y-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

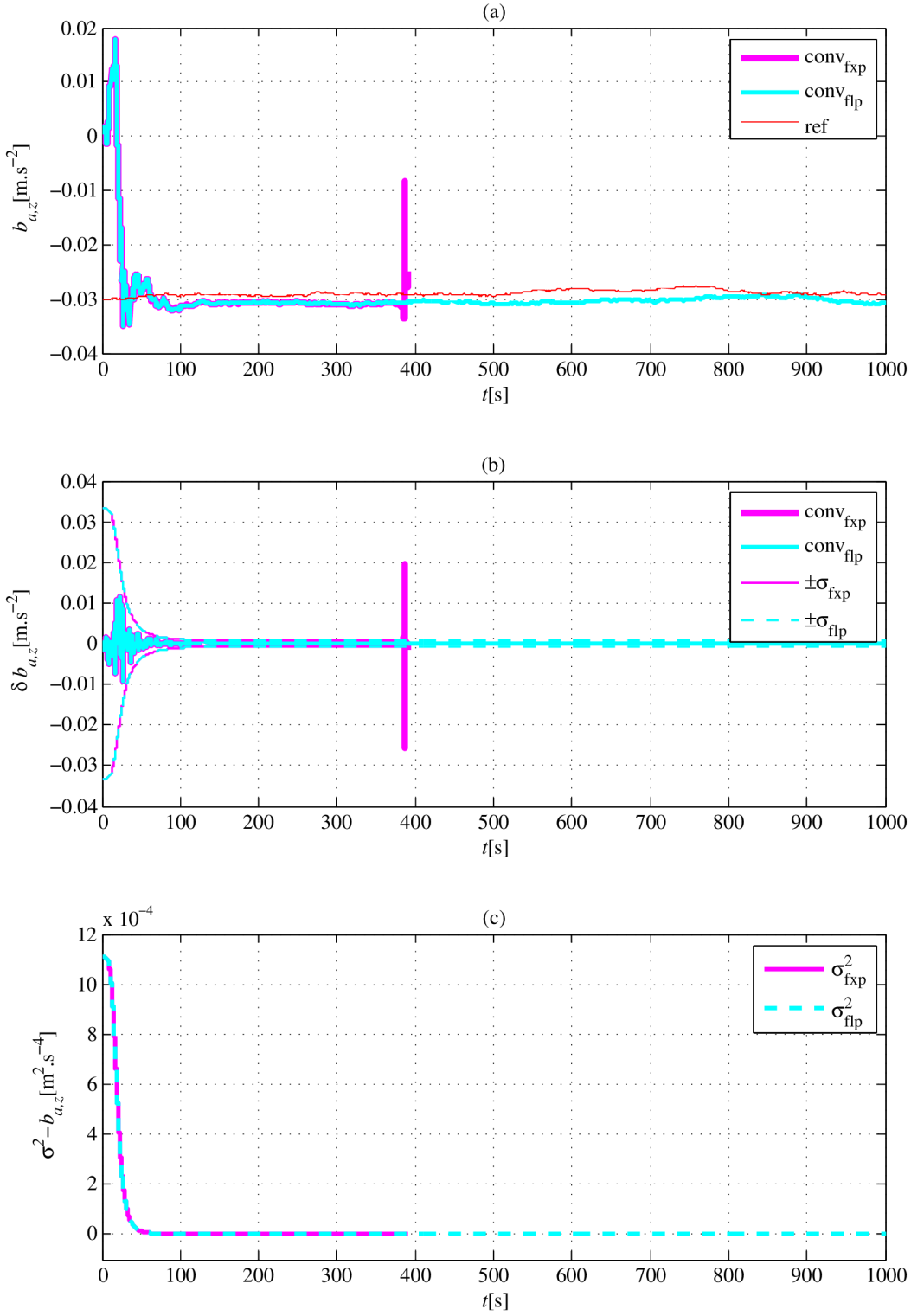


Fig. 5.21: Estimates of the accelerometers' z-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

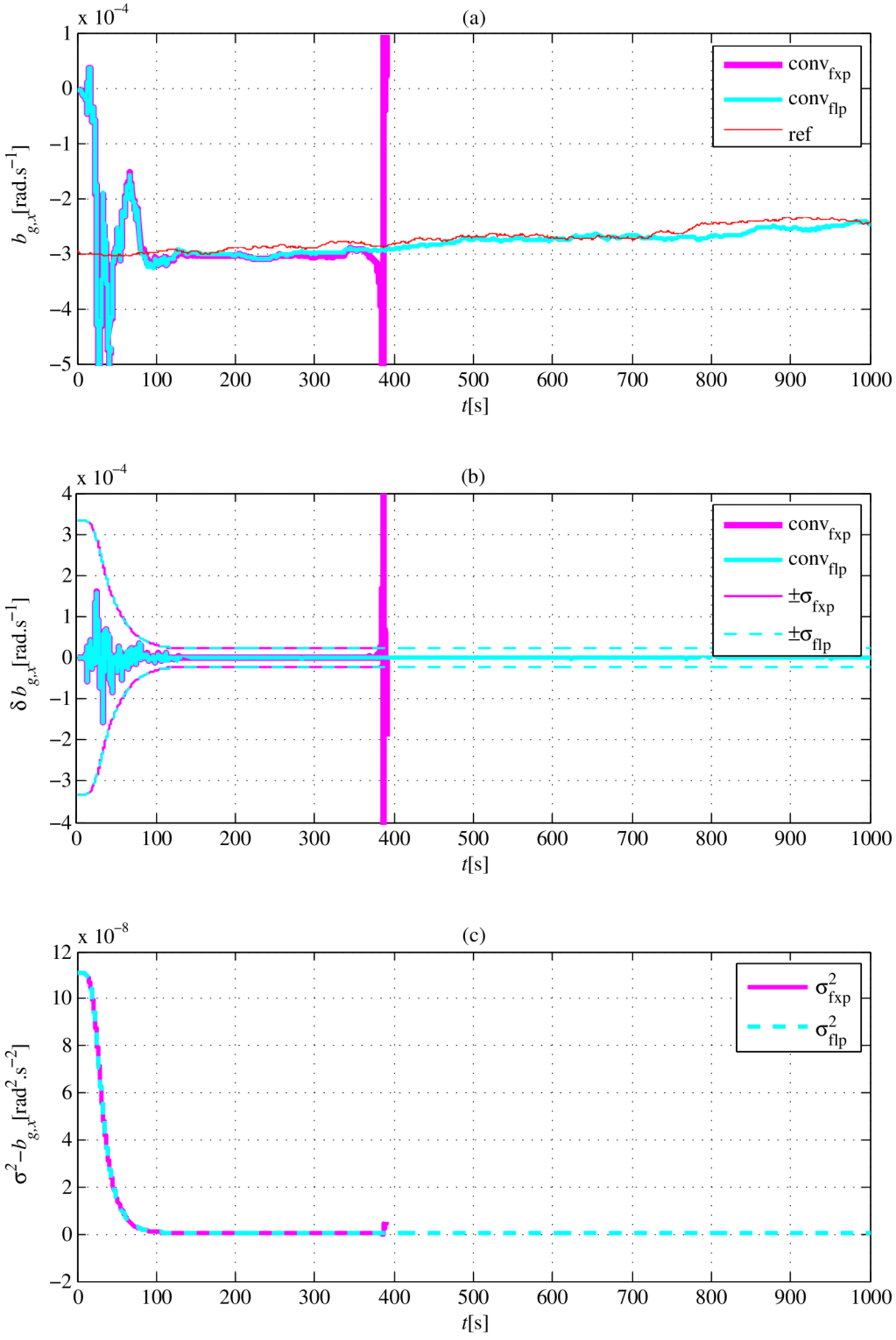


Fig. 5.22: Estimates of the gyroscopes' x-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

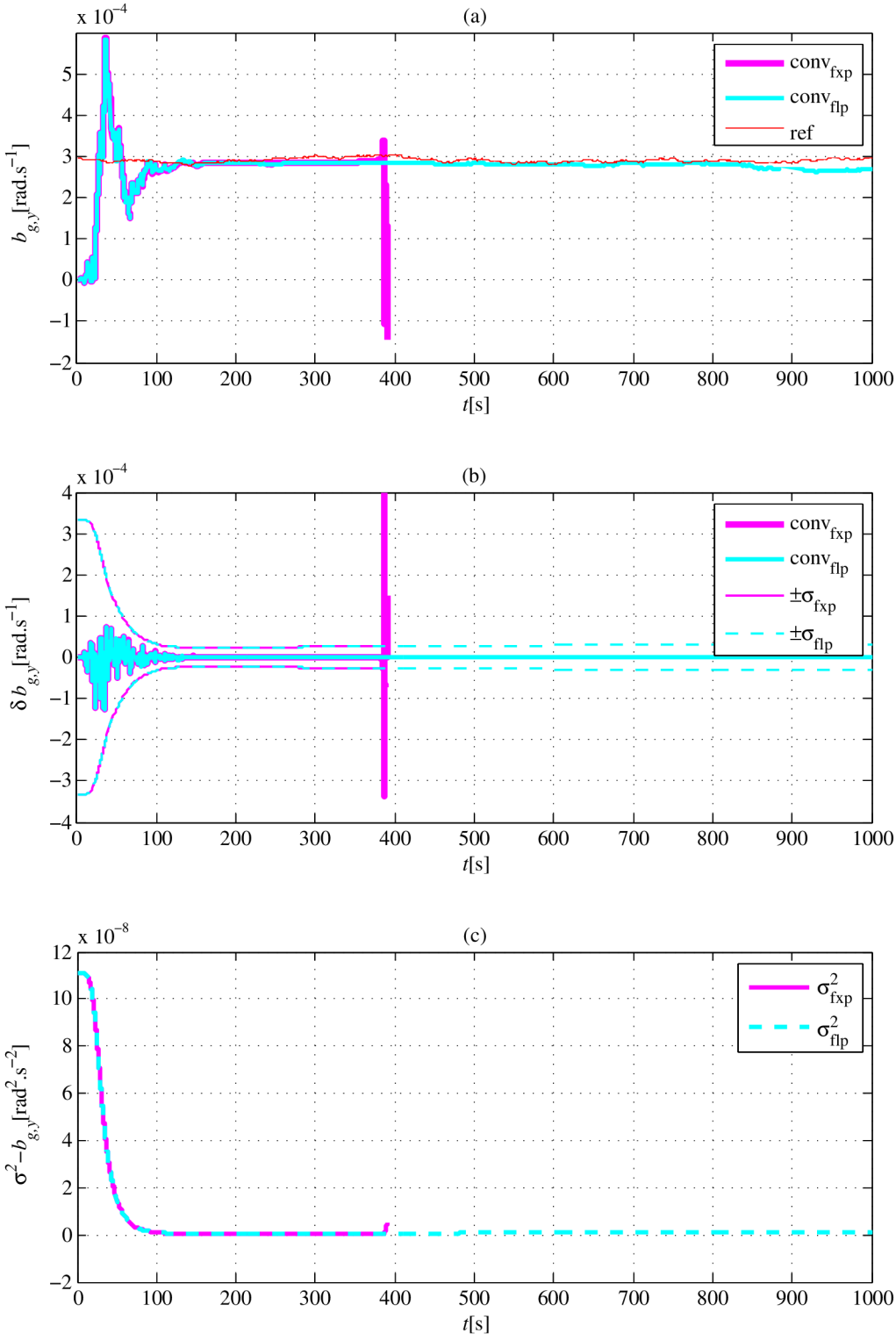


Fig. 5.23: Estimates of the gyroscopes' y-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

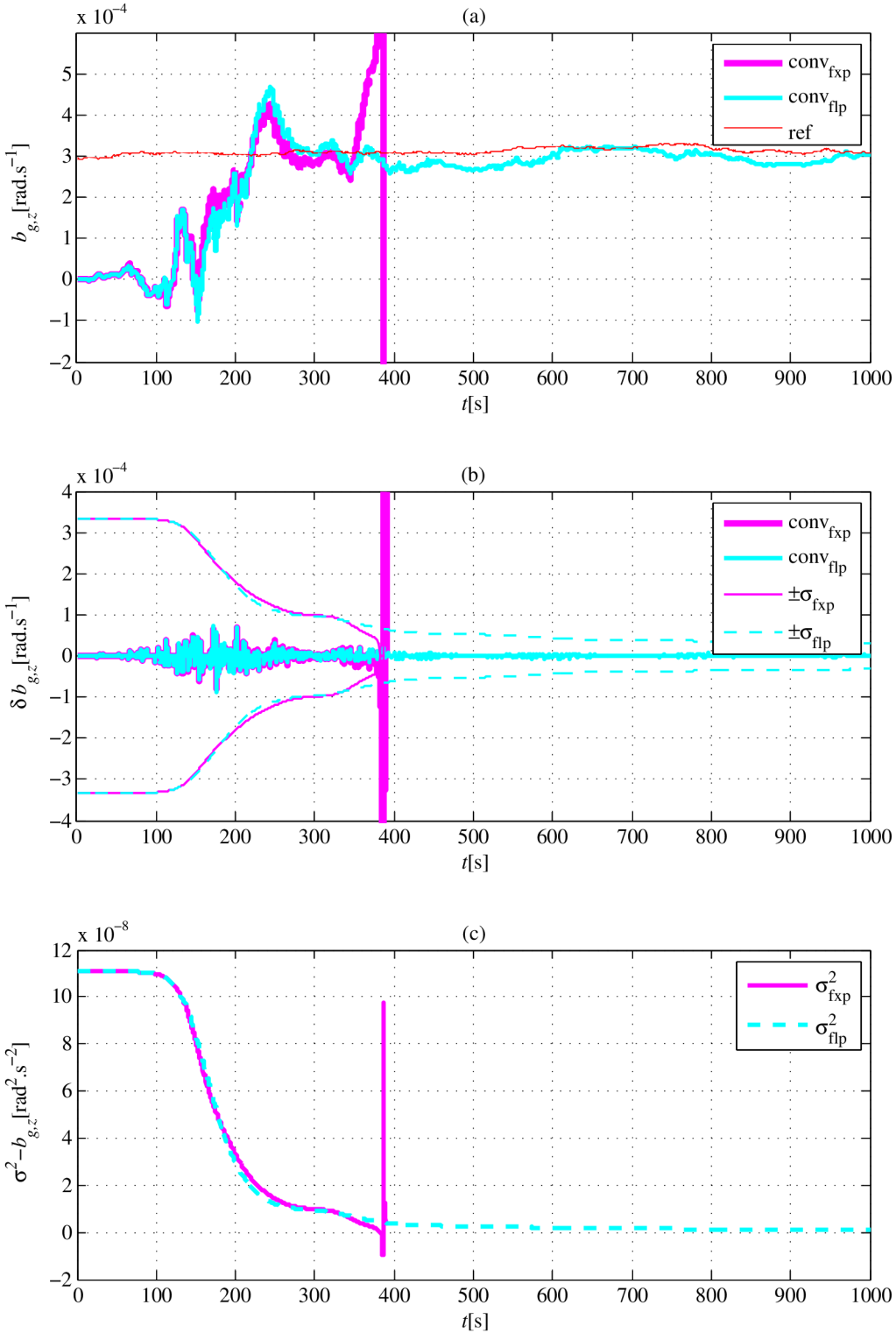


Fig. 5.24: Estimates of the gyroscopes' z-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the conventional Kalman filter compared with respect to its double precision floating-point version. The red line represents a reference trajectory.

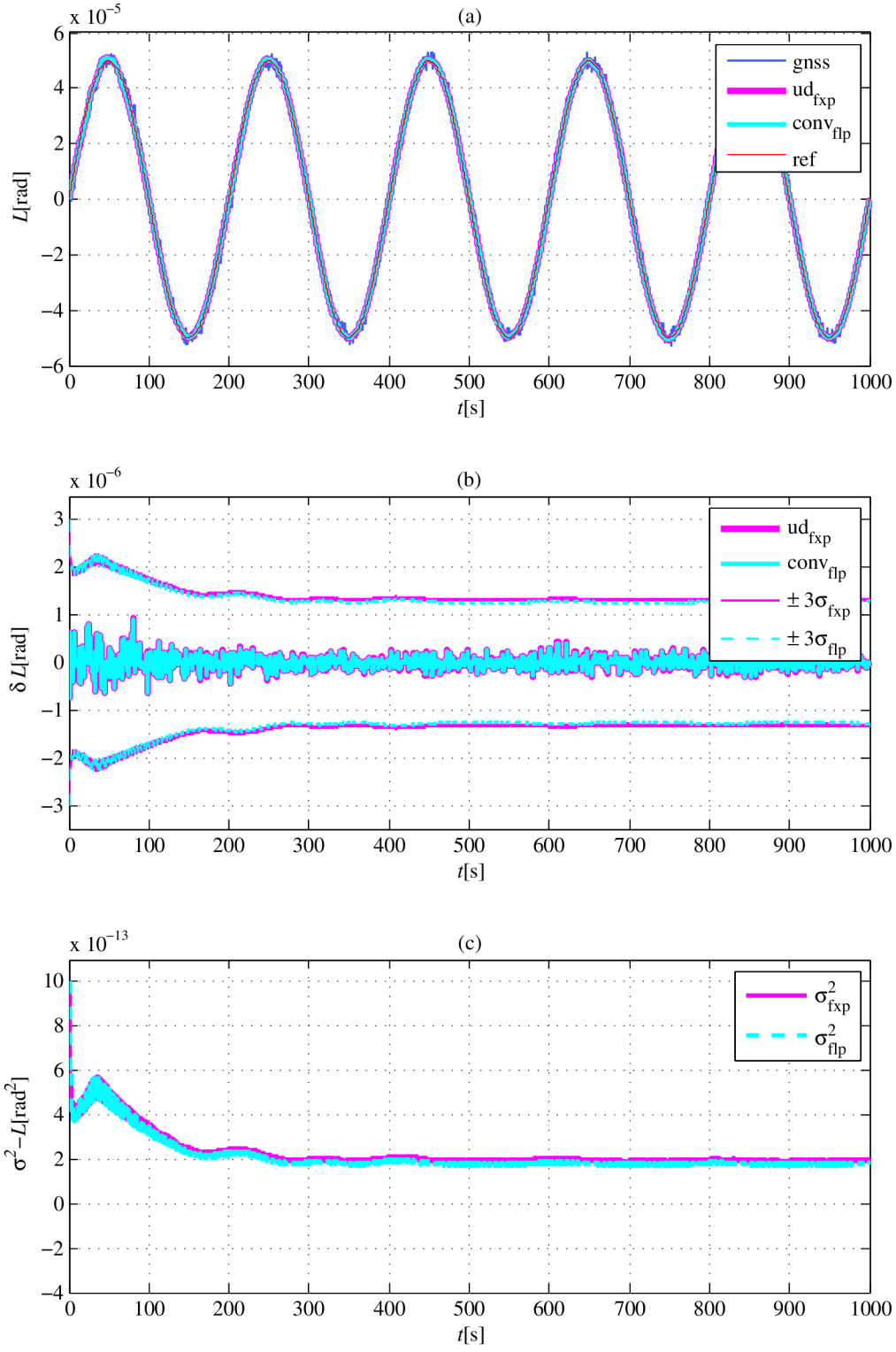


Fig. 5.25: Estimates of the latitude (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

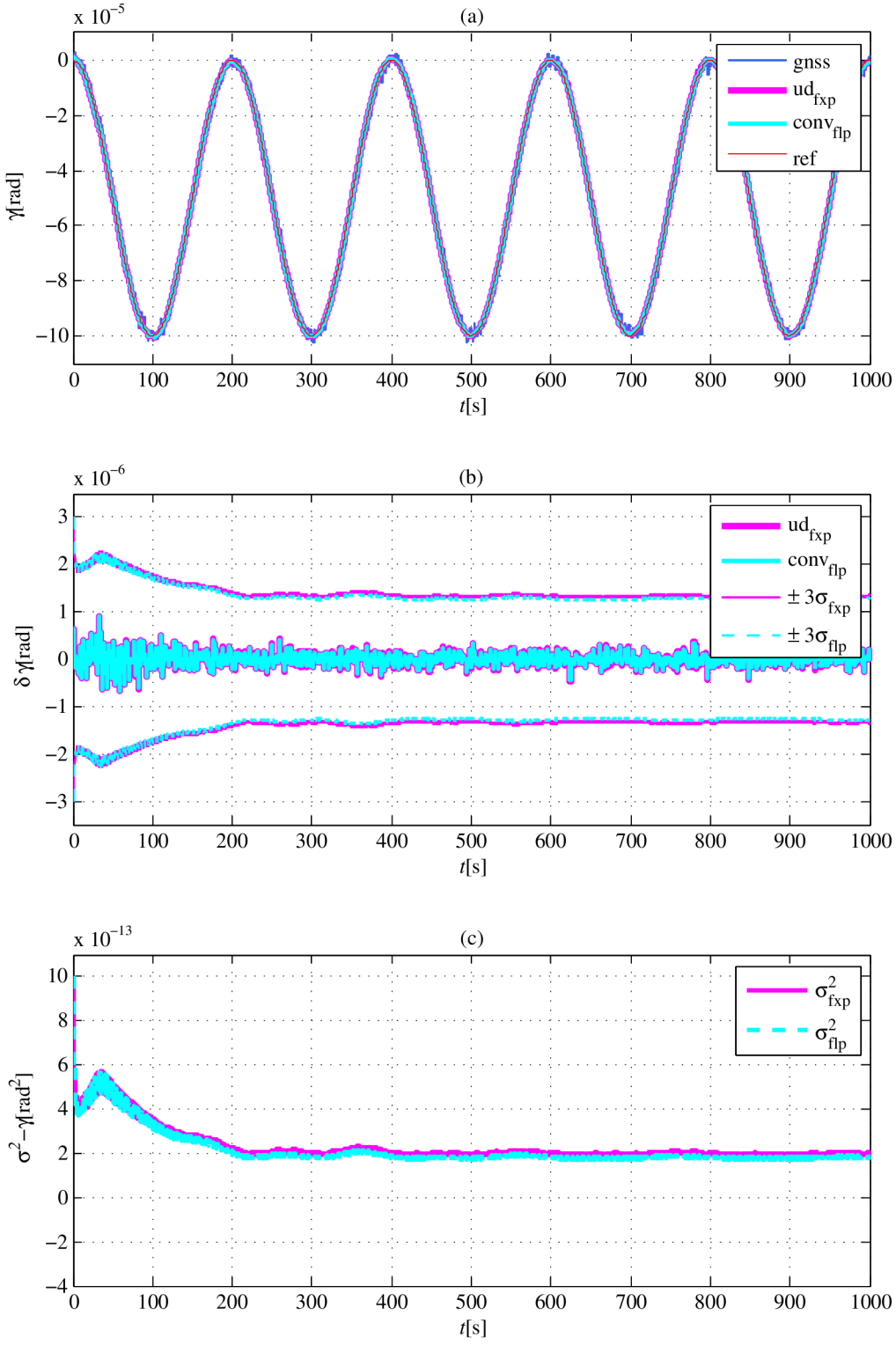


Fig. 5.26: Estimates of the longitude (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

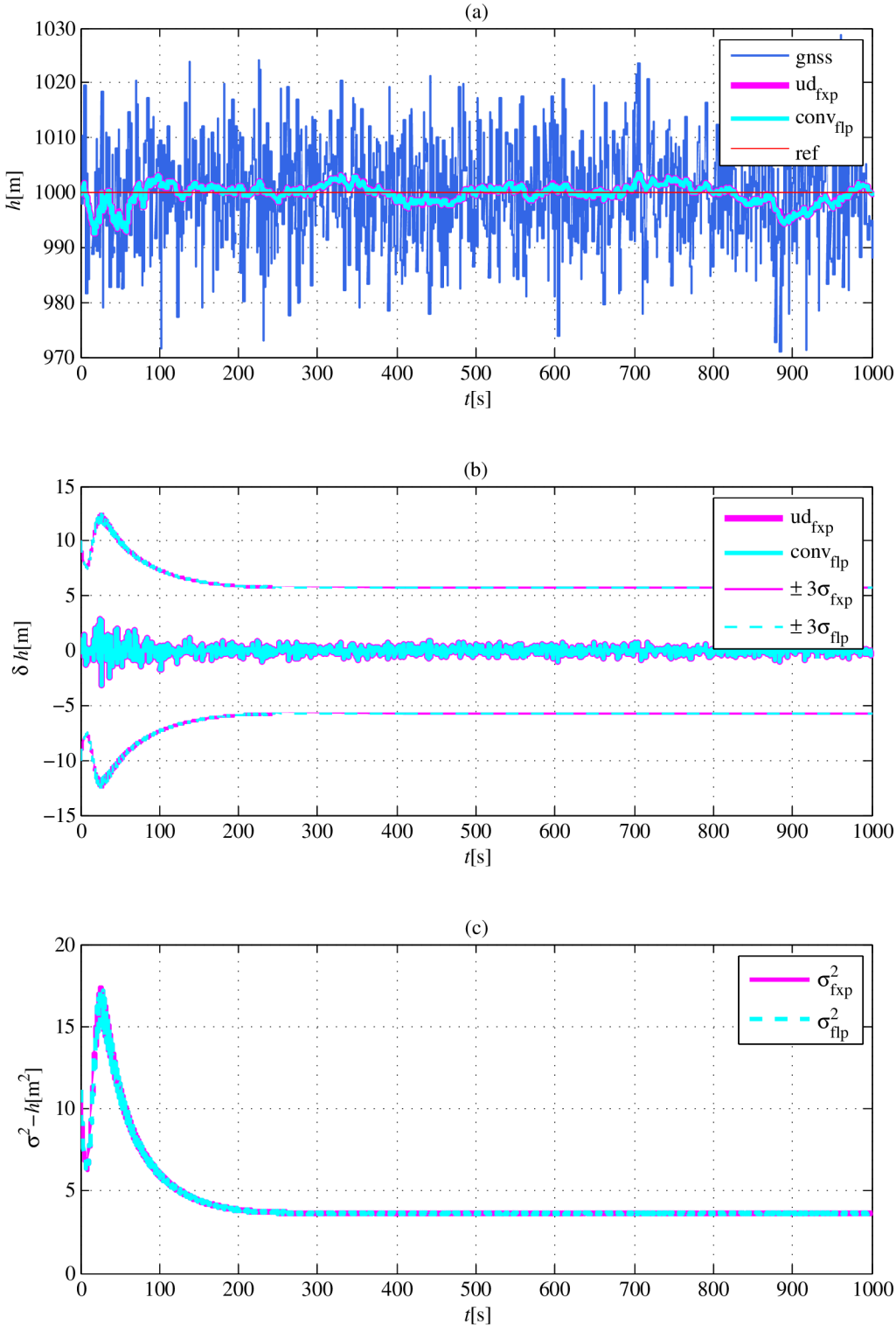


Fig. 5.27: Estimates of the height (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

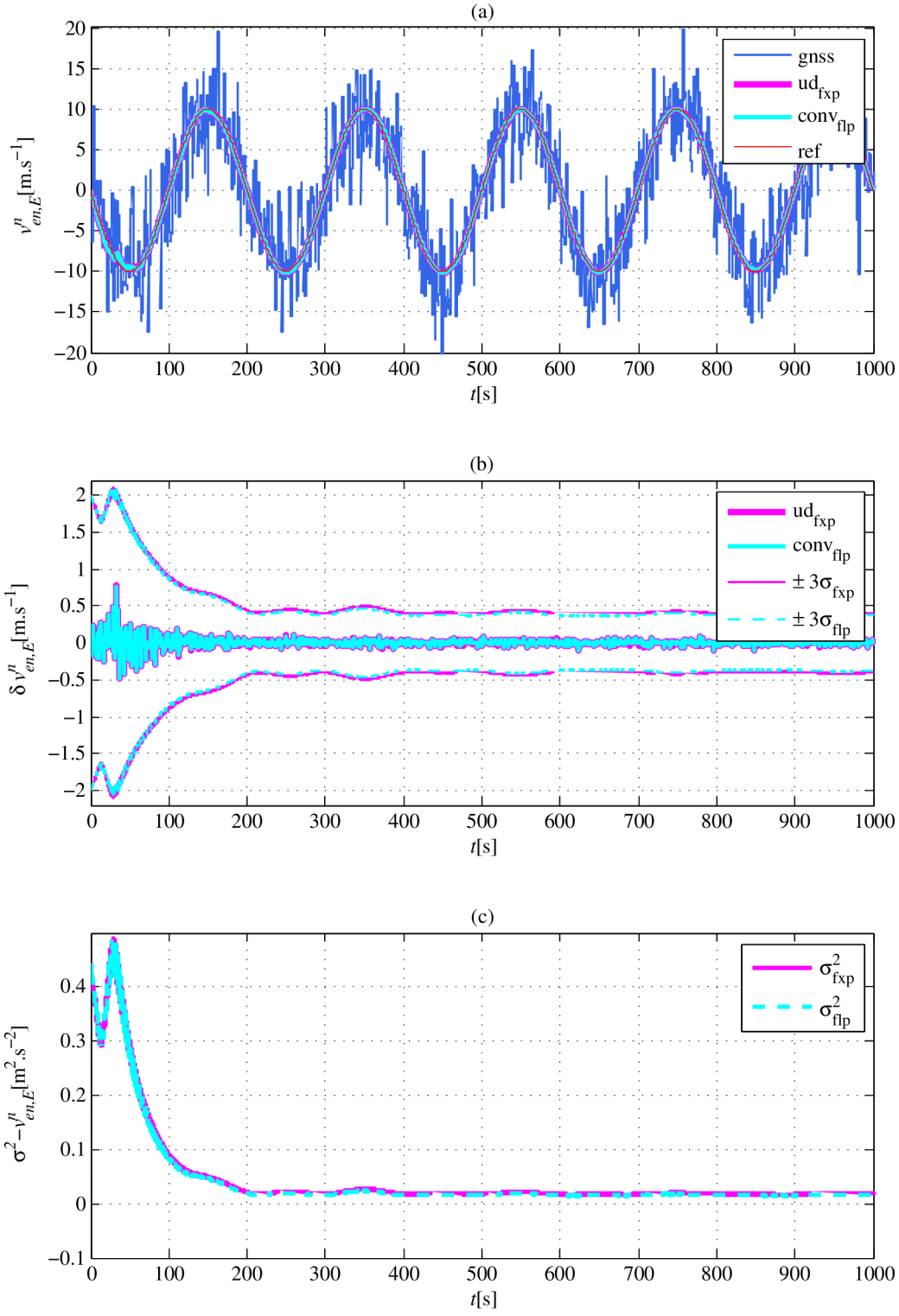


Fig. 5.28: Estimates of the east velocity (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

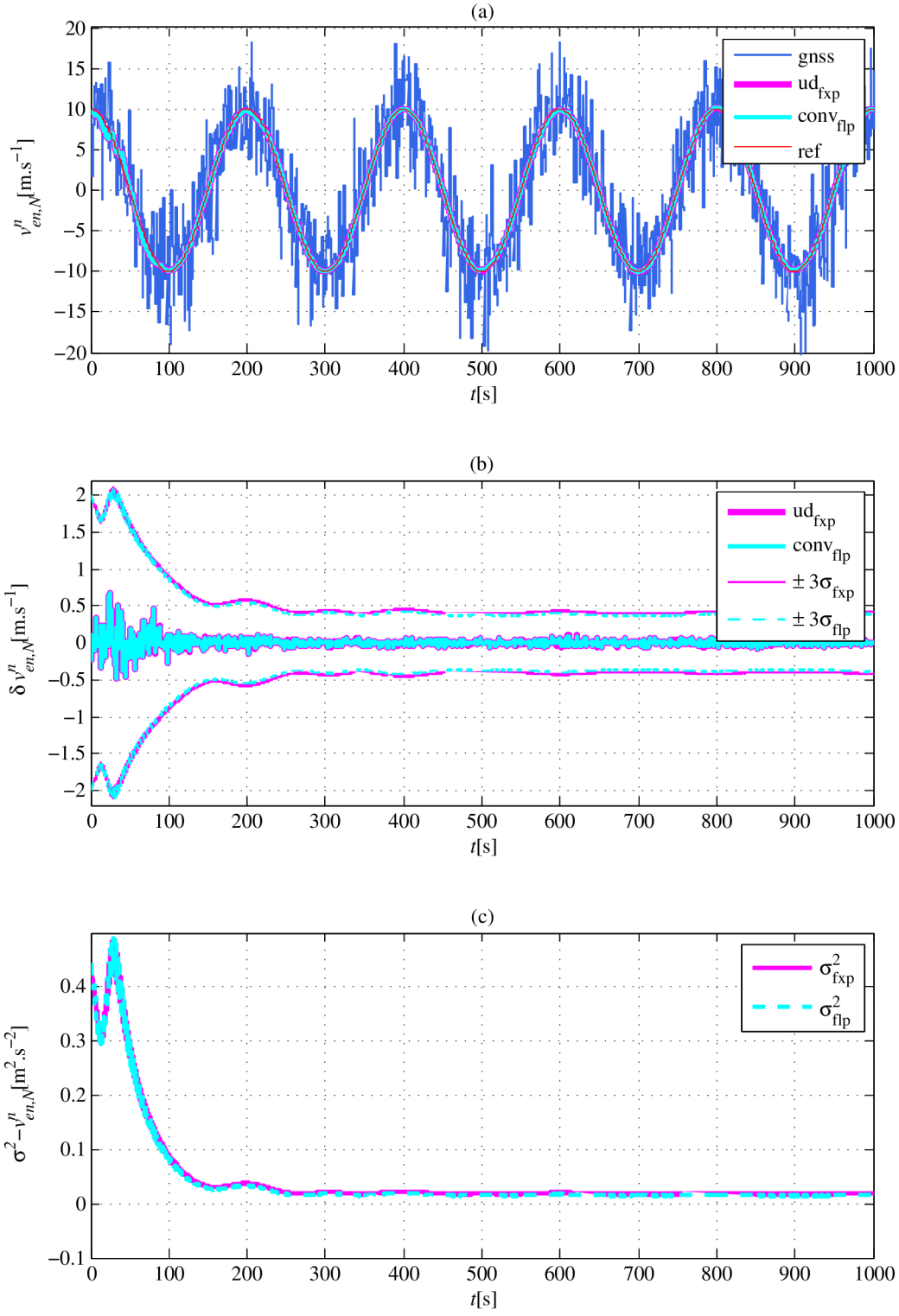


Fig. 5.29: Estimates of the north velocity (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

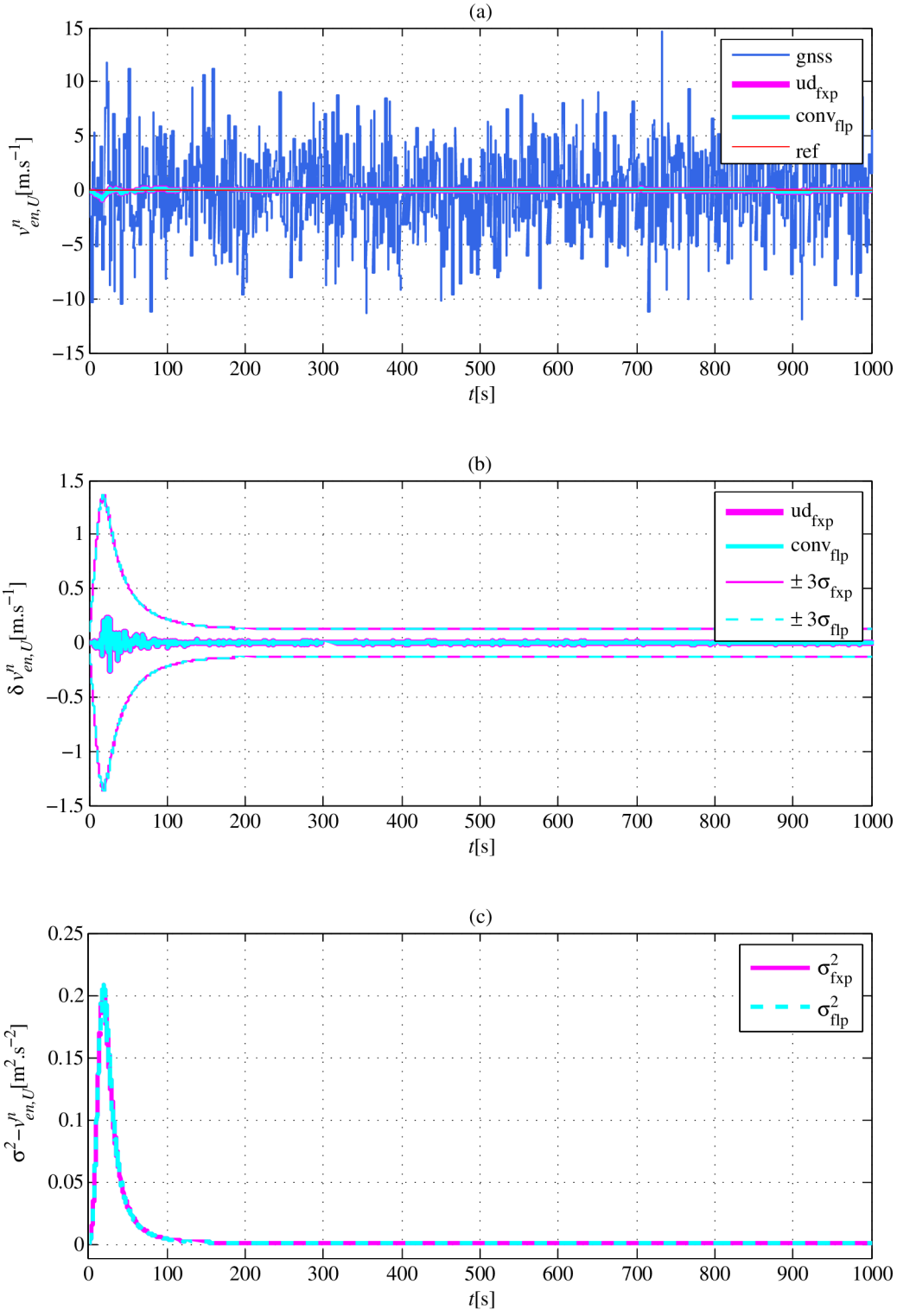


Fig. 5.30: Estimates of the up velocity (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory and blue line the GNSS measurements.

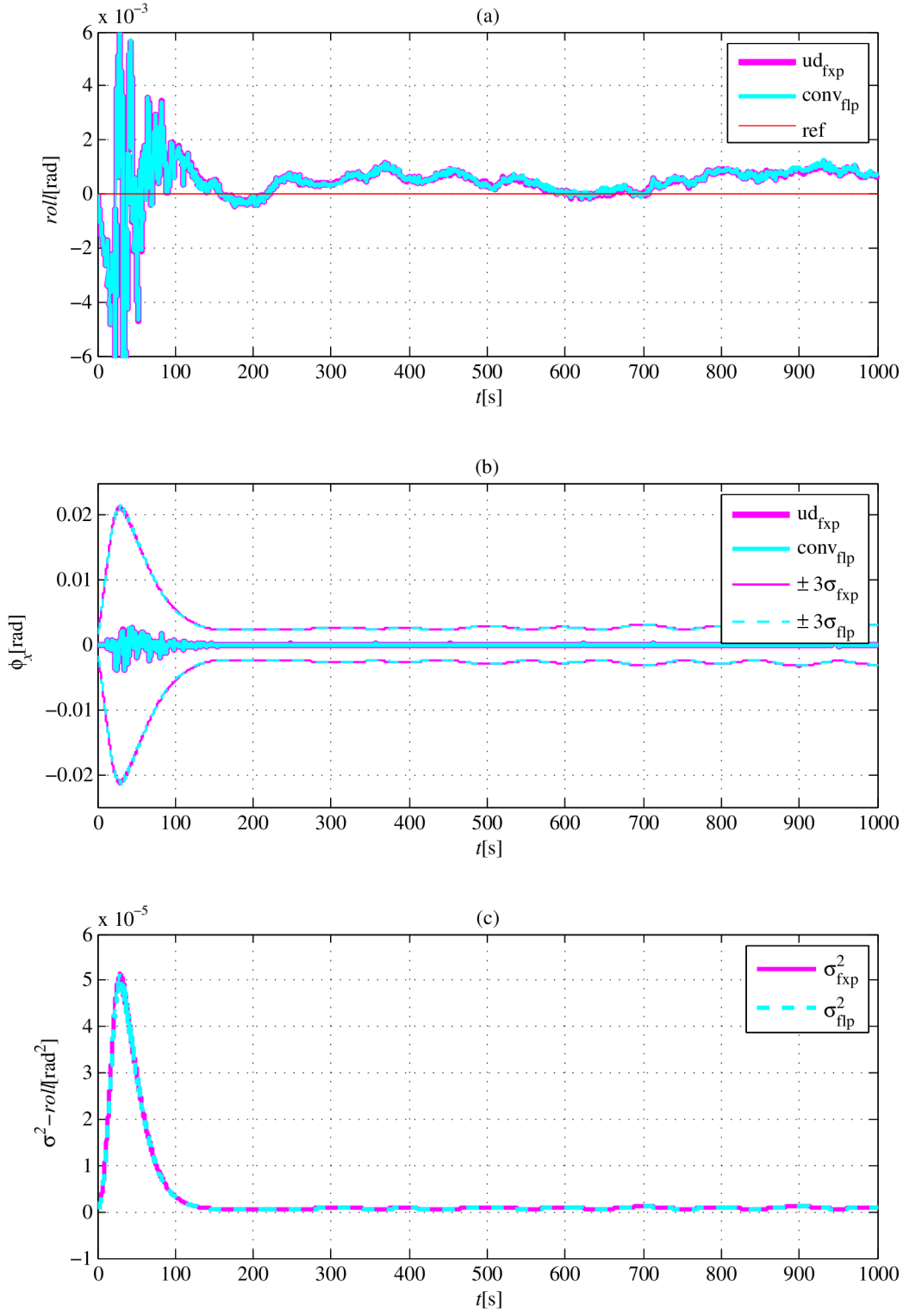


Fig. 5.31: Estimates of the roll (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

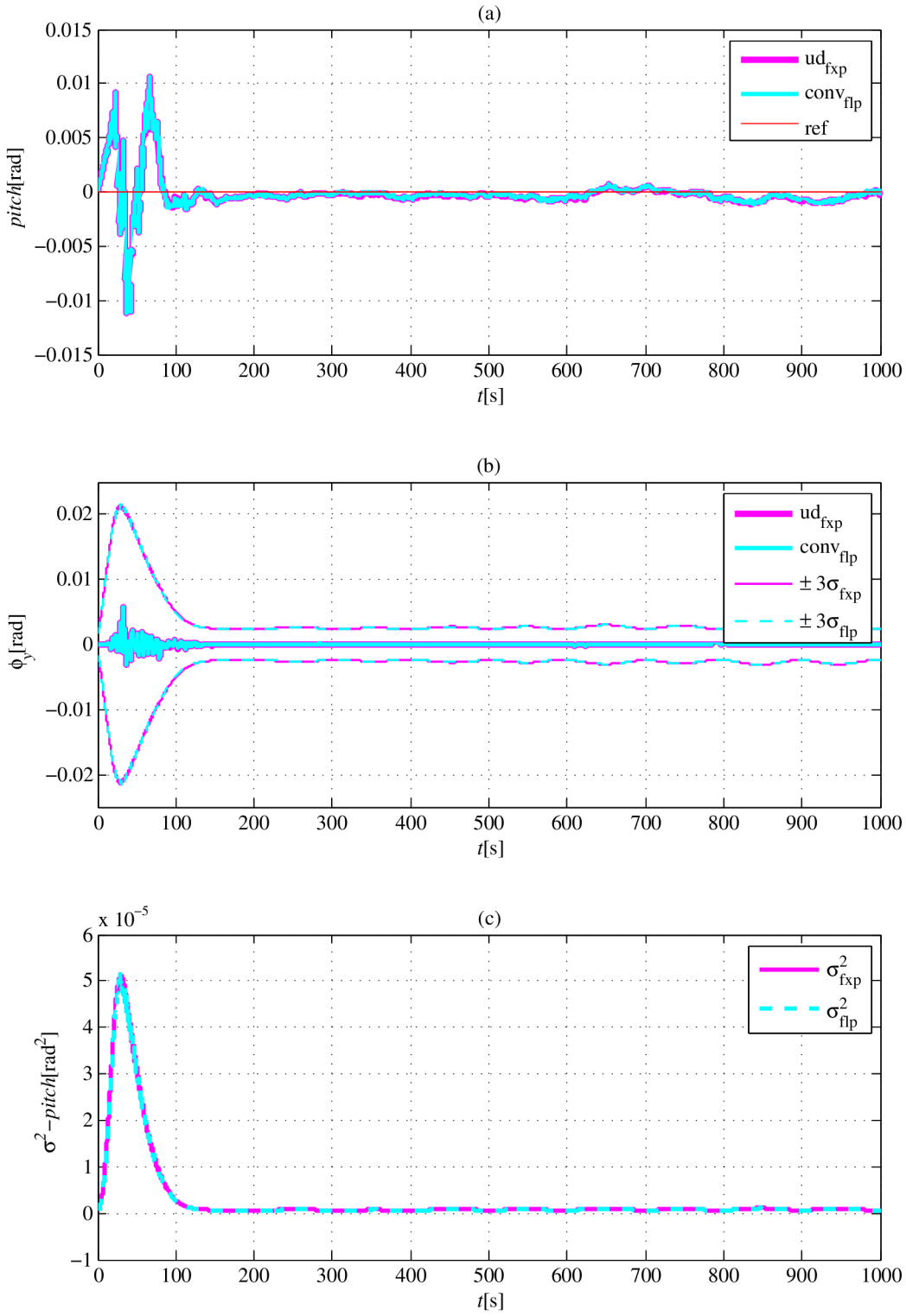


Fig. 5.32: Estimates of the pitch (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

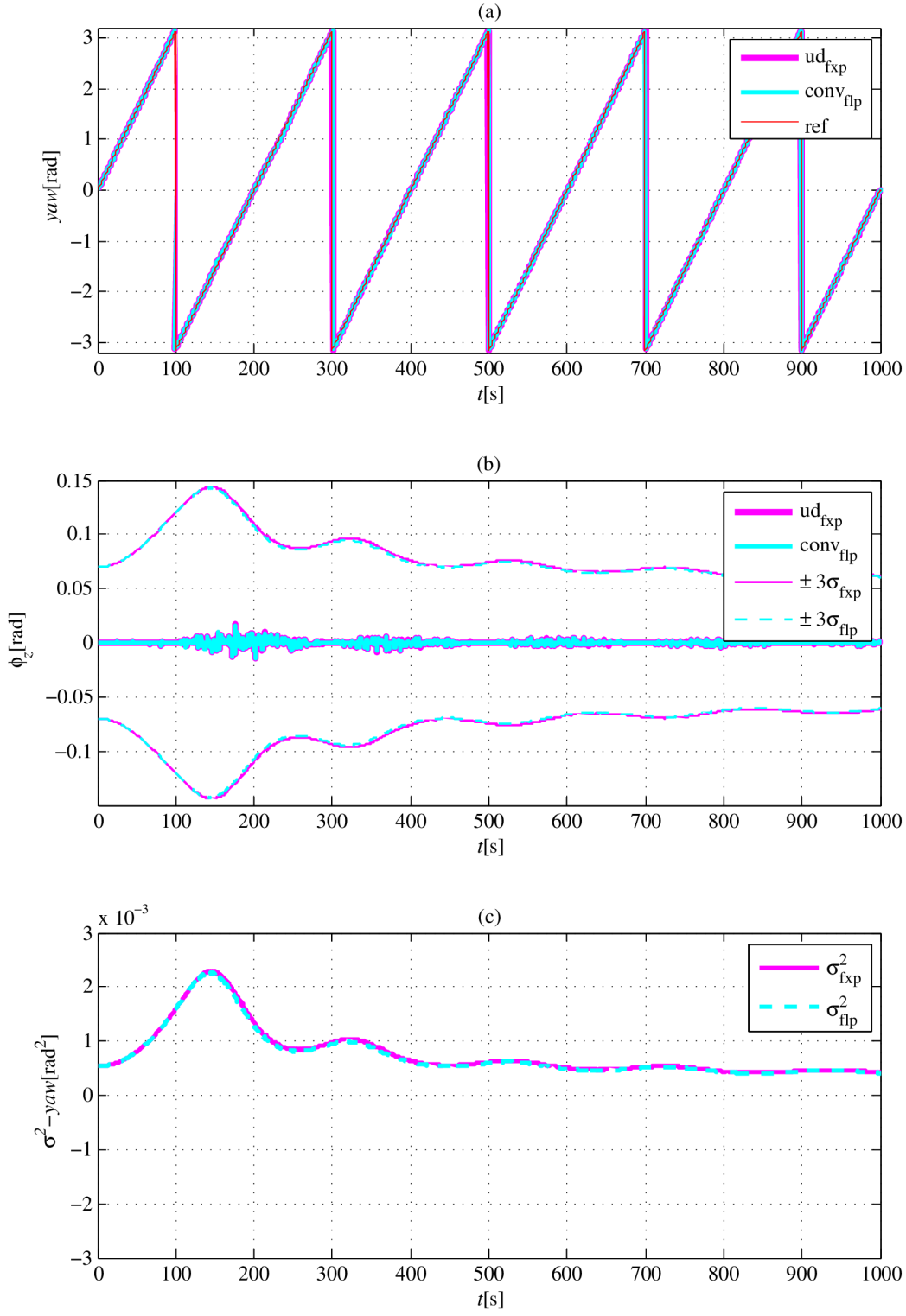


Fig. 5.33: Estimates of the yaw (a), corresponding error with 3-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

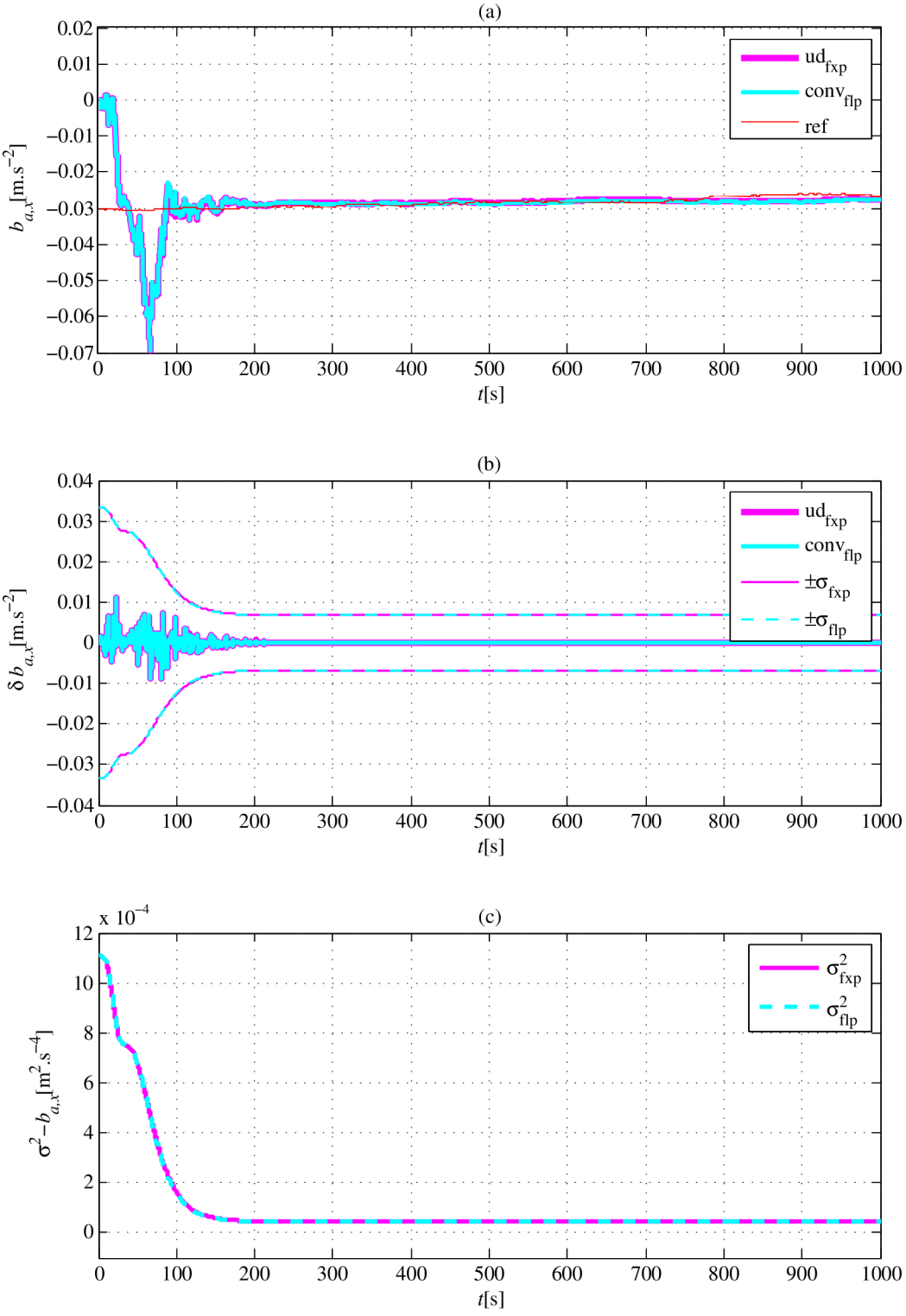


Fig. 5.34: Estimates of the accelerometers' x-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

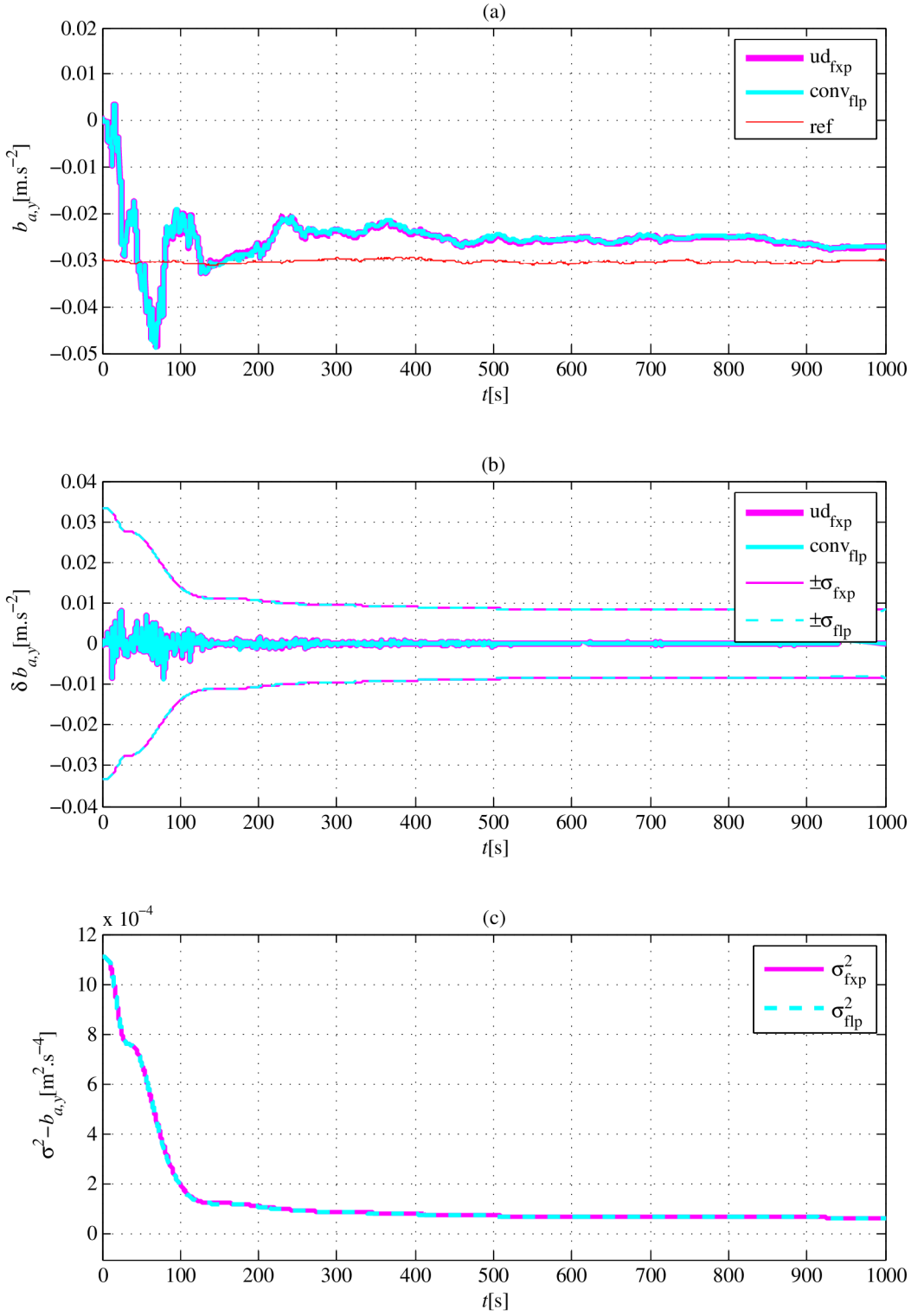


Fig. 5.35: Estimates of the accelerometers' y-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

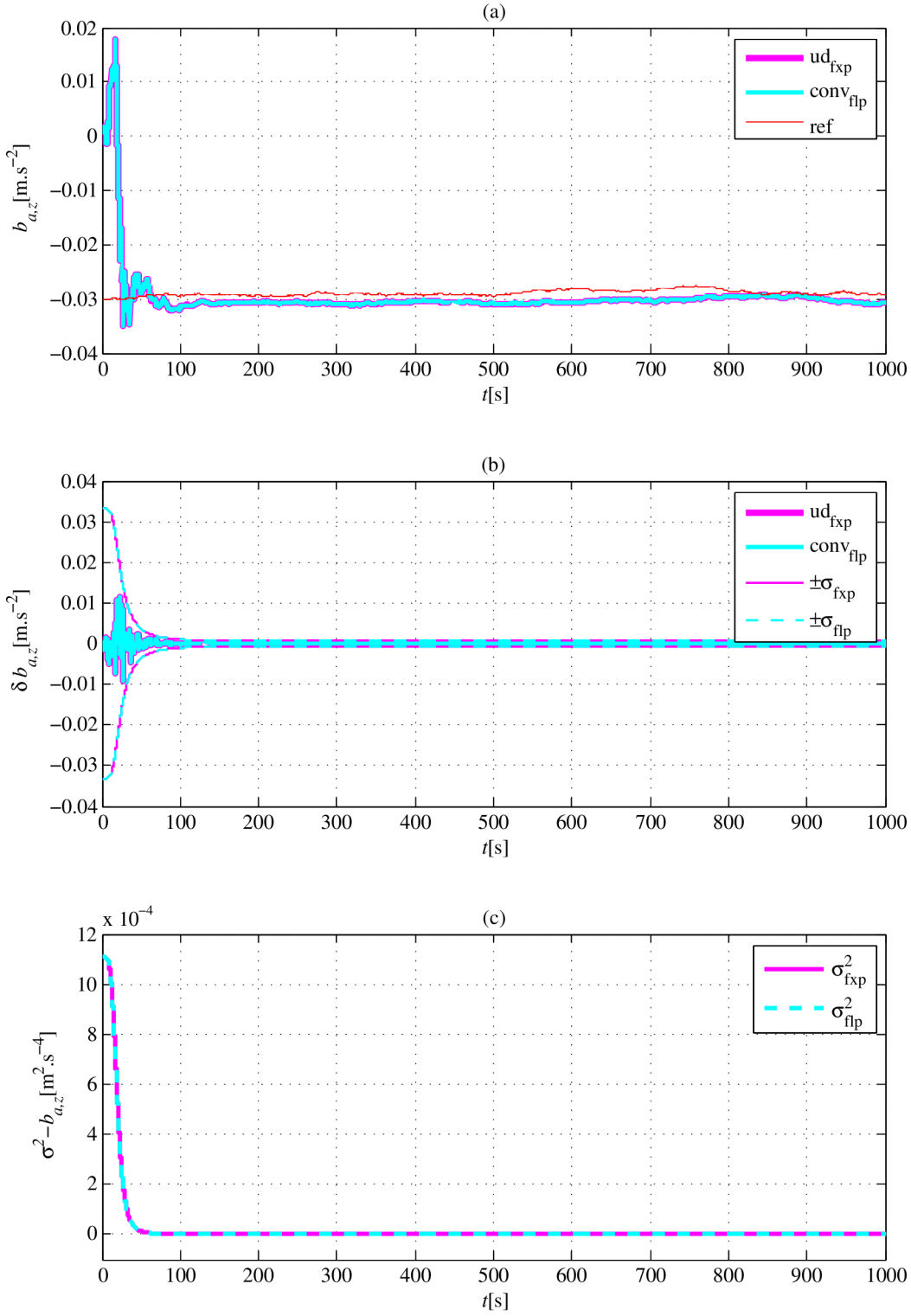


Fig. 5.36: Estimates of the accelerometers' z-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

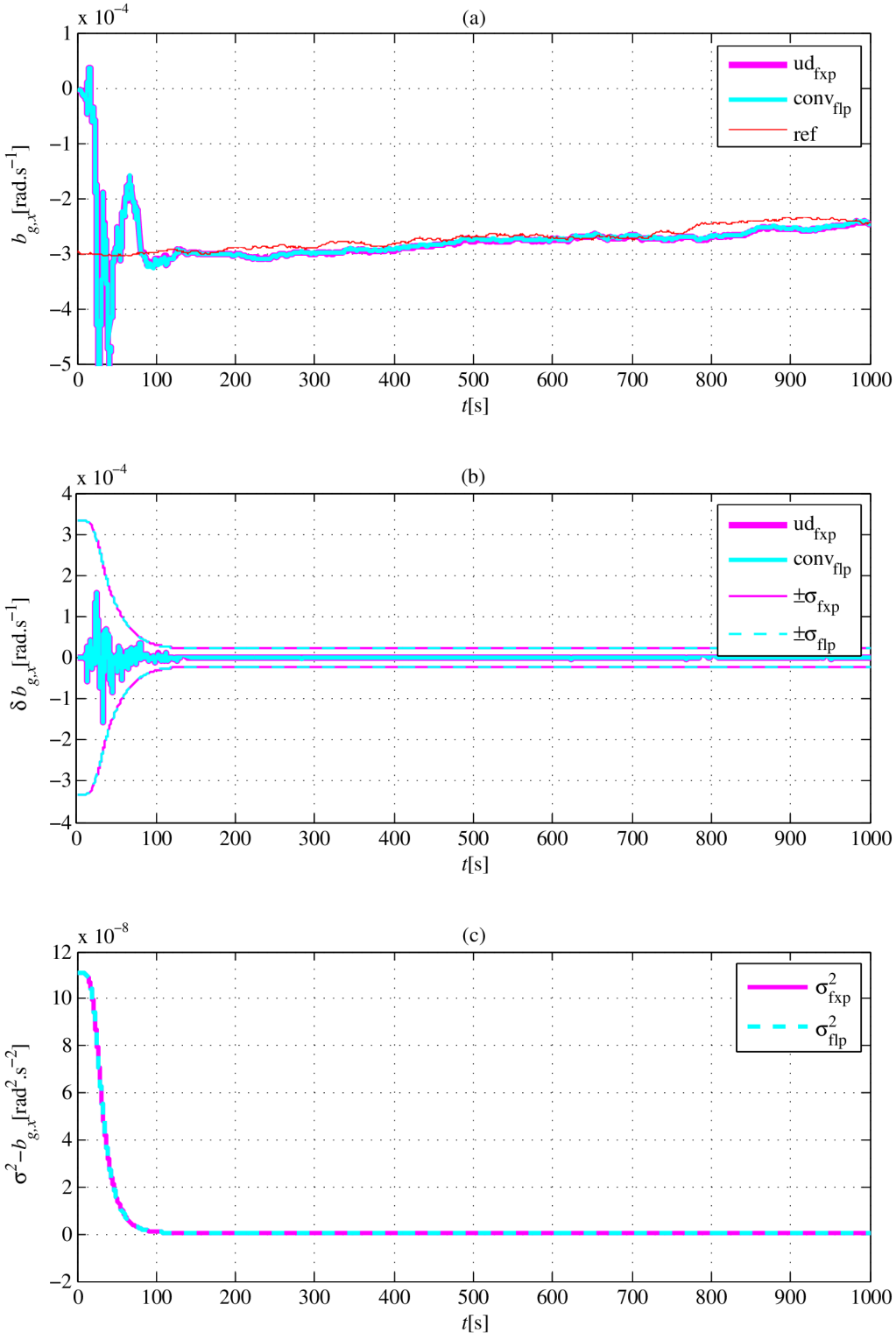


Fig. 5.37: Estimates of the gyroscopes' x-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

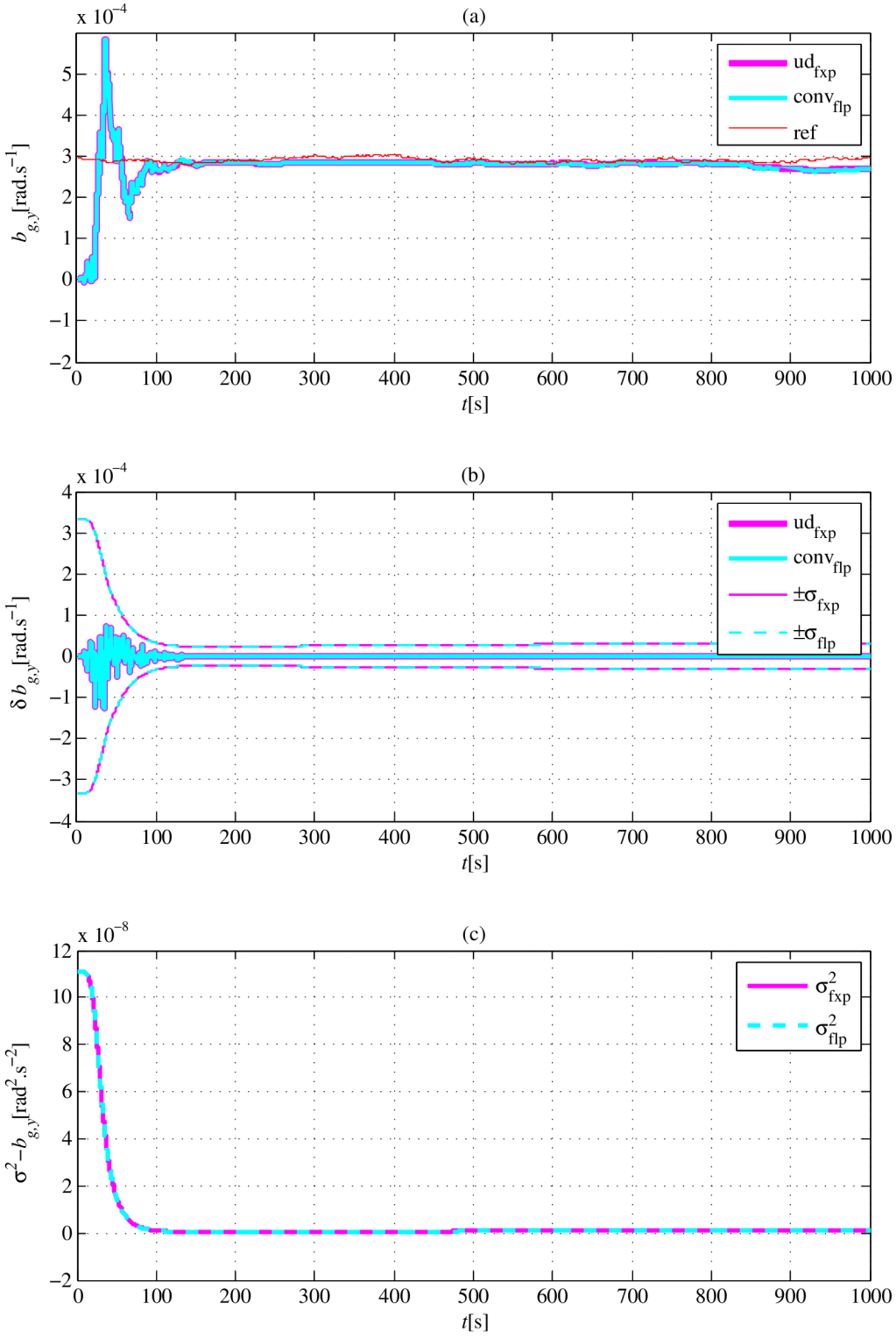


Fig. 5.38: Estimates of the gyroscopes' y-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

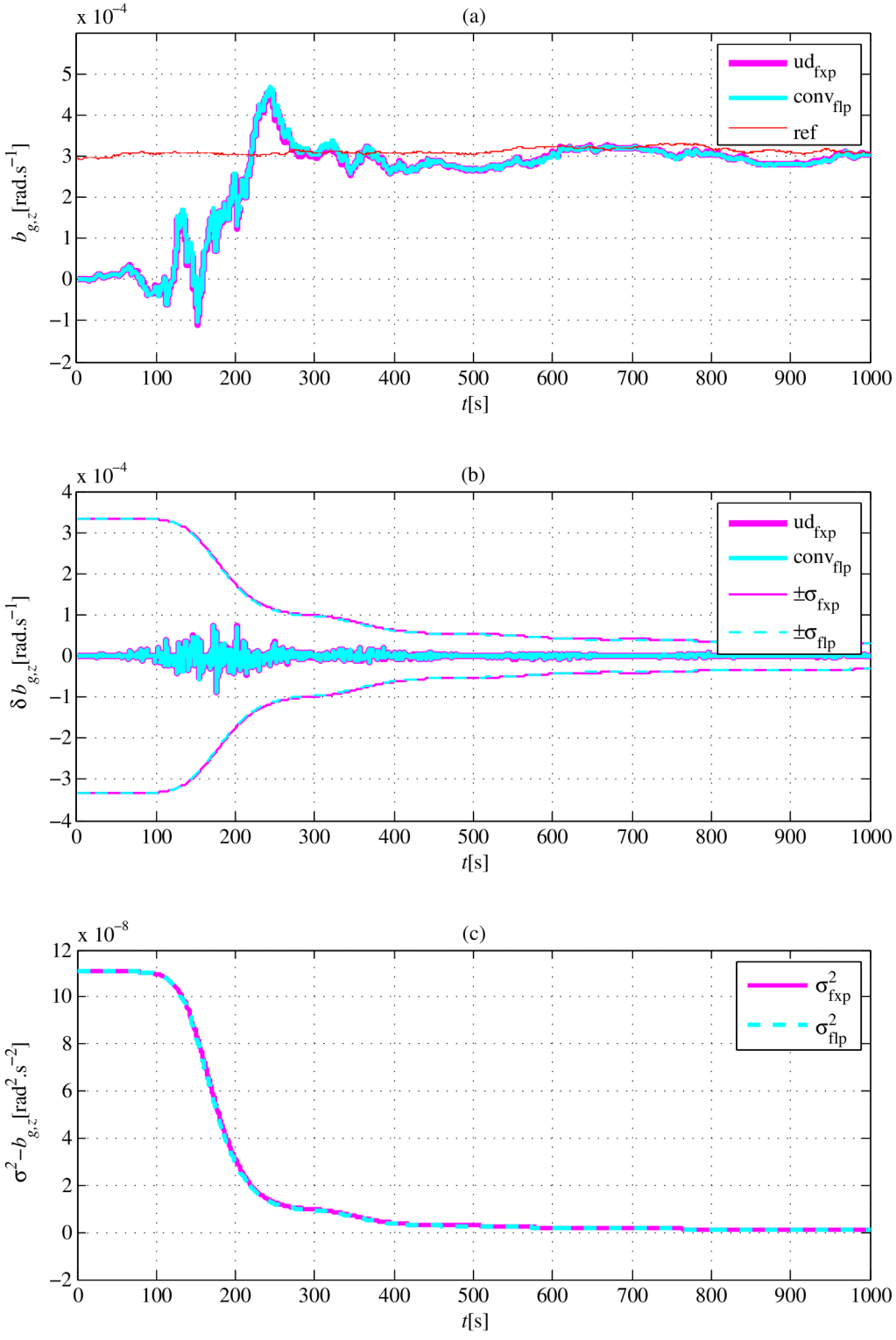


Fig. 5.39: Estimates of the gyroscopes' z-axis bias (a), corresponding error with 1-sigma bounds (b) and variance (c) computed in the fixed-point arithmetic, which has 24bits in the integer part and 48bits in the fractional part, for the UD factorized Kalman filter compared with respect to the conventional Kalman filter computed in double precision floating-point arithmetic. The red line represents a reference trajectory.

6 CONCLUSION

Main objective of this thesis was to compare different numerical implementations of the Kalman filter applied in an inertial navigation system which is expressed in the local navigation frame and which is implemented with using the Loosely Coupled integration approach and the Phi-angle error model. The implementations of the Kalman filter, investigated through this thesis, were the conventional Kalman filter, Kalman filter with the Joseph's stabilized form of the a posteriori covariance matrix, Square Root Kalman filter and UD factorized Kalman filter. All algorithms were derived due to a better understanding and certainty about their integration into a simulation. The performance of the all Kalman filter implementations was compared in a sense of an evaluation of the criteria function, the covariance matrix or its factors conditioning and in an ability to trace the reference trajectory.

The presented results show that using the Square Root and UD factorized Kalman filters leads to a better robustness of the inertial navigation system. A closer look at the evaluated criteria function shows that a reduction in the number of bits is not much significant. However, one need to consider the results of the covariance matrices or their factors conditioning. These indicates a possibility that the conventional and Joseph implementations of the covariance matrix may become ill-conditioned as the trends of the condition number significantly fluctuates. On the other hand the trends of the condition number of the Square Root and UD factors are smooth, which directly shows their better numerical performance. It can be stated that 24bits in the integer part and 54bits in the fractional part of the fixed-point arithmetic is sufficient for the all Kalman filter implementations. Further, decreasing the number of bits of the fractional part to 48bits cases a failure of the conventional and Joseph's implementations while the Square Root and UD implementations run well even for the 45bits of the fractional part. However, it is obvious from the results of the criteria function, that the both of these implementations, thus the Square Root and UD implementations, starts to be suboptimal from the 49bits of the fractional part. It can be excluded that this suboptimality is caused due to rounding in the measurement noise covariance matrix as it is represented by high numbers. On the other hand, the entries of the process covariance matrix are very small numbers, however the 45bit length of the fractional part is still sufficient for their interpretation. This is supported by the fact, that the Square Root filter do not uses directly the process noise covariance matrix, but its square root factor, which has two times better precision. Despite this fact, the suboptimal performance of the Square Root implementation is obvious. The only possible explanation, to the best author's knowledge, is that the suboptimality can be caused by the rounding in the system transition matrix, because if we take a closer look at the matrices W

(equations 3.57 and 3.87), then can be seen that the system transition matrix can be affected by rounding before its use in the Gram-Schmidt algorithm. Therefore, even if we use the Square Root or UD factors, then there is still a possible source of rounding errors, thus the rounding in the system transition matrix. However, this statement can not be generalized since it is a matter of the investigated system model. This statement holds only for such a model which is numerically badly conditioned, thus a model where is a significant difference in the magnitude of numbers, which is exactly our case as we tray to estimate the position error expressed in radians of the geodetic coordinates and the velocity errors expressed in the meters per second. The figures, which depict the estimated trajectory and error variables for the conventional Kalman filter implementation, show that the most affected part of the error model is the velocity part due to its most rapid divergence.

It was stated in the thesis objectives, if there is a possibility to use a length of the fixed-point arithmetic word, which is less or comparable to the length of the double precision floating-point arithmetic. The 69bit length of the fixed-point arithmetic word is the minimal possible value since the 45bit of the fractional part can interpret the entries of the process noise covariance matrix. However, this length is sufficient only for the Square Root and UD factorized implementations. The 69bit length is comparable to the 64bit length of the double precision, however this 69bits cause the previously mentioned suboptimality.

A possible extension of this thesis lies in an investigation of the same navigation algorithm with using better integration methods for the position and velocity updating. Another extension can be an investigation of the other numerical implementations of the Kalman filter e.g. the information Kalman filter, information Square Root filter etc. Further, the Psi-angle error model can be another interesting extension. It is an unexpected result that the implementation which uses the Joseph's stabilized version of the a posteriori covariance matrix do not brings any improvement with respect to the conventional Kalman filter. Their results are nearly identical, thus some diagonal entries of the Joseph's form of the a posteriori covariance matrix becomes negative as same as in the case of the conventional a posteriori covariance matrix. This phenomena deserves a deeper explanation, however this is out of the time portion for this thesis, therefore it will be a subject of further analysis.

BIBLIOGRAPHY

References

[1] KAMINSKI, P. G. *Square root filtering and smoothing for discrete processes* [online]. Stanford, California, 1971. [retrieved. 2013-01-04]. Available at: <`http://ntrs.nasa.gov/archive`>. Ph.D. thesis. Stanford University.

[2] BIERMAN, G. J., THORNTON, C. L. *Numerical comparison of discrete Kalman filter algorithms: orbit determination case study.* In: Proceedings of the 1976 IEEE Conference on Decision and Control [online]. 1976, Volume 15, p. 859-872 [retrieved. 2013-01-04]. Available at: <`http://www.ieee.org`>.

[3] THORNTON, C. L. *Triangular covariance factorizations for Kalman filtering.* In: NASA/JPL Technical Memorandum 33-798 [online]. California, 1976 [retrieved. 2013-01-04] Available at: <`http://ntrs.nasa.gov/archive`>. Ph.D. Thesis. University of California at LA, School of Engeneering.

[4] GHANBARPOUR ASL, H. *UD Covariance Factorization for Unscented Kalman Filter using Sequential Measurements Update.* In: World Academy of Science, Engineering and Technology [online]. 2007, Volume 34, p. 368-376 [retrieved. 2013-01-04]. Available at: <`http://www.citeseerx.ist.psu.edu`>.

[5] YATES, R. *Fixed-Point Arithmetic: An Introduction* [online]. 2009. [retrieved. 2012-05-20]. Available at: <`http://www.digitalsignallabs.com`>.

[6] IEEE 754. *IEEE Standard for Floating-Point Arithmetic.* New York, NY 10016-5997: USA: The Institute of Electrical and Electronics Engineers, 2008. Available at: <`http://www.ieee.org`>.

[7] NIKOLIC, Z., NGUYEN, H. T., FRANTZ G. *Design and Implementation of Numerical Linear Algebra Algorithms on Fixed Point DSPs.* EURASIP Journal on Advances in Signal Processing [online]. 2012, Volume 2007, p. 22 [retrieved. 2013-01-20]. Available at: <`http://www.asp.eurasipjournals.com`>.

[8] ŠTECHA, J., HAVLENA, V. *Teorie dynamických systémů.* Praha: Vydavatelství ČVUT, 2002. 247 pages. ISBN 80-01-01971-3.

[9] SIMON, D. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches.* New Jersey: Wiley-Interscience, 2006. 526 pages. ISBN 0 471-70858-5.

[10] THORNTON, C. L., BIERMAN, G. J. *Gram-Schmidt Algorithms for Covariance Propagation.* In: Proceedings of the 1975 IEEE Conference on Decision

and Control [online]. 1977, Volume 14, p. 489-498 [retrieved. 2012-05-14]. Available at: <http://www.ieee.org>.

[11] KALMAN, R. E. *A New Approach to Linear Filtering and Prediction Problems.* In: ASME: Journal of Basic Engineering [online]. 1960, p. 12 [retrieved. 2012-04-05]. Available at: <http://www.cs.unc.edu/~welch>.

[12] BIERMAN, G. J. *Measurement Updating Using the U-D Factorization.* In: Proceedings of the 1975 IEEE Conference on Decision and Control [online]. 1975, Volume 14, p. 10 [retrieved. 2012-05-14]. Available at: <http://www.ieee.org>.

[13] SCHMIDT, G. T., PHILLIPS, R. E. *INS/GPS Integration Architectures* [online]. 2011. [retrieved. 2012-05-20]. Available at: <http://ftp.rta.nato.int/public/>.

[14] GILL, P. E. *Methods for Modifying Matrix Factorizations.* In: Mathematics of Computation [online]. 1974, Volume 28, p. 31 [retrieved. 2012-05-14]. Available at: <http://www.stanford.edu/group/SOL/papers/ggms74.pdf>.

[15] GROVES, P. D. *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems.* Boston, London: Artech House, 2008. 540 pages. ISBN 978-1-58053-255-6.

[16] ROGERS, R. M. *Applied Mathematics In Integrated Navigation Systems.* Reston, Virginia: American Institute of Aeronautics and Astronautics, 2003. 326 pages. ISBN 1563476568.

[17] MAYBECK, P. S. *Stochastic Models, Estimation and Control: Volume1.* New York: Academic Press, 1979. 444 pages. ISBN 0124110428.

[18] ZHANG, W., GHOGHO, M., YUAN, B. *Mathematical Model and Matlab Simulation of Strapdown Inertial Navigation System.* In: Modelling and Simulation in Engineering, Volume 2012 [online]. 2012, p. 25 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[19] WENDEL, J., SCHLAILE, C., TROMMER, G.F. *Direct Kalman Filtering of GPS/INS for Aerospace Applications.* In: International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation (KIS2001) [online]. 2001, p. 144-149 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[20] BENSON, D. O. *A Comparison of Two Approaches to Pure-Inertial and Doppler-Inertial Error Analysis.* In: IEEE Transactions on Aerospace and Electronic Systems [online]. 1975, Volume AES-11, no. 4, p. 447-455 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[21] CRASSIDIS, J. L. *Sigma-Point Kalman Filtering for Integrated GPS and Inertial Navigation.* In: IEEE Transactions on Aerospace and Electronic Systems [online]. 2006, p. 24 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[22] SCHERZINGER, B.M., BLAKE, D.B. *Modified Strapdown Inertial Navigator Error Models.* In: Position Location and Navigation Symposium - PLANS 1994 [online]. 1994, p. 426-430 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[23] SCHERZINGER, B.M. *Inertial navigator error models for large heading uncertainty.* In: Position Location and Navigation Symposium - PLANS 1996 [online]. 1996, p. 447-484 [retrieved. 2013-01-20]. Available at: <http://www.ieee.org>.

[24] NASSAR, S. *Improving the Inertial Navigation System (INS) Error Model for INS and INS/DGPS Applications* [online]. Calgary, 2003. [retrieved. 2013-01-21]. Available at: <http://www.ucalgary.ca>. PhD thesis. University of Calgary.

[25] WEI, G., QI, N., GUOFU, Z., HUI, J. *Gyroscope Drift Estimation in Tightly-coupled INS/GPS Navigation System.* In: Industrial Electronics and Applications, 2007. ICIEA 2007 [online]. 2007, p. 391-396 [retrieved. 2012-05-14]. Available at: <http://www.ieee.org>.

List of Author's Publications

Published

[26] PAPEŽ, M. *Numerical Aspects of Inertial Navigation.* In: STUDENT EEICT Proceedings of the 19th Conference [online]. 2013, Volume 2, p. 96-98 [retrieved. 2013-05-15]. Available at: <http://www.feec.vutbr.cz/EEICT/>.

Submitted

[27] PAPEŽ, M. *Numerical Aspects of Inertial Navigation.*(extended paper) In: Proceedings of 12th IFAC/IEEE International Conference on Programmable Devices and Embedded Systems PDeS 2013. 2013, 6 pages. [retrieved. 2013-05-15].

LIST OF SYMBOLS, PHYSICAL CONSTANTS AND ABBREVIATIONS

INS	Inertial Navigation System
DSP	Digital Signal Processor
FPGA	Field-Programmable Gate Arrays
GNSS	Global Navigation Satellite System
IEEE	Institute of Electrical and Electronics Engineers
MAC	Multiply and ACcumulate
DCM	Direction Cosine Matrix
ECI	Earth Centered Inertial frame
ECEF	Earth Centered Earth Fixed frame
ENU	East-North-Up navigation frame
NED	North-East-Down navigation frame
e	Earth frame, Earth's eccentricity
i	Inertial frame
b	Body frame
n	Navigation frame, State space dimension
$x^{\alpha}_{\beta\gamma}$	γ-frame to β-frame related vector, expressed in α-frame
$x^{\alpha}_{\beta\gamma,\delta}$	δ entry of the γ-frame to β-frame related vector, expressed in α-frame
$\omega^{\alpha}_{\beta\gamma}$	Turn rate of the γ-frame with respect to the β frame, expressed in α-frame
$\Omega^{\alpha}_{\beta\gamma}$	Skew-symmetric matrix of the $\omega^{\alpha}_{\beta\gamma}$ vector
$(\omega^{\alpha}_{\beta\gamma}\times)$	Skew-symmetric matrix of the $\omega^{\alpha}_{\beta\gamma}$ vector
C^{β}_{α}	α-frame to β-frame DCM
ω_{ie}	Earth's angular rate
L	Latitude
λ	Longitude
h	Height
R_e	Earth radius
R_N	Meridian radius
R_E	Transverse radius
γ	Roll angle
θ	Pitch angle
ψ	Yaw angle
g_0	The gravity magnitude at zero altitude

LIST OF APPENDICES

A APPENDIX

A.1 Contents of the Attached CD

The attached CD contains source codes and simulation files of the all algorithms presented in this thesis and an electronic version of this thesis. The algorithms are implemented in the Matlab/Simulink environment with the Fixed-Point Toolbox.