Low-cost MEMS-INS/GPS Integration using Nonlinear Filtering Approaches

von der Naturwissenschaftlich-Technischen Fakultät der
Universität Siegen
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
(Dr.-Ing.)

von

M.Sc. Junchuan Zhou

1. Gutachter: Professor Dr. -Ing. habil. Otmar Loffeld
2. Gutachter: Privatdozent Dr.-Ing. habil. Stefan Knedlik

Datum der mündlichen Prüfung: 18.04.2013

Acknowledgments

Special thanks go to my first supervisor, Prof. Dr.-Ing. habil. Otmar Loffeld, for his lecture of stochastic modeling and estimation theory and the constructive feedback throughout my postgraduate research. His encouragement has been a great source of my confidence and motivation.

I also appreciate the support and guidance from my second supervisor, Priv.-Doz. Dr.-Ing. habil. Stefan Knedlik, for his technical advices and letting me be involved in the project “Attitude and Position Determination (AtPos) for Bistatic SAR Experiments", which was funded by the German Research Foundation (DFG).

I should thank all the members in navigation group, who are Mr. Zhen Dai, Ms. Miao Zhang, Mr. Ezzaldeen Edwan, Ms. Jieying Zhang for the frequent discussions on scientific and non-scientific matters.

I extend my gratitude to Dr.-Ing. Holger Nies, Dr.-Ing. Klaus Hartmann and Mrs. Silvia Niet-Wunram for their administrative suggestions and helps. I appreciate Mr. Arne Stadermann for his support on computer related matters, and Ms. Amaya Medrano-Ortiz for organizing many interesting social events. Many thanks go to all the colleagues in the Center for Sensor Systems (ZESS) for providing such a wonderful environment for research.

Finally, I would like to give my thanks to the technical committee of the Institute of Navigation (ION) for awarding me the student paper award and financially (i.e., 2000 USD) supporting me to orally present my work in the 24th international technical meeting of the satellite division of the Institute of Navigation (ION GNSS 2011) in Portland, Oregon, USA, 2011.

Contents

List of Figures

List of Tables

Acronyms

BPF	Bootstrap particle filter
DCM	Direction cosine matrix
DLL	Delay lock loop
DSP	Digital signal processor
ECEF	Earth-centered earth-fixed
ECI	Earth center inertial
EKF	Extended Kalman filter
FLL	Frequency lock loop
GNSS	Global navigation satellite system
GPS	Global positioning system
IMU	Inertial measurement unit
INS	Inertial navigation system
KF	Kalman filter
FPGA	Field-programmable gate array
MCS	Master control segment
MEMS	Micro-electromechanical systems
MSE	Mean-square-error
NED	North earth down
PF	Particle filter
PLL	Phase lock loop
RMS	Root mean square
SISR PF	Sequential importance sampling re-sampling particle filter
UAV	Unmanned aerial vehicle
UKF	Unscented Kalman filter
UPF	Unscented particle filter

Abstract

Some important key issues in GNSS/INS integration mainly arise in the field of creating and developing low-cost, robust and at the same time highly accurate navigation systems, putting a focus of interest onto powerful sensor fusion algorithms. The so-called tightly-coupled integration is one of the most promising approaches to fuse the GNSS (global navigation satellite systems) data with INS (inertial navigation system) measurements. However, when modeling the underlying problem, the system process and observation models turn out to be nonlinear, and the GNSS stochastic measurement errors are often non-Gaussian distributed (e.g., due to multipath effects). Among other estimation approaches, the so-called particle filter (PF) as a nonlinear/non-Gaussian estimation method is especially theoretically attractive to be used in this field. However, its large computational burden usually limits its practical usage. In order to reduce the computational burden without degrading the system estimation accuracy, recently, an unscented particle filter (UPF) has been proposed, which combines the PF with the unscented Kalman filter (UKF). In this thesis, only one UKF is used in the algorithm, and the re-sampling step is not required anymore. Thus, the number of particles can be largely reduced, and the implementation of the PF on a hardware platform turns out to be feasible.

Kurzfassung

Aktuelle Entwicklungen auf dem Gebiet der Fusion von inertialer Navigation und satellitengestützten Positionierungsverfahren zielen klar auf kosteneffiziente, robuste und gleichzeitig hochpräzise Lösungen ab. Leistungsfähige Sensordatenfusionsansätze spielen hier eine Schlüsselrolle, wobei die sogenannte „Tightly Coupled Integration“ zur Fusion der satellitengestützten Navigationsdaten mit den Messdaten eines inertialen Systems besonders vielversprechend erscheint. Als erschwerender Umstand ergeben sich hier allerdings nichtlineare Prozess- und Beobachtungsmodelle, die in Verbindung mit nicht länger gaußverteilten Beobachtungsfehlern, beispielsweise aufgrund von Mehrwegeausbreitung, nichtlineare, möglichst optimale Datenfusionsverfahren, wie beispielsweise Partikelfilter-Ansätze erfordern. Theoretisch elegant und leistungsfähig auf der einen Seite, benötigen diese Ansätze in der praktischen Realisierung vielfach eine ungemein hohe Anzahl von einzelnen „Partikeln“, so dass der hierdurch verursachte Berechnungsaufwand die praktische Einsatzfähigkeit unter Echtzeitbedingungen vielfach entweder im Hinblick auf die Filterperformance oder auf die Taktzeit limitiert. Ein Ansatz zur Lösung dieser Problematik besteht in der Kombination eines Partikelfilters mit einem Unscented Kalman Filter. Hierbei wird der sonst bei Partikelfiltern übliche, aber zeitaufwändige, Resampling Schritt nicht mehr benötigt. Auch die Anzahl der benötigten Partikel kann stark reduziert werden, so dass eine Realisierung auf einer Signalprozessorplattform möglich wird.

Outline

In this thesis, the content is organized as follows.

In chapter one, the single point L1 GPS receiver data processing is introduced, which is used throughout of this thesis. Inertial navigation system (INS) principle is overviewed and INS strapdown mechanization equations are formulated. The INS/GPS loosely-coupled and tightly-coupled integration approaches are given in detail. A field experiment is made to show the advantages of INS/GPS system with respect to that from GPS alone device.

In chapter two, the knowledge on estimation theory in solving nonlinear filtering problems is overviewed. The recursive Bayesian filter is introduced, and its difficulties in handling practical tracking problems are pointed out. The UKF and PF algorithms are given. As an important contribution of this thesis, the UPF algorithm is used, which combines the best from the UKF and PF, yielding a robust and highly accurate solution in handling nonlinear and non-Gaussian problems. A simulation is conducted to show its merits with respect to the EKF, UKF and conventional PF.

In the third chapter, nonlinear filtering approaches (i.e., EKF, UKF and UPF) are applied on INS/GPS tightly-coupled integration using quaternions as the representation of attitude. Three field experiments are made and numerical results are compared and analyzed.

In chapter four, the focus is moved from land-based navigation to UAV-based high dynamic navigation applications. For accurately estimating and tracking the dynamics of a flying vehicle, methods on correctly handling the carrier phase derived delta range measurements (i.e., a type of integrated measurement) are introduced. Sequential processing is successfully applied for reducing the computational burden.

Last but not least, in the appendix A, some basics on quaternions are overviewed, especially focusing on the relationship between quaternions and rotation vector. In appendix B, the equations for transforming the quaternion covariance to Euler angle covariance are given. In appendix C, the numerical operations involved in the matrix inversion is given. In appendix D, the reduced numerical operations using sequential processing are counted, where the Joseph covariance update formula is used.

Motivation

INS/GPS integration

For decades, GPS (only) receivers have dominated the field of positioning. The emergence of micro-electromechanical systems (MEMS) technology has brought low-cost INS/GPS integration approaches into reality. Due to the complementary nature of INS and GPS principles, such an integrated navigation system combines the best of two worlds, working in all environments, and constituting, for example, a potential and powerful alternative to the GPS alone navigation devices.

The conventional INS/GPS integration system

In a conventional INS/GPS integration system, the INS and GPS data are usually integrated in a loosely-coupled manner, where the position and velocity are exploited in the integration KF. In this way, off-the-shelf navigation devices can be used, and independent redundant navigation solutions are available from the GPS receiver and INS. However, the flaws are that typically four satellites have to be in view to obtain position and velocity update from GPS receiver. Besides, if one KF is used in GPS data processing, another is used for integration purpose, the mutual feedback of estimation errors from both KFs will cause cascaded filtering problem [1, 2]. In this thesis, a tightly-coupled integration approach is used, in which a centralized KF is employed. In this method, all systematic errors and noise sources of the distributed sensors are modeled in the same filter, which ensures that all error correlations are accounted for. Moreover, using the pseudorange and Doppler measurements, if there are less than 4 satellites in view, the measurements from the remaining satellites can still be used to update the INS estimates, which improve the system robustness.

Cost-effective sensors

The advent of cheaper MEMS-based sensor has opened new horizons for inertial navigation systems. The MEMS technology drives the heavy and inflexible inertial sensor system to small, cost-effective, light-weight, portable and low-power silicon-based inertial devices. Although the cheap MEMS-based sensors do not exhibit highly accurate navigation performance with respect to the higher level sensors, they can be used to meet the requirements of many navigation applications when aided with GPS devices. Regarding single point low-cost L1 GPS receivers, their prices are usually at the range of 20 € to 300 €. For the GPS receiver chip, which is mounted inside of the mobile phone with an onboard GPS antenna, it's unit price reaches only 5 € and this price is still dropping [3].

Regarding the INS/GPS integration system, the off-the-shelf products are mainly implemented in loosely-coupled manner. The tightly-coupled integration system is seldom to be seen in the market yet, and its price is usually high, which bases on the level of inertial sensor in use. In this dissertation, the low-cost MEMS-IMU (i.e. consumer grade) and single point L1 GPS receivers are used. The concentration is on the development of sophisticated sensor fusion algorithm. Numerical results show that the integration of a GPS receiver with a low level inertial sensor can present competitive estimation results with respect to the integration using high level inertial sensors.

Reasons for applying PF on INS/GPS tightly-coupled integration

The INS/GPS tightly-coupled integration is a typical nonlinear filtering problem. Besides, the GPS pseudorange measurement is often affected by the multipath effects, yielding its error to be non-Gaussian distributed. Recently, there have been proposed approaches based on statistical processing theory that try to overcome the multipath effects. The PF as a nonlinear non-Gaussian estimation method may constitute a better solution and hence shows its great suitability to be used here [4, 5]. In order to reduce its processing load without degrading system estimation accuracy, some researchers propose to combine the PF with other filters (i.e., EKF or UKF). Such an approach presents robust system performances using only a small number of particles.

Using quaternions as the representation of attitude

Regarding the representation of attitude, various parameterizations can be used, such as Euler angles, direction cosine matrix (DCM), quaternions, etc. Quaternions were firstly introduced by Hamilton [6]. In this thesis, the quaternion-based approaches are applied in the INS/GPS tightly-coupled integration using EKF, UKF and UPF algorithms, which do not exhibit the singularity problems inherited in Euler angle-based approaches, and the quaternion vector involves only 4 elements instead of 9 elements in DCM-based approaches.

1. INS/GPS Integration Principles

1.1 Introduction

In this chapter, the single point L1 GPS receiver data processing is presented in detail. It includes the compensation of GPS measurement errors (e.g., ionospheric errors, tropospheric errors, etc.), the computation of satellite positions in Earth-Centered, Earth-Fixed (ECEF) coordinate based on ephemeris orbital data, and the computation of user position and velocity fixes using a least-squares estimation method. Besides, the INS principle and INS strapdown mechanization process are introduced. The derivation is in the navigation frame, which will be used afterwards in INS/GPS integration. In the INS/GPS integration, the error states are used and the system models are derived from the linearization process. The INS/GPS loosely-coupled and tightly-coupled integrations are introduced in detail, and their differences are discussed. A field experiment is conducted to demonstrate the outperformance of an INS/GPS system with respect to a GPS alone device.

1.2 GPS data processing

Global Positioning System (GPS) is a radio-based Global Navigation Satellite System (GNSS) established by the U.S Department of Defense. Details on GPS related topics have been well introduced in many references (e.g., [1, 2, 7-11]). The GPS receiver offers three kinds of measurements, i.e., pseudorange, Doppler and carrier phase. In the following subsections, we first look at the GPS measurements collected from a low-cost GPS receiver (u-blox Antaris 4) based on a train ride with two GPS outage periods. And then, the observation models of the pseudorange and Doppler are given. Their error compensation processes are highlighted. Moreover, a least-squares estimation method is applied here for computing the user position and velocity fixes using error compensated GPS pseudorange and Doppler measurements.

1.2.1 L1 GPS measurements

Two types of GPS measurements are in the focus of this chapter, namely, code pseudorange and Doppler, which are often used to compute the position and velocity fixes of the receiver antenna phase center. A set of field collected measurements (i.e., code pseudorange, Doppler and carrier phase data) from ublox Antaris 4 are plotted in **Figure 1-1**, where different colors in the figure represent measurements from different satellites. This dataset is collected based on a train ride with two tunnels (GPS outage environments). The trajectory lasts 1400 s. Details on this experiment can be found in Section 3.2.1.

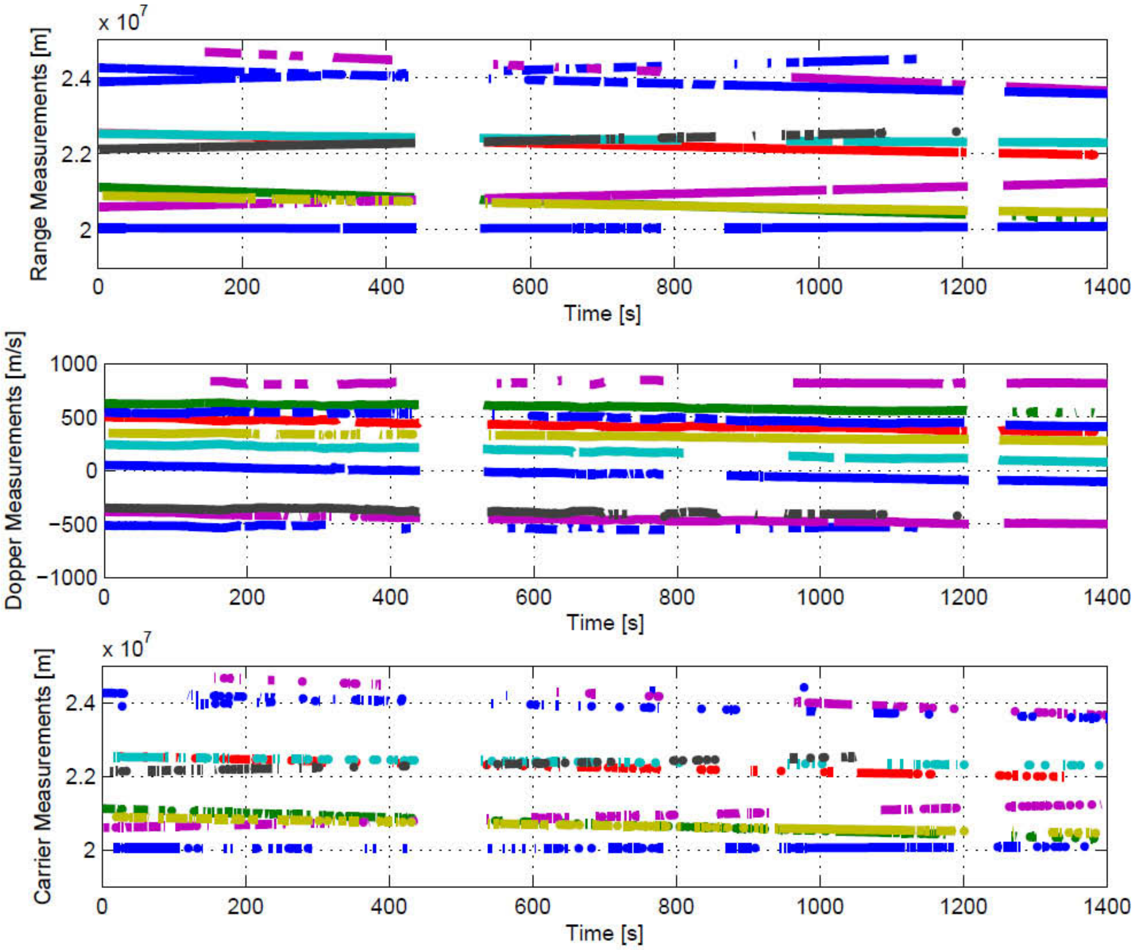


Figure 1-1. GPS measurements

As shown in the figure, the pseudorange and Doppler measurements are continuous and robust. However, for the carrier phase data, they are more easily to be disturbed by external influences, for instance, buildings (i.e. railway stations), trees and hills nearby. **Figure 1-2** shows the number of measurements during the path, where the number of carrier phase data drops below 4 frequently. In this thesis, for land-based navigation, the pseudorange and Doppler are used in the INS/GPS integration.

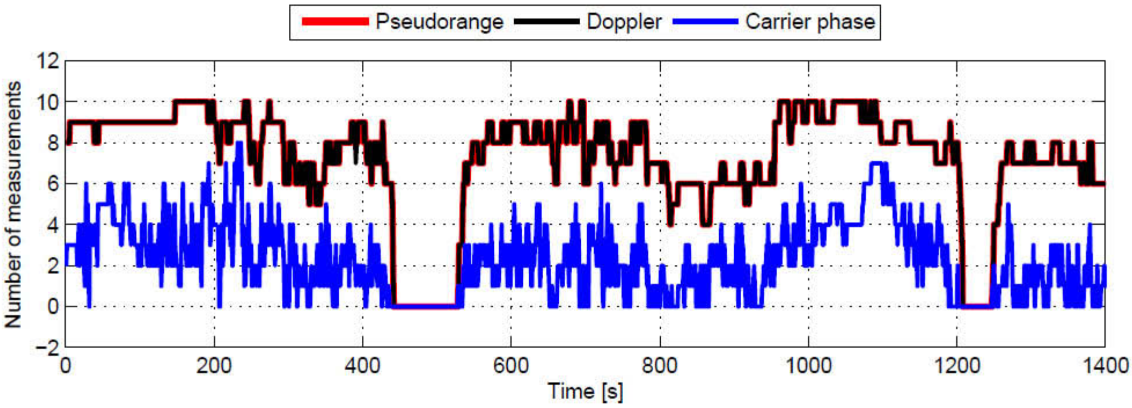


Figure 1-2. Number of measurements

1.2.2 Pseudorange measurement model

Code pseudorange measurements can be described by the multiplication of the apparent signal transit time with the speed of light. The apparent signal transit time is defined as the difference between signal reception time determined by the receiver and the transmission time at the satellite, which is marked on the signal. The measured pseudorange is biased due to the fact that the satellite and receiver clocks are not synchronized, and each keeps time independently. Therefore, the receiver and satellite clock biases must be considered in the data processing.

Moreover, the ionospheric and tropospheric delays must be correctly modeled and compensated from the measured pseudorange data. Besides, the interference from signals reflected from the surfaces in the vicinity of the GPS antenna will also significantly deteriorate the received measurement. Considering all these aspects, the biased pseudorange measurement from one satellite vehicle (SV) at one time instance is formulated as:

$$\tilde{\rho} = \rho + c\left(t_u - t_s\right) + T_{iono} + T_{tropo} + \varepsilon_\rho \tag{1.1}$$

where

$\tilde{\rho}$ Measured user to satellite range

ρ	True user to satellite range
t_u	Receiver clock bias
t_s	Satellite clock bias
T_{iono}	Ionospheric delay
T_{tropo}	Tropospheric delay
ε_ρ	Other un-modeled errors, i.e., multipath delay

For single-point positioning, the modeling and compensation of pseudorange measurement errors are critical. Some of them must be handled in an iterative way. A flowchart of the measurement error compensation is given in **Figure 1-3**.

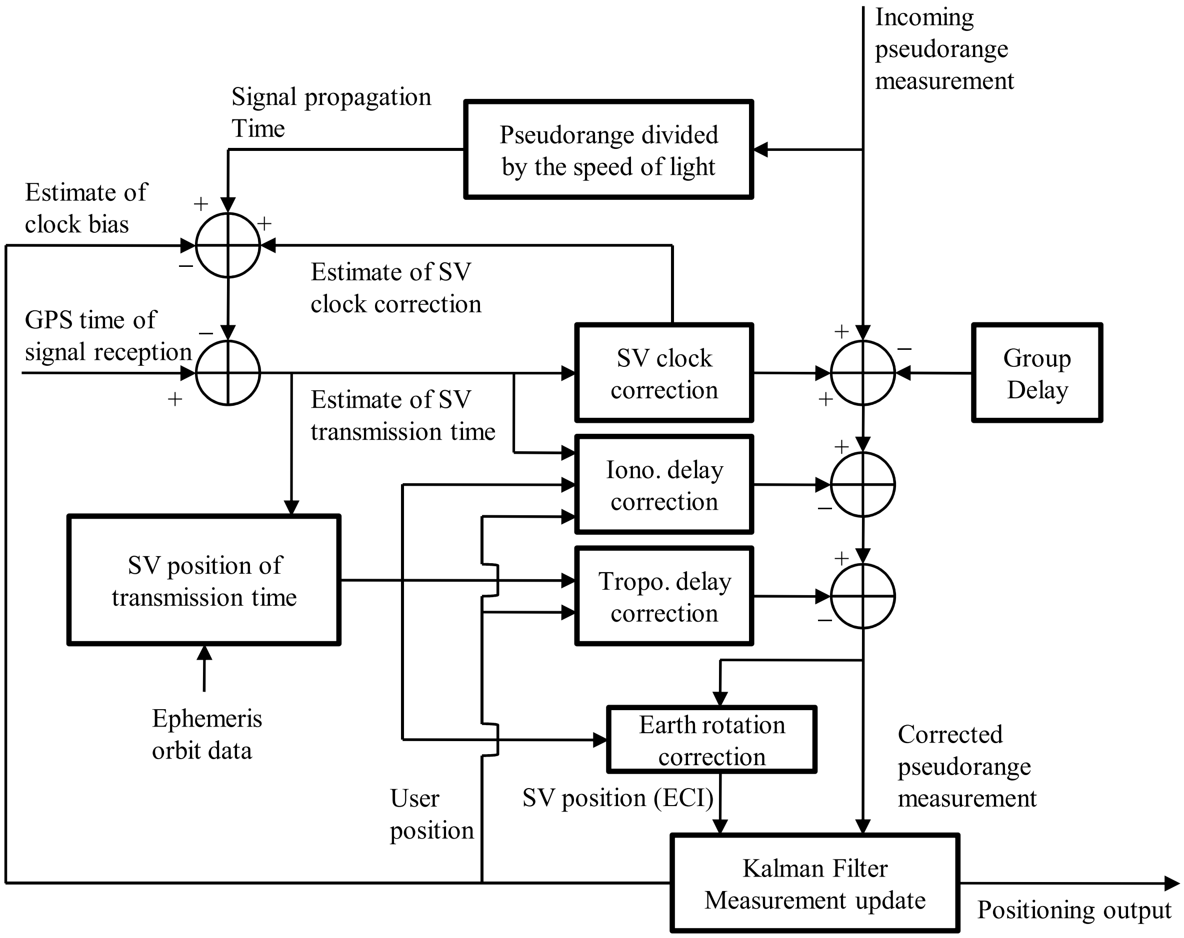


Figure 1-3. Flowchart of measurement error compensation process

The error compensation process mainly involves the following computations:

- Satellite signal transmission time
- Satellite clock correction including the relativistic effects
- Satellite position at the transmission time based on the ephemeris orbital data

- Ionospheric errors
- Tropospheric errors
- Errors introduced by the earth rotation

1.2.2.1 Computation of satellite signal transmission time

The navigation message broadcasted by each satellite carries time stamps in accordance with the satellite clock using the so-called Z-count, which increments in units of 1.5 s. A GPS satellite transmits navigation messages at 50Hz, and the message is formatted into frames of 1500 bits. It takes 30 s to transit a frame. Each frame has 5 sub-frames. Each sub-frame is 6 s long, containing ten 30-bit words. The second word in each sub-frame named “HOW” specifies the number of sub-frames elapsed (i.e., how many 6 s) since the beginning of the week (transition between Saturday and Sunday) using Z-counts [1, 7, 11].

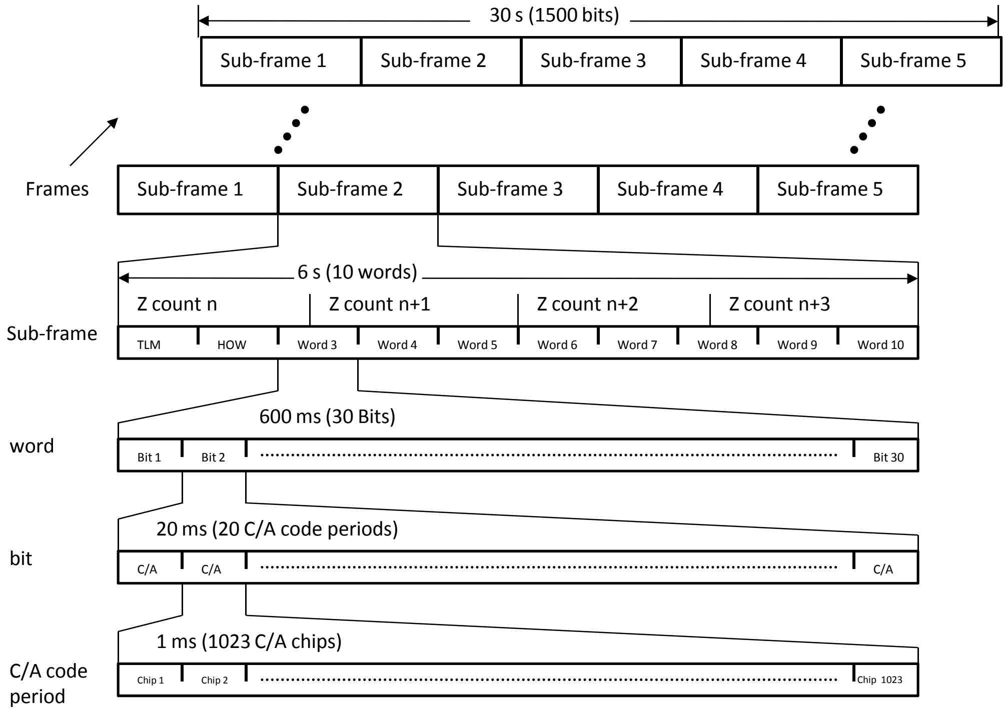


Figure 1-4. GPS navigation message organization and timing relationship

In order to determine the satellite clock time associated with the current measurement epoch, one needs to know the Z-count associated with the current sub-frame, plus the satellite time elapsed since the beginning of the sub-frame. This elapsed time is computed as the sum of following terms:

1. Number of navigation message data bits transmitted in the current sub-frame
2. Number of C/A code periods since the beginning of the current message bit
3. Number of whole chips in the current C/A-code cycle
4. The fraction of the current C/A-code chip

The 1 and 2 are counted by the receiver software. The remaining two terms are provided by the receiver delay lock loop (DLL) though correlation process.

The timing relationship between the satellite signal transmission time, receiver signal reception time, clocks offsets and pseudorange time equivalent are illustrated in **Figure 1-5**. That is, the satellite signal transmission time is equal to subtracting the ideal pseudorange divided by speed of light from the receiver's time-tag for the measurement, and compensating clock bias terms.

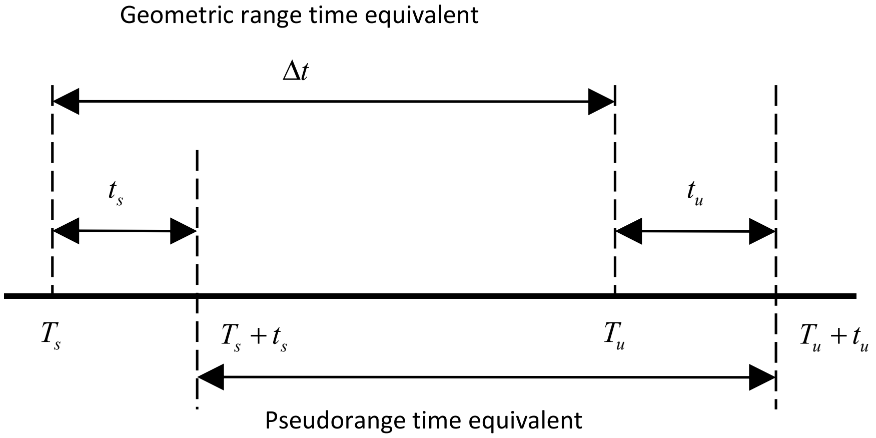


Figure 1-5. Pseudorange measurement timing relationship

This timing relationship can be formulated in Equation (1.2).

$$T_s = T_u - \left(\frac{\tilde{\rho}}{c} - t_u + t_s\right) \tag{1.2}$$

where:

T_s	Satellite signal transmission time
T_u	Receiver signal reception time
t_s	Offset of satellite clock from GPS reference time
t_u	Offset of receiver clock from GPS reference time
$T_s + t_s$	Satellite clock reading at the time of signal transmission
$T_u + t_u$	Receiver clock reading at the time the signal reception
c	Speed of light

1.2.2.2 Computation of satellite clock correction

The satellites contain atomic clocks that control on-board timing operations. The master control segment (MCS) consistently determines and transmits clock correction parameters to the satellites for rebroadcast in the navigation message. The correction parameter is made by the receiver using a second-order polynomial [7]:

$$\begin{aligned} &F = -2\sqrt{\mu}/c^2 \\ &\Delta t_r = Fe\sqrt{a}\sin E_k \\ &t_s = a_{f0} + a_{f1}(T_s - t_{oc}) + a_{f2}(T_s - t_{oc})^2 + \Delta t_r \end{aligned} \tag{1.3}$$

where:

E_k :	Eccentric anomaly of the satellite orbit
Δt_r :	Correction due to the relativistic effects
a_{f0} :	Clock bias parameter
a_{f1} :	Clock drift parameter
a_{f2} :	Frequency drift parameter
t_{oc} :	Ephemeris clock data reference time
T_s :	Signal transmission time

The satellite group delays are initially calculated by the master control segment, and the values for each satellite are updated to reflect the actual on-orbit group delay. These values are given in the navigation data message. Therefore, corrections can be directly applied as:

$$t_s^{gd} = t_s - T_{gd} \tag{1.4}$$

where

T_{gd}	The group delay of the satellite expressed in seconds
t_s^{gd}	Satellite clock offset with group delay compensated

1.2.2.3 Computation of satellite position based on ephemeris orbital parameters

The ability to accurately predict the position of a satellite at the instant of signal transmission is vital to the operation of GPS. The satellite position can be computed in Earth-Centered, Earth-Fixed (ECEF) coordinate at its time of transmission using the broadcasted ephemeris orbital data. The ephemeris parameters describe the orbit of the space vehicle, and provide the best trajectory fit for each specific fit interval. The orbital parameters using Keplerian orbital parameters are given in **Table 1-1**. Constant parameters are given in **Table 1-2**. And the computation of satellite position at signal transmission time is shown in **Table 1-3**. A detailed explanation can also be found in [1, 7].

Table 1-1. Ephemeris orbital data

Data	Physical meanings
M_0	Mean Anomaly at Reference Time
Δn	Mean Motion Difference From Computed Value
e	Eccentricity
$\sqrt{a}$	Square Root of the Semi-Major Axis
Ω_0	Longitude of Ascending Node of Orbit Plane at Weekly Epoch
i_0	Inclination Angle at Reference Time
ω	Argument of Perigee
$\dot{\Omega}$	Rate of Right Ascension
di/dt	Rate of Inclination Angle
C_{uc}	Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude
C_{us}	Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude
C_{rc}	Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius
C_{rs}	Amplitude of the Sine Harmonic Correction Term to the Orbit Radius
C_{ic}	Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination
C_{is}	Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination
t_{oe}	Reference Time Ephemeris

Table 1-2. Constant parameters

Constant parameters	Physical meanings
$\mu = 3.986005 \times 10^{14}$ m^3/s^2	Earth universal gravitational parameter (WGS 84)
$\dot{\Omega}_e = 7.2921151467 \times 10^{-5}$ rad/s	Earth rotation rate (WGS 84)
$c = 2.99792458 \times 10^{8}$ m/s	Speed of light

Table 1-3. Calculation of satellite position using ephemeris orbital data

Computations		Physical meanings of computations
1)	$a = \left(\sqrt{a}\right)^2$	Semi-major axis
2)	$n_0 = \sqrt{\frac{\mu}{a^3}}$	Computed mean motion (rad/sec)
3)	$t_k = T_s - t_{oe}$	Time from ephemeris reference epoch
4)	$n = n_0 + \Delta n$	Corrected mean motion
5)	$M_k = M_0 + n_{t_k}$	Mean anomaly
6)	$\begin{cases} E_k = M_k ; E_k' = E_k ; f_1 = 1; f_2 = 0; \\ while\ f_1 > 10^{-13} \\ \qquad E_k = M_k + e \sin E_k ; \\ \qquad f_1 = \lVert E_k - E' \rVert ; E' = E_k ; f_2 = f_2 + 1; \\ \qquad if\ f_2 > 10, break, end \\ end \end{cases}$	Kepler's Equation for Eccentric Anomaly (must be solved iteratively for E_k).
7)	$v_k = \tan^{-1}\left\{ \frac{\sqrt{1-e^2}\sin E_k \big/ (1 - e\cos E_k)}{\left(\cos E_k - e\right) \big/ (1 - e\cos E_k)} \right\}$	True Anomaly
8)	$E_k = \cos^{-1}\left\{ \frac{e + \cos v_k}{1 + e\cos v_k} \right\}$	Eccentric Anomaly
9)	$\Phi_k = v_k + \omega$	Argument of Latitude
10)	$\begin{cases} \delta u_k = C_{us}\sin 2\Phi_k + C_{uc}\cos 2\Phi_k \\ \delta r_k = C_{rs}\sin 2\Phi_k + C_{rc}\cos 2\Phi_k \\ \delta i_k = C_{is}\sin 2\Phi_k + C_{ic}\cos 2\Phi_k \end{cases}$	Argument of Latitude Correction Radius Correction Inclination Correction
11)	$u_k = \Phi_k + \delta u_k$	Corrected Argument of Latitude
12)	$r_k = a\left(1 - e\cos E_k\right) + \delta r_k$	Corrected Radius
13)	$i_k = i_0 + \delta i_k + \left(di/dt\right) t_k$	Corrected Inclination
14)	$\begin{cases} x_k' = r_k \cos u_k \\ y_k' = r_k \sin u_k \end{cases}$	Positions in orbital plane
15)	$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e) t_k - \dot{\Omega}_e t_{oe}$	Corrected longitude of ascending node
16)	$\begin{cases} x_k = x_k' \cos\Omega_k - y_k' \cos i_k \sin\Omega_k \\ y_k = x_k' \sin\Omega_k + y_k' \cos i_k \cos\Omega_k \\ z_k = y_k' \sin i_k \end{cases}$	Satellite position in ECEF frame

1.2.2.4 Computation of the ionospheric error

In this dissertation, the ionospheric delay corrections are computed from the broadcasted model parameters in the navigation message according to GPS-ICD-200 [7]. It is estimated that the use of this model will provide at least a 50 percent reduction in the single-frequency user's root mean square (RMS) error due to ionospheric propagation effects [1, 7].

Table 1-4. Computation of ionospheric correction

Computations	
1)	$E = el / \pi$
2)	$F = 1.0 + 16[0.53 - E]^3$
3)	$\psi = \frac{0.0137}{E + 0.11} - 0.022$
4)	$\begin{cases} if\ \lvert\phi_i\rvert \le 0.416,\ \phi_i = \phi_u + \psi\cos A; \\ if\ \phi_i > 0.416,\ \phi_i = 0.416; \\ if\ \phi_i < -0.416,\ \phi_i = -0.416; \end{cases}$
5)	$\lambda_i = \lambda_u + \frac{\psi \sin A}{\cos\phi_i}$
6)	$\phi_m = \phi_i + 0.064\cos(\lambda_i - 1.617)$
7)	$t = 4.32\times10^4\lambda_i + GPStime$
8)	$if\ t \ge 86400,\ t = t - 86400;$ $if\ t < 0,\ t = t + 86400;$
9)	$\begin{cases} if\ PER \ge 72000,\ PER = \beta_1 + \beta_2\phi_m + \beta_3\phi_m{}^2 + \beta_4\phi_m{}^3; \\ if\ PER < 72000,\ PER = 72000; \end{cases}$
10)	$x = \frac{2\pi(t - 50400)}{PER}$
11)	$\begin{cases} if\ AMP \ge 0,\ AMP = \alpha_1 + \alpha_2\phi_m + \alpha_3\phi_m{}^2 + \alpha_4\phi_m{}^3; \\ if\ AMP < 0,\ AMP = 0; \end{cases}$
12)	$\begin{cases} f\ \lvert x\rvert < 1.57,\ T_{iono} = F\left[5\times10^{-9} + AMP\left(1 - \frac{x^2}{2} + \frac{x^4}{24}\right)\right]; \\ if\ \lvert x\rvert \ge 1.57,\ T_{iono} = F * 5\times10^{-9}; \end{cases}$

where

α_n	The coefficients of a cubic equation representing the amplitude of the vertical delay
β_n	The coefficient of a cubic equation representing the period of the model

E	Elevation angle between the user and satellite in semi-circles
A	Azimuth angle between the user and satellite, measured clockwise positive from the true North in semi-circles
ϕ_u	User geodetic latitude in semi-circles (WGS-84)
λ_u	User geodetic longitude in semi-circles (WGS-84)
x	Phase in radians
t	local time
ϕ_m	Geomagnetic latitude of the earth projection of the ionospheric intersection point
λ_i	Geodetic longitude of the earth projection of the ionospheric intersection point
ϕ_i	Geodetic latitude of the earth projection of the ionospheric intersection point
Ψ	Earth's central angle between the user position and the earth projection of ionospheric intersection point

1.2.2.5 Computation of the tropospheric errors

The tropospheric error is often modeled as including both a dry (hydrostatic) and wet (non-hydrostatic) components. The dry component, which arises from the dry air, gives rise to about 90% of the tropospheric delay and can be predicted accurately [9-11]. The wet component, which arises from the water vapor, is more difficult to be predicted due to uncertainties in the atmospheric distribution (e.g., based on local weather condition and may change dramatically over time). The approach to cope with dry delay is usually handed by computing its delay in zenith direction (e.g., using Hopfield model), with a map function for considering the elevation angle [9, 10], as shown in Equation (1.5).

$$T_{tropo} = \frac{1e-6}{5} \cdot \frac{77.64 \cdot p/T}{\sin\sqrt{el^2 + 6.25}} \cdot \left[40136 - 148.72(T - 273.16)\right] \tag{1.5}$$

where "*el*" is the elevation angle; "*p*" is the atmospheric pressure in millibar (mb) and "*T*" is the temperature in Kelvin.

As shown in Equation (1.5), in order to accurately compute the dry delay, the accurate local surface temperature and pressure measurements should be given. However, for navigation applications, such information may not always be available. Thus, a simplified model can be used (derived empirically) as:

$$T_{tropo} = \frac{2.47}{\sin(el) + 0.0121} \tag{1.6}$$

The estimated tropospheric dry delay with respect to different elevation angles (i.e., from 1 deg to 90 deg) are compared using Equation (1.5) and (1.6). For Hopfield's model, we consider “p” as standard atmosphere (i.e., 1013.25 mb), and temperature as 298.15 Kelvin (i.e., 25° Celsius). The computed dry delays are shown in **Figure 1-6**.

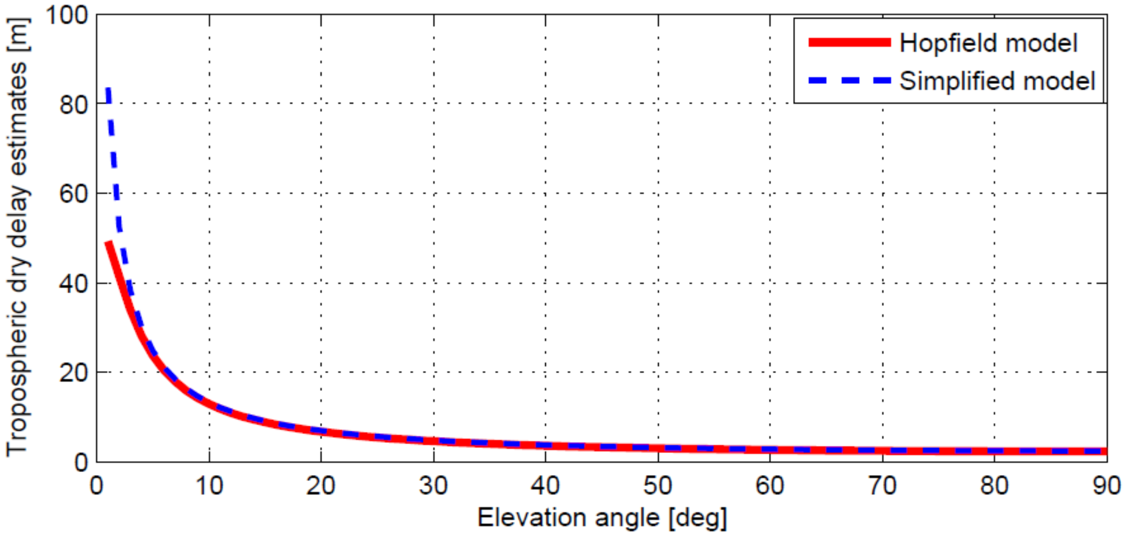


Figure 1-6: Tropospheric delay estimates comparison using different models

As depicted in the figure, major differences appear when the elevation angle is smaller than 5 degrees. However, in the GPS data processing, usually a mask elevation angle (e.g., >10 deg) is introduced to avoid using the measurements from the satellite with very low elevation angle. These measurements are often heavily deteriorated by interferes (e.g., multipath effects). For large elevation angles, differences in the estimated dry delays are very small, e.g., 0.1 m at 90 deg elevation angle and 0.3 m with 10 deg elevation angle.

1.2.2.6 Errors introduced from earth rotation

Due to the rotation of the earth during the time of signal transmission, a relativistic error is introduced, which is known as the *Sagnac effect*. That is, during the signal propagation period, the earth experiences a finite rotation with respect to an earth center inertial (ECI) coordinate system. If the satellite and receiver coordinates are expressed in ECEF frame, the earth rotation during the signal propagation cannot be taken into consideration (coordinate rotates with the earth). Therefore errors are introduced.

One approach to avoid the *Sagnac effect* is to work within an ECI coordinate frame for satellite and user position computations. An ECI frame can be artificially obtained by freezing an ECEF frame at every time instant when the pseudorange measurements are made by the receiver to the set of visible satellites. At these time instances, the ECI and ECEF frames are overlapped. For computing the user positions, they are the same. However, the satellite coordinates in ECEF frame needs to be transformed into ECI considering the earth rotation during the signal propagation period. This transformation can be calculated in Equation (1.7).

$$\begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix}_{ECI} = \begin{bmatrix} \cos\dot{\Omega}(T_u - T_s) & \sin\dot{\Omega}(T_u - T_s) & 0 \\ -\sin\dot{\Omega}(T_u - T_s) & \cos\dot{\Omega}(T_u - T_s) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix}_{ECEF} \tag{1.7}$$

The $T_u - T_s$ can be calculated as the corrected pseudorange measurements divided by speed of light. After coordinate transformation, both satellite and receiver coordinates are expressed in the same frame. They are either in ECI or in ECEF, because these two coordinates are overlapped at the time instance, when measurements are made by the receiver [1].

1.2.2.7 Multipath errors

Multipath refers to the phenomenon of a signal reaching an antenna via two or more paths. Typically, one antenna receives the direct (i.e., line-of-sight) signal and one or more of its reflections from structures in the vicinity and from the ground. The subsequent pseudorange measurements are of the sum of all the received signals.

The pseudorange measurement multipath error is based on the strength of the reflected signal and the delay between the direct and reflected signals. It is probably the dominant source of error in GPS-based high-precision applications since it can introduce a bias up to a hundred of meters when employing a 1-chip wide (standard) Delay Lock Loop (DLL) to track the PRN code [11]. The method to reduce the multipath effect is either to locate the antenna away from reflectors (not always practical), or improve the receiver antenna design and signal processing steps in tracking loops, which can be only achieved by the receiver manufactures.

1.2.3 Position determination using pseudorange measurements

In order to determine the user position and receiver clock offset, usually four pseudorange measurements are required, which relate the user position coordinates as:

$$\rho_j = f(x_u, y_u, z_u, t_u) = \sqrt{\left(x_j - x_u\right)^2 + \left(y_j - y_u\right)^2 + \left(z_j - z_u\right)^2} + ct_u \tag{1.8}$$

The "j" represents the *j*-th satellite, and (x_j, y_j, z_j) denotes the *j*-th satellite's position coordinates in ECEF. The true user position and receiver clock offset (x_u, y_u, z_u, t_u) are considered to include approximate components ($\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u$) and incremental components ($\delta x_u, \delta y_u, \delta z_u, \delta t_u$) as:

$$\begin{aligned} x_u &= \hat{x}_u - \delta x_u \\ y_u &= \hat{y}_u - \delta y_u \\ z_u &= \hat{z}_u - \delta z_u \\ t_u &= \hat{t}_u - \delta t_u \end{aligned} \tag{1.9}$$

Thus, equation (1.8) can be reformulated as:

$$\begin{aligned} \rho_j &= f(\hat{x}_u - \delta x, \hat{y}_u - \delta y, \hat{z}_u - \delta z, \hat{t}_u - \delta t) \\ &= f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u) - \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{x}_u}\delta x - \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{y}_u}\delta y \\ &- \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{z}_u}\delta z - \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{t}_u}\delta t + h.o.t \end{aligned} \tag{1.10}$$

We denote satellite position vector as $\mathbf{x}_j = [x_j, y_j, z_j]^T$, and the estimated receiver position vector as $\mathbf{x}_u = [x_u, y_u, z_u]^T$. $\left\|\mathbf{x}_j - \mathbf{x}_u\right\|$ represents the estimated satellite-to-user distance. The partial derivative parameters in Equation (1.10) can be computed as:

$$\begin{aligned} \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{x}_u} &= -\frac{x_j - \hat{x}_u}{\left\|\mathbf{x}_j - \mathbf{x}_u\right\|} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{y}_u} &= -\frac{y_j - \hat{y}_u}{\left\|\mathbf{x}_j - \mathbf{x}_u\right\|} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{z}_u} &= -\frac{z_j - \hat{z}_u}{\left\|\mathbf{x}_j - \mathbf{x}_u\right\|} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{t}_u} &= c \end{aligned} \tag{1.11}$$

We denote $\mathbf{l}_j = \frac{1}{\left\|\mathbf{x}_j - \mathbf{x}_u\right\|}\left[x_j - \hat{x}_u, y_j - \hat{y}_u, z_j - \hat{z}_u\right]^T$, which is the estimated line-of-sight unit vector pointing from the initial estimate of the user position to the *j*-th satellite. By substituting $\hat{\rho}_j = f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)$ in Equation (1.10) and ignoring the higher order terms (i.e., "h.o.t" in Equation (1.10)). For n pseudoranges, we have:

$$\Delta\boldsymbol{\rho} = \begin{bmatrix} \hat{\rho}_1 - \rho_1 \\ \hat{\rho}_2 - \rho_2 \\ \vdots \\ \hat{\rho}_n - \rho_n \end{bmatrix} = \begin{bmatrix} \left(-\mathbf{l}_1^T\right)_{1\times3}, 1 \\ \left(-\mathbf{l}_2^T\right)_{1\times3}, 1 \\ \vdots \\ \left(-\mathbf{l}_n^T\right)_{1\times3}, 1 \end{bmatrix} \begin{bmatrix} \delta x_u \\ \delta y_u \\ \delta z_u \\ c\delta t_u \end{bmatrix} + \boldsymbol{\varepsilon}_\rho \tag{1.12}$$

We define:

$$\delta\mathbf{p}_u = \begin{bmatrix} \delta x_u \\ \delta y_u \\ \delta z_u \end{bmatrix}, \ \mathbf{H} = \begin{bmatrix} \left(-\mathbf{l}_1^T\right)_{1\times3}, 1 \\ \left(-\mathbf{l}_2^T\right)_{1\times3}, 1 \\ \vdots \\ \left(-\mathbf{l}_n^T\right)_{1\times3}, 1 \end{bmatrix} \tag{1.13}$$

Thus, for 4 satellites in view, we can solve Equation (1.12) directly as:

$$\begin{bmatrix} \delta\mathbf{p}_u \\ c\delta t_u \end{bmatrix} = \mathbf{H}^{-1} \cdot \Delta\boldsymbol{\rho} \tag{1.14}$$

After obtaining the incremental components, we can update the estimates of user's coordinates and clock offset using Equation (1.9). This process (i.e. from Equation (1.9) to (1.14)) should be reiterated until $\left\|\delta\mathbf{p}_u\right\|$ is sufficiently small, which is decided by the user's accuracy requirements. For more than four satellites available, a least-squares estimation technique can be applied. In this way, the incremental components are computed as:

$$\begin{bmatrix} \delta\mathbf{p}_u \\ c\delta t_u \end{bmatrix} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1} \mathbf{H}^T \cdot \Delta\boldsymbol{\rho} \tag{1.15}$$

1.2.4 Doppler measurement model

The relative motion of a satellite and the user results in changes in the observed frequency of the satellite signal. Given the satellite velocity (dividing the difference of satellite positions from adjacent epochs by the time interval), the Doppler measurements can be used to estimate the user velocity. However, they are biased by the receiver clock drift errors. The Doppler measurement from one satellite at one time instance can be formulated as [11]:

$$\tilde{\dot{\rho}} = \dot{\rho} + c\left(\dot{t}_u - \dot{t}_s\right) + \dot{T}_{iono} + \dot{T}_{tropo} \tag{1.16}$$

where

$\tilde{\dot{\rho}}$	The measured user to satellite range rate
$\dot{\rho}$	The true user to satellite range rate
$\dot{t}_u$	Receiver clock drift
$\dot{t}_s$	Satellite clock drift
$\dot{T}_{iono}$	Rate of change in ionospheric delay
$\dot{T}_{tropo}$	Rate of change in tropospheric delay

The Doppler shift can also be written as a projection of the relative velocity vector on the satellite line-of-sight vector as [11]:

$$\tilde{\dot{\rho}} = \left(\mathbf{l}_u^s\right)^T (\mathbf{v}_s - \mathbf{v}_u) + c\dot{t}_u + \varepsilon_{\dot{\rho}} \tag{1.17}$$

where

$\mathbf{l}_u^s$	User-to-satellite line-of-sight unit vector
$\mathbf{v}_s$	Satellite velocity vector (e.g., in ECEF frame)
$\mathbf{v}_u$	Receiver velocity vector (e.g., in ECEF frame)
$\varepsilon_{\dot{\rho}}$	Combined error due to changes during the measurement interval in the satellite clock, ionosphere, and troposphere.

1.2.5 Velocity determination using Doppler measurements

As shown in Equation (1.17), with the knowledge of user location, the following relationship exists for the *j*-th satellite:

$$\tilde{\dot{\rho}}_j - \left(\mathbf{l}_j\right)^T \mathbf{v}_j = -\left(\mathbf{l}_j\right)^T \mathbf{v}_u + c\dot{t}_u + \varepsilon_{\dot{\rho}} \tag{1.18}$$

The quantities on the left side of Equation (1.18) are either measurement or already computed. The $\mathbf{v}_j = \left[\mathbf{v}_{j,x}, \mathbf{v}_{j,y}, \mathbf{v}_{j,z}\right]^T$ from *j*-th satellite can be derived from ephemeris orbital data. Therefore, we can denote the left side of Equation (1.18) as d_j, and Equation (1.18) is formulated as:

$$d_j = -\left(\mathbf{l}_j\right)^T \mathbf{v}_u + c\dot{t}_u + \varepsilon_{\dot{\rho}} \tag{1.19}$$

For n Doppler measurements, we have:

$$\mathbf{d} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} \left(-\mathbf{l}_1^T\right)_{1\times 3}, & 1 \\ \left(-\mathbf{l}_2^T\right)_{1\times 3}, & 1 \\ \vdots, & 1 \\ \left(-\mathbf{l}_n^T\right)_{1\times 3}, & 1 \end{bmatrix} \begin{bmatrix} v_{u,x} \\ v_{u,y} \\ v_{u,z} \\ c\dot{t}_u \end{bmatrix} + \boldsymbol{\varepsilon}_{\dot{\rho}} \tag{1.20}$$

Using the geometric matrix $\mathbf{H}$, as introduced in Equation (1.13), when 4 satellites are in view, user velocity estimates are computed as:

$$\begin{bmatrix} \mathbf{v}_u \\ c\dot{t}_u \end{bmatrix} = \mathbf{H}^{-1} \cdot \mathbf{d} \tag{1.21}$$

For more than 4 satellites in view, the least-squares estimation method can be utilized.

$$\begin{bmatrix} \mathbf{v}_u \\ c\dot{t}_u \end{bmatrix} = \left(\mathbf{H}^T \mathbf{H}\right)^{-1} \mathbf{H}^T \cdot \mathbf{d} \tag{1.22}$$

Using error states, similar like in former section, we define the true user velocity and receiver clock drift ($\dot{x}_u$, $\dot{y}_u$, $\dot{z}_u$, $\dot{t}_u$) are considered to include approximate components ($\hat{\dot{x}}_u, \hat{\dot{y}}_u, \hat{\dot{z}}_u, \hat{\dot{t}}_u$) and incremental components ($\delta\dot{x}_u, \delta\dot{y}_u, \delta\dot{z}_u, \delta\dot{t}_u$) as:

$$\begin{aligned} \dot{x}_u &= \hat{\dot{x}}_u - \delta\dot{x}_u \\ \dot{y}_u &= \hat{\dot{y}}_u - \delta\dot{y}_u \\ \dot{z}_u &= \hat{\dot{z}}_u - \delta\dot{z}_u \\ \dot{t}_u &= \hat{\dot{t}}_u - \delta\dot{t}_u \end{aligned} \tag{1.23}$$

We denote $\mathbf{v}_u = [\dot{x}_u, \dot{y}_u, \dot{z}_u]^T$, $\hat{\mathbf{v}}_u = \left[\hat{\dot{x}}_u, \hat{\dot{y}}_u, \hat{\dot{z}}_u\right]^T$ and $\delta\mathbf{v}_u = [\delta\dot{x}_u, \delta\dot{y}_u, \delta\dot{z}_u]^T$. Thus, we have $\mathbf{v}_u = \hat{\mathbf{v}}_u - \delta\mathbf{v}_u$. Substitute them into Equation (1.17) yields:

$$\tilde{\dot{\rho}}_j = \left(\mathbf{l}_j\right)^T (\mathbf{v}_j - \hat{\mathbf{v}}_u + \delta\mathbf{v}_u) + c\hat{\dot{t}}_u - c\delta\dot{t}_u + \varepsilon_{\dot{\rho}_j} \tag{1.24}$$

Rearranging Equation (1.24) as:

$$\tilde{\dot{\rho}}_j - \left[\left(\mathbf{l}_j \right)^T (\mathbf{v}_j - \hat{\mathbf{v}}_u) + c\hat{\dot{t}}_u \right] = \left(\mathbf{l}_j \right)^T \delta \mathbf{v}_u - c\delta \dot{t}_u + \varepsilon_{\dot{\rho}_j} \tag{1.25}$$

We denote $\hat{\dot{\rho}}_j = \left(\mathbf{l}_j \right)^T (\mathbf{v}_j - \hat{\mathbf{v}}_u) + c\hat{\dot{t}}_u$, and we have:

$$\hat{\dot{\rho}}_j - \tilde{\dot{\rho}}_j = \left(-\mathbf{l}_j \right)^T \delta \mathbf{v}_u + c\delta \dot{t}_u + \varepsilon_{\dot{\rho}_j} \tag{1.26}$$

Thus, for n Doppler measurements, we have:

$$\Delta \dot{\boldsymbol{\rho}} = \begin{bmatrix} \hat{\dot{\rho}}_1 - \tilde{\dot{\rho}}_1 \\ \hat{\dot{\rho}}_2 - \tilde{\dot{\rho}}_2 \\ \vdots \\ \hat{\dot{\rho}}_n - \tilde{\dot{\rho}}_n \end{bmatrix} = \begin{bmatrix} \left(-\mathbf{l}_1^T \right)_{1\times 3}, & 1 \\ \left(-\mathbf{l}_2^T \right)_{1\times 3}, & 1 \\ \vdots, & 1 \\ \left(-\mathbf{l}_n^T \right)_{1\times 3}, & 1 \end{bmatrix} \begin{bmatrix} \delta \dot{x}_u \\ \delta \dot{y}_u \\ \delta \dot{z}_u \\ c\delta \dot{t}_u \end{bmatrix} + \boldsymbol{\varepsilon}_{\dot{\rho}} \tag{1.27}$$

1.3 INS principle

Inertial navigation is based on Newtonian physics and is affected by gravity. That is, the object will remain in uniform motion unless disturbed by an external force. It involves a blend of inertial measurements, mathematics, control system design and geodesy [12]. The external force generates acceleration on the object, which can be measured by the inertial sensor. After integration of the measured accelerations, the change in velocity and position with respect to the initial conditions can be determined. A conventional inertial measurement unit (IMU) consists of three gyroscopes for measuring angular rates and three accelerometers for measuring accelerations. They are mounted in triads so that the sensitive axes of sensors are mutually orthogonal, setting up a Cartesian reference frame.

For the accelerometer, it measures the total accelerations encountered by the object, and it cannot distinguish between the accelerations caused by gravity or by inertial motions. An accelerometer at rest relative to the surface of the earth will sense the force due to gravity and the centrifugal force caused by the earth's rotation. A non-rotating accelerometer which is in free fall, accelerating at the rate of gravity, will sense nothing. Therefore, using an accelerometer, the user must compensate the specific force caused by gravity. In order to do so, the tilt of the platform, on which accelerometers are mounded in triad, with respect to the local vertical needs to be known. Therefore, in an IMU, gyroscopes are needed.

The gyroscopes are used to measure the attitude. They measure the angular rates from the body frame to the inertial frame. After integration, the changes in angle with respect to an initial condition can be determined. **Figure 1-7** is a photograph showing the size of an MEMS-based IMU (Landmark™20 eXT). The mounting of this IMU in field experiments is shown in **Figure 1-8**.

Figure 1-7. Landmark™20 eXT MEMS-based IMU

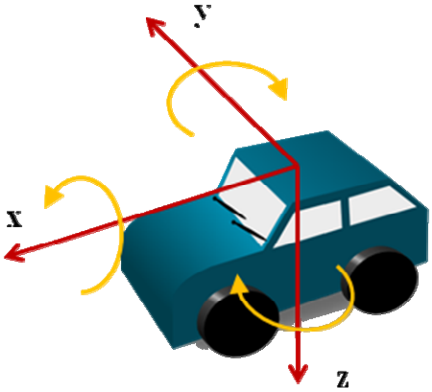


Figure 1-8. Tri-axial IMU aligned in the vehicle body frame

The IMU coordinate is usually aligned with the vehicle body frame coordinate, where the sensor x-axis often points to the forward direction, the y-axis points to the lateral direction, and the z-axis points to the vertical down direction forming a right-handed orthogonal coordinate (e.g., for Landmark™20 eXT). Different IMUs may use different coordinates.

1.3.1 INS strapdown mechanizations

For strapdown inertial sensors, the inertial sensor assembly is mounted directly on the vehicle platform. The sensor raw data are processed to yield navigation solutions (i.e., position, velocity, and attitude), and this process is named strapdown processing. A system that contains an IMU and a processing unit for computing navigation solutions is called an inertial navigation system (INS).

1.3.1.1 Inertial frame mechanization

For navigation in the vicinity of Earth, we usually derive the position, velocity and attitude estimates of the vehicle with respect to an Earth-fixed frame. However, due to the rotation of the Earth, additional apparent forces will be acting which are functions of the reference frame motion (i.e., theorem of Coriolis). Thus, the inertial velocity of vehicle $\mathbf{v}_i$ can be computed from its ground velocity with respect to earth $\mathbf{v}_e$ plus the Coriolis term $\boldsymbol{\omega}_{ie} \times \mathbf{r}$, owning to the earth rotation.

$$\mathbf{v}_i = \mathbf{v}_e + \boldsymbol{\omega}_{ie} \times \mathbf{r} \tag{1.28}$$

where $\mathbf{r}$ is the position vector of the vehicle with respect to the origin of the reference frame (i.e., center of earth); "×" denotes a vector cross product; $\boldsymbol{\omega}_{ie}$ represents the turn rate of the earth frame with respect to the inertial frame.

Differentiating Equation (1.28) with respect to inertial frame yields:

$$\dot{\mathbf{v}}_i = \dot{\mathbf{v}}_e\big|_i + \dot{\boldsymbol{\omega}}_{ie}\big|_i \times \mathbf{r} + \boldsymbol{\omega}_{ie} \times \left.\frac{d\mathbf{r}}{dt}\right|_i \tag{1.29}$$

In Equation (1.29), the $\dot{\mathbf{v}}_i$ represents the acceleration encountered by the vehicle with respect to the inertial frame. It equals to the sum of specific forces measured by the accelerometers and the force caused by the mass attraction gravitation (i.e., $\dot{\mathbf{v}}_i = \mathbf{f} + \mathbf{g}$). The $\left.\frac{d\mathbf{r}}{dt}\right|_i$ denotes the derivative of the position vector of the vehicle with respect to the origin of the inertial reference frame, which equals to $\mathbf{v}_i$. By substituting Equation (1.28) into (1.29), and assuming the earth rotation rate to be constant $\dot{\boldsymbol{\omega}}_{ie}\big|_i = 0$, we arrive at:

$$\dot{\mathbf{v}}_e\big|_i = \mathbf{f} - \boldsymbol{\omega}_{ie} \times \mathbf{v}_e + \left[\mathbf{g} - \boldsymbol{\omega}_{ie} \times (\boldsymbol{\omega}_{ie} \times \mathbf{r})\right] \tag{1.30}$$

Equation (1.30) shows that, the acceleration encountered by the vehicle with respect to Earth expressed in inertial axes is equal to the specific force measured by the accelerometers $\mathbf{f}$ compensated for the Coriolis acceleration $\boldsymbol{\omega}_{ie} \times \mathbf{v}_e$ (due to the velocity of the vehicle over a rotating Earth) plus the local gravitational acceleration, which arises through mass attraction $\mathbf{g}$ and the centripetal acceleration $\boldsymbol{\omega}_{ie} \times (\boldsymbol{\omega}_{ie} \times \mathbf{r})$ owning to the rotation of the earth.

1.3.1.2 Navigation frame mechanization

A navigation frame is often used for travelling around the Earth, where coordinates are defined in terms of north, east and down directions. For applications which cover large distances around the Earth, the rotation of the navigation frame with respect to the Earth-fixed frame (i.e., transport rate) needs to be considered. The vehicle navigation equations with respect to the navigation axes can be computed from the inertial frame mechanization equation (Equation (1.30)). That is, the ground acceleration of the vehicle expressed in the navigation frame is equal to the ground acceleration expressed in the inertial frame compensated the Coriolis terms encountered by the rotations of the navigation frame with respect to the Earth-fixed frame, and the Earth-fixed frame with respect to the inertial frame. This relationship is given in Equation (1.31).

$$\dot{\mathbf{v}}_e\big|_n = \dot{\mathbf{v}}_e\big|_i - (\boldsymbol{\omega}_{ie} + \boldsymbol{\omega}_{en}) \times \mathbf{v}_e \tag{1.31}$$

Substituting $\dot{\mathbf{v}}_e\big|_i$ from Equation (1.30), and expressing all the terms in navigation axes yields:

$$\begin{aligned}\dot{\mathbf{v}}_e^n &= \mathbf{f}^n - \boldsymbol{\omega}_{ie}^n \times \mathbf{v}_e^n + \mathbf{g}^n - \boldsymbol{\omega}_{ie}^n \times \left(\boldsymbol{\omega}_{ie}^n \times \mathbf{r}\right) - (\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \mathbf{v}_e^n \\ &= \mathbf{R}_b^n \mathbf{f}_{ib}^b - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \mathbf{v}_e^n + \mathbf{g}^n - \boldsymbol{\omega}_{ie}^n \times \left(\boldsymbol{\omega}_{ie}^n \times \mathbf{r}\right)\end{aligned} \tag{1.32}$$

where superscript “n” denotes that the quantities are expressed in the navigation frame; $\mathbf{R}_b^n$ is the direction cosine matrix used to transform the measured specific force vector into navigation axes, and $\mathbf{f}_{ib}^b$ represents the specific force measured by a perfect triad accelerometers mounted on the vehicle body frame. The second term on the right side is a correction caused by the Coriolis acceleration due to vehicle’s velocity over the surface of a rotating Earth. The third term is the apparent gravitational force acting on the vehicle. The fourth term represents the centripetal force owning to the rotation rate of the Earth.

For low-cost MEMS-based IMU and short distance navigation applications, simplifications can be made to the navigation frame mechanization model. That is, when the gyroscope errors are significantly in excess of the rotation rate of the Earth, and the accelerometer bias errors are much larger than the centripetal forces introduced by the earth rotation, we can assume $\boldsymbol{\omega}_{ie}^n = \mathbf{0}$. Besides, for short distance navigation applications, the rotation of the navigation frame with respect to the Earth-fixed frame does not need to be considered. Then we will have $\boldsymbol{\omega}_{en}^{n} = \mathbf{0}$. Taking these two aspects in consideration, Equation (1.32) can be simplified as:

$$\dot{\mathbf{v}}_{e}^{n} = \mathbf{R}_{b}^{n}\mathbf{f}_{ib}^{b} + \mathbf{g}^{n} \tag{1.33}$$

The rotation rate of body frame with respect to navigation frame can be related with gyroscope angular rate raw measurements as:

$$\begin{aligned} \boldsymbol{\omega}_{nb}^{b} &= \boldsymbol{\omega}_{ib}^{b} - \boldsymbol{\omega}_{in}^{b} \\ &= \boldsymbol{\omega}_{ib}^{b} - \mathbf{R}_{n}^{b}(\boldsymbol{\omega}_{ie}^{n} + \boldsymbol{\omega}_{en}^{n}) \end{aligned} \tag{1.34}$$

where $\boldsymbol{\omega}_{ib}^{b}$ is the angular rate measured by the body-mounted gyroscopes; $\boldsymbol{\omega}_{in}^{b}$ represents the summing rotation rates of the Earth with respect to the inertial frame plus the turn rate of navigation frame with respect to the Earth.

With ignoring the Earth rotation rate and transport rate (i.e., $(\boldsymbol{\omega}_{ie}^{n} + \boldsymbol{\omega}_{en}^{n}) = \mathbf{0}$), the following relationship exists for connecting the derivative of Euler angles $\dot{\boldsymbol{\Psi}} = \left[\dot{\alpha}, \dot{\beta}, \dot{\gamma}\right]^{T}$ and gyroscope angular rate measurements [13] as:

$$\begin{aligned} \dot{\boldsymbol{\Psi}} &= \mathbf{E}_{b}^{n} \cdot \boldsymbol{\omega}_{nb}^{b} \\ &= \mathbf{E}_{b}^{n}\left[\boldsymbol{\omega}_{ib}^{b} - \mathbf{R}_{n}^{b}(\boldsymbol{\omega}_{ie}^{n} + \boldsymbol{\omega}_{en}^{n})\right] \\ &= \mathbf{E}_{b}^{n} \cdot \boldsymbol{\omega}_{ib}^{b} \end{aligned} \tag{1.35}$$

with

$$\mathbf{E}_{b}^{n} = \begin{bmatrix} 1 & S\alpha T\beta & C\alpha T\beta \\ 0 & C\alpha & -S\alpha \\ 0 & S\alpha / C\beta & C\alpha / C\beta \end{bmatrix} \tag{1.36}$$

where CX, SX and TX represent the trigonometric operations of cosine, sine and tangent of X. The α, β, γ denote the roll, pitch and yaw respectively.

Having the derivative of Euler angles and initial attitude information, we can compute the attitude estimates (e.g., Euler angles) by using integral operations. And the body frame to navigation frame direction cosine matrix $\mathbf{R}_{b}^{n}$ can be formulated, as shown in Equation (1.37).

It is the transpose of $\mathbf{R}_n^b$, which is formed by the rotation sequence of Z-Y-X from the navigation frame to the body frame. It is worth mentioning that, different sequence of rotations yields a different presentation of direction cosine matrix.

$$\mathbf{R}_b^n = \begin{bmatrix} C\gamma C\beta & C\gamma S\beta S\alpha - S\gamma C\alpha & C\gamma S\beta C\alpha + S\gamma S\alpha \\ S\gamma C\beta & S\gamma S\beta S\alpha + C\gamma C\alpha & S\gamma S\beta C\alpha - C\gamma S\alpha \\ -S\beta & C\beta S\alpha & C\beta C\alpha \end{bmatrix} \tag{1.37}$$

From Equation (1.33) to (1.37), we can form the INS strapdown process model for the low-cost MEMS-based IMU in discrete time domain as:

$$\begin{aligned} \mathbf{p}_{k+1}^n &= \mathbf{p}_k^n + \mathbf{v}_k^n \cdot \Delta t \\ \mathbf{v}_{k+1}^n &= \mathbf{v}_k^n + \left(\mathbf{R}_{b,k}^n \cdot \mathbf{f}_{ib,k}^b + \mathbf{g}^n\right) \cdot \Delta t \\ \mathbf{\Psi}_{k+1} &= \mathbf{\Psi}_k + \mathbf{E}_{b,k}^n \cdot \boldsymbol{\omega}_{ib,k}^b \cdot \Delta t \end{aligned} \tag{1.38}$$

where k is the time instant; $\mathbf{f}_{ib,k}^b$ is the ideal triad accelerometer measurement vector; $\boldsymbol{\omega}_{ib,k}^b$ represents the ideal triad gyroscope measurement vector; $\mathbf{g}^n$ is the gravity vector without considering the local centripetal forces; $\mathbf{\Psi}_k$ is the Euler angles; $\mathbf{R}_{b,k}^n$ is the frame rotation matrix from body frame to north east and down (NED) navigation frame, and $\mathbf{E}_{b,k}^n$ is the rotation rate transformation matrix between body and navigation frame.

A flow-chart of the general strapdown mechanization process in navigation frame for all levels of inertial sensors (i.e. without simplifications) is depicted in **Figure 1-9**.

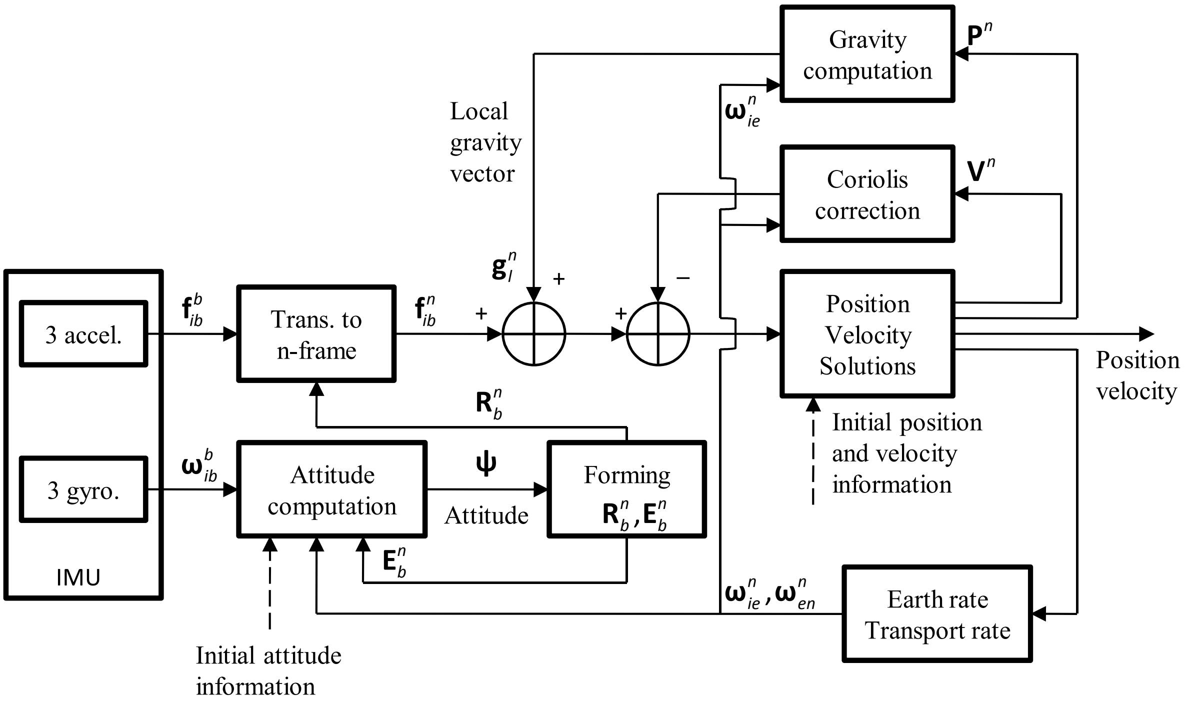


Figure 1-9. Strapdown mechanization in navigation frame.

1.4 INS/GPS integration

The GPS receiver offers long-term stable absolute positioning information with output rate at around 1 to 10 Hz. However, the system performance depends on the signal environment. It provides navigation solutions usually when more than four satellites are in view. In an INS, angular rate and specific force measurements from the IMU are processed to yield position, velocity and attitude solutions. Such systems can act autonomously and provide measurements at a higher data rate (e.g., 100 Hz). However, similar as other dead reckoning sensor systems, in an INS, the IMU sensor errors, such as sensor bias, scale factor error and noise will cause an accumulation in navigation solution errors over time. For example, the tilt errors caused by the integration of gyroscope sensor errors blur the distinction between the acceleration measured by the vehicles motion and that due to the gravity, which yields inaccurate velocity and position solutions. An integrated INS/GPS system combines the advantages of both sides and can provide accurate and uninterrupted navigation results. In such an integration system, the GPS data is used to provide absolute positioning information for frequently updating the INS estimates. And the INS data is used to provide the short-term solutions during GPS outage signal environments.

The primary methods used to fuse the INS and GPS data are the loosely-coupled or tightly-coupled integration. For the deeply-coupled (or ultra-tightly coupled) integration, the GPS receiver tracking loops need to be accessed, which is usually not provided by hardware manufacturers. Therefore, in this dissertation, it will not be discussed.

1.4.1 Loosely-coupled integration

The loosely-coupled integration has a decentralized estimation architecture, which uses the output information of the navigation solutions from a GPS receiver and an INS. One example using error states is given in **Figure 1-10**, where INS and GPS estimates (i.e., position and velocity) are compared, the resulting differences forming the measurement input to the integration Kalman filter. The filter yields estimates of the inertial sensor errors, which will be used to compensate the inertial sensor measurements. Besides, the compensation of estimated navigation solution errors from the INS estimated solution should be conducted after every Kalman filter measurement update.

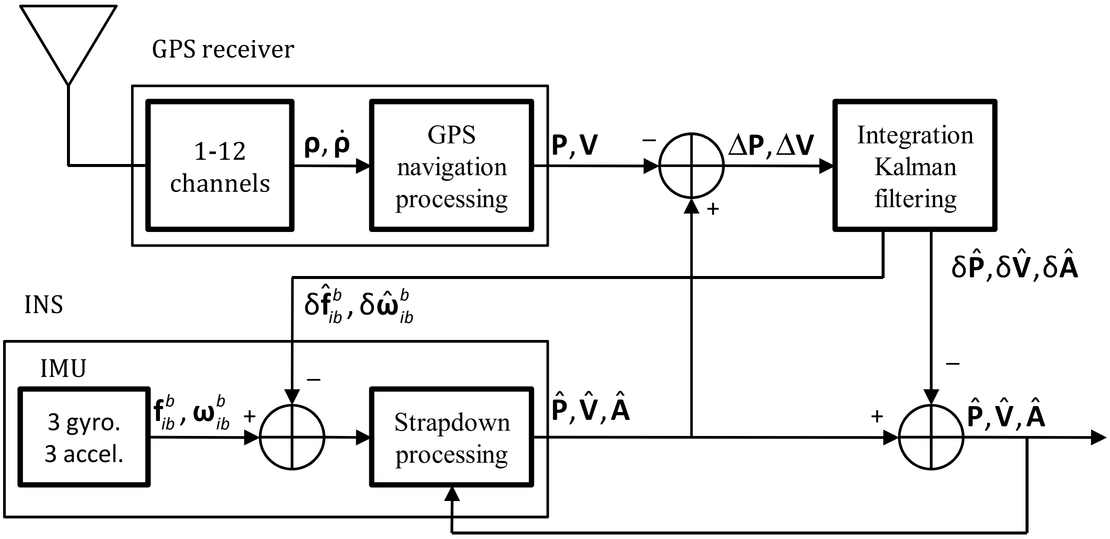


Figure 1-10. INS/GPS Loosely-coupled integration (indirect feedback)

The main advantages of using loosely-coupled integration can be summarized as:

- The system observation model is simpler with respect to that in tightly-coupled manner, and accordingly it requires much less computational burden in the integration KF.

- The number of measurement input for the Kalman filter is fixed (i.e. position and velocity).
- Redundant GPS navigation solutions are available.

The disadvantages can be summarized as:

- In case of using two separate KFs (i.e., one for GPS navigation processing, and the other for integration purpose), it opens the possibility of presenting instable navigation solutions caused by mutual feedbacks of estimation errors, which is coined as cascaded filtering problem.
- Usually more than 4 satellites are required to form and maintain a GPS navigation solution.
- For improving the system estimation accuracy, the integration KF can take advantage of covariance information from the GPS navigation processor outputs, which vary with satellite geometry. However, for many GPS receivers, these covariance data are simply not available.

1.4.2 Tightly-coupled integration

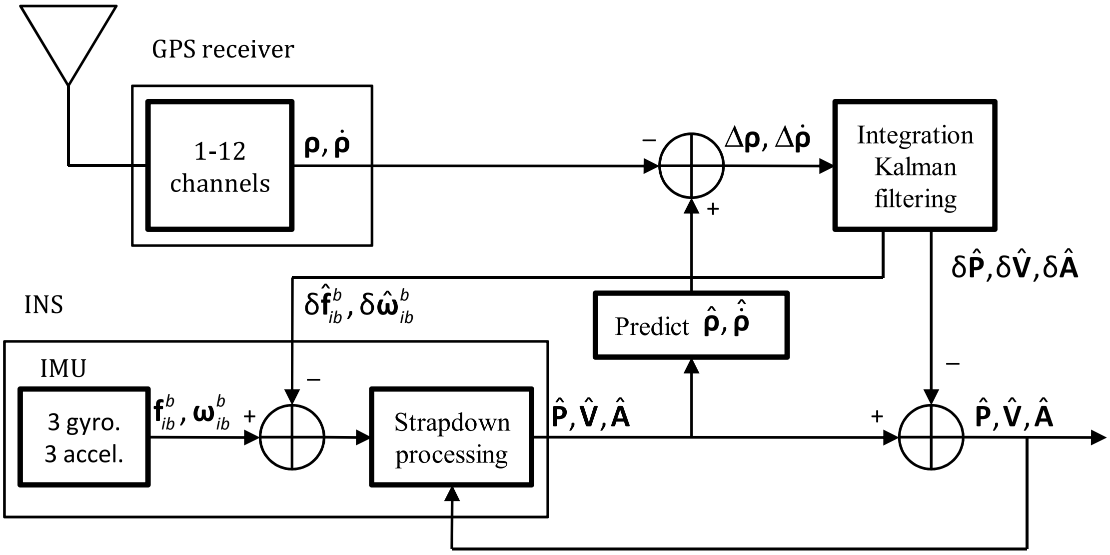


Figure 1-11. INS/GPS Tightly-coupled integration (indirect feedback)

In tightly-coupled integration, only a centralized KF is used, and the pseudorange and delta range (Doppler) measurements are directly used in the filter as shown in **Figure 1-11**. In such a case, the GPS pseudorange and delta range measurements are compared with predicted quantities made by the inertial system. The differences form the measurement input to the integration KF to generate estimates of the INS navigation errors and sensor errors, which are used to correct the inertial system estimates after each measurement update. The corrected INS navigation solution forms the integrated navigation solution. The advantages of applying the tightly-coupled integration with respect to the loosely-coupled manner are mainly in the following aspects:

- The cascaded filtering problem arising through the mutual feedbacks of the estimation errors between two separate KFs is eliminated.
- The system does not require a GPS navigation solution to aid the INS. Even less than 4 satellites in view, the remaining satellite measurements can still be used in the algorithm, which promote the robustness of the navigation system.
- All systematic errors and noise sources of the distributed sensors are modeled in the same filter, which ensures that all error correlations are accounted for.

The disadvantage of this method arises in the increased dimension of the observation vector. And the number of GPS measurements as input for the KF is varying during the navigation applications, which bases on the signal environments.

1.4.3 INS/GPS state space models using error states

In the integration of measurements coming from a single GPS receiver antenna and an INS without redundant attitude information (e.g., from magnetometers, or multi-antenna GPS systems), the INS attitude (i.e., heading) and sensor bias errors are weakly observable, which are based on the dynamics of the platform [14-17]. They are essentially corrected by the GPS updates through the off-diagonal parameters in their error covariance matrices in the filter. It is the nonlinear system model that relates the attitude errors, sensor bias errors with the position and velocity errors. Therefore, the nonlinearity of the INS/GPS integrated system model should be carefully treated. The most common application of the KF to nonlinear systems is the extended Kalman filter (EKF), which will be used in this chapter.

1.4.3.1 System process model based on the first-order linearization

In this section, we derive the models from the first-order linearization of nonlinear equations. The nonlinear system dynamic equations in navigation frame for low-cost MEMS-based IMU are given in Equation (1.39).

$$\begin{aligned}
\mathbf{p}_{k+1}^{n} &= \mathbf{p}_{k}^{n} + \mathbf{v}_{k}^{n} \cdot \Delta t \\
\mathbf{v}_{k+1}^{n} &= \mathbf{v}_{k}^{n} + \left[\mathbf{R}_{b,k}^{n} \cdot \hat{\mathbf{f}}_{ib,k}^{b} + \mathbf{g}^{n} \right] \cdot \Delta t \\
\mathbf{\Psi}_{k+1} &= \mathbf{\Psi}_{k} + \mathbf{E}_{b,k}^{n} \cdot \hat{\boldsymbol{\omega}}_{ib,k}^{b} \cdot \Delta t \\
\mathbf{f}_{ib,k+1}^{b,error} &= \mathbf{f}_{ib,k}^{b,error} + \mathbf{w}_{\mathbf{f},k} \\
\boldsymbol{\omega}_{ib,k+1}^{b,error} &= \boldsymbol{\omega}_{ib,k}^{b,error} + \mathbf{w}_{\boldsymbol{\omega},k}
\end{aligned} \tag{1.39}$$

where $\mathbf{p}_{k+1}^{n}$ denotes position in navigation frame at epoch k+1; $\mathbf{v}_{k+1}^{n}$ is the velocity in navigation frame at epoch k+1; $\mathbf{\Psi}_{k+1}$ is the attitude which includes roll, pitch and yaw estimates at epoch k+1; $\mathbf{f}_{ib,k+1}^{b,error}$ and $\boldsymbol{\omega}_{ib,k+1}^{b,error}$ are the accelerometer and gyroscope sensor errors at epoch k+1, expressed in body frame.

Here we define $\hat{\mathbf{f}}_{ib,k}^{b} = \tilde{\mathbf{f}}_{ib,k}^{b} - \mathbf{f}_{ib,k}^{b,error}$ and $\hat{\boldsymbol{\omega}}_{ib,k}^{b} = \tilde{\boldsymbol{\omega}}_{ib,k}^{b} - \boldsymbol{\omega}_{ib,k}^{b,error}$. The linearization is conducted at local (i.e., k-th epoch) estimated position, velocity, attitude, accelerometer and gyroscope sensor errors. The results are given in Equation (1.40). The error states are formed from the linearization process.

$$\begin{bmatrix} \delta\mathbf{p}_{k+1}^{n} \\ \delta\mathbf{v}_{k+1}^{n} \\ \delta\mathbf{\Psi}_{k+1} \\ \delta\mathbf{f}_{ib,k+1}^{b,error} \\ \delta\boldsymbol{\omega}_{ib,k+1}^{b,error} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3}\Delta t & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} \\ \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{F}_{23,k} & -\mathbf{R}_{b,k}^{n}\Delta t & \mathbf{O}_{3\times3} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{F}_{33,k} & \mathbf{O}_{3\times3} & -\mathbf{E}_{b,k}^{n}\Delta t \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{O}_{3\times3} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{p}_{k}^{n} \\ \delta\mathbf{v}_{k}^{n} \\ \delta\mathbf{\Psi}_{k} \\ \delta\mathbf{f}_{ib,k}^{b,error} \\ \delta\boldsymbol{\omega}_{ib,k}^{b,error} \end{bmatrix} + \mathbf{w}_{k} \tag{1.40}$$

where $\mathbf{R}_{b,k}^{n}$ and $\mathbf{E}_{b,k}^{n}$ are the transformation matrices shown in Equation (1.36) and (1.37) at the time instance k. The sub-matrices $\mathbf{F}_{23,k}$ and $\mathbf{F}_{33,k}$ are computed in Equation (1.41) and (1.42).

$$\mathbf{F}_{23,k}=\left[\begin{array}{c|c|c} \begin{array}{l}\hat{f}_y(S\alpha S\gamma+C\alpha C\gamma S\beta)\\ +\hat{f}_z(C\alpha S\gamma+C\gamma S\beta S\alpha)\end{array} & \begin{array}{l}\hat{f}_z C\beta C\alpha C\gamma\\ -\hat{f}_x C\gamma S\beta\\ +\hat{f}_y C\beta C\gamma S\alpha\end{array} & \begin{array}{l}\hat{f}_z(C\gamma S\alpha+C\alpha S\beta S\gamma)\\ -\hat{f}_y(C\alpha C\gamma-S\beta S\alpha S\gamma)\\ -\hat{f}_x C\beta S\gamma\end{array} \\ \hline \begin{array}{l}-\hat{f}_z(S\beta S\alpha S\gamma+C\alpha C\gamma)\\ -\hat{f}_y(C\gamma S\alpha-C\alpha S\beta S\gamma)\end{array} & \begin{array}{l}\hat{f}_z C\beta C\alpha S\gamma\\ -\hat{f}_x S\beta S\gamma\\ +\hat{f}_y C\beta S\alpha S\gamma\end{array} & \begin{array}{l}\hat{f}_z(S\alpha S\gamma+C\alpha C\gamma S\beta)\\ -\hat{f}_y(C\alpha S\gamma-C\gamma S\beta S\alpha)\\ +\hat{f}_x C\beta C\gamma\end{array} \\ \hline \hat{f}_y C\beta C\alpha-\hat{f}_z C\beta S\alpha & \begin{array}{l}-\hat{f}_x C\beta\\ -\hat{f}_z C\alpha S\beta\\ -\hat{f}_y S\beta S\alpha\end{array} & 0 \end{array}\right]_{3\times 3}\cdot\Delta t \tag{1.41}$$

and

$$\mathbf{F}_{33,k}=\left[\begin{array}{c|c|c} 1+\left(\begin{array}{l}\hat{\omega}_y C\alpha T\beta\\ +\hat{\omega}_z T\beta S\alpha\end{array}\right)\Delta t & \left[\begin{array}{l}\hat{\omega}_z C\alpha(T^2\beta+1)\\ +\hat{\omega}_y S\alpha(T^2\beta+1)\end{array}\right]\Delta t & 0 \\ \hline -\left[\begin{array}{l}\hat{\omega}_z C\alpha\\ +\hat{\omega}_y S\alpha\end{array}\right]\Delta t & 1 & 0 \\ \hline \left[\dfrac{\hat{\omega}_y C\alpha-\hat{\omega}_z S\alpha}{C\beta}\right]\Delta t & \left[\dfrac{C\alpha S\beta\hat{\omega}_z+S\beta S\alpha\hat{\omega}_y}{C^2\beta}\right]\Delta t & 1 \end{array}\right]_{3\times 3} \tag{1.42}$$

where C^2X, T^2X represent the square of cosine and tangent operations of X, and $\hat{f}_x=\tilde{f}^b_{ib,x}-f^{b,error}_{ib,x}$, $\hat{f}_y=\tilde{f}^b_{ib,y}-f^{b,error}_{ib,y}$, $\hat{f}_z=\tilde{f}^b_{ib,z}-f^{b,error}_{ib,z}$, $\hat{\omega}_y=\tilde{\omega}^b_{ib,y}-\omega^{b,error}_{ib,y}$, $\hat{\omega}_z=\tilde{\omega}^b_{ib,z}-\omega^{b,error}_{ib,z}$ are the IMU raw data compensated with the estimated sensor errors at time instance k.

Equation (1.40) can be used as the system process model for the loosely-coupled INS/GPS integration. However, for the tightly-coupled approach, the receiver clock bias and clock drift errors need to be modeled. We form the system process model for tightly-coupled integration in Equation (1.43), where the range-rate equivalent of the clock drift error is modeled as a constant plus a random walk process, while the range equivalent of the receiver clock bias error is the integral of the clock drift error.

$$\begin{bmatrix} \delta\mathbf{p}^n_{k+1} \\ \delta\mathbf{v}^n_{k+1} \\ \delta\mathbf{\Psi}_{k+1} \\ \delta\mathbf{f}^{b,error}_{ib,k+1} \\ \delta\boldsymbol{\omega}^{b,error}_{ib,k+1} \\ c\delta t_{k+1} \\ c\delta\dot{t}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3}\Delta t & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{F}_{23,k} & -\mathbf{R}^n_{b,k}\Delta t & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{F}_{33,k} & \mathbf{O}_{3\times3} & -\mathbf{E}^n_{b,k}\Delta t & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 1 & \Delta t \\ \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{p}^n_{k} \\ \delta\mathbf{v}^n_{k} \\ \delta\mathbf{\Psi}_{k} \\ \delta\mathbf{f}^{b,error}_{ib,k} \\ \delta\boldsymbol{\omega}^{b,error}_{ib,k} \\ c\delta t_{k} \\ c\delta\dot{t}_{k} \end{bmatrix} + \mathbf{w}_k \tag{1.43}$$

1.4.3.2 *Observation model for loosely-coupled and tightly-coupled integration*

For the loosely-coupled integration, the position and velocity measurements from the GPS receiver are compared with the INS estimated quantities, forming the new measurement input for the integration KF. Its observation model is given as:

$$\mathbf{y}_k = \begin{bmatrix} \hat{\mathbf{p}}^n_k - \tilde{\mathbf{p}}^n_k \\ \hdashline \hat{\mathbf{v}}^n_k - \tilde{\mathbf{v}}^n_k \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} \\ \hdashline \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} \end{bmatrix}_{6\times15} \cdot \begin{bmatrix} \delta\mathbf{p}^n_{k} \\ \delta\mathbf{v}^n_{k} \\ \delta\mathbf{\Psi}_{k} \\ \delta\mathbf{f}^{b,error}_{ib,k} \\ \delta\boldsymbol{\omega}^{b,error}_{ib,k} \end{bmatrix} + \boldsymbol{\varepsilon}_k \tag{1.44}$$

In the tightly-coupled integration, the system observation model is formulated in Equation (1.45), which is the combination of Equation (1.12) and (1.27) introduced in former sections.

$$\mathbf{y}_k = \begin{bmatrix} \hat{\rho}_{1,k} - \tilde{\rho}_{1,k} \\ \vdots \\ \hat{\rho}_{j,k} - \tilde{\rho}_{j,k} \\ \hdashline \hat{\dot{\rho}}_{1,k} - \tilde{\dot{\rho}}_{1,k} \\ \vdots \\ \hat{\dot{\rho}}_{j,k} - \tilde{\dot{\rho}}_{j,k} \end{bmatrix} = \begin{bmatrix} \left(-\mathbf{l}^T_{1,k}\right)_{1\times3} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \left(-\mathbf{l}^T_{j,k}\right)_{1\times3} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 1 & 0 \\ \hdashline \mathbf{0}^T_{3\times1} & \left(-\mathbf{l}^T_{1,k}\right)_{1\times3} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0}^T_{3\times1} & \left(-\mathbf{l}^T_{j,k}\right)_{1\times3} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & \mathbf{0}^T_{3\times1} & 0 & 1 \end{bmatrix}_{2j\times17} \cdot \begin{bmatrix} \delta\mathbf{p}^n_{k} \\ \delta\mathbf{v}^n_{k} \\ \delta\mathbf{\Psi}_{k} \\ \delta\mathbf{f}^{b,error}_{ib,k} \\ \delta\boldsymbol{\omega}^{b,error}_{ib,k} \\ c\delta t_{k} \\ c\delta\dot{t}_{k} \end{bmatrix} + \boldsymbol{\varepsilon}_k \tag{1.45}$$

where ‘j’ denotes the number of satellites in view.

By looking at the dimension of the observation matrix in Equation (1.44) and (1.45), we observe that in the tightly-coupled integration, the dimension of observation matrix varies during the time, which is based on the number of tracked satellites. If a large number of satellites are in view, the observation matrix will be bulky and causes large computational burden. For instance, if 11 satellites are in view, the observation matrix has the dimension of 22x17. However, for the loosely-coupled integration, the dimension of the observation matrix is fixed (i.e., 6x15), which simplifies the implementation and reduces the computational burden.

1.5 Field experiment

In order to verify the algorithm from the INS/GPS integrated system, an experiment based on a train ride is made. We consider the tightly-coupled integration (due to its advantages as introduced in Section 1.4.2) as the approach to blend the INS and GPS data. The trajectory experienced frequent GPS outage environments (i.e., going through tunnels). The path starts from the city of Betzdorf to Siegen in the North West of Germany. The trajectory lasts 800 s, as shown in **Figure 1-12**. The number of tracked satellites is given in **Figure 1-13**.

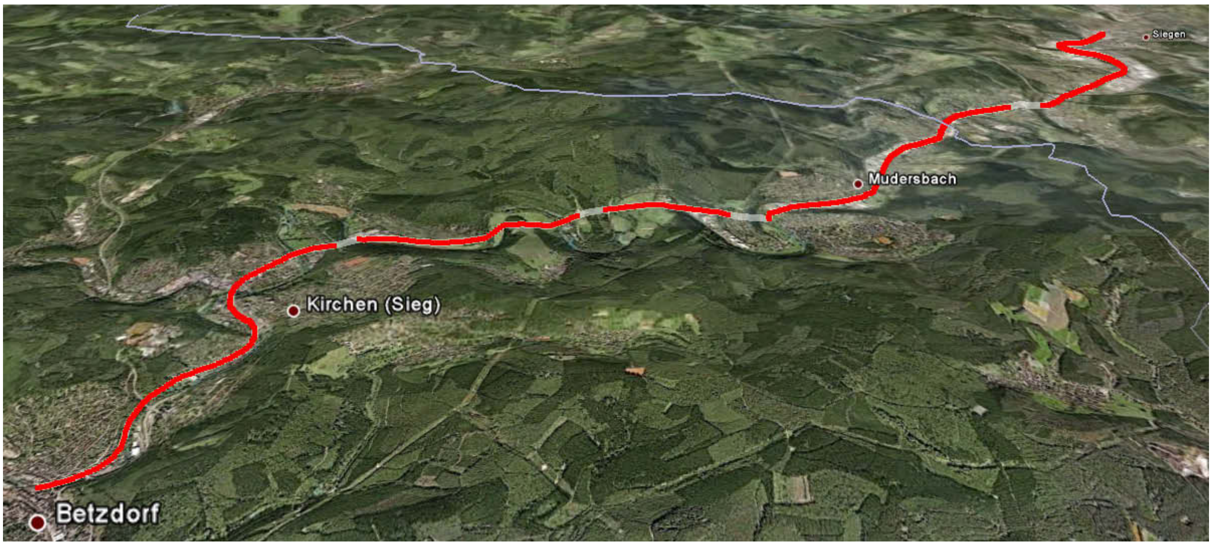


Figure 1-12. A train ride trajectory computed from processing the L1 GPS pseudorange measurements using a least-squares estimation method (plotted in Google earth)

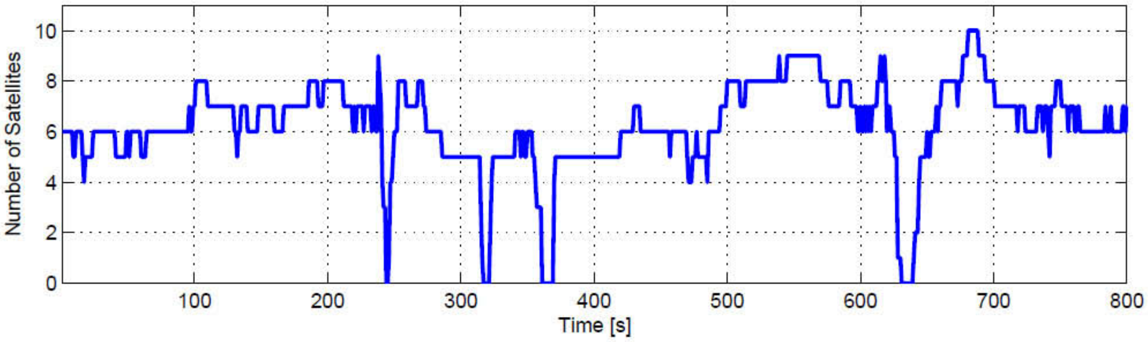


Figure 1-13. Number of satellites in view.

One Landmark™20 eXT MEMS-based IMU (100 Hz) and one u-blox Antaris 4 (1 Hz) GPS receiver are used. The main sensor errors of the Landmark™20 eXT MEMS-based IMU are given in **Table 1-5**.

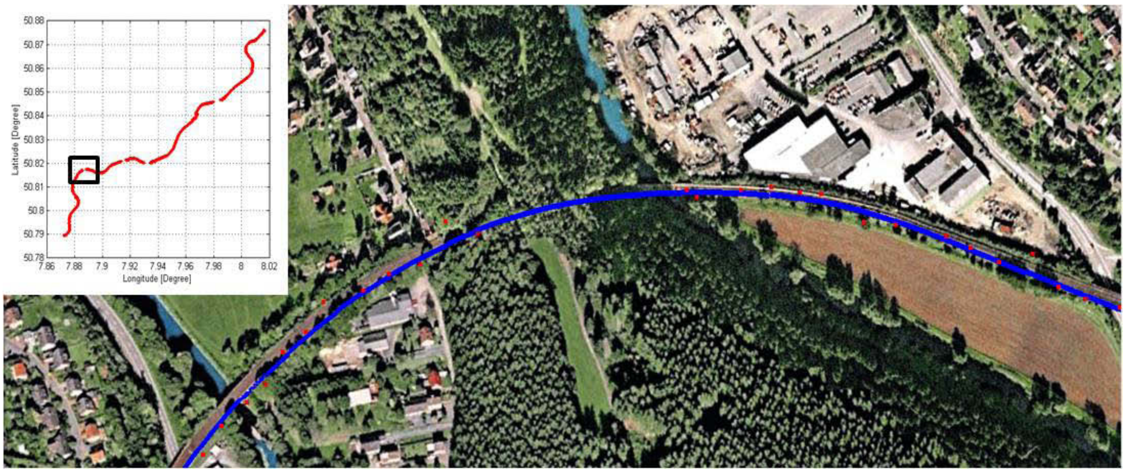
Table 1-5. Landmark™20 eXT MEMS-based IMU performance specification.

Gyroscope	Bias in-run stability	Noise (ARW)	Scale Factor Error
(Angular rates)	20 [°/h] (1σ)	0.035[°/s/√Hz] (1σ)	≤1000 [ppm]
Accelerometer	Bias in-run stability	Noise (VRW)	Scale Factor Error
(Specific forces)	20 [μg] (1σ)	40 [μg/√Hz] (1σ)	≤1000 [ppm]

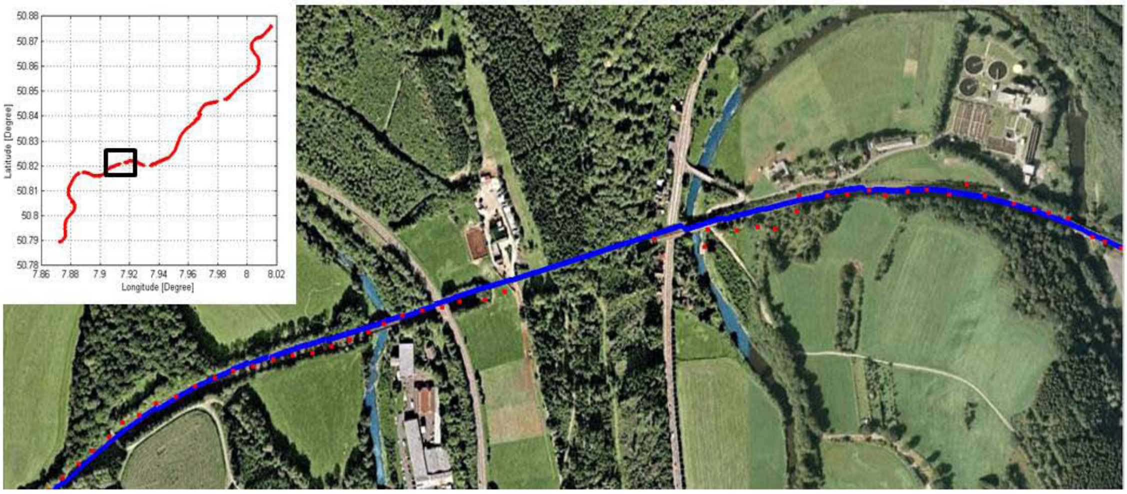
The train went through tunnels 4 times during the trajectory. The durations of the GPS outage environments are given in **Table 1-6**. The system performances from tightly-coupled integration (blue curves) and GPS alone navigation solutions based on least-squares estimation method (red points) are depicted in **Figure 1-14**. In these periods, the INS bridges the GPS outages, and presents continuous and robust positioning results.

Table 1-6. GPS outage environments

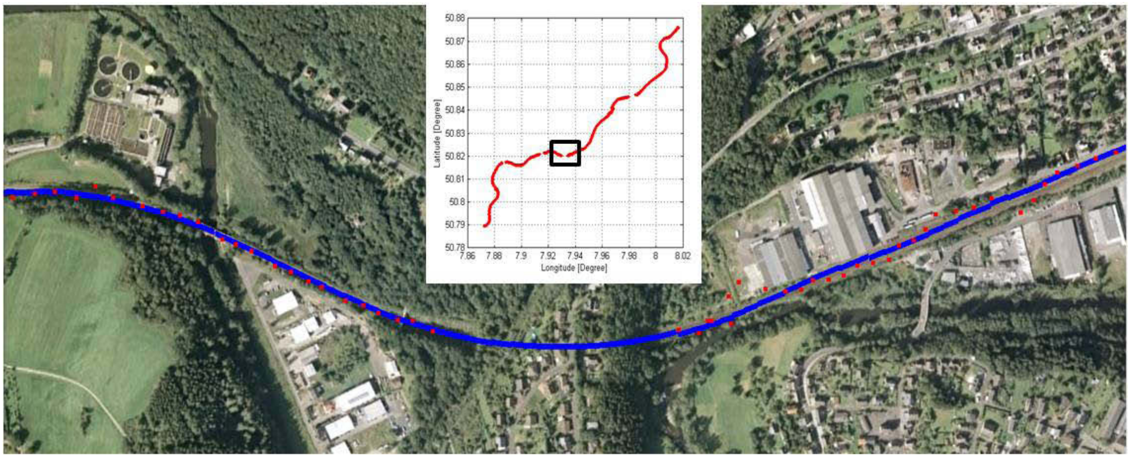
GPS outages	< 4 satellites in view	no satellites in view
a)	5 Seconds	2 Seconds
b)	8 Seconds	5 Seconds
c)	13 Seconds	9 Seconds
d)	16 Seconds	9 Seconds
Sum	42 Seconds	25 Seconds



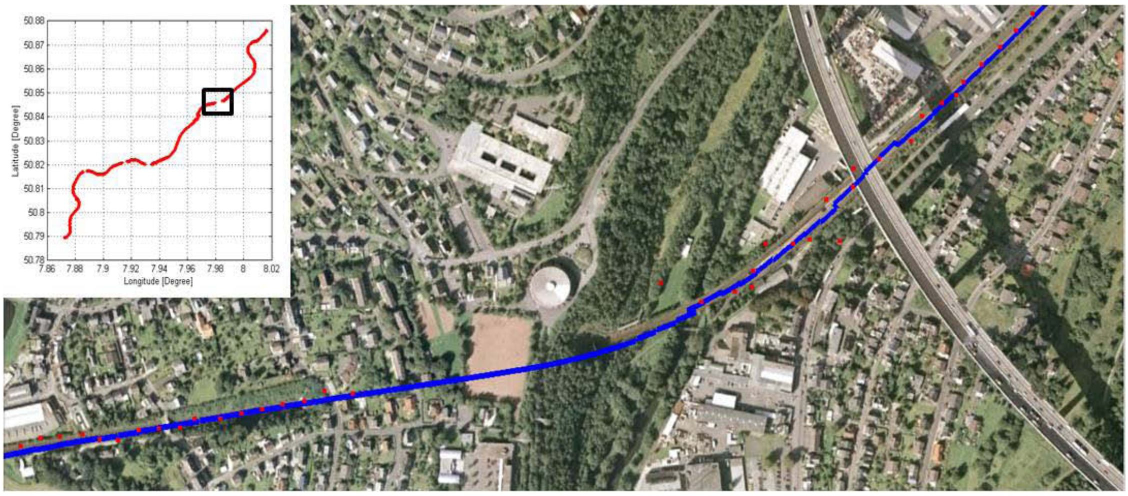
a) GPS outage with 2 s (no Sat.), blue (integrated), red (GPS alone)



b) GPS outage with 5 s (no Sat.), blue (integrated), red (GPS alone)



c) GPS outage with 9 s (no Sat.), blue (integrated), red (GPS alone)



d) GPS outage with 9 s (no Sat.), blue (integrated), red (GPS alone)

Figure 1-14. System performance comparison during tunnels (plotted in Google earth)

Other than GPS outage environments, the multipath effects will significantly influence the GNSS positioning accuracy (e.g., caused by the elevated road and large building nearby, as shown in **Figure 1-14** (d)). In this case, the INS estimates can smooth the noisy GNSS measurement so that the INS/GPS integrated solution presents a more accurate and robust navigation solution.

1.6 Summary

In this chapter, the GPS data processing, INS principles and INS strapdown processing have been introduced in detail. For the INS/GPS integration, the main advantages and disadvantages from the loosely-coupled and tightly-coupled integrations were discussed. The system process and observation models for both approaches were given. the model is derived from a first-order linearization process.

One field experiment has been made based on a train ride to verify the advantage of INS/GPS integrated system with respect to GPS alone devices. Numerical results show that in GPS outage and highly reflective signal environments the integrated solution presents more accurate and robust navigation performance.

In the following chapters, more advanced nonlinear filtering approaches will be applied on INS/GPS integration. Furthermore, quaternions will be employed as the representation of attitude, which does not exhibit the singularity problem.

2. Nonlinear Filtering Methods

2.1 Introduction

"Filtering consists of recursively estimating and tracking of states governed by a linear or nonlinear stochastic system state space model based on a set of noisy observation data" [18]. Navigation is one of many areas, in which estimation theory can be successfully applied. In the navigation field, different methodologies will be used to estimate the time varying position, velocity and attitude of moving objects on the basis of measurements derived from navigation sensor systems [13].

In order to estimate the system dynamics recursively, one has to define a system propagation model, which describes the evolution of state over time, and an observation model, which relates the incoming measurements to the estimated states.

In the Bayesian recursive estimation approach, all the states are regarded as random variables which can be completely described by their probability distributions. Thus, the estimation problem turns to be the recursive quantifying of the *a posteriori* state density function given all available observables. And hence, the conditional probability density functions of random variables will be of primary importance in following derivations. In the following sections, a few important basic concepts will be overviewed in the probability theory, and then the general recursive Bayesian filter will be introduced.

The general recursive Bayesian filter provides a conceptual solution for nonlinear non-Gaussian applications. In order to apply it on INS/GPS integration, assumptions should be made, which leads to the UKF and PF algorithms.

As one contribution of this thesis, a UPF algorithm is proposed, which combines the best of UKF and PF. A simulation test is made to show its merits with respect to other nonlinear filtering approaches.

2.2 Basics in probability theory

Given a random variable vector $\mathbf{x}=[x_1,x_2,.....x_n]^T$, the probability distribution function F_x can be formulated as a scalar function of the dummy vector $\boldsymbol{\zeta}=[\zeta_1,\zeta_1,\cdots\zeta_n]^T$ as:

$$\begin{aligned}F_{\mathbf{x}}(\boldsymbol{\zeta})&=F_{x_1,x_2,\cdots,x_n}(\zeta_1,\zeta_2,\cdots\zeta_n)\\&=P\left(\left\{\omega:x_1(\omega)\le\zeta_1,x_2(\omega)\le\zeta_2,\cdots,x_n(\omega)\le\zeta_n\right\}\right)\end{aligned}\tag{2.1}$$

where ω denotes the elementary outcome of the experiment in sample space; $\mathbf{x}(\omega)$ is the realization of the random variable mapping ω from sample space to real values in Euclidean space $\mathbb{R}^n$; $\boldsymbol{\zeta}$ is a dummy vector consists of real values for integration; $P(\cdot)$ is the probability function which assigns the probabilities to the event described in the parenthesis.

It can be calculated as:

$$\begin{aligned}F_{\mathbf{x}}(\boldsymbol{\zeta})&=F_{x_1,x_2,\cdots,x_n}(\zeta_1,\zeta_2,\cdots,\zeta_n)\\&=\int_{-\infty}^{\zeta_1}\int_{-\infty}^{\zeta_2}\cdots\int_{-\infty}^{\zeta_n}f_{x_1,x_2,.....x_n}(\xi_1,\xi_2,\cdots,\xi_n)\,d\xi_1d\xi_2\cdots d\xi_n\end{aligned}\tag{2.2}$$

where $\boldsymbol{\xi}=[\xi_1,\xi_2,\cdots,\xi_n]^T$ is a dummy vector for random variable $\mathbf{x}$, which is used for integration.

Equation (2.2) can be formulated in its vector expression for convenience as:

$$F_{\mathbf{x}}(\boldsymbol{\zeta})=\int_{-\infty}^{\boldsymbol{\zeta}}f_{\mathbf{x}}(\boldsymbol{\xi})\,d\boldsymbol{\xi}\tag{2.3}$$

where $f_{\mathbf{x}}(\boldsymbol{\xi})$ is called the probability density function.

In the stochastic modeling and estimation, we often need to estimate the random variable vector $\mathbf{x}$ given the measurement vector $\mathbf{y}$ at its specific realization $\boldsymbol{\gamma}$. The concept of the conditional expectation arises, which is calculated as:

$$E[\mathbf{x}\,|\,\mathbf{y}=\boldsymbol{\gamma}]=\int_{-\infty}^{+\infty}\boldsymbol{\xi}\cdot f_{\mathbf{x}|\mathbf{y}}(\boldsymbol{\xi}\,|\,\boldsymbol{\gamma})\,d\boldsymbol{\xi}\tag{2.4}$$

Naturally, the expectation of certain functions of a random variable vector $\mathbf{z}=g(\mathbf{x})$ on the basis of the measurement vector $\mathbf{y}$ at its specific realization of $\boldsymbol{\gamma}$ is formulated as:

$$E[\mathbf{z} \mid \mathbf{y} = \boldsymbol{\gamma}] = \int_{-\infty}^{+\infty} g(\boldsymbol{\xi}) \cdot f_{\mathbf{x}|\mathbf{y}}(\boldsymbol{\xi} \mid \boldsymbol{\gamma})\, d\boldsymbol{\xi} \tag{2.5}$$

For a Gaussian distributed random variable vector $\mathbf{x} = [x_1, x_2, ...x_n]^T$, its probability density function is formulated as:

$$f_{\mathbf{x}}(\boldsymbol{\xi}) = \frac{1}{(2\pi)^{n/2} \left|\mathbf{P}_{\mathbf{x}}\right|^{1/2}} \exp\left\{ -\frac{1}{2} [\boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}}]^T \, \mathbf{P}_{\mathbf{x}}^{-1} [\boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}}] \right\} \tag{2.6}$$

where $\mathbf{m}_{\mathbf{x}}$ and $\mathbf{P}_{\mathbf{x}}$ are the mean vector and covariance matrix of random variable vector $\mathbf{x}$; n is the dimension of the random variable vector; $|\cdot|$ denotes the determinant of a matrix, and $\exp\{\cdot\}$ denotes the exponential function.

Because the probability density function of a Gaussian random variable can be completely described by its mean and covariance parameters. Therefore, a shorthand notation is often used as:

$$\mathbf{x} \sim \mathcal{N}(\mathbf{m}_{\mathbf{x}}, \mathbf{P}_{\mathbf{x}}) \tag{2.7}$$

The joint probability density function of Gaussian random variable vectors $\mathbf{x} = [x_1, x_2, ...x_n]^T$ and $\mathbf{y} = [y_1, y_2, ...y_m]^T$ can be written as:

$$f_{\mathbf{x},\mathbf{y}}(\boldsymbol{\xi}, \boldsymbol{\gamma}) = \frac{1}{(2\pi)^{(n+m)/2} \left\| \left[\begin{array}{c|c} \mathbf{P}_{\mathbf{xx}} & \mathbf{P}_{\mathbf{xy}} \\ \hline \mathbf{P}_{\mathbf{yx}} & \mathbf{P}_{\mathbf{yy}} \end{array}\right] \right\|^{1/2}} \exp\left\{ -\frac{1}{2} \left[\begin{array}{c} \boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}} \\ \hline \boldsymbol{\gamma} - \mathbf{m}_{\mathbf{y}} \end{array}\right]^T \left[\begin{array}{c|c} \mathbf{P}_{\mathbf{xx}} & \mathbf{P}_{\mathbf{xy}} \\ \hline \mathbf{P}_{\mathbf{yx}} & \mathbf{P}_{\mathbf{yy}} \end{array}\right]^{-1} \left[\begin{array}{c} \boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}} \\ \hline \boldsymbol{\gamma} - \mathbf{m}_{\mathbf{y}} \end{array}\right] \right\} \tag{2.8}$$

where $\boldsymbol{\xi}$ and $\boldsymbol{\gamma}$ are dummy vectors which denote the realizations of random vectors **x** and **y**; $\mathbf{m}_{\mathbf{x}}, \mathbf{m}_{y}$ are their expectations respectively.

According to Bayes' rule, $f_{\mathbf{x}|\mathbf{y}}(\boldsymbol{\xi} \mid \boldsymbol{\gamma}) = f_{\mathbf{x},\mathbf{y}}(\boldsymbol{\xi}, \boldsymbol{\gamma}) / f_{\mathbf{y}}(\boldsymbol{\gamma})$, the conditional density function $f_{\mathbf{x}|\mathbf{y}}(\boldsymbol{\xi} \mid \boldsymbol{\gamma})$ is computed as:

$$f_{\mathbf{x}|\mathbf{y}}(\boldsymbol{\xi} \mid \boldsymbol{\gamma}) = \frac{1}{(2\pi)^{n/2} \left|\mathbf{P}_{\mathbf{x}|\mathbf{y}}\right|^{1/2}} \exp\left\{ -\frac{1}{2} \left[\boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}|\mathbf{y}}\right]^T \mathbf{P}_{\mathbf{x}|\mathbf{y}}^{-1} \left[\boldsymbol{\xi} - \mathbf{m}_{\mathbf{x}|\mathbf{y}}\right] \right\} \tag{2.9}$$

where

$$\begin{aligned}\mathbf{m}_{\mathbf{x}|\mathbf{y}} &= \mathbf{m}_{\mathbf{x}} + \mathbf{P}_{\mathbf{xy}}\mathbf{P}_{\mathbf{yy}}^{\mathbf{-1}}\left(\boldsymbol{\gamma} - \mathbf{m}_{\mathbf{y}}\right) \\ \mathbf{P}_{\mathbf{x}|\mathbf{y}} &= \mathbf{P}_{\mathbf{xx}} - \mathbf{P}_{\mathbf{xy}}\mathbf{P}_{\mathbf{yy}}^{\mathbf{-1}}\mathbf{P}_{\mathbf{yx}}\end{aligned} \tag{2.10}$$

In Equation (2.10), if we assume $\mathbf{x}$ represents the random variable vector of interest, and $\mathbf{y}$ denotes the new information coming from the external sensor measurements into the system. The $f_{\mathbf{x}|\mathbf{y}}\left(\boldsymbol{\xi}\,|\,\boldsymbol{\gamma}\right)$ represents the conditional density for the random variable vector conditioned on the information obtained from the sensor output.

If we assign $\mathbf{K} = \mathbf{P}_{\mathbf{xy}}\mathbf{P}_{\mathbf{yy}}^{-1}$, we have:

$$\begin{aligned}\mathbf{m}_{\mathbf{x}|\mathbf{y}} &= \mathbf{m}_{\mathbf{x}} + \mathbf{K}\left(\boldsymbol{\gamma} - \mathbf{m}_{\mathbf{y}}\right) \\ \mathbf{P}_{\mathbf{x}|\mathbf{y}} &= \mathbf{P}_{\mathbf{xx}} - \mathbf{K}\mathbf{P}_{\mathbf{yy}}\mathbf{K}^{T}\end{aligned} \tag{2.11}$$

They are the Kalman filter equations derived from the conditional probability density function based on Gaussian distributions.

2.3 Recursive Bayesian state estimator

Linear systems do practically not really exist. They are approximations within a limited range in practical applications. For many problems, an estimate is required every time when a measurement is received. In this case, a recursive filter is a convenient solution. A recursive Bayesian approach to dynamic state estimation problems involves the construction of the probability function of the current state for an evolving system, given the accumulated observation history. It provides a rigorous general framework for dynamic state estimation problems [19]. In order to derive the recursive Bayesian equations, we assume that the discrete-time stochastic system state space models are as follows:

$$\begin{aligned}\mathbf{x}_k &= f_{k-1}(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}) \\ \mathbf{y}_k &= h_k(\mathbf{x}_k, \mathbf{v}_k)\end{aligned} \tag{2.12}$$

where k is the time instant; $\mathbf{x}_k$ is the system state vector; $\mathbf{y}_k$ represents the measurement vector; $\mathbf{w}_k$ and $\mathbf{v}_k$ are the system process and measurement noise vectors, which are assumed to be independent and white with known probability density functions; $f_{\mathrm{k}}(.)$ and $h_{\mathrm{k}}(.)$ are the time varying nonlinear stochastic system and observation models.

As shown in Equation (2.12), in order to analyze and make inference about a dynamic system, at least two models are required. The system process model links the current state vector to the previous state vector, which provides the predictive conditional density of the current state given the previous *a posteriori* state. The observation model relates the observation data to the current state vector, which specifies the likelihood density function of the current observation vector given the current states. The objective of the Bayesian estimator is to derive the *a posteriori* probability density function of $\mathbf{x}_k$ conditioned upon the measurements at the current and all previous epochs, $f_{\mathbf{x}_k|\mathbf{y}_1,\mathbf{y}_{2,}\cdots,\mathbf{y}_k}(\boldsymbol{\xi}_k \,|\, \boldsymbol{\gamma}_1,\boldsymbol{\gamma}_{2,}\cdots,\boldsymbol{\gamma}_k)$. This density embodies all of the information used for the estimation purpose. For simplicity, it is denoted as $f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \,|\, \boldsymbol{\gamma}_{1:k})$, where $\mathbf{Y}_k \sim \mathbf{y}_1,\mathbf{y}_{2,}\cdots,\mathbf{y}_k$, and $\boldsymbol{\gamma}_{1:k}$ represents the realization of $\mathbf{Y}_k$.

In order to propagate this probability density function from one time instant to the next, we need to treat it in a recursive way. The *a priori* density function $f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \,|\, \boldsymbol{\gamma}_{1:k-1})$ can be formulated as:

$$f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \,|\, \boldsymbol{\gamma}_{1:k-1}) = \int_{-\infty}^{+\infty} f_{(\mathbf{x}_k,\mathbf{x}_{k-1})|\mathbf{Y}_{k-1}}\left[(\boldsymbol{\xi}_k,\boldsymbol{\xi}_{k-1}) \,|\, \boldsymbol{\gamma}_{1:k-1}\right] d\boldsymbol{\xi}_{k-1} \tag{2.13}$$

According to the Bayes's rule, the following relationship exists, where the dummy parameters are omitted for simplification.

$$\begin{aligned} f_{(\mathbf{x}_k,\mathbf{x}_{k-1})|\mathbf{Y}_{k-1}} &= \frac{f_{\mathbf{x}_k,\mathbf{x}_{k-1},\mathbf{Y}_{k-1}}}{f_{\mathbf{Y}_{k-1}}} \\ &= \frac{f_{\mathbf{x}_k,\mathbf{x}_{k-1},\mathbf{Y}_{k-1}}}{f_{\mathbf{x}_{k-1},\mathbf{Y}_{k-1}}} \cdot \frac{f_{\mathbf{x}_{k-1},\mathbf{Y}_{k-1}}}{f_{\mathbf{Y}_{k-1}}} \\ &= f_{\mathbf{x}_k|(\mathbf{x}_{k-1},\mathbf{Y}_{k-1})} \cdot f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}} \end{aligned} \tag{2.14}$$

As shown in system propagation model from Equation (2.12), $\mathbf{x}_k$ is determined by $\mathbf{x}_{k-1}$. Therefore, in Equation (2.14), we have:

$$f_{\mathbf{x}_k|(\mathbf{x}_{k-1},\mathbf{Y}_{k-1})} = f_{\mathbf{x}_k|\mathbf{x}_{k-1}} \tag{2.15}$$

Thus, Equation (2.13) turns to be:

$$f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\xi_k \mid \gamma_{1:k-1}) = \int_{-\infty}^{+\infty} f_{\mathbf{x}_k|\mathbf{x}_{k-1}}\left(\xi_k \mid \xi_{k-1}\right) \cdot f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}\left(\xi_{k-1} \mid \gamma_{1:k-1}\right) d\xi_{k-1} \tag{2.16}$$

For the *a posteriori* density function $f_{\mathbf{x}_k|\mathbf{Y}_k}$, according to Bayes' rule, we have:

$$\begin{aligned} f_{\mathbf{x}_k|\mathbf{Y}_k} &= \frac{f_{\mathbf{x}_k,\mathbf{Y}_k}}{f_{\mathbf{Y}_k}} \\ &= \frac{f_{\mathbf{y}_k|(\mathbf{x}_k,\mathbf{Y}_{k-1})} \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}} \cdot f_{\mathbf{Y}_{k-1}}}{f_{\mathbf{y}_k|\mathbf{Y}_{k-1}} \cdot f_{\mathbf{Y}_{k-1}}} \\ &= \frac{f_{\mathbf{y}_k|\mathbf{x}_k} \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}}{f_{\mathbf{y}_k|\mathbf{Y}_{k-1}}} \end{aligned} \tag{2.17}$$

where, $\mathbf{y}_k$ is completely determined by $\mathbf{x}_k$, that is $f_{\mathbf{y}_k|(\mathbf{x}_k,\mathbf{Y}_{k-1})} = f_{\mathbf{y}_k|\mathbf{x}_k}$.

The density function $f_{\mathbf{y}_k|\mathbf{Y}_{k-1}}$ in denominator of Equation (2.17) can be further formulated as:

$$\begin{aligned} f_{\mathbf{y}_k|\mathbf{Y}_{k-1}} &= \int_{-\infty}^{+\infty} f_{(\mathbf{x}_k,\mathbf{y}_k)|\mathbf{Y}_{k-1}}\left[(\xi_k,\gamma_k) \mid \gamma_{1:k-1}\right] d\xi_k \\ &= \int_{-\infty}^{+\infty} f_{\mathbf{y}_k|(\mathbf{x}_k,\mathbf{Y}_{k-1})}\left[\gamma_k \mid (\xi_k,\gamma_{1:k-1})\right] \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}\left[\xi_k \mid \gamma_{1:k-1}\right] d\xi_k \end{aligned} \tag{2.18}$$

Again, $\mathbf{y}_k$ is completely determined by $\mathbf{x}_k$, and we have:

$$f_{\mathbf{y}_k|\mathbf{Y}_{k-1}} = \int_{-\infty}^{+\infty} f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k \mid \xi_k\right) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}\left[\xi_k \mid \gamma_{1:k-1}\right] d\xi_k \tag{2.19}$$

Substituting Equation (2.19) into Equation (2.17) with considering the dummy parameters yields:

$$f_{\mathbf{x}_k|\mathbf{Y}_k}\left(\xi_k \mid \gamma_{1:k}\right) = \frac{f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k \mid \xi_k\right) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}\left(\xi_k \mid \gamma_{1:k-1}\right)}{\int_{-\infty}^{+\infty} f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k \mid \xi_k\right) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}\left(\xi_k \mid \gamma_{1:k-1}\right) d\xi_k} \tag{2.20}$$

It is obvious that the denominator is the integral of the density function in the numerator. Thus, it is a normalization constant. We assign:

$$c = \left(\int_{-\infty}^{+\infty} f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k \mid \xi_k\right) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}\left(\xi_k \mid \gamma_{1:k-1}\right) d\xi_k\right)^{-1} \tag{2.21}$$

And Equation (2.20) turns to be:

$$f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k}) = c \cdot f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k-1}) \tag{2.22}$$

The $f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k)$ in Equation (2.22) is called "likelihood function". It is computed based on the system observation model. If we assume that the observation noise is zero mean, additive and Gaussian distributed, the likelihood function can be derived as follows.

With observation model:

$$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{v}_k,\ \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k) \tag{2.23}$$

where $\mathbf{R}_k$ is the covariance matrix of the measurement error vector $\mathbf{v}_k$.

We compute the probability of $\mathbf{y}_k$ equals to its realization $\boldsymbol{\gamma}_k$, given the premise that $\mathbf{x}_k$ is equal to its realization $\boldsymbol{\xi}_k$ over a infinitesimal interval from $\boldsymbol{\gamma}_k$ to $\boldsymbol{\gamma}_k + d\boldsymbol{\gamma}_k$ as:

$$P\{\mathbf{y}_k \in (\boldsymbol{\gamma}_k, \boldsymbol{\gamma}_k + d\boldsymbol{\gamma}_k] \mid \mathbf{x}_k = \boldsymbol{\xi}_k\} = f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot d\boldsymbol{\gamma}_k \tag{2.24}$$

Substituting Equation (2.23) into Equation (2.24) yields:

$$\begin{aligned} &P\{[h_k(\mathbf{x}_k) + \mathbf{v}_k] \in (\boldsymbol{\gamma}_k, \boldsymbol{\gamma}_k + d\boldsymbol{\gamma}_k] \mid \mathbf{x}_k = \boldsymbol{\xi}_k\} = f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot d\boldsymbol{\gamma}_k \\ \Rightarrow &P\{\mathbf{v}_k \in (\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k), \boldsymbol{\gamma}_k + d\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k)] \mid \mathbf{x}_k = \boldsymbol{\xi}_k\} = f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot d\boldsymbol{\gamma}_k \end{aligned} \tag{2.25}$$

From Equation (2.23), we have $\mathbf{v}_k = \mathbf{y}_k - h_k(\mathbf{x}_k)$. Thus, $\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)$ is actually the realization of measurement noise $\mathbf{v}_k$. Therefore, we have:

$$P\{\mathbf{v}_k \in (\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k), \boldsymbol{\gamma}_k + d\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k)] \mid \mathbf{x}_k = \boldsymbol{\xi}_k\} = f_{\mathbf{v}_k|\mathbf{x}_k}[\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k) \mid \mathbf{x}_k = \boldsymbol{\xi}_k] \cdot d\boldsymbol{\gamma}_k \tag{2.26}$$

Now we use the independence of the noise $\mathbf{v}_k$ from $\mathbf{x}_k$ to write:

$$f_{\mathbf{v}_k|\mathbf{x}_k}[\boldsymbol{\gamma}_k - h_k(\mathbf{x}_k) \mid \mathbf{x}_k = \boldsymbol{\xi}_k] \cdot d\boldsymbol{\gamma}_k = f_{\mathbf{v}_k}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)] \cdot d\boldsymbol{\gamma}_k \tag{2.27}$$

Comparing the right sides in Equation (2.25) and (2.27) yields:

$$\begin{aligned} &f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot d\boldsymbol{\gamma}_k = f_{\mathbf{v}_k}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)] \cdot d\boldsymbol{\gamma}_k \\ \Rightarrow &f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) = f_{\mathbf{v}_k}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)] \end{aligned} \tag{2.28}$$

Because we assume $\mathbf{v}_k$ is additive, zero mean and Gaussian distributed with covariance matrix $\mathbf{R}_k$. Thus, the likelihood density function is formulated as:

$$
\begin{aligned}
f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) &= f_{\mathbf{v}_k}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)] \\
&= \frac{1}{(2\pi)^{m/2}|\mathbf{R}_k|^{1/2}} \exp\left\{-\frac{1}{2}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)]^T \mathbf{R}_k^{-1}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)]\right\}
\end{aligned}
\tag{2.29}
$$

With Equations (2.16), (2.22) and (2.29), the recursive Bayesian state estimation equations are presented. An initial state density function $f_{\mathbf{x}_0|\mathbf{y}_0}$ should be given, so that $f_{\mathbf{x}_k|\mathbf{y}_k}$ can be computed recursively at any time instant k. The recursive equations are summarized as follows.

Table 2-1. Recursive Bayesian state estimator algorithm

Discrete-time stochastic system state space models	$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, \mathbf{w}_{k-1})$ $\mathbf{y}_k = h_k(\mathbf{x}_k, \mathbf{v}_k)$

1) with initial density function: $f_{\mathbf{x}_0|\mathbf{y}_0}$

For k=1,2,...

2) *The a priori* density function is computed as:

$$f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k-1}) = \int_{-\infty}^{+\infty} f_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\boldsymbol{\xi}_k \mid \boldsymbol{\xi}_{k-1}) \cdot f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_{k-1} \mid \boldsymbol{\gamma}_{1:k-1})\, d\boldsymbol{\xi}_{k-1}$$

3) *The a posteriori* density function is computed as:

$$f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k}) = c \cdot f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k-1})$$

where

$$c = \left(\int_{-\infty}^{+\infty} f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k-1})\, d\boldsymbol{\xi}_k\right)^{-1}$$

If the observation noise is additive, zero mean and Gaussian distributed:

$$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{v}_k,\ \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$$

The likelihood density function can be computed as:

$$f_{\mathbf{y}_k|\mathbf{x}_k}(\boldsymbol{\gamma}_k \mid \boldsymbol{\xi}_k) = \frac{1}{(2\pi)^{m/2}|\mathbf{R}_k|^{1/2}} \exp\left\{-\frac{1}{2}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)]^T \mathbf{R}_k^{-1}[\boldsymbol{\gamma}_k - h_k(\boldsymbol{\xi}_k)]\right\}$$

It is worth mentioning that the recursive propagation of the *a posteriori* density algorithm shown in **Table 2-1** is served as a conceptual solution in that, in general, it cannot be determined analytically. Only in a limited number of cases, that is, for linear Gaussian models where the probability density functions can be completely described by their mean and covariance parameters. The analytical calculation can be carried out in terms of the Kalman filter equations. In general, for nonlinear, non-Gaussian models, there is no simple way to proceed.

2.4 Recursive Bayesian state estimator with Gaussian assumptions

The recursive Bayesian filter makes no assumptions on the linearity of system state space models and the statistical distributions on the process and observation noises. However, before the filter can be implemented, one has to quantify the specific probabilities for the state *a priori*, *a posteriori* and likelihood density functions for having optimal estimation solutions, as shown in **Table 2-1**. If the noise in the respective process or observation models cannot be modeled as Gaussian distributed, quantification of these density functions can sometimes be difficult, or even impossible. Thus, practically, the Gaussian assumptions are often held, which leads to the algorithm of recursive Bayesian state estimator with Gaussian assumptions.

In this section, we simplify the problem, and consider the noises as additive and Gaussian distributed with zero mean and covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$. The system state space models are written as:

$$\begin{aligned}&\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1}\\&\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{v}_k\\&\text{with } \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)\end{aligned} \tag{2.30}$$

The initial density function $f_{\mathbf{x}_0|\mathbf{y}_0}$ is assumed to be Gaussian distributed with mean $\hat{\mathbf{x}}_0^+$ and covariance $\mathbf{P}_0^{\mathbf{x}+}$, which is denoted as:

$$f_{\mathbf{x}_0|\mathbf{y}_0}(\xi_0 \mid \gamma_0) = \mathcal{N}(\xi_0; \hat{\mathbf{x}}_0^+, \mathbf{P}_0^{\mathbf{x}+}) \tag{2.31}$$

The right side of Equation (2.31) represents that for random variable $\mathbf{x}$ at initial time instant, the realization of it (i.e., ξ_0) is distributed according to Gaussian density function $\mathcal{N}(\hat{\mathbf{x}}_0^+, \mathbf{P}_0^{\mathbf{x}+})$.

Assume that one has the *a posteriori* state $\hat{\mathbf{x}}_{k-1}^{+}$ with density $f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}$ at time instance k-1, its Gaussian density function can be described by its mean $\hat{\mathbf{x}}_{k-1}^{+}$ and covariance $\mathbf{P}_{k-1}^{\mathbf{x}+}$ as:

$$f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}} = \mathcal{N}(\xi_{k-1}; \hat{\mathbf{x}}_{k-1}^{+}, \mathbf{P}_{k-1}^{\mathbf{x}+}) \tag{2.32}$$

Based on the system propagation model, the expectation of *a priori* state at time instant k can be calculated as:

$$\hat{\mathbf{x}}_{k}^{-} = \int_{-\infty}^{+\infty} f_{k-1}(\xi_{k-1}) \cdot \mathcal{N}(\xi_{k-1}; \hat{\mathbf{x}}_{k-1}^{+}, \mathbf{P}_{k-1}^{\mathbf{x}+}) d\xi_{k-1} \tag{2.33}$$

The *a priori* covariance parameter is calculated as:

$$\mathbf{P}_{k}^{\mathbf{x}-} = \int_{-\infty}^{+\infty} f_{k-1}(\xi_{k-1}) \cdot f_{k-1}^{T}(\xi_{k-1}) \cdot \mathcal{N}(\xi_{k-1}; \hat{\mathbf{x}}_{k-1}^{+}, \mathbf{P}_{k-1}^{\mathbf{x}+}) d\xi_{k-1} - \hat{\mathbf{x}}_{k}^{-} \cdot \left(\hat{\mathbf{x}}_{k}^{-}\right)^{T} + \mathbf{Q}_{k-1} \tag{2.34}$$

Having *a priori* state mean and covariance estimates derived in Equation (2.33) and (2.34), we can form the *a priori* density $f_{\mathbf{x}_{k}|\mathbf{Y}_{k-1}}$ as:

$$f_{\mathbf{x}_{k}|\mathbf{Y}_{k-1}} = \mathcal{N}(\xi_{k}; \hat{\mathbf{x}}_{k}^{-}, \mathbf{P}_{k}^{\mathbf{x}-}) \tag{2.35}$$

Based on the observation model, the expected value of $\hat{\mathbf{y}}_{k}$ is computed as:

$$\begin{aligned} \hat{\mathbf{y}}_{k} &= \int_{-\infty}^{+\infty} h_{k}(\xi_{k}) \cdot f_{\mathbf{x}_{k}|\mathbf{Y}_{k-1}}(\xi_{k} \mid \gamma_{1:k-1}) d\xi_{k} \\ &= \int_{-\infty}^{+\infty} h_{k}(\xi_{k}) \cdot \mathcal{N}(\xi_{k}; \hat{\mathbf{x}}_{k}^{-}, \mathbf{P}_{k}^{\mathbf{x}-}) d\xi_{k} \end{aligned} \tag{2.36}$$

We form the difference quantity as:

$$\begin{aligned} \boldsymbol{\varepsilon}_{k}^{\mathbf{y}_{k}} &= \mathbf{y}_{k} - \hat{\mathbf{y}}_{k} \\ &= h_{k}(\mathbf{x}_{k}) + \mathbf{v}_{k} - \hat{\mathbf{y}}_{k} \end{aligned} \tag{2.37}$$

Its covariance is computed as:

$$\begin{aligned} \mathbf{P}_{k}^{y} &= E\left[\boldsymbol{\varepsilon}_{k}^{\mathbf{y}_{k}} \cdot \left(\boldsymbol{\varepsilon}_{k}^{\mathbf{y}_{k}}\right)^{T}\right] \\ &= \int_{-\infty}^{+\infty} h_{k}(\xi_{k}) \cdot h_{k}^{T}(\xi_{k}) \cdot \mathcal{N}(\xi_{k}; \hat{\mathbf{x}}_{k}^{-}, \mathbf{P}_{k}^{\mathbf{x}-}) d\xi_{k} - \hat{\mathbf{y}}_{k} \cdot \left(\hat{\mathbf{y}}_{k}\right)^{T} + \mathbf{R}_{k} \end{aligned} \tag{2.38}$$

The cross-covariance matrix is computed as:

$$\begin{aligned}\mathbf{P}_k^{\mathbf{xy}} &= E\left[\left(\boldsymbol{\xi}_k - \hat{\mathbf{x}}_k^-\right)\cdot\left(\boldsymbol{\varepsilon}_k^{\mathbf{y}_k}\right)^T\right] \\ &= \int_{-\infty}^{+\infty} \boldsymbol{\xi}_k \cdot h_k^T(\boldsymbol{\xi}_k)\cdot \mathcal{N}(\boldsymbol{\xi}_k;\hat{\mathbf{x}}_k^-,\mathbf{P}_k^{\mathbf{x}-})d\boldsymbol{\xi}_k - \hat{\mathbf{x}}_k^- \cdot \left(\hat{\mathbf{y}}_k\right)^T\end{aligned} \tag{2.39}$$

According to Equation (2.9) and (2.10), we denote $\mathbf{K}_k = \mathbf{P}_k^{\mathbf{xy}}\left(\mathbf{P}_k^{\mathbf{y}}\right)^{-1}$ and a Gaussian approximation of *a posteriori* density function can be constructed as:

$$f_{\mathbf{x}_k|\mathbf{Y}_k} = \mathcal{N}(\boldsymbol{\xi}_k;\hat{\mathbf{x}}_k^+,\mathbf{P}_k^{\mathbf{x}+}) \tag{2.40}$$

where its mean and covariance parameters are given by:

$$\begin{aligned}\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\boldsymbol{\gamma}_k - \hat{\mathbf{y}}_k\right) \\ \mathbf{P}_k^{\mathbf{x}+} &= \mathbf{P}_k^{\mathbf{x}-} - \mathbf{K}_k\mathbf{P}_k^{\mathbf{y}}\mathbf{K}_k^{\ T}\end{aligned} \tag{2.41}$$

In this derivation, we have made the assumption of Gaussian distributed system process and observation noises, and assume the *a priori* and *a posteriori* states are Gaussian distributed. Nevertheless, we have not made assumptions on the nonlinearity of the system state space models, which are included in the computational of integrals from Equation (2.33) to (2.39). For nonlinear equations, these integrals cannot be solved in closed form. Therefore, approximation must be made to evaluate these integrals.

The Unscented Kalman filter (UKF) provides one efficient way to cope with these integrals by employing a set of supporting points (sigma points) generated on the basis of the first and second moments of the *a priori* and *a posteriori* probability density functions. The supporting points are passed through the nonlinear equations directly, and their outcomes are summed to approximate the results of these integral operations.

2.5 Unscented Kalman filter

The unscented Kalman filter algorithm was proposed by Julier [20] in 1995. It is derived based on the principles of recursive Bayesian state estimator with Gaussian assumptions, which can be easily applied on practical tracking problems. It bases on the assumption that it is much easier to approximate a Gaussian distribution, rather than to simulate the arbitrary nonlinear function. The filter tracks the first two moments of states using a set of supporting points (i.e., sigma point) passing through the nonlinear process and observation models directly with integrals replaced by discrete finite sums of weighted supporting points. For some problems, the UKF has been shown to give better performance than a standard EKF since it better estimates the parameters of the Gaussian approximation to the state distributions [21]. According to the derivation in [20, 22-28], its state estimation accuracy can reach up to the third order of Taylor series expansion for any nonlinearity with Gaussian inputs.

The realization of UKF algorithm can be handled in many ways (e.g., [22-25]). In this thesis, for the discrete-time stochastic system models, as shown in Equation (2.30), the UKF algorithm according to [22, 26] is introduced here (using 2n equally weighted sigma points), which is summarized as follows.

Table 2-2. Summary of UKF algorithm

(1) Initialization	$\hat{\mathbf{x}}_0^{+}, \mathbf{P}_0^{+}$
(2) Draw sigma points	$\boldsymbol{\chi}_{k,i} = \hat{\mathbf{x}}_{k-1}^{+} + \left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T, \boldsymbol{\chi}_{k,i+n} = \hat{\mathbf{x}}_{k-1}^{+} - \left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T, i = 1,\ldots,n$
(3) Time update	$\hat{\mathbf{x}}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n} f_{k-1}(\boldsymbol{\chi}_{k,i}),$
	$\mathbf{P}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left[f_{k-1}(\boldsymbol{\chi}_{k,i}) - \hat{\mathbf{x}}_k^{-}\right]\left[f_{k-1}(\boldsymbol{\chi}_{k,i}) - \hat{\mathbf{x}}_k^{-}\right]^T + \mathbf{Q}_{k-1}$
(4) Draw sigma points	$\boldsymbol{\chi}_{k,i} = \hat{\mathbf{x}}_k^{-} + \left(\sqrt{n\mathbf{P}_k^{-}}\right)_i^T, \boldsymbol{\chi}_{k,i+n} = \hat{\mathbf{x}}_k^{-} - \left(\sqrt{n\mathbf{P}_k^{-}}\right)_i^T, i = 1,\ldots,n$
(5) Predict measurements	$\hat{\mathbf{y}}_k = \frac{1}{2n}\sum_{i=1}^{2n} h_k(\boldsymbol{\chi}_{k,i})$
(6) Update covariance	$\mathbf{P}_k^{y} = \frac{1}{2n}\sum_{i=1}^{2n}\left[h_k(\boldsymbol{\chi}_{k,i}) - \hat{\mathbf{y}}_k\right]\left[h_k(\boldsymbol{\chi}_{k,i}) - \hat{\mathbf{y}}_k\right]^T + \mathbf{R}_k$
	$\mathbf{P}_k^{xy} = \frac{1}{2n}\sum_{i=1}^{2n}\left[\boldsymbol{\chi}_{k,i} - \hat{\mathbf{x}}_k^{-}\right]\left[h_k(\boldsymbol{\chi}_{k,i}) - \hat{\mathbf{y}}_k\right]^T$
(7) Measurement update	$\mathbf{K}_k = \mathbf{P}_k^{xy}(\mathbf{P}_k^{y})^{-1}, \hat{\mathbf{x}}_k^{+} = \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k), \mathbf{P}_k^{+} = \mathbf{P}_k^{-} - \mathbf{K}_k\mathbf{P}_k^{y}\mathbf{K}_k^T$

where '*n*' denotes the dimension of the system state vector; $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)$ is the matrix square root of $n\mathbf{P}_{k-1}^{+}$, such as $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)^T\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right) = n\mathbf{P}_{k-1}^{+}$, which can be obtained from the Cholesky factorization 'CHOL' in MATLAB; $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i$ is the *i*-th row of $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)$; $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T$ is the transpose of $\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i$; $\mathbf{Q}_k$ is the covariance matrix of the additive system process noise; $\mathbf{R}_k$ is the covariance matrix of the additive measurement noise.

2.6 Particle filter

With recent massive increases in computational power, the PF turns to be more attractive due to its great advantage of not being subject to the assumption of linearity and Gaussianess in the models [29]. The PF are sequential Monte Carlo methods based upon point mass (i.e., particles) representations of probability densities, which can be applied to any state space models [21, 29-33]. The derivations of the sequential Monte Carlo filter are known variously as bootstrap filtering [19], the condensation algorithm [34], interacting particle approximations [35] and the survival of the fittest [36]. These variations are based on two basic principles, namely the Monte Carlo approximation, and the importance sampling [21, 29, 31, 32].

2.6.1 Monte Carlo approximation

As shown in Equations (2.31) to (2.41), the integral over a nonlinear function multiplying a density is required to be calculated recursively, which in most cases cannot be solved analytically. Unlike the UKF, which utilizes a set of determinately chosen supporting points, in the particle filter, the numerical approximation is based on the Monte Carlo approximation. That is, a particle filter works by providing a Monte Carlo approximation to the probability density function which can be easily updated to incorporate new information as it arrives. At time instant k, if N samples (denoted as $\boldsymbol{\chi}_{k,i}$, with i=1 to N) are randomly, independently taken from $f_{\mathbf{x}_k|\mathbf{Y}_k}$, the expectation of certain function of random variable vector $\mathbf{z}_k = g(\mathbf{x}_k)$ on the basis of measurements $\mathbf{Y}_k$ is formulated as:

$$\begin{aligned} E[g(\mathbf{x}_k) \mid \mathbf{Y}_k = \boldsymbol{\gamma}_{1:k}] &= \int_{-\infty}^{+\infty} g(\boldsymbol{\xi}_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k})\, d\boldsymbol{\xi}_k \\ &\approx \frac{1}{N}\sum_{i=1}^{N} \mathrm{g}(\boldsymbol{\chi}_{k,i}) \end{aligned} \tag{2.42}$$

where $\boldsymbol{\chi}_{k,i}$ denotes the *i*-th particle drawn from $f_{\mathbf{x}_k|\mathbf{Y}_k}$ at time instant k.

The concept of Monte Carlo approximation is to turn the integral over probability density function into its discrete approximation using a set of weighted points to ensemble the distribution of $f_{\mathbf{x}_k|\mathbf{Y}_k}$. When the number of points approaches infinity, this discrete approximated calculation of integral over density function approaches its true value.

2.6.2 Sequential Importance Sampling (SIS)

In Monte Carlo approximation, we assume that we can draw particles from a known *a posteriori* probability density function $f_{\mathbf{x}_k|\mathbf{Y}_k}$. However, in many cases, this density function $f_{\mathbf{x}_k|\mathbf{Y}_k}$ may not always be expressed in its analytical expression. Therefore, generating particles from $f_{\mathbf{x}_k|\mathbf{Y}_k}$ could be challenging. The concept of importance sampling is to draw particles from another density function $q_{\mathbf{x}_k|\mathbf{Y}_k}$ (e.g., a Gaussian density function). This density function $q_{\mathbf{x}_k|\mathbf{Y}_k}$ is called importance density function, which is expected to be proportional to the original distribution $f_{\mathbf{x}_k|\mathbf{Y}_k}$ at every particle with a different scaling factor at each particle. If we assume here $\boldsymbol{\xi}_k$ as particles ideally drawn from density $f_{\mathbf{x}_k|\mathbf{Y}_k}$, and $\boldsymbol{\chi}_k$ is its corresponding particles drawn from $q_{\mathbf{x}_k|\mathbf{Y}_k}$, we have:

$$f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k}) = \mathrm{w}(\boldsymbol{\chi}_k) \cdot q_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\chi}_k \mid \boldsymbol{\gamma}_{1:k}) \tag{2.43}$$

The idea behind Equation (2.43) is that, one can use the particles drawn from $q_{\mathbf{x}_k|\mathbf{Y}_k}$ (e.g., a Gaussian density function) multiplying different weight on each particle to approximate the particle generated from the original density function $f_{\mathbf{x}_k|\mathbf{Y}_k}$ (e.g., multimodal).

As shown in Equation (2.20), the $f_{\mathbf{x}_k|\mathbf{Y}_k}$ is a normalized density function. The integral over a normalized probability density function returns 1. Therefore, Equation (2.42) can be expressed in terms of importance density function $q_{\mathbf{x}_k|\mathbf{Y}_k}$ as:

$$\begin{aligned} E[g(\mathbf{x}_k) \mid \mathbf{Y}_k = \boldsymbol{\gamma}_{1:k}] &= \frac{\int_{-\infty}^{+\infty} g(\boldsymbol{\xi}_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k})\, d\boldsymbol{\xi}_k}{\int_{-\infty}^{+\infty} f_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\xi}_k \mid \boldsymbol{\gamma}_{1:k})\, d\boldsymbol{\xi}_k} \\ &= \frac{\int_{-\infty}^{+\infty} g(\boldsymbol{\chi}_k) \cdot \mathrm{w}(\boldsymbol{\chi}_k) \cdot q_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\chi}_k \mid \boldsymbol{\gamma}_{1:k})\, d\boldsymbol{\chi}_k}{\int_{-\infty}^{+\infty} \mathrm{w}(\boldsymbol{\chi}_k) \cdot q_{\mathbf{x}_k|\mathbf{Y}_k}(\boldsymbol{\chi}_k \mid \boldsymbol{\gamma}_{1:k})\, d\boldsymbol{\chi}_k} \end{aligned} \tag{2.44}$$

where $\int_{-\infty}^{+\infty} f_{\mathbf{x}_k|\mathbf{Y}_k}(\xi_k \mid \gamma_{1:k})\, d\xi_k = 1$.

Substituting Equation (2.42) into Equation (2.44) yields:

$$E\left[g(\mathbf{x}_k) \mid \mathbf{Y}_k = \gamma_{1:k}\right] \approx \frac{\frac{1}{N}\sum_{i=1}^{N} \mathrm{g}(\chi_{k,i}) \cdot \mathrm{w}(\chi_{k,i})}{\frac{1}{N}\sum_{i=1}^{N} \mathrm{w}(\chi_{k,i})} \approx \sum_{i=1}^{N} \bar{\mathrm{w}}(\chi_{k,i}) \cdot \mathrm{g}(\chi_{k,i}) \tag{2.45}$$

where $\bar{\mathrm{w}}(\chi_{k,i})$ is denoted as the normalized importance weight, which is computed as:

$$\bar{\mathrm{w}}(\chi_{k,i}) = \frac{\mathrm{w}(\chi_{k,i})}{\frac{1}{N}\sum_{i=1}^{N} \mathrm{w}(\chi_{k,i})} \tag{2.46}$$

The computations in Equations (2.44) to (2.46) are correct only in case that the weight $\mathrm{w}(\chi_k)$ introduced in Equation (2.43) exists, which can proportionally relate the particles drawn from the importance density function $q_{\mathbf{x}_k|\mathbf{Y}_k}$ to the original *a posteriori* density function $f_{\mathbf{x}_k|\mathbf{Y}_k}$. These weights are computed as follows [29, 32]:

$$\mathrm{w}(\chi_k) = \frac{f_{\mathbf{x}_k|\mathbf{Y}_k}(\xi_k \mid \gamma_{1:k})}{q_{\mathbf{x}_k|\mathbf{Y}_k}(\chi_k \mid \gamma_{1:k})} = \frac{c \cdot f_{\mathbf{y}_k|\mathbf{x}_k}(\gamma_k \mid \xi_k) \cdot f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}(\xi_k \mid \gamma_{1:k-1})}{q_{\mathbf{x}_k|\mathbf{Y}_k}(\chi_k \mid \gamma_{1:k})} \tag{2.47}$$

The $q_{\mathbf{x}_k|\mathbf{Y}_k}(\chi_k \mid \gamma_{1:k})$ in Equation (2.47) can be formulated as:

$$\begin{aligned} q_{\mathbf{x}_k|\mathbf{Y}_k}(\chi_k \mid \gamma_{1:k}) &= \int_{-\infty}^{+\infty} q_{(\mathbf{x}_k,\mathbf{x}_{k-1})|\mathbf{Y}_k}\left[(\chi_k, \chi_{k-1}) \mid \gamma_{1:k}\right] d\chi_{k-1} \\ &= \int_{-\infty}^{+\infty} q_{\mathbf{x}_k|(\mathbf{x}_{k-1},\mathbf{Y}_k)}\left[\chi_k \mid (\chi_{k-1}, \gamma_{1:k})\right] \cdot q_{\mathbf{x}_{k-1}|\mathbf{Y}_k}(\chi_{k-1} \mid \gamma_{1:k})\, d\chi_{k-1} \end{aligned} \tag{2.48}$$

where the *a priori* state vector $\mathbf{x}_k$ is only determined by $\mathbf{x}_{k-1}$ according to the system propagation model, thus $q_{\mathbf{x}_k|(\mathbf{x}_{k-1},\mathbf{Y}_k)} = q_{\mathbf{x}_k|\mathbf{x}_{k-1}}$; the *a posteriori* state vector $\mathbf{x}_{k-1}$ is only determined based on $\mathbf{Y}_{k-1}$ instead of $\mathbf{Y}_k$. Thus we have $q_{\mathbf{x}_{k-1}|\mathbf{Y}_k} = q_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}$.

By Substituting Equation (2.48) and (2.13) into Equation (2.47) yields:

$$\mathrm{w}(\chi_k) = \frac{c \cdot f_{\mathbf{y}_k|\mathbf{x}_k}(\gamma_k \mid \xi_k) \cdot \int_{-\infty}^{+\infty} f_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\xi_k \mid \xi_{k-1}) \cdot f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}(\xi_{k-1} \mid \gamma_{1:k-1})\, d\xi_{k-1}}{\int_{-\infty}^{+\infty} q_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\chi_k \mid \chi_{k-1}) \cdot q_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}(\chi_{k-1} \mid \gamma_{1:k-1})\, d\chi_{k-1}} \tag{2.49}$$

Evaluating the weights on individual particles, we have:

$$\begin{aligned} \mathrm{w}(\chi_{k,i}) &\propto \frac{f_{\mathbf{y}_k|\mathbf{x}_k}(\gamma_k \mid \xi_{k,i}) \cdot f_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\xi_{k,i} \mid \xi_{k-1,i})}{q_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\chi_{k,i} \mid \chi_{k-1,i})} \cdot \frac{f_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}(\xi_{k-1,i} \mid \gamma_{1:k-1})}{q_{\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}}(\chi_{k-1,i} \mid \gamma_{1:k-1})} \\ &\propto \frac{f_{\mathbf{y}_k|\mathbf{x}_k}(\gamma_k \mid \xi_{k,i}) \cdot f_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\xi_{k,i} \mid \xi_{k-1,i})}{q_{\mathbf{x}_k|\mathbf{x}_{k-1}}(\chi_{k,i} \mid \chi_{k-1,i})} \cdot \mathrm{w}(\chi_{k-1,i}) \end{aligned} \tag{2.50}$$

The "$\propto$" represents the meaning of "proportional to", because we omit the normalization constant "c". Note that in Equation (2.50), the subscript "i" denotes that the weight currently under estimation is for the *i*-th particle. That is, unlike in Equation (2.49), the terms in Equation (2.50) are not the density functions, but the numerical values evaluated by the density functions with respect to the *i*-th particle. After normalization as conducted in Equation (2.46), the normalized importance weights are simply numerical values between 0 and 1.

Equation (2.50) also reveals that the $\mathrm{w}(\chi_{k,i})$ is a function of $\mathrm{w}(\chi_{k-1,i})$. That is, the importance weights of particles can be computed in a recursive manner. The sequential importance Sampling algorithm consists of the recursive propagation of the weights and support points when measurements are received sequentially. In order to accomplish this, we have to firstly quantify the likelihood density function $f_{\mathbf{y}_k|\mathbf{x}_k}$, the *a priori* density function $f_{\mathbf{x}_k|\mathbf{x}_{k-1}}$ and the importance density function $q_{\mathbf{x}_k|\mathbf{x}_{k-1}}$. If the noise in the system process and observation models cannot be modeled as Gaussian, the quantification of these density functions can sometime be very difficult.

In order to simplify the computation, the system *a priori* density $f_{\mathbf{x}_k|\mathbf{x}_{k-1}}$ is often used as the importance density:

$$q_{\mathbf{x}_k|\mathbf{x}_{k-1}} = f_{\mathbf{x}_k|\mathbf{x}_{k-1}} \tag{2.51}$$

This approach eliminates two density terms in Equation (2.50), and the computation of particle importance weight is simplified as:

$$\mathrm{w}(\chi_{k,i}) \propto f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k \mid \xi_{k,i}\right) \cdot \mathrm{w}(\chi_{k-1,i}) \tag{2.52}$$

The implementation of PF in this manner is named the bootstrap particle filter (BPF) [19, 37]. It can be successfully applied to any tracking problems in case that the likelihood density function can be analytically formulated, the noise statistics are known and the *a priori* density is available for sampling [19]. However, the flaws are that the filter is highly dependent on the initialization estimates and can quickly diverge if the initialization mean of the state vector is far from the true state vector, because the observations are only used in the likelihood function, and there is no measurement update in the filter. Besides, a large number of particles are required for having optimal estimation results.

2.6.3 Re-sampling

In the BPF algorithm, the importance weights of particles are computed based on the likelihood density functions. During the recursions, some particles may be significantly weighted, while others are much less weighted. After several recursions, majority of particles may have negligible weights except only one with significant weight. It has been revealed in [32] that, the variance of the importance weights will increase over time, and it is impossible to avoid the degeneracy phenomenon. This problem implies that a majority of computation burden is devoted to updating particles whose contribution is almost zero [21]. Clearly, this problem is an undesirable effect in the algorithm.

The brute force approach to reducing this effect is to use a very large number of particles. However, it is impractical, and degeneracy problem may eventually still take place.

Therefore, we may rely on two other methods: 1) Take a good choice of importance density function for generating particles intelligently (such as UPF, which will be introduced in section 2.7), and 2) use of re-sampling approach [21]. The re-sampling idea is formally justified in [37], which removes the less weighted particles and multiply the significantly weighted particles, keeping the number of particles constant, leading to the formation of the sequential importance sampling re-sampling particle filter. However, re-sampling also brings drawback. That is, the significantly weighted particles will be statistically selected time after time, which leads to the problem, known as sample impoverishment. That is, although we still have a large number of particles, but the information contained in the particles may come from only one significantly weighted original particle, which leads to a loss of diversity among particles. In order to avoid the problem, the re-sampling step should be added only when the degeneracy phenomenon is high. For evaluating the degeneracy problem, the concept of effective sample size can be introduced [38, 39], which is usually computed as $s_{eff} = 1/\sum_{i=1}^{N} \mathrm{w}(\chi_{k,i})^2$. Small s_{eff} indicates severe degeneracy. If s_{eff} is bigger than certain threshold sample number (e.g., $N_s = 2N/3$), the particles remain. Otherwise, re-sampling step takes place.

Among many re-sampling approaches, in this chapter, we introduce one often used re-sampling method based on the importance weights of particles [19, 33]. Its idea is illustrated in **Figure 2-1**, and the algorithm is summarized in **Table 2-3**.

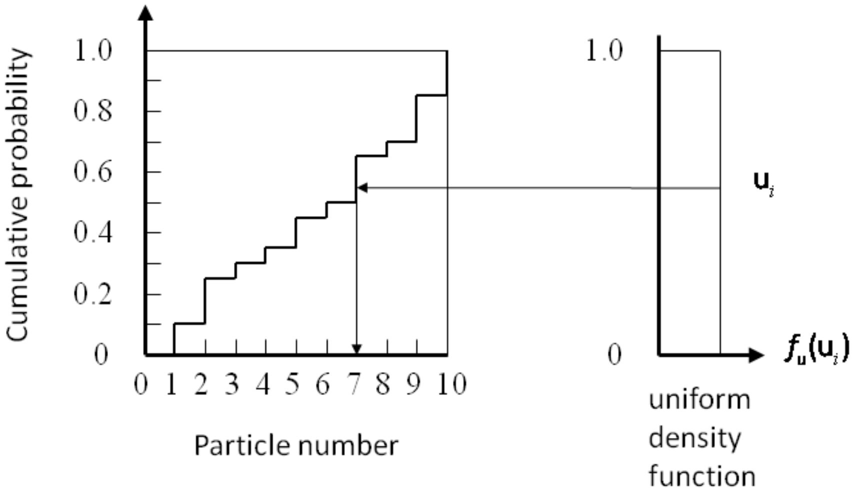


Figure 2-1. Re-sampling approach based on importance weights

Table 2-3. Re-sampling based on importance weights

```
For i=1,2,...,N
(1) Draw random number "u" from uniform distribution between (0,1]
(2) For a set of a posteriori particles from 1 to N. Accumulate their importance weights
    into a sum to meet the following conditions:
    $\sum_{i=1}^{j} \mathrm{w}(\chi_{k,i}) \geq u, and \sum_{i=1}^{j-1} \mathrm{w}(\chi_{k,i}) < u$
(3) The new particle $\chi_{k,i}$ is assigned to be the original particle $\chi_{k,j}$ with importance
    weight $\mathrm{w}(\chi_{k,i}) = 1/N$.
```

As shown in the left side of **Figure 2-1**, the horizontal axis is the particle number, and the vertical axis is the cumulative probability, which is the sum of the normalized importance weights of particles. The normalized importance weight for each particle will be a number between 0 to 1. If we add them together, as shown in the axis of cumulative probability, the significantly weighted particle will occupy more space (e.g., from 0.4 to 0.8 for particle number 7,), while the insignificantly weighted particle will occupy less space (e.g., from about 0.28 to 0.3 for particle number 3). In this way, when we draw random number from a uniform distribution between 0 to 1, the new generated random number will be more frequently located at the area belonging to particle number 7 instead of the area belonging to particle number 3. Thus, after re-sampling process, the information from the significantly weighted particle (e.g., particle number 7) will be inherited by many new generated particles, while the information from insignificantly weighted particle (e.g., particle number 3) will be seldom inherited.

Other efficient re-sampling schemes such as order statistics [40, 41], stratified sampling and residual sampling [38], and systematic re-sampling [42] may be applied as alternatives to this algorithm.

2.6.4 Sequential Importance Sampling Re-sampling (SISR) particle filter

In this section, we summarize the sequential importance sampling re-sampling PF algorithm in **Table 2-4**.

Table 2-4. Summary of SISR PF algorithm

Randomly generate N particles $\chi_{0,i}$ (i=1,...,N) from $f_{\mathbf{x}_0|\mathbf{Y}_0}$ with equal weights $1/N$.

For k=1,2,...

(1) Perform the time propagation step to obtain the *a priori* particles $\chi_{k,i}$ as:

$$\chi_{k,i}=f_{k-1}(\chi_{k-1,i},\mathbf{w}_{k-1,i})$$

(2) Compute the importance weight $\mathrm{w}(\chi_{k,i})$ recursively as:

$$\mathrm{w}(\chi_{k,i})\propto\frac{f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k\mid\xi_{k,i}\right)\cdot f_{\mathbf{x}_k|\mathbf{x}_{k-1}}\left(\xi_{k,i}\mid\xi_{k-1,i}\right)}{q_{\mathbf{x}_k|\mathbf{x}_{k-1}}\left(\chi_{k,i}\mid\chi_{k-1,i}\right)}\cdot\mathrm{w}(\chi_{k-1,i})$$

If one takes the *a priori* density as the importance density function (i.e., bootstrap PF), the importance weights are updated as:

$$\mathrm{w}(\chi_{k,i})=\mathrm{w}(\chi_{k-1,i})\cdot f_{\mathbf{y}_k|\mathbf{x}_k}\left(\gamma_k\mid\xi_{k,i}\right).$$

(3) Normalize the importance weight as:

$$\overline{\mathrm{w}}(\boldsymbol{\chi}_{k,i}) = \frac{\mathrm{w}(\boldsymbol{\chi}_{k,i})}{\sum_{i=1}^{N} \mathrm{w}(\boldsymbol{\chi}_{k,i})}, \text{with } \sum_{i=1}^{N} \overline{\mathrm{w}}(\boldsymbol{\chi}_{k,i}) = 1$$

(4) Compute the effective sample size: $s_{eff} = 1/\sum_{i=1}^{N} \mathrm{w}(\boldsymbol{\chi}_{k,i})^2$

(5) If s_{eff} is bigger than the given threshold sample number (e.g., $N_s = 2N/3$), particles remain with their original weights. Otherwise, re-sampling should take place:
   1) Draw a set of new particles on the basis of importance weights $\overline{\mathrm{w}}(\boldsymbol{\chi}_{k,i})$
   2) Assign equal weights $1/N$ to $\overline{\mathrm{w}}(\boldsymbol{\chi}_{k,i})$

(6) Compute any desired statistical measure based on *a posteriori* particles. Usually, we are only interested in the mean and covariance parameters.

Move from time instant k to k+1.

2.7 Unscented particle filter

As mentioned in former sections, a proper choice of importance density function in the PF is critical, which yields different variations of PF algorithms (e.g., BPF, Laplace's PF [43], etc.). In this section, we introduce the KF measurement update in the PF algorithm. The UKF presents highly accurate estimation results for nonlinear filtering tasks. Therefore, its state *a posteriori* density is employed as the importance density function for generating particles more accurately. This combination yields a new filter which is coined the unscented particle filter (UPF). In the UPF algorithm, the particles are newly generated in each recursion, which are not reused. Thus, the importance weights of particles can be computed in a non-recursive manner (as shown in Equation (2.47)). And the problems of system degeneracy and sample impoverishment will not happen.

In the proposed UPF algorithm, for quantifying the density functions, we describe all distributions using their mean and covariance parameters as Gaussian distributed. Applying these assumptions in PF leads to a type of recursive Gaussian particle filter, which is asymptotically optimal in the number of particles when the Gaussian assumption holds true. However, this Gaussian assumption can be relaxed in case these density functions can be approximated as Gaussian distributed.

In our implementation of UPF algorithm, we assume the *a priori* density $f_{\mathbf{x}_k|\mathbf{Y}_{k-1}}$ as Gaussian distributed: $\mathcal{N}(\hat{\mathbf{x}}_k^-, \mathbf{P}_k^{\mathrm{x}-})$. The likelihood density function is formulated in Equation (2.29), where measurement errors are approximated as Gaussian distributed with zero mean, covariance **R**. This approximation can be removed if we know the true statistics of measurement errors and if its density function can be analytically formulated. Otherwise, a Gaussian assumption may be a good approximation. For the importance density function $q_{\mathbf{x}_k|\mathbf{Y}_k}$, we use the *a posteriori* density obtained from the UKF measurement update. Thus, Equation (2.47) turns to be:

$$\mathrm{w}(\boldsymbol{\chi}_{k,i}) \propto \frac{f_{\mathbf{y}_k|\mathbf{x}_k}\left(\boldsymbol{\gamma}_k \mid \boldsymbol{\chi}_{k,i}\right) \cdot \mathcal{N}(\boldsymbol{\chi}_{k,i}; \hat{\mathbf{x}}_k^-, \mathbf{P}_k^{\mathbf{x}-})}{\mathcal{N}(\boldsymbol{\chi}_{k,i}; \hat{\mathbf{x}}_k^+, \mathbf{P}_k^{\mathbf{x}+})} \tag{2.53}$$

Note that the terms in Equation (2.53) are simply numerical values evaluated by the density functions with respect to the *i*-th particle. For instance, $\mathcal{N}(\boldsymbol{\chi}_{k,i}; \hat{\mathbf{x}}_k^-, \mathbf{P}_k^{\mathbf{x}-})$ represents a value evaluated by the density $\mathcal{N}(\hat{\mathbf{x}}_k^-, \mathbf{P}_k^{\mathbf{x}-})$ with respect to the particle $\boldsymbol{\chi}_{k,i}$.

The main advantages of UPF in this realization with respect to the sequential sampling importance re-sampling bootstrap particle filter are summarized as:

1) The re-sampling step is not required, and computational complexity is reduced.
2) Latest measurements are used to improve the importance density function for generating particles more intelligently.
3) A small number of particles is required.
4) Filter converges rapidly due to the UKF measurement update.
5) The likelihood function, *a priori* and *a posteriori* densities are described as Gaussian distributions, which simplifies the implementation. However, the filter can handle non-Gaussian cases, if these Gaussian approximations can be made.

The principle of UPF in our implementation is illustrated in **Figure 2-2**, and its algorithm is summarized in **Table 2-5**.

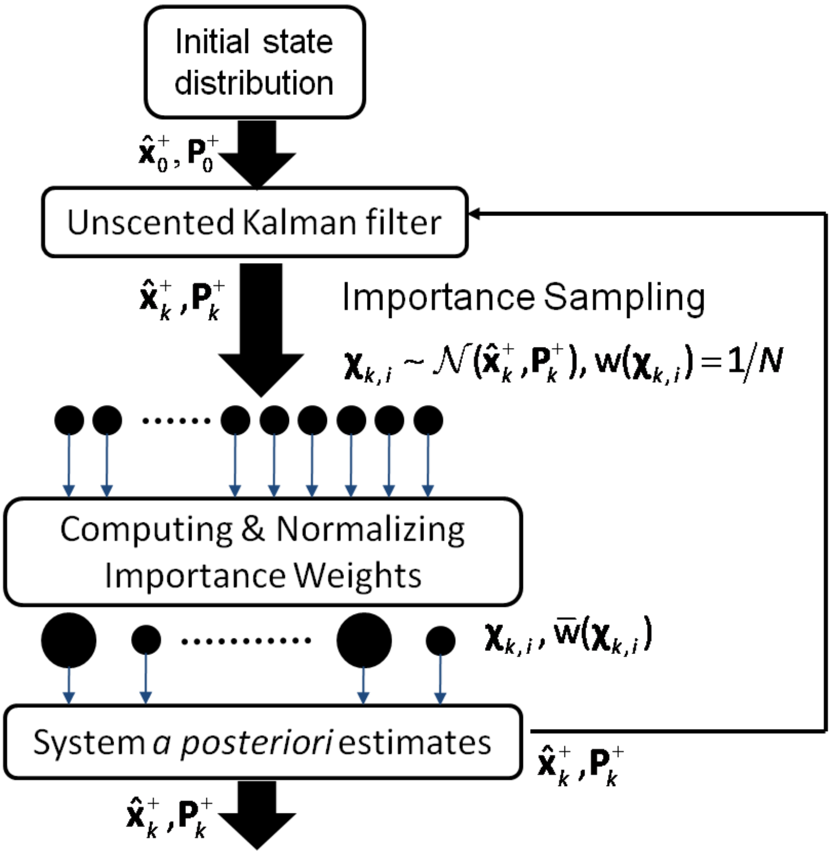


Figure 2-2. Unscented Particle filter principle flowchart

The principle of UPF algorithms introduced in [22, 28] is depicted in **Figure 2-3**:

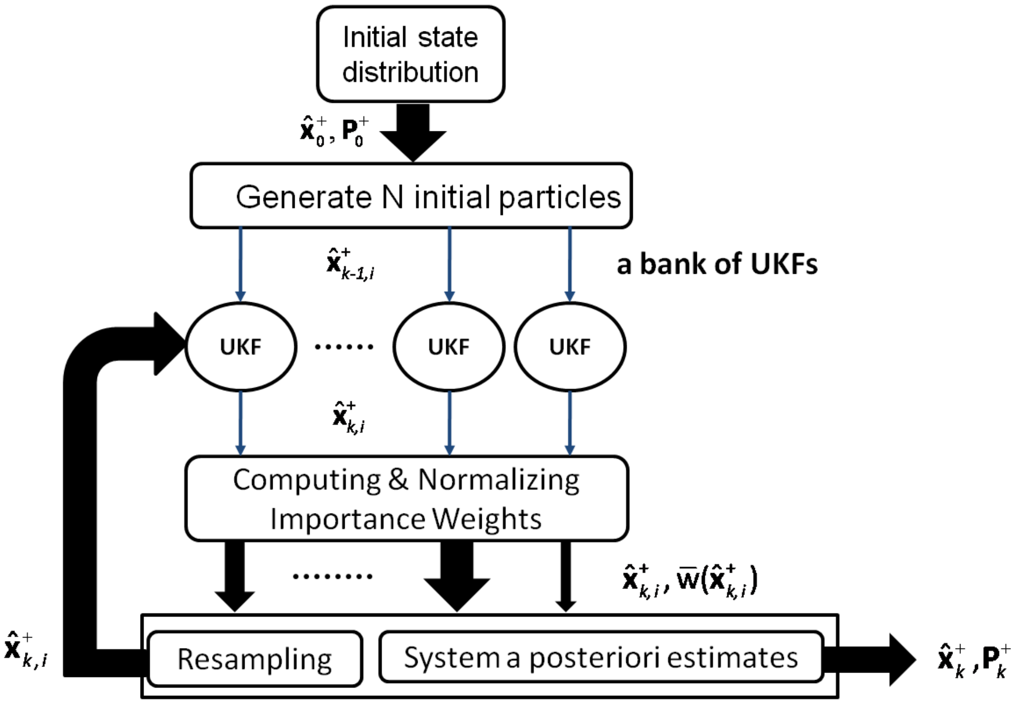


Figure 2-3. The other realization of the unscented particle filter algorithm

Table 2-5. Summary of UPF algorithm

(1) Initialization	$\hat{\mathbf{x}}_0^+, \mathbf{P}_0^+$
(2) Draw sigma points	$\chi_{k,i} = \hat{\mathbf{x}}_{k-1}^+ + \left(\sqrt{n\mathbf{P}_{k-1}^+}\right)_i^T, \chi_{k,i+n} = \hat{\mathbf{x}}_{k-1}^+ - \left(\sqrt{n\mathbf{P}_{k-1}^+}\right)_i^T, i = 1,\ldots,n$
(3) Time update	$\hat{\mathbf{x}}_k^- = \frac{1}{2n}\sum_{i=1}^{2n} f_{k-1}(\chi_{k,i}),$ $\mathbf{P}_k^- = \frac{1}{2n}\sum_{i=1}^{2n}\left[f_{k-1}(\chi_{k,i}) - \hat{\mathbf{x}}_k^-\right]\left[f_{k-1}(\chi_{k,i}) - \hat{\mathbf{x}}_k^-\right]^T + \mathbf{Q}$
(4) Draw sigma points	$\chi_{k,i} = \hat{\mathbf{x}}_k^- + \left(\sqrt{n\mathbf{P}_k^-}\right)_i^T, \chi_{k,i+n} = \hat{\mathbf{x}}_k^- - \left(\sqrt{n\mathbf{P}_k^-}\right)_i^T, i = 1,\ldots,n$
(5) Predict measurements	$\hat{\mathbf{y}}_k = \frac{1}{2n}\sum_{i=1}^{2n} h_k(\chi_{k,i})$
(6) Update covariance	$\mathbf{P}_k^y = \frac{1}{2n}\sum_{i=1}^{2n}\left[h_k(\chi_{k,i}) - \hat{\mathbf{y}}_k\right]\left[h_k(\chi_{k,i}) - \hat{\mathbf{y}}_k\right]^T + \mathbf{R}$ $\mathbf{P}_k^{xy} = \frac{1}{2n}\sum_{i=1}^{2n}\left[\chi_{k,i} - \hat{\mathbf{x}}_k^-\right]\left[h_k(\chi_{k,i}) - \hat{\mathbf{y}}_k\right]^T$
(7) Measurement update	$\mathbf{K}_k = \mathbf{P}_k^{xy}(\mathbf{P}_k^y)^{-1}, \hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k), \mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{P}_k^y\mathbf{K}_k^T$
(8) Draw particles	$\chi_{k,i} \sim \mathcal{N}(\hat{\mathbf{x}}_k^+, \mathbf{P}_k^+)$, and $i = 1,\ldots,N$,
(9) Compute importance weights	$\mathrm{w}(\chi_{k,i}) \propto \frac{f_{\mathbf{y}_k\mid\mathbf{x}_k}(\gamma_k \mid \chi_{k,i}) \cdot \mathcal{N}(\chi_{k,i}; \hat{\mathbf{x}}_k^-, \mathbf{P}_k^{\mathbf{x}-})}{\mathcal{N}(\chi_{k,i}; \hat{\mathbf{x}}_k^+, \mathbf{P}_k^{\mathbf{x}+})}$
(10) Normalize importance weights	$\overline{\mathrm{w}}(\chi_{k,i}) = \frac{\mathrm{w}(\chi_{k,i})}{\sum_{i=1}^{N}\mathrm{w}(\chi_{k,i})}, \text{ } with \sum_{i=1}^{N}\overline{\mathrm{w}}(\chi_{k,i}) = 1$
(11) Update the *a posteriori* state mean and covariance estimates	$\hat{\mathbf{x}}_k^+ = \sum_{i=1}^{N}\overline{\mathrm{w}}(\chi_{k,i})\chi_{k,i}$ $\mathbf{P}_k^+ = \sum_{i=1}^{N}\overline{\mathrm{w}}(\chi_{k,i})\left[\chi_{k,i} - \hat{\mathbf{x}}_k^+\right]\left[\chi_{k,i} - \hat{\mathbf{x}}_k^+\right]^T$

As shown in **Figure 2-3**, a bank of UKFs (one for each particle) is used. Their *a posteriori* estimates are weighted. The weighted *a posteriori* estimates $\hat{\mathbf{x}}_{k,i}^+$ are then re-sampled to generate a new set of particles with equal weights. In the next recursion, each particle is fed into an individual UKF to yield the *a posteriori* estimate. This process repeats until the end of the trajectory.

Apparently, the main disadvantage of this method is its high computational load caused by running a set of parallel UKFs. Even with 50 particles, running 50 UKFs at the same time is already impractical for many real-time applications. Therefore, in our contribution, we use only one UKF in the UPF algorithm.

In the next section, a simulation is conducted to show the behavior of the proposed UPF algorithm using a nonlinear non-Gaussian problem.

2.8 Simulation test

In this section, we verify the system performance of the proposed UPF algorithm with respect to that of other nonlinear filters, i.e., EKF, UKF (using 2n equally weighted sigma points) and Bootstrap Particle Filter (BPF). A nonlinear and non-Gaussian estimation problem is employed, which was firstly used in [28]. Its system process and observation models are:

$$x_{k+1} = 1 + \sin(0.04\pi k) + 0.5x_k + w_k \qquad (2.54)$$
$$y_k = \begin{cases} 0.2x_k^2 + v_k, & k \le 30 \\ 0.5x_k - 2 + v_k, & k > 30 \end{cases}$$

where the system process noise is gamma distributed with shape parameter k = 3 and scale parameter θ = 0.5. The measurement noise is Gaussian distributed with zero mean and variance 1e-5.

The experiment is repeated for 100 times with random re-initialization for each run. **Table 2-6** shows the performances of different filters, where the mean and variance of the root-mean-square-error (RMSE) of state estimates from 100 independent runs are given.

Table 2-6. Comparison of estimation results of nonlinear filters

Nonlinear filtering methods	Mean of RMSE	Variance of RMSE
EKF	0.3658	0.0116
UKF	0.3376	0.0120
BPF (200 particles)	0.4191	0.0699
BPF (500 particles)	0.2487	0.0466
BPF (1000 particles)	0.1568	0.0368
BPF (5000 particles)	0.0575	0.0190
UPF (50 particles)	0.2201	0.0171
UPF (100 particles)	0.2051	0.0154
UPF (200 particles)	0.1846	0.0085
UPF (500 particles)	0.1783	0.0091

As shown in the table, for the BPF, when the number of particles approaches infinity, the estimation errors will theoretically approach zero. However, in order to achieve this performance, the distributions of the system process and measurement noises must be known, which will be challenging for navigation applications.

The UPF presents highly accurate estimation results with a dramatically reduced number of particles. However, using a larger number of particles will not lead the filter to present optimal estimation performance. Because in the filter, we assumed the system process noise as Gaussian distributed, which is in fact gamma distributed. In the implementation of UPF in [28], the author uses the exact gamma density function in the algorithm. Thus, the filter makes no assumption on the noises, and it will be asymptotically optimal in the number of particles.

For the EKF and UKF, they all treat noises as Gaussian distributed, therefore the UPF we proposed does not make any extra assumptions. But, it presents improved estimation results, as shown in the table.

The reason why the UPF presents highly accurate results with a small number of particles is further investigated by conducting one single run of the test, where 200 particles are used for both UPF and BPF. The state estimates from different filters are depicted in **Figure 2-4**.

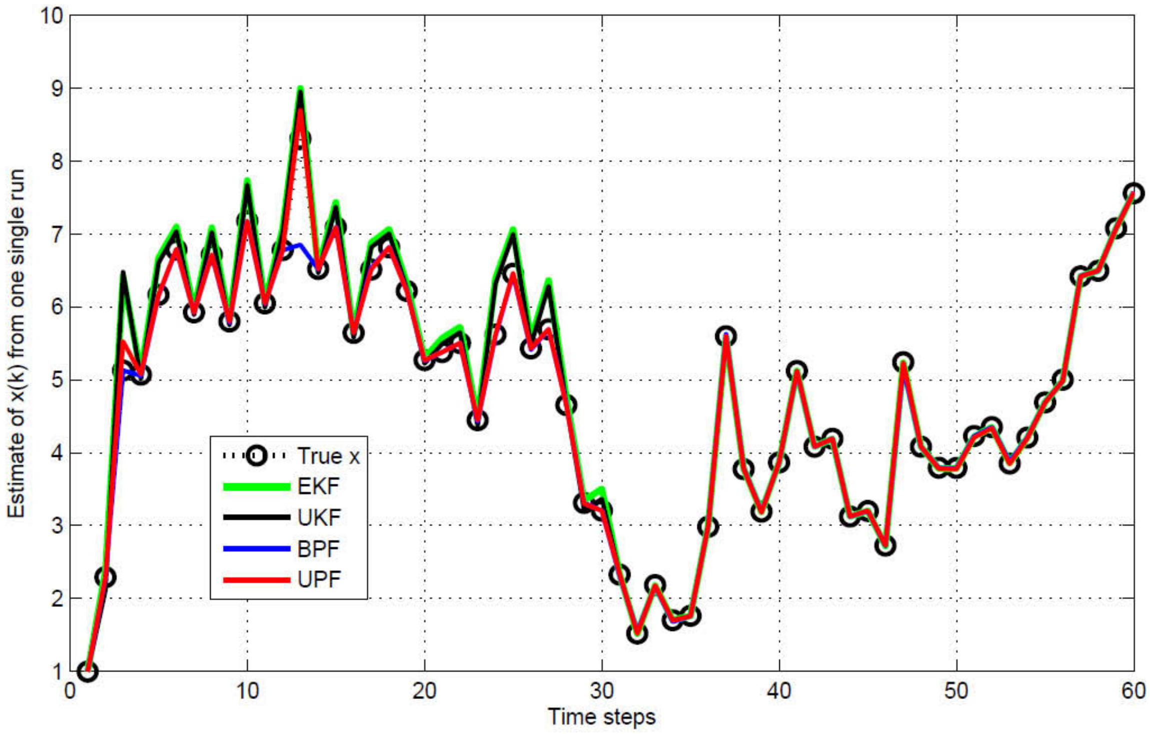


Figure 2-4. State estimates of nonlinear filters from one single run

For the BPF, at certain time instant (e.g., at the 13th epoch in **Figure 2-4**), the state estimation accuracy could be very poor. This is because that, at this time instant, even the most accurate *a priori* particle is far away from the true value of the state. For solving this problem, usually a larger amount of particles should be employed.

The outperformance of UPF stems from drawing particles from the *a posteriori* density function. In this way, *a posteriori* particles are more likely to be in the region of high likelihood (i.e., a proper choice of importance density function may drastically improve the efficiency of the filter). To show this effect, in **Figure 2-5**, the numbers of heavily weighted particles in BPF and UPF are given. For 200 equally weighted particles, their normalized weights are 0.005. Thus, in the figure, the particles with normalized weights larger than 0.01 are considered as heavily weighted ones.

Not surprisingly, the number from the UPF is statistically larger than that of the BPF. Although the numbers for both filters are small due to the fact that we have a very narrow shape likelihood density function (i.e., $v_k \sim \mathcal{N}(0, 1\mathrm{e}-5)$).

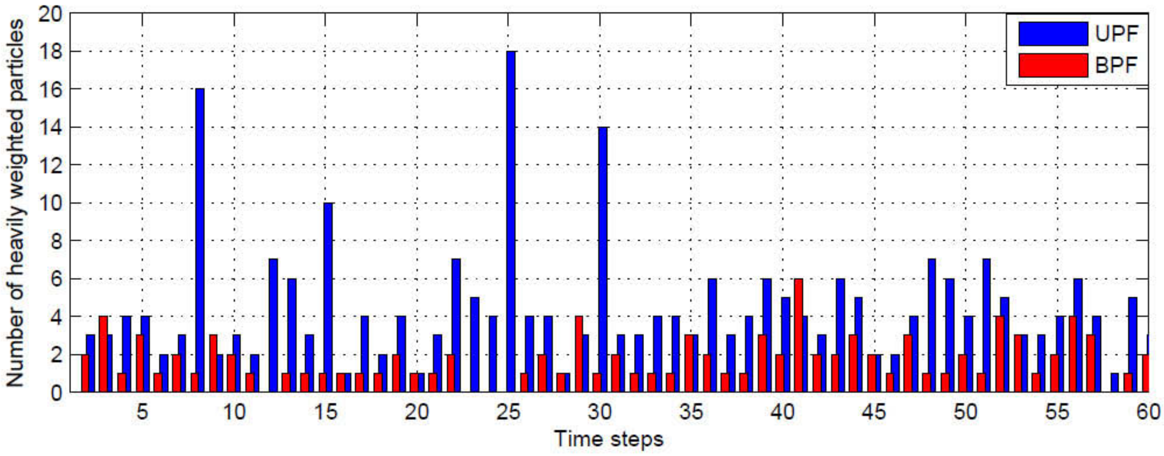


Figure 2-5. Number of heavily weighted particles

The difference of UPF and BPF can be further illustrated in **Figure 2-6**, **Figure 2-7**. The computation of weights through the likelihood density function in BPF is illustrated in **Figure 2-6**.

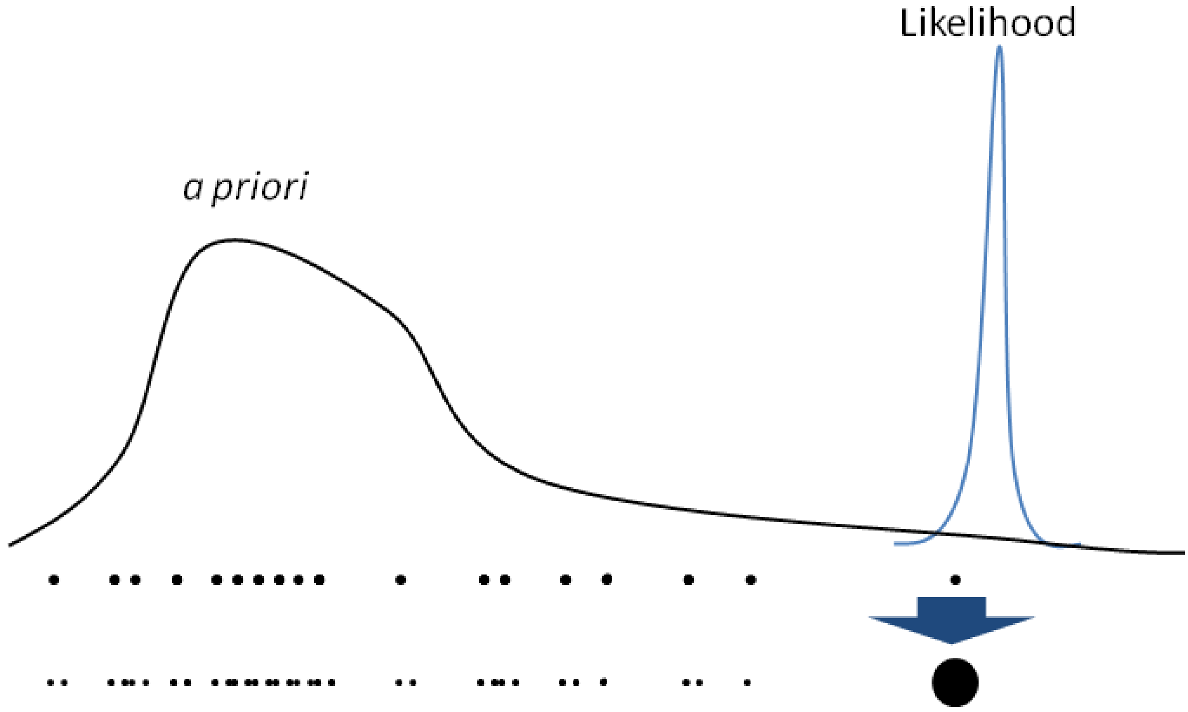


Figure 2-6. Computation of weights through likelihood density function in bootstrap particle filter algorithm

In BPF algorithm, the paramount importance of the likelihood often happens at the tail of the *a priori* density function [2], because the filter does not have measurement update process. Thus, a large amount of particles are needed to ensure that there will be certain number of particles located in the region of high likelihood. Otherwise the accuracy of the particle filter will be largely decreased. Or simply after a few recursions, the filter diverges. That is, after several recursions in the BPF, only one or two particles are significantly weighted, which are obviously inadequate for representing the probability distribution of state.

For the UPF algorithm, as shown in **Figure 2-7**, the measurement update moves *a posteriori* particles closer to the region of high likelihood. And correspondingly, the number of particles can be reduced. However, unlike the BPF, where the importance weights are only determined by the likelihood density function, in UPF, the *a priori*, *a posteriori* and likelihood density functions all involve in the computation of importance weights. (i.e., step 9 in **Table 2-5**).

Usually, if the filter puts more confidence on measurements, after measurement update, the weights of *a posteriori* particles evaluated on the basis of *a priori* density (i.e., $\mathcal{N}(\boldsymbol{\chi}_{k,i};\hat{\mathbf{x}}_k^-,\mathbf{P}_k^-)$) are small. In order to prevent mathematical problems, a small dummy weight can be introduced and added onto these weights. Thus, the importance weights of particles will mainly depend on the *a posteriori* (i.e., $\mathcal{N}(\boldsymbol{\chi}_{k,i};\hat{\mathbf{x}}_k^+,\mathbf{P}_k^+)$) and likelihood density functions $f_{\mathbf{y}_k|\mathbf{x}_k}\left(\boldsymbol{\gamma}_k \mid \boldsymbol{\chi}_{k,i}\right)$.

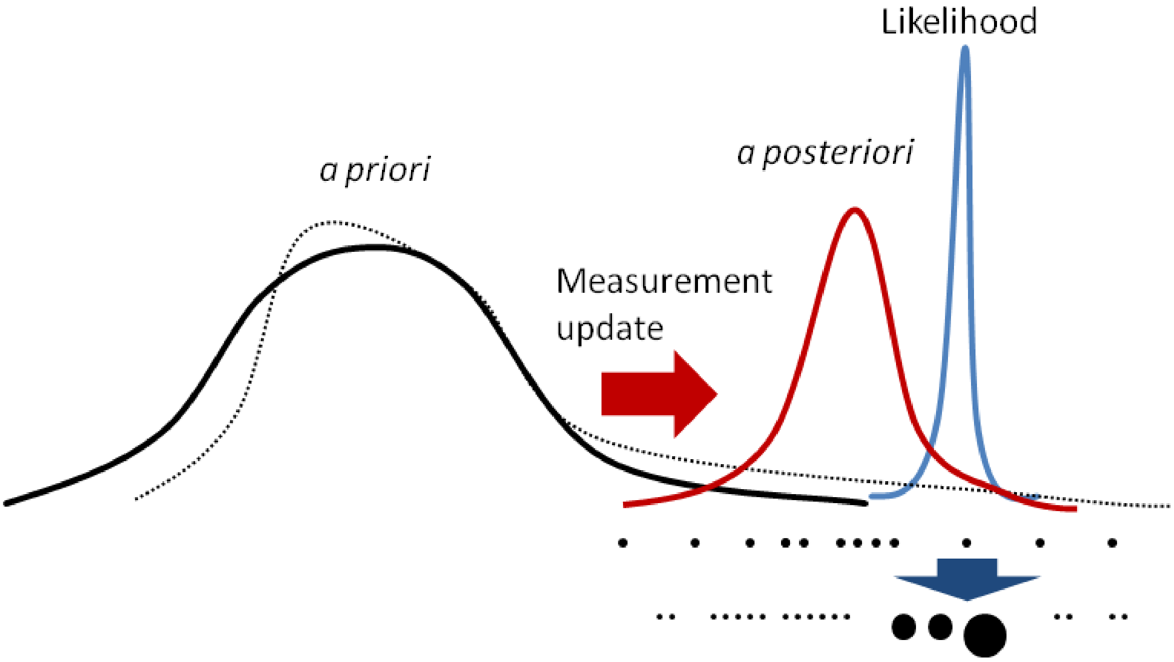


Figure 2-7. Computation of weights through likelihood density function in unscented particle filter algorithm

If the *a posteriori* density has a very small covariance parameter, the UPF could present the same estimation result as that of the UKF algorithm. This is because, if the paramount importance of *a posteriori* and likelihood density functions are at tails of each other. All *a posteriori* particles will be negligibly weighted. Thus, the mean of these equally negligibly weighted particles returns the mean of *a posteriori* distribution (i.e., the UKF *a posteriori* mean estimate). This problem can be solved by giving a relatively larger measurement error covariance parameter **R** in the measurement update phase of UKF to enlarge the variance of its *a posteriori* density function. Doing so may cause the shift of *a posteriori* density function a little bit away from the paramount region of likelihood. Nevertheless, by increasing the variance, the probability of having particles in the region of high likelihood is increased.

2.9 Summary

In this chapter, a UPF algorithm is used which combines the PF with a UKF algorithm. The UKF *a posteriori* density function is used as the importance density for drawing particles intelligently. The re-sampling step is not required, and the filter can present highly accurate estimation results using a small number of particles. Nevertheless, the UPF is still a type of recursive Gaussian particle filter, which approximates densities in the algorithm as Gaussian distributed. This Gaussian assumption can be relaxed and in general be non-Gaussian if the importance density can be approximated as Gaussian distribution, and the likelihood density function can be quantified.

A nonlinear non-Gaussian estimation problem has been used to verify the outperformance of UPF. In the next chapter, the UPF algorithm will be applied on INS/GPS integration system.

3. INS/GPS using Quaternion-based Nonlinear Filtering Methods

3.1 Introduction

For the integration of INS/GPS, we follow a tightly-coupled approach in this chapter.

The most common application of the KF on nonlinear systems is the extended Kalman filter (EKF) [44, 45], which is based on a first-order linearization of nonlinear stochastic system models with the assumption of Gaussian distributed noises. Although the EKF maintains the elegant and computationally efficient update form of the KF, it suffers from a number of drawbacks. That is, the linearized transformations are only reliable, if the error propagation can be well approximated by a linear function. That is, the small error tolerances of the EKF can cause inconsistency of the covariance update while using first order approximations and lead to filter instability in the presence of higher order effects [46, 47]. Besides, the derivation of Jacobian matrix can be a complicated mathematical task. In certain circumstances, more robust and improved system performance is demanded.

Since 1995, the unscented Kalman filter (UKF) has been proposed as an alternative. It can accurately capture the mean and covariance estimates up to the 3rd order of a Taylor series expansion for any nonlinearity [20, 23, 26] with Gaussian inputs, leading to faster convergence from inaccurate initial conditions in estimation problems [47]. Many researchers have investigated the UKF-based INS/GPS systems and reported their performances [27, 47-61].

Lately, the Particle Filter (PF) as a nonlinear/non-Gaussian estimation method [21] has become attractive to be used in navigation and tracking applications, which has the potential to constitute a better solution in these field. An idea of hardware implementation of PF on FPGA has been introduced in [62]. The INS/GPS using PF-based approaches can be found in [63-65], and a reduced-INS/GPS using PF for land vehicle navigation is presented in [66]. For reducing the processing load of PF without degrading much system estimation accuracy, some researchers also proposed to combine the PF with other filters (i.e., EKF and UKF) to form the extended particle filter (EPF) and the unscented particle filter (UPF) [18, 22, 28, 30, 55, 67-69]. Such approaches can present robust system performances with only a small number of particles.

Regarding the attitude, the quaternions are used as the representation of attitude in this chapter. We present two approaches in the propagation of the quaternion vector over time. In the first method, the quaternion vector is considered as a usual vector on which the vector addition operation is directly applied. In the second method, the quaternion vector is transformed into the rotational space to preserve the nonlinear nature of the unit vector [51, 70, 71]. In this way, the quaternion vector is updated using the quaternion product chain rule, having a natural way of maintaining the normalization constraint. That is, the successive rotation can be accomplished using quaternion multiplication in the same order as the direction cosine matrix multiplication [72]. We apply different nonlinear filtering approaches using these two quaternion propagation methods. In the EKF algorithm, the first approach is used. In the UKF and UPF algorithms, the second approach is used.

In the remainder of this chapter, we first formulate the system stochastic models of the quaternion-based INS/GPS using EKF, UKF and UPF algorithms. And then, three field experiments are conducted. Numerical results are compared and analyzed. The following topics are in the focus.

- What is the advantage of a quaternion-based approach with respect to the Euler angle-based approach?
- Do different quaternion propagation approaches yield different attitude estimation results?
- How about the nonlinearity in the quaternion-based system models?
- Using different levels of IMU, will the integrated system present significantly different estimation results?
- What is the system performance using UPF if a small number of particles is employed?
- Does UPF presents different estimation results with respect to other nonlinear filtering approaches?

3.2 Quaternion-based INS/GPS using Extended Kalman filter

3.2.1 Algorithm

Among many definitions of a quaternion vector, in this thesis, the quaternion vector is denoted as $\mathbf{q} = [q_1\ \vec{\mathbf{q}}^T]^T$, where $\vec{\mathbf{q}} = [q_2\ q_3\ q_4]^T$. It is used to represent the rotation from the navigation frame to body frame.

The attitude differential equation in terms of the quaternion vector $\mathbf{q}$ is given in Equation (3.1). For a detailed derivation, the reader is referred to [6, 13].

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{Q}_{\boldsymbol{\omega}_{bn}^b}\mathbf{q}, \text{ with } \mathbf{Q}_{\boldsymbol{\omega}_{bn}^b} = \left[\begin{array}{c|c} 0 & -\left[\boldsymbol{\omega}_{bn}^b\right]^T \\ \hline \boldsymbol{\omega}_{bn}^b & [\boldsymbol{\omega}_{bn}^b \times] \end{array}\right] \tag{3.1}$$

where $\boldsymbol{\omega}_{bn}^b$ is the angular rate measurement vector from the navigation frame to the body frame, expressed in the body frame, which is equal to:

$$\boldsymbol{\omega}_{bn}^b = \boldsymbol{\omega}_{in}^b - \boldsymbol{\omega}_{ib}^b \tag{3.2}$$

where $\boldsymbol{\omega}_{ib}^b$ is the rotational rate vector of the body frame relative to the inertial frame, expressed in the body frame (i.e., IMU gyroscope raw measurements); $\boldsymbol{\omega}_{in}^b$ represents the sum of the rotation of the earth with respect to the inertial frame plus the turn rate of the navigation frame with respect to the earth, expressed in the body frame, i.e., $\boldsymbol{\omega}_{in}^b = \mathbf{R}_n^b\left(\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n\right)$.

Using a low-cost MEMS-based IMU, the earth rotation is often buried in sensor errors, and cannot be detected by the sensor. Thus, the Coriolis and centrifugal terms are not considered in the following. Moreover, for short distance applications, the transport rate is negligible. Considering these effects, we have $\boldsymbol{\omega}_{in}^b = \mathbf{0}$. Thus, Equation (3.2) turns to be:

$$\boldsymbol{\omega}_{bn}^b = \mathbf{0} - \boldsymbol{\omega}_{ib}^b \Rightarrow \boldsymbol{\omega}_{bn}^b = -\boldsymbol{\omega}_{ib}^b \tag{3.3}$$

where $\boldsymbol{\omega}_{ib}^b = \left[\omega_{ib,x}^b \quad \omega_{ib,y}^b \quad \omega_{ib,z}^b\right]^T$ is the gyroscope raw data, resolved in the body frame.

Using Equations (3.2) and (3.3), Equation (3.1) can be approximated as:

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{Q}_{\boldsymbol{\omega}_{bn}^b}\mathbf{q}, \text{ with } \mathbf{Q}_{\boldsymbol{\omega}_{bn}^b} = \left[\begin{array}{c|c} 0 & -\left[\boldsymbol{\omega}_{bn}^b\right]^T \\ \hline \boldsymbol{\omega}_{bn}^b & [\boldsymbol{\omega}_{bn}^b \times] \end{array}\right] \approx \left[\begin{array}{c|c} 0 & \left[\boldsymbol{\omega}_{ib}^b\right]^T \\ \hline -\boldsymbol{\omega}_{ib}^b & [-\boldsymbol{\omega}_{ib}^b \times] \end{array}\right] \tag{3.4}$$

The simplified mechanization model for the IMU can be expressed in the navigation frame, as shown in Equation (3.5). More sophisticated models can be found in [45, 73, 74].

$$\begin{aligned} \dot{\mathbf{p}}_n &= \mathbf{v}_n \\ \dot{\mathbf{v}}_n &= \mathbf{R}_b^n(\mathbf{q})\mathbf{f}_{ib}^b + \mathbf{g}_n \\ \dot{\mathbf{q}} &= \frac{1}{2}\mathbf{Q}_{\boldsymbol{\omega}_{bn}^b}\mathbf{q} \end{aligned} \tag{3.5}$$

Here $\mathbf{g}_n$ represents gravity indicated in the navigation frame, which is assumed to be constant. The rotational transformation matrix $\mathbf{R}_b^n(\mathbf{q})$ from the body frame to the navigation frame is expressed using quaternions as:

$$\mathbf{R}_b^n(\mathbf{q}) = \begin{bmatrix} q_1^2+q_2^2-q_3^2-q_4^2 & 2(q_2q_3-q_1q_4) & 2(q_1q_3+q_2q_4) \\ 2(q_2q_3+q_1q_4) & q_1^2-q_2^2+q_3^2-q_4^2 & 2(q_3q_4-q_1q_2) \\ 2(q_2q_4-q_1q_3) & 2(q_1q_2+q_3q_4) & q_1^2-q_2^2-q_3^2+q_4^2 \end{bmatrix}^T \tag{3.6}$$

The IMU specific force and angular rate measurement errors (e.g., sensor biases) are modeled as constants superseded by random walk as:

$$\begin{aligned} \dot{\mathbf{f}}_b^{bias} &= \mathbf{w}_f \\ \dot{\boldsymbol{\omega}}_b^{bias} &= \mathbf{w}_\omega \end{aligned} \tag{3.7}$$

where $\mathbf{w}_f$ and $\mathbf{w}_\omega$ are assumed to be zero mean, Gaussian distributed.

It is worth mentioning that using Equation (3.5) as the INS/GPS system propagation model, the IMU incoming measurements should be compensated by the current estimate of the sensor biases before further processing, i.e., $\mathbf{f}_{ib}^b = \tilde{\mathbf{f}}_{ib}^b - \mathbf{f}_b^{bias} - \mathbf{n}_f$, $\boldsymbol{\omega}_{ib}^b = \tilde{\boldsymbol{\omega}}_{ib}^b - \boldsymbol{\omega}_b^{bias} - \mathbf{n}_w$ ($\mathbf{n}_f$ and $\mathbf{n}_w$ are the remaining noises which are assumed to be zero-mean, Gaussian distributed).

In the discrete-time domain with a sufficiently small time interval (e.g., 0.01s IMU update rate), and for low dynamic applications, we have:

$$\begin{aligned} \mathbf{p}_{n,k+1} &= \mathbf{p}_{n,k} + \mathbf{v}_{n,k} \cdot T \\ \mathbf{v}_{n,k+1} &= \mathbf{v}_{n,k} + \left[\mathbf{R}_b^n(\mathbf{q}_k)\mathbf{f}_{ib,k}^b + \mathbf{g}_n\right] \cdot T \\ \mathbf{q}_{k+1} &= \mathbf{q}_k + \left(\frac{1}{2}\mathbf{Q}_{\boldsymbol{\omega}_{bn,k}^b} \cdot T\right)\mathbf{q}_k \\ \mathbf{f}_{b,k+1}^{bias} &= \mathbf{f}_{b,k}^{bias} + \mathbf{w}_{f,k} \\ \boldsymbol{\omega}_{b,k+1}^{bias} &= \boldsymbol{\omega}_{b,k}^{bias} + \mathbf{w}_{\omega,k} \end{aligned} \tag{3.8}$$

where “*T*” is system propagation time interval.

In Equation (3.8), a part of the system propagation model is nonlinear, e.g., $\mathbf{R}_b^n(\mathbf{q})$ contains quadratic terms of quaternion elements. Therefore, the linearization process should be conducted in order to apply the KF equations on it. Besides, in the scope of a tightly-coupled integration approach, the receiver clock errors need to be modeled. The range-rate equivalent of the clock drift error is modeled as a constant plus a random walk process, while the range equivalent of the receiver clock bias error is the integral of the clock drift error. Thus, the linearized system propagation model used for INS/GPS tightly-coupled integration is formulated in Equation (3.9), where error states are employed.

$$\begin{bmatrix} \delta\mathbf{p}_{n,k+1} \\ \delta\mathbf{v}_{n,k+1} \\ \delta\mathbf{q}_{k+1} \\ \delta\mathbf{f}_{b,k+1}^{bias} \\ \delta\boldsymbol{\omega}_{b,k+1}^{bias} \\ c\delta t_{k+1} \\ c\delta\dot{t}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3}\cdot T & \mathbf{O}_{3\times4} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{F}_{23,k}\cdot T & -\mathbf{R}_b^n(\mathbf{q}_k)\cdot T & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{4\times3} & \mathbf{O}_{4\times3} & \mathbf{I}_{4\times4}+\frac{1}{2}\mathbf{Q}_{\boldsymbol{\omega}_{bn,k}^{b}}\cdot T & \mathbf{O}_{4\times3} & \frac{1}{2}\boldsymbol{\Omega}_{\mathbf{q}_k}\cdot T & \mathbf{0}_{4\times1} & \mathbf{0}_{4\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times4} & \mathbf{I}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times4} & \mathbf{O}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{4\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & 1 & T \\ \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{4\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{p}_{n,k} \\ \delta\mathbf{v}_{n,k} \\ \delta\mathbf{q}_{k} \\ \delta\mathbf{f}_{b,k}^{bias} \\ \delta\boldsymbol{\omega}_{b,k}^{bias} \\ c\delta t_{k} \\ c\delta\dot{t}_{k} \end{bmatrix} + \mathbf{w}_k \tag{3.9}$$

where

$$\boldsymbol{\Omega}_{\mathbf{q}_k} = \left[\begin{array}{c} -\vec{\mathbf{q}}^T \\ \hline q_1\mathbf{I}_{3\times3} - [\vec{\mathbf{q}}\times] \end{array}\right]_k = \left[\begin{array}{ccc} -q_2 & -q_3 & -q_4 \\ \hline q_1 & q_4 & -q_3 \\ -q_4 & q_1 & q_2 \\ q_3 & -q_2 & q_1 \end{array}\right]_k \tag{3.10}$$

and $\mathbf{F}_{23,k}$ is computed as:

$$\mathbf{F}_{23,k} = 2\cdot\begin{bmatrix} q_1\hat{f}_x + q_4\hat{f}_y - q_3\hat{f}_z & q_2\hat{f}_x + q_3\hat{f}_y + q_4\hat{f}_z & -q_3\hat{f}_x + q_2\hat{f}_y - q_1\hat{f}_z & q_1\hat{f}_y - q_4\hat{f}_x + q_2\hat{f}_z \\ q_1\hat{f}_y - q_4\hat{f}_x + q_2\hat{f}_z & q_3\hat{f}_x - q_2\hat{f}_y + q_1\hat{f}_z & q_2\hat{f}_x + q_3\hat{f}_y + q_4\hat{f}_z & -q_1\hat{f}_x - q_4\hat{f}_y + q_3\hat{f}_z \\ q_3\hat{f}_x - q_2\hat{f}_y + q_1\hat{f}_z & -q_1\hat{f}_y + q_4\hat{f}_x - q_2\hat{f}_z & q_1\hat{f}_x + q_4\hat{f}_y - q_3\hat{f}_z & q_2\hat{f}_x + q_3\hat{f}_y + q_4\hat{f}_z \end{bmatrix}_k \tag{3.11}$$

In Equation (3.11), $\hat{f}_x = \left(\tilde{f}_{ib,x}^{b} - \hat{f}_{b,x}^{bias}\right)_k$, $\hat{f}_y = \left(\tilde{f}_{ib,y}^{b} - \hat{f}_{b,y}^{bias}\right)_k$, $\hat{f}_z = \left(\tilde{f}_{ib,z}^{b} - \hat{f}_{b,z}^{bias}\right)_k$ are the IMU raw data compensated by the current estimate of the sensor biases expressed in the body frame.

For the INS/GPS tightly-coupled observation model, Equation (1.45) is used. Thus we have completed the description of system models. As opposed to the Euler angle-based EKF model (i.e., Equation (1.43)), Equation (3.9) does not involve trigonometric operations, which are not only error prone when implementing, but also exhibit the potential of singularities. In addition, the quaternion-based approach potentially provides speed improvements due to the reduction of computational complexity and accuracy improvements when applied to a programmable processor. We will now make a field test based on a train ride to compare the estimation accuracy of attitude and gyro bias errors from the quaternion-based and Euler angle-based EKF algorithms.

3.2.2 Field experiment: comparison between quaternion-based and Euler angle-based INS/GPS using EKF

The trajectory of this field test starts from Haiger to Siegen in the North West of Germany. It lasts about 1400 s, as shown in **Figure 3-1**. The number of tracked satellites is depicted in **Figure 3-2**. where the first GPS outage environment lasts 93 s, and the second one takes 43 s. The Landmark™20 MEMS-based IMU and a u-blox Antaris 4 receiver are used in this test.

Figure 3-1. A train ride trajectory computed from processing the L1 GPS pseudorange measurements using a least-squares estimation method (plotted in Google earth)

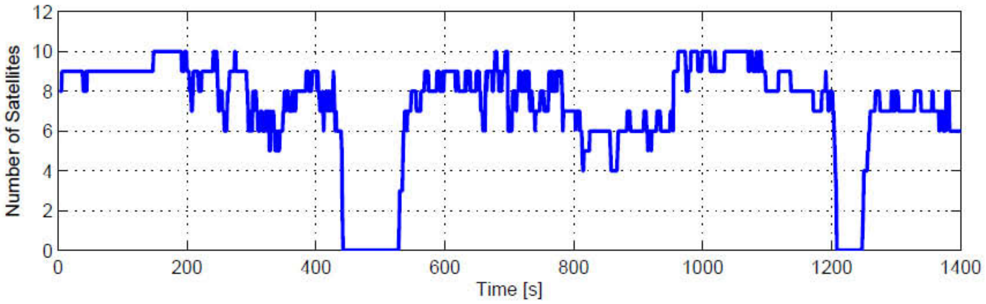


Figure 3-2. Number of satellites in view

The attitude estimation comparison (e.g., mean and variance) is shown in **Figure 3-3** and **Figure 3-4**, while the gyro bias estimation comparison (e.g., mean and variance) is given in **Figure 3-5** and **Figure 3-6**.

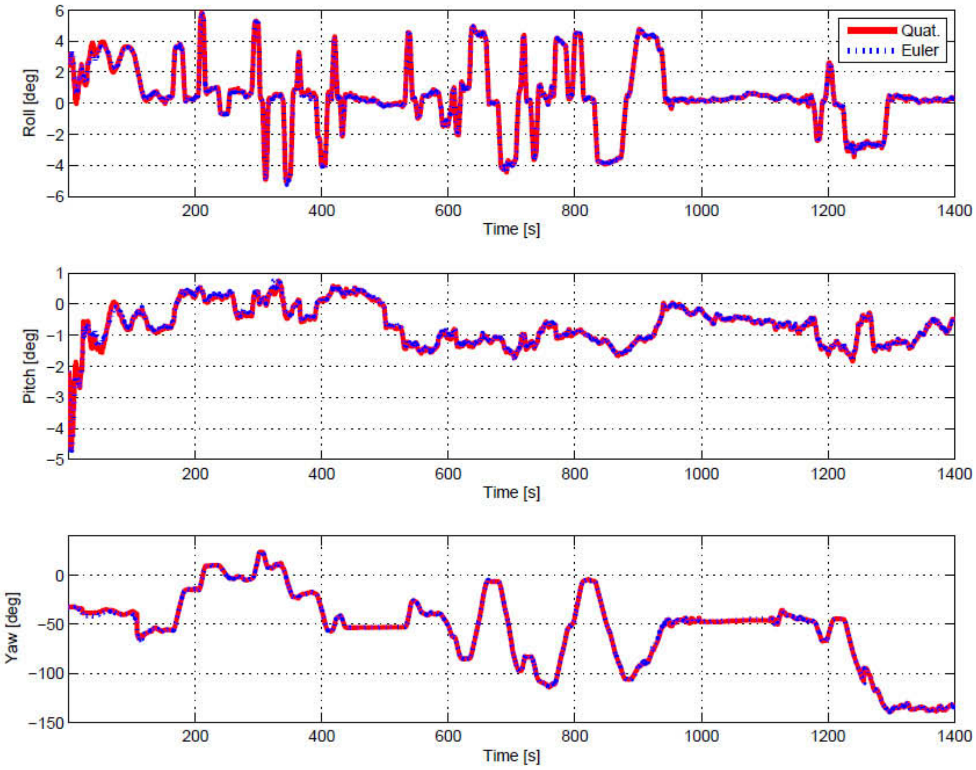


Figure 3-3. Attitude estimation comparison between quaternion-based and Euler angle-based EKF algorithms

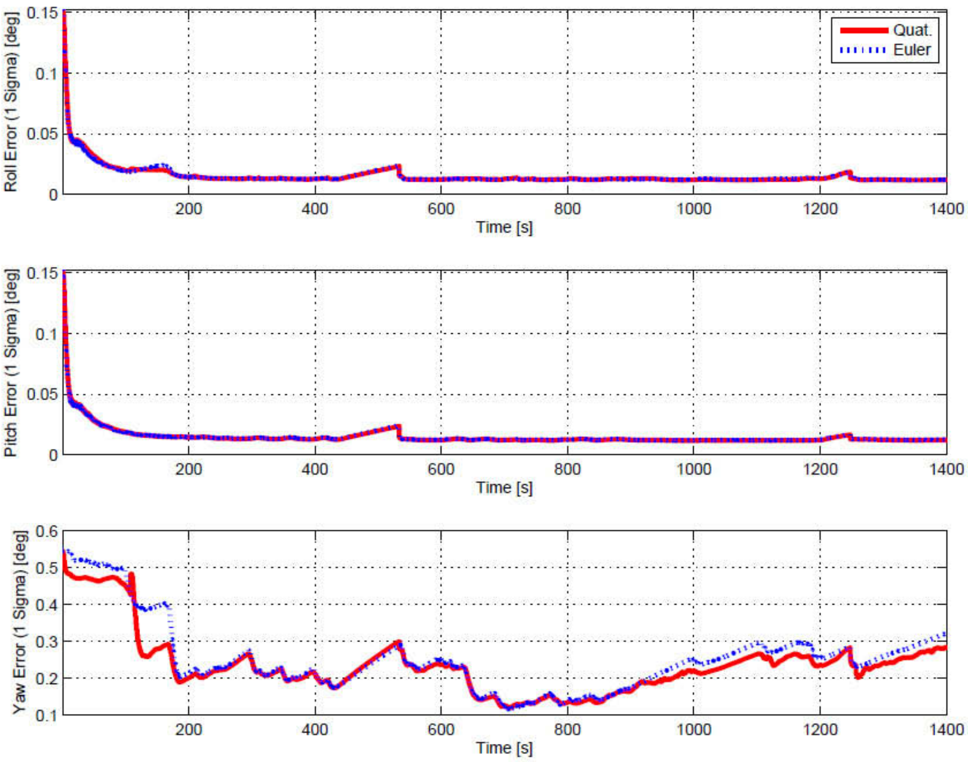


Figure 3-4. Attitude variance (1 sigma) estimation comparison between quaternion-based and Euler angle-based EKF algorithms

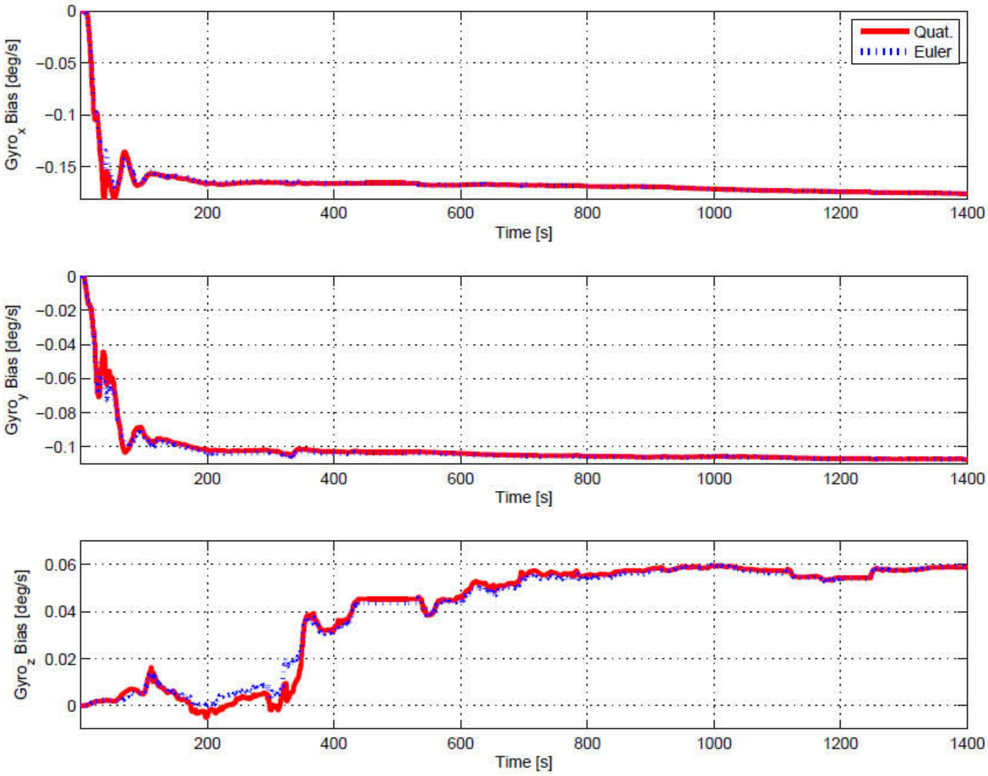


Figure 3-5. Gyro bias estimation comparison between quaternion-based and Euler angle-based EKF algorithms

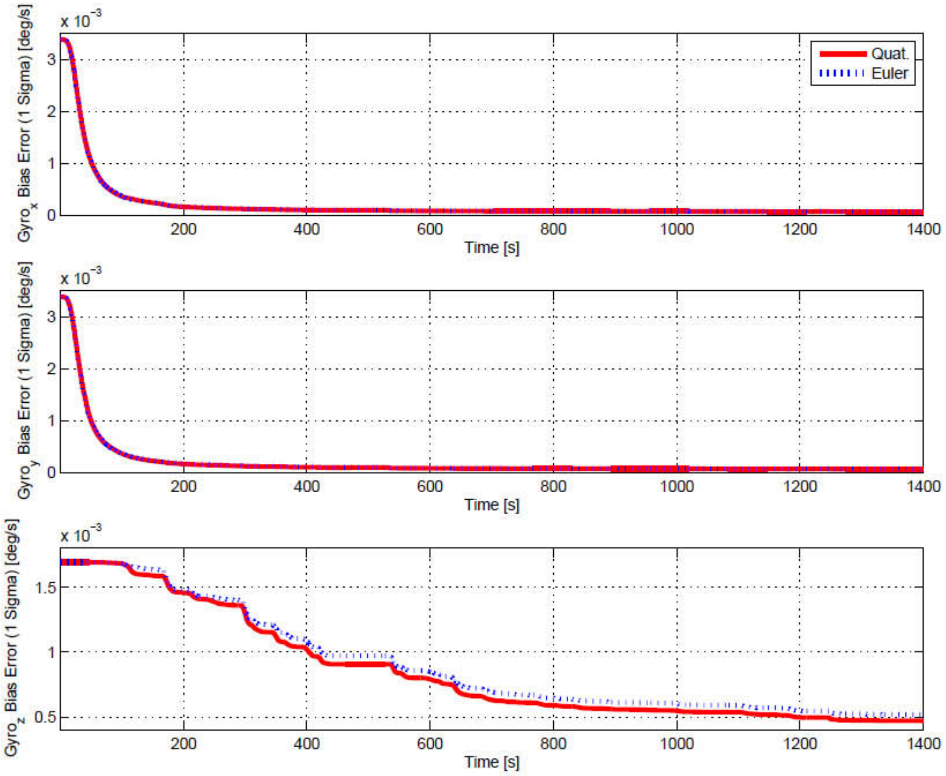


Figure 3-6. Gyro bias variance (1 sigma) estimation comparison between quaternion-based and Euler angle-based EKF algorithms

For the land-based navigation, the roll and pitch angles usually change little and both approaches present almost identical results. However, on the estimation of heading, small differences can be observed. By comparing their estimated variance parameters, we notice that the quaternion-based approach is more confident on the estimation of heading.

On the estimation of Gyro bias errors, the initial Gyro bias on 3 axes are set to be 0, because we have not conducted sensor calibration before the test starts. It is a nice opportunity to compare the filter convergence performances from both approaches. As shown in **Figure 3-5** and **Figure 3-6**, the difference mainly locates at the estimation of Gyro bias at z-axis (mainly affecting the heading). By looking at the variance parameter, the one estimated from quaternions is smaller than the one from Euler angle, which hints that the quaternion-based approach is more confident on the estimation of Gyro z-axis bias error than the Euler angle-based approach.

It is worth mentioning that in this comparison we have initialized both filters with exactly the same initial parameters (initial states and their covariance) and the same tuning parameters. The transformation of quaternion covariance to Euler angle covariance is derived in Appendix B. Besides, both approaches use EKF. Therefore, the difference we observed should come from the different representations of attitude in the system stochastic models.

From this field test, we observe that the quaternion-based approach is more confident on the estimation of heading related states than the Euler angle-based approach (i.e., without trigonometric terms, and the estimation of attitude converges faster). In the next section, we introduce the quaternion-based UKF algorithm.

3.3 Quaternion-based INS/GPS using Unscented Kalman filter

3.3.1 Algorithm

In Equation (3.8), the quaternion vector is updated using vector addition. It is fine in case the time interval is very small. However, this may cause errors, as the usual definitions of vector addition and scaling can normally not be applied directly, due to the fact that the unit sphere defined by quaternions is not an Euclidean vector space [13]. In the UKF algorithm, we will apply the quaternion product chain rule to update the quaternion vector.

In this approach, the quaternion vector is transformed into the rotational space to preserve the nonlinear nature of the unit quaternion. This transformation will be carried out throughout the derivation given below. In this way, the quaternion vector is updated using the quaternion product chain rule, leading to a natural way of maintaining the normalization constraint. That is, the successive rotations can be accomplished using quaternion multiplication in the same order as the direction cosine matrix multiplication [72, 75].

Unlike in Equation (3.8), now the system position, velocity and attitude updates in the discrete time domain are formulated in Equation (3.12).

$$\begin{aligned} \mathbf{p}_{n,k+1} &= \mathbf{p}_{n,k} + \mathbf{v}_{n,k} \cdot T \\ \mathbf{v}_{n,k+1} &= \mathbf{v}_{n,k} + \left[\mathbf{R}_b^n(\mathbf{q}_k)\mathbf{f}_{ib,k}^b + \mathbf{g}_n \right] \cdot T \\ \mathbf{q}_{k+1} &= \Delta\mathbf{q}_k \otimes \mathbf{q}_k \end{aligned} \tag{3.12}$$

where ⊗ denotes the quaternion product; $\Delta\mathbf{q}_k = \begin{bmatrix} \cos(0.5\|\boldsymbol{\theta}_k\|) \\ \dfrac{\sin(0.5\|\boldsymbol{\theta}_k\|)}{0.5\|\boldsymbol{\theta}_k\|} 0.5\boldsymbol{\theta}_k \end{bmatrix}$; $\boldsymbol{\theta}_k$ represents the integral of the IMU body frame angular rate measurements over IMU measurement update interval (e.g., k-1 to k).

The sensor biases, e.g., $\mathbf{f}_{b,k}^{bias}$ and $\boldsymbol{\omega}_{b,k}^{bias}$, are modeled as constants superseded by random walks as:

$$\begin{aligned} \mathbf{f}_{b,k+1}^{bias} &= \mathbf{f}_{b,k}^{bias} + \mathbf{w}_f \\ \boldsymbol{\omega}_{b,k+1}^{bias} &= \boldsymbol{\omega}_{b,k}^{bias} + \mathbf{w}_\omega \end{aligned} \tag{3.13}$$

where $\mathbf{w}_f$ and $\mathbf{w}_\omega$ are assumed to be zero mean, Gaussian distributed.

The incoming specific force and angular rate raw data are compensated with the estimated sensor biases and used in Equation (3.12). The range-rate-type clock drift error is modeled as a constant plus a random walk process, while the receiver clock bias is modeled as the integral of the clock drift error.

$$\begin{bmatrix} c\Delta t_{k+1} \\ c\Delta \dot{t}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c\Delta t_k \\ c\Delta \dot{t}_k \end{bmatrix} + \mathbf{w}_{clk} \tag{3.14}$$

Equation (3.12), (3.13) and (3.14) form the system propagation model. For the observation model, the nonlinear equations are directly employed.

The predicted pseudorange measurement based on the current position estimate is formulated as:

$$\hat{\rho}_k^j = \sqrt{(x_{n,k}^j - \hat{x}_{n,k})^2 + (x_{e,k}^j - \hat{x}_{e,k})^2 + (x_{d,k}^j - \hat{x}_{d,k})^2} + c\Delta\hat{t}_k \tag{3.15}$$

where $\hat{\rho}_k^j$ is the predicted pseudorange measurement from the j-th satellite; $x_{n,k}^j, x_{e,k}^j, x_{d,k}^i$ are the j-th satellite position coordinates expressed in the NED navigation frame; $\hat{x}_{n,k}, \hat{x}_{e,k}, \hat{x}_{d,k}$ are the vehicle position estimates resolved in the NED navigation frame.

The predicted pseudorange-rate measurements are related to the velocity estimates as:

$$\hat{\dot{\rho}}_k^j = a_{n,k}^j(\dot{x}_{n,k}^j - \hat{\dot{x}}_{n,k}) + a_{e,k}^j(\dot{x}_{e,k}^j - \hat{\dot{x}}_{e,k}) + a_{d,k}^j(\dot{x}_{d,k}^j - \hat{\dot{x}}_{d,k}) + c\Delta\hat{\dot{t}}_k \tag{3.16}$$

where $a_{n,k}^j = \dfrac{x_{n,k}^j - \hat{x}_{n,k}}{\hat{d}_k^j}, a_{e,k}^j = \dfrac{x_{e,k}^j - \hat{x}_{e,k}}{\hat{d}_k^j}, a_{d,k}^j = \dfrac{x_{d,k}^j - \hat{x}_{d,k}}{\hat{d}_k^j}$

$$\hat{d}_k^j = \sqrt{(x_{n,k}^j - \hat{x}_{n,k})^2 + (x_{e,k}^j - \hat{x}_{e,k})^2 + (x_{d,k}^j - \hat{x}_{d,k})^2}$$

where $\hat{\dot{\rho}}_k^j$ is the predicted delta range measurement from the *j*-th satellite; $\dot{x}_{n,k}^j, \dot{x}_{e,k}^j, \dot{x}_{d,k}^j$ are the *j*-th satellite velocity coordinates expressed in NED navigation frame; $\hat{\dot{x}}_{n,k}, \hat{\dot{x}}_{e,k}, \hat{\dot{x}}_{d,k}$ are the vehicle velocity estimates in NED navigation frame.

Given the system propagation and observation models, the UKF algorithm can now be applied. Among several UKF algorithms, the approach employing 2n equally weighted sigma points is used, which can be found in [20, 22]. Due to the quaternion normalization constraint, the degree of freedom of a quaternion vector is three rather than four. Thus, if we use quaternion vector elements as states, the dimension of the state vector is 18×1, but the dimension of state error covariance matrix is 17×17. This dimensional mismatch can be solved by transforming the quaternion vector error into its corresponding rotation vector in the rotation space. We define the state vector and its associated errors (i.e., estimate minus truth) as:

$$\hat{\mathbf{x}}_k^+ = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^+ \\ \hat{\mathbf{v}}_{n,k}^+ \\ \hat{\mathbf{q}}_k^+ \\ \hat{\mathbf{f}}_{b,k}^{bias+} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias+} \\ c\Delta\hat{t}_k^+ \\ c\Delta\hat{\dot{t}}_k^+ \end{bmatrix}_{18\times1}, \delta\hat{\mathbf{x}}_k^+ = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^+ - \mathbf{p}_{n,k} \\ \hat{\mathbf{v}}_{n,k}^+ - \mathbf{v}_{n,k} \\ \hat{\boldsymbol{\varphi}}_k^+ \\ \hat{\mathbf{f}}_{b,k}^{bias+} - \mathbf{f}_{b,k}^{bias} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias+} - \boldsymbol{\omega}_{b,k}^{bias} \\ c\Delta\hat{t}_k^+ - c\Delta t_k \\ c\Delta\hat{\dot{t}}_k^+ - c\Delta\dot{t}_k \end{bmatrix} = \begin{bmatrix} \delta\hat{\mathbf{p}}_{n,k}^+ \\ \delta\hat{\mathbf{v}}_{n,k}^+ \\ \hat{\boldsymbol{\varphi}}_k^+ \\ \delta\hat{\mathbf{f}}_{b,k}^{bias+} \\ \delta\hat{\boldsymbol{\omega}}_{b,k}^{bias+} \\ c\delta\hat{t}_k^+ \\ c\delta\hat{\dot{t}}_k^+ \end{bmatrix}_{17\times1} \tag{3.17}$$

where $\hat{\boldsymbol{\varphi}}_k^+$ is the rotation vector corresponding to $\hat{\mathbf{q}}_k^+ \otimes \left(\mathbf{q}_k\right)^{-1}$.

The state error covariance matrix can be formulated as:

$$\mathbf{P}_k^+ = \begin{bmatrix} \left(\sigma_{\delta\hat{\mathbf{p}}}^2\right)_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \left(\sigma_{\delta\hat{\mathbf{v}}}^2\right)_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \left(\sigma_{\hat{\boldsymbol{\varphi}}}^2\right)_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \left(\sigma_{\delta\hat{\mathbf{f}}}^2\right)_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \mathbf{O}_{3\times3} & \left(\sigma_{\delta\hat{\boldsymbol{\omega}}}^2\right)_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \left(\sigma_{c\delta\hat{t}}^2\right)_{1\times1} & 0_{1\times1} \\ \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & \mathbf{0}_{3\times1}^T & 0_{1\times1} & \left(\sigma_{c\delta\hat{\dot{t}}}^2\right)_{1\times1} \end{bmatrix}_{17\times17} \tag{3.18}$$

We generate 2n (n=17) equally weighted sigma points as:

$$\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T = \begin{bmatrix} \Delta\hat{\mathbf{p}}_{n,k-1,i}^{+} \\ \Delta\hat{\mathbf{v}}_{n,k-1,i}^{+} \\ \hat{\boldsymbol{\varphi}}_{k-1,i}^{+} \\ \Delta\hat{\mathbf{f}}_{b,k-1,i}^{bias+} \\ \Delta\hat{\boldsymbol{\omega}}_{b,k-1,i}^{bias+} \\ \Delta c\Delta\hat{t}_{k-1,i}^{+} \\ \Delta c\Delta\hat{\dot{t}}_{k-1,i}^{+} \end{bmatrix}, \boldsymbol{\chi}_{k-1,i}^{+} = \begin{bmatrix} \hat{\mathbf{p}}_{n,k-1}^{+} + \Delta\hat{\mathbf{p}}_{n,k-1,i}^{+} \\ \hat{\mathbf{v}}_{n,k-1}^{+} + \Delta\hat{\mathbf{v}}_{n,k-1,i}^{+} \\ \delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}) \otimes \hat{\mathbf{q}}_{k-1}^{+} \\ \hat{\mathbf{f}}_{b,k-1}^{bias+} + \Delta\hat{\mathbf{f}}_{b,k-1,i}^{bias+} \\ \hat{\boldsymbol{\omega}}_{b,k-1}^{bias+} + \Delta\hat{\boldsymbol{\omega}}_{b,k-1,i}^{bias+} \\ c\Delta\hat{t}_{k-1}^{+} + \Delta c\Delta\hat{t}_{k-1,i}^{+} \\ c\Delta\hat{\dot{t}}_{k-1}^{+} + \Delta c\Delta\hat{\dot{t}}_{k-1,i}^{+} \end{bmatrix}, \boldsymbol{\chi}_{k-1,i+n}^{+} = \begin{bmatrix} \hat{\mathbf{p}}_{n,k-1}^{+} - \Delta\hat{\mathbf{p}}_{n,k-1,i}^{+} \\ \hat{\mathbf{v}}_{n,k-1}^{+} - \Delta\hat{\mathbf{v}}_{n,k-1,i}^{+} \\ \delta\mathbf{q}\left(-\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}\right) \otimes \hat{\mathbf{q}}_{k-1}^{+} \\ \hat{\mathbf{f}}_{b,k-1}^{bias+} - \Delta\hat{\mathbf{f}}_{b,k-1,i}^{bias+} \\ \hat{\boldsymbol{\omega}}_{b,k-1}^{bias+} - \Delta\hat{\boldsymbol{\omega}}_{b,k-1,i}^{bias+} \\ c\Delta\hat{t}_{k-1}^{+} - \Delta c\Delta\hat{t}_{k-1,i}^{+} \\ c\Delta\hat{\dot{t}}_{k-1}^{+} - \Delta c\Delta\hat{\dot{t}}_{k-1,i}^{+} \end{bmatrix} \tag{3.19}$$

where $i = 1,\ldots,n,$ and we denote:

$$\delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}) = \begin{bmatrix} \cos(0.5\left\|\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}\right\|) \\ \sin(0.5\left\|\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}\right\|)\dfrac{\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}}{\left\|\hat{\boldsymbol{\varphi}}_{k-1,i}^{+}\right\|} \end{bmatrix} \tag{3.20}$$

In the time update, we pass sigma points through the nonlinear system propagation model, and compute the mean as:

$$\boldsymbol{\chi}_{k,i}^{-} = f_{k,k-1}(\boldsymbol{\chi}_{k-1,i}^{+}) = \begin{bmatrix} \hat{\mathbf{p}}_{n,k,i}^{-} \\ \hat{\mathbf{v}}_{n,k,i}^{-} \\ \hat{\mathbf{q}}_{k,i}^{-} \\ \hat{\mathbf{f}}_{b,k,i}^{bias-} \\ \hat{\boldsymbol{\omega}}_{b,k,i}^{bias-} \\ c\Delta\hat{t}_{k,i}^{-} \\ c\Delta\hat{\dot{t}}_{k,i}^{-} \end{bmatrix}, \hat{\mathbf{x}}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\boldsymbol{\chi}_{k,i}^{-} = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^{-} \\ \hat{\mathbf{v}}_{n,k}^{-} \\ \hat{\mathbf{q}}_{k}^{-} \\ \hat{\mathbf{f}}_{b,k}^{bias-} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias-} \\ c\Delta\hat{t}_{k}^{-} \\ c\Delta\hat{\dot{t}}_{k}^{-} \end{bmatrix} \tag{3.21}$$

Due to the fact that the unit quaternion is not mathematically closed for addition and scalar multiplications, the renormalization must be conducted, which is handled by replacing $\hat{\mathbf{q}}_k^{-}$ with $\dfrac{\hat{\mathbf{q}}_k^{-}}{\left\|\hat{\mathbf{q}}_k^{-}\right\|}$

For updating the state error covariance matrix, we calculate it as:

$$\mathbf{P}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left[\Delta\hat{\mathbf{x}}_k^{-}\right]\left[\Delta\hat{\mathbf{x}}_k^{-}\right]^T + \mathbf{Q}_{k-1} \tag{3.22}$$

where $\Delta\hat{\mathbf{x}}_k^- = \boldsymbol{\chi}_{k,i}^- - \hat{\mathbf{x}}_k^- = \begin{bmatrix} \hat{\mathbf{p}}_{n,k,i}^- - \hat{\mathbf{p}}_{n,k}^- \\ \hat{\mathbf{v}}_{n,k,i}^- - \hat{\mathbf{v}}_{n,k}^- \\ \hat{\boldsymbol{\varphi}}_{k,i}^- \\ \hat{\mathbf{f}}_{b,k,i}^{bias-} - \hat{\mathbf{f}}_{b,k}^{bias-} \\ \hat{\boldsymbol{\omega}}_{b,k,i}^{bias-} - \hat{\boldsymbol{\omega}}_{b,k}^{bias-} \\ c\Delta\hat{t}_{k,i}^- - c\Delta\hat{t}_k^- \\ c\Delta\hat{\dot{t}}_{k,i}^- - c\Delta\hat{\dot{t}}_k^- \end{bmatrix}$, and $\hat{\boldsymbol{\varphi}}_{k,i}^-$ is the rotation vector corresponding to $\hat{\mathbf{q}}_{k,i}^- \otimes \left(\hat{\mathbf{q}}_k^-\right)^{-1}$.

Having the *a priori* mean and covariance matrix, we generate the sigma points for measurement updates as:

$$\left(\sqrt{n\mathbf{P}_k^-}\right)_i^T = \begin{bmatrix} \Delta\hat{\mathbf{p}}_{n,k,i}^- \\ \Delta\hat{\mathbf{v}}_{n,k,i}^- \\ \hat{\boldsymbol{\varphi}}_{k,i}^- \\ \Delta\hat{\mathbf{f}}_{b,k,i}^{bias-} \\ \Delta\hat{\boldsymbol{\omega}}_{b,k,i}^{bias-} \\ \Delta c\Delta\hat{t}_{k,i}^- \\ \Delta c\Delta\hat{\dot{t}}_{k,i}^- \end{bmatrix}, \boldsymbol{\chi}_{k,i}^- = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^- + \Delta\hat{\mathbf{p}}_{n,k,i}^- \\ \hat{\mathbf{v}}_{n,k}^- + \Delta\hat{\mathbf{v}}_{n,k,i}^- \\ \delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k,i}^-) \otimes \hat{\mathbf{q}}_k^- \\ \hat{\mathbf{f}}_{b,k}^{bias-} + \Delta\hat{\mathbf{f}}_{b,k,i}^{bias-} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias-} + \Delta\hat{\boldsymbol{\omega}}_{b,k,i}^{bias-} \\ c\Delta\hat{t}_k^- + \Delta c\Delta\hat{t}_{k,i}^- \\ c\Delta\hat{\dot{t}}_k^- + \Delta c\Delta\hat{\dot{t}}_{k,i}^- \end{bmatrix}, \boldsymbol{\chi}_{k,i+n}^- = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^- - \Delta\hat{\mathbf{p}}_{n,k,i}^- \\ \hat{\mathbf{v}}_{n,k}^- - \Delta\hat{\mathbf{v}}_{n,k,i}^- \\ \delta\mathbf{q}\left(-\hat{\boldsymbol{\varphi}}_{k,i}^-\right) \otimes \hat{\mathbf{q}}_k^- \\ \hat{\mathbf{f}}_{b,k}^{bias-} - \Delta\hat{\mathbf{f}}_{b,k,i}^{bias-} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias-} - \Delta\hat{\boldsymbol{\omega}}_{b,k,i}^{bias-} \\ c\Delta\hat{t}_k^- - \Delta c\Delta\hat{t}_{k,i}^- \\ c\Delta\hat{\dot{t}}_k^- - \Delta c\Delta\hat{\dot{t}}_{k,i}^- \end{bmatrix} \tag{3.23}$$

where $i = 1,\ldots,n,$ and $\delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k,i}^-) = \begin{bmatrix} \cos(0.5\left\|\hat{\boldsymbol{\varphi}}_{k,i}^-\right\|) \\ \sin(0.5\left\|\hat{\boldsymbol{\varphi}}_{k,i}^-\right\|)\dfrac{\hat{\boldsymbol{\varphi}}_{k,i}^-}{\left\|\hat{\boldsymbol{\varphi}}_{k,i}^-\right\|} \end{bmatrix}$.

We pass them through the nonlinear measurement models and compute the predicted measurement and covariance matrices as:

$$\begin{aligned} \hat{\mathbf{y}}_k &= \frac{1}{2n}\sum_{i=1}^{2n} h_k(\boldsymbol{\chi}_{k,i}^-) \\ \mathbf{P}_k^{yy} &= \frac{1}{2n}\sum_{i=1}^{2n}\left[h_k(\boldsymbol{\chi}_{k,i}^-) - \hat{\mathbf{y}}_k\right]\left[h_k(\boldsymbol{\chi}_{k,i}^-) - \hat{\mathbf{y}}_k\right]^T + \mathbf{R}_k \\ \mathbf{P}_k^{xy} &= \frac{1}{2n}\sum_{i=1}^{2n}\left[\boldsymbol{\chi}_{k,i}^- - \hat{\mathbf{x}}_k^-\right]\left[h_k(\boldsymbol{\chi}_{k,i}^-) - \hat{\mathbf{y}}_k\right]^T \end{aligned} \tag{3.24}$$

The Kalman gain and the state correction terms are computed as:

$$\mathbf{K}_k = \mathbf{P}_k^{xy}(\mathbf{P}_k^{yy})^{-1}, \Delta\hat{\mathbf{x}}_k^+ = \mathbf{K}_k(\tilde{\mathbf{y}}_k - \hat{\mathbf{y}}_k) = \begin{bmatrix} \Delta\hat{\mathbf{p}}_{n,k}^+ \\ \Delta\hat{\mathbf{v}}_{n,k}^+ \\ \hat{\boldsymbol{\varphi}}_k^+ \\ \Delta\hat{\mathbf{f}}_{b,k}^{bias+} \\ \Delta\hat{\boldsymbol{\omega}}_{b,k}^{bias+} \\ \Delta c\Delta\hat{t}_k^+ \\ \Delta c\Delta\hat{\dot{t}}_k^+ \end{bmatrix} \tag{3.25}$$

The last step in the UKF algorithm is to apply the corrections.

$$\hat{\mathbf{x}}_k^+ = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^- + \Delta\hat{\mathbf{p}}_{n,k}^+ \\ \hat{\mathbf{v}}_{n,k}^- + \Delta\hat{\mathbf{v}}_{n,k}^+ \\ \mathbf{q}(\hat{\boldsymbol{\varphi}}_k^+) \otimes \hat{\mathbf{q}}_k^- \\ \hat{\mathbf{f}}_{b,k}^{bias-} + \Delta\hat{\mathbf{f}}_{b,k}^{bias+} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias-} + \Delta\hat{\boldsymbol{\omega}}_{b,k}^{bias+} \\ c\Delta\hat{t}_k^- + \Delta c\Delta\hat{t}_k^+ \\ c\Delta\hat{\dot{t}}_k^- + \Delta c\Delta\hat{\dot{t}}_k^+ \end{bmatrix}, \ \mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{P}_k^{yy}\mathbf{K}_k^T \tag{3.26}$$

3.3.2 Field experiment: comparison between quaternion-based INS/GPS using EKF and UKF

We use the same field experiment as introduced in the last section to compare the quaternion-based UKF and EKF algorithms. It is worth mentioning that the difference may not only come from the different handlings of nonlinearity, but also the different quaternion time update.

Both approaches use the quaternion vector in the algorithm. Therefore, we can directly compare the estimated quaternion elements, which are shown in **Figure 3-7**. In the first subplot, the quaternion elements q1 to q4 are depicted, while in the 2nd subplot, the q1 to q4 residuals between UKF and EKF are plotted.

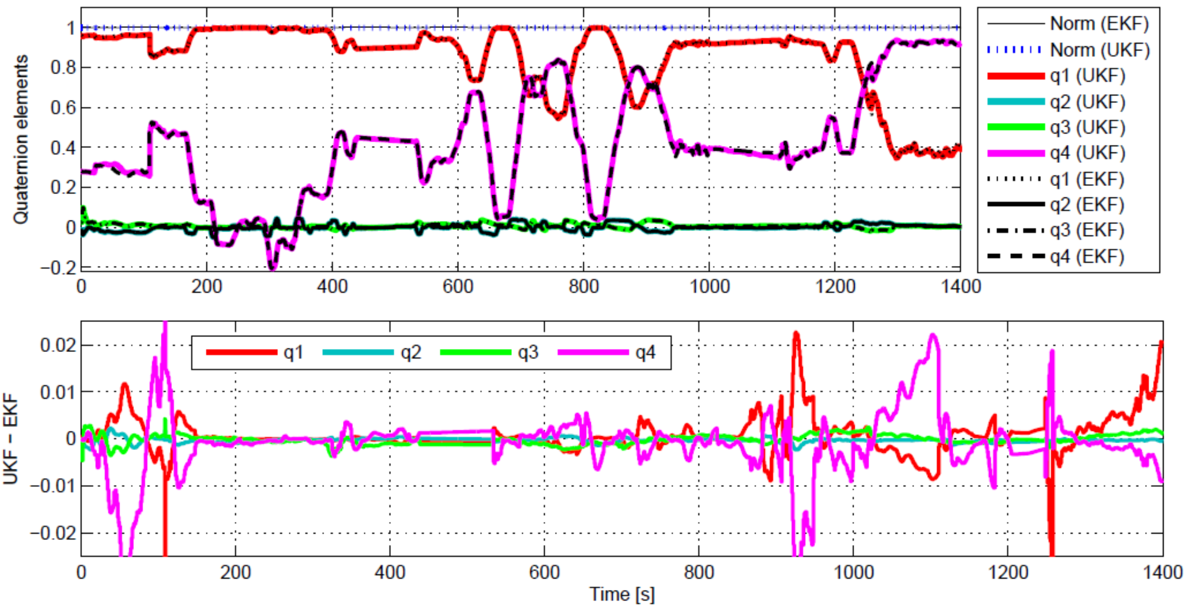


Figure 3-7. Quaternion elements estimation results comparison (EKF vs. UKF)

As shown in the figure, the quaternion estimates from the EKF and UKF are very similar. The main differences come from the estimation of quaternion elements q1 and q4, which contain the heading information.

In this field experiment, the trajectory contains a long GPS outage period, as shown in **Figure 3-2**. In this period, the quaternion elements are propagated over time without GPS measurement updates. Due to the fact that the trajectory path is straight forward, therefore the heading remains unchanged over this period. **Figure 3-7** shows that during this GPS outage, almost no differences can be observed from both filters, proposing that it is the measurement update which is responsible for the deviations of attitude estimates from both approaches. In order to provide a more tangible interpretation of the results, **Figure 3-8** shows the transformed Euler angle estimates corresponding to the quaternion plots given in **Figure 3-7**.

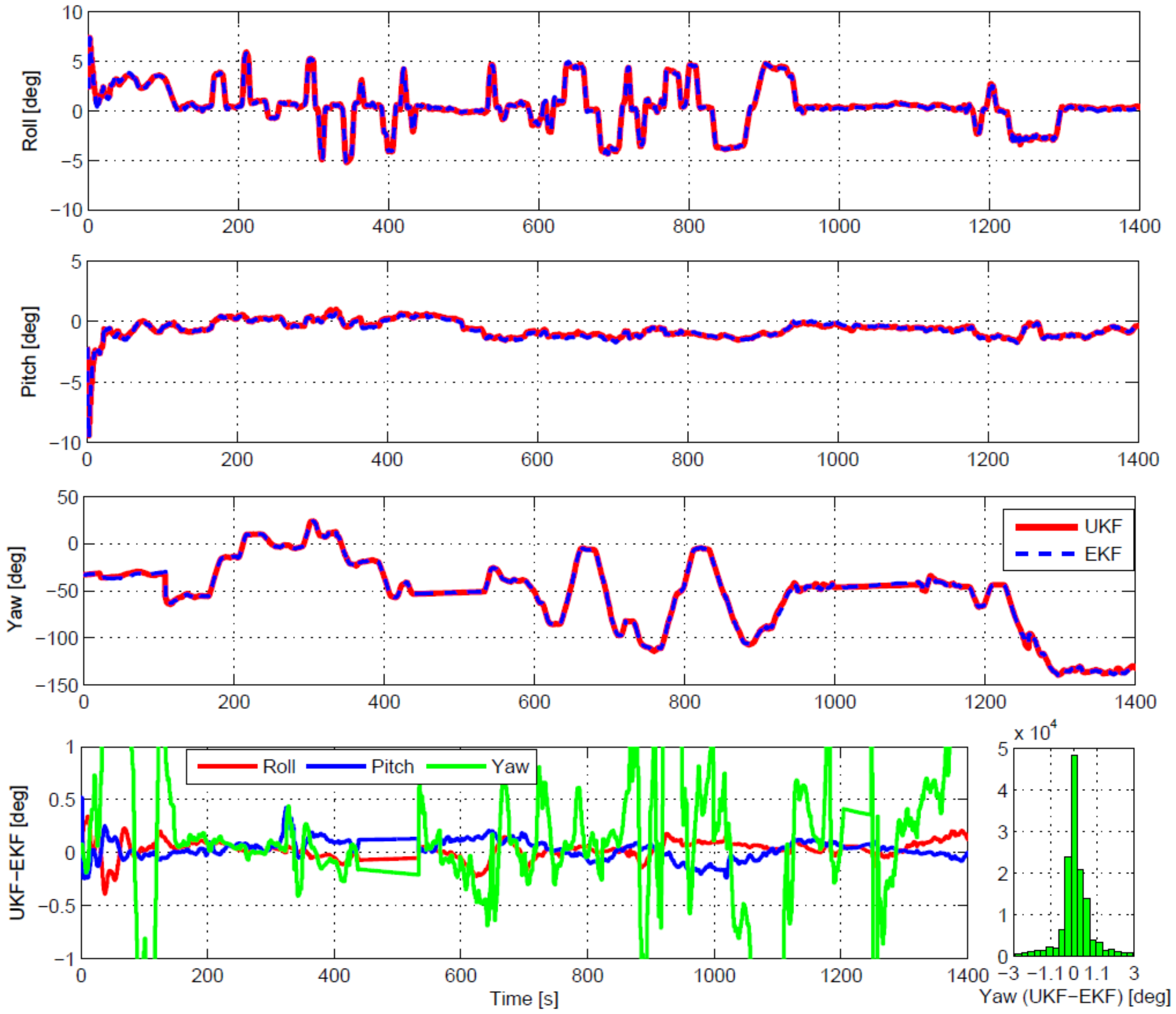


Figure 3-8. Attitude estimation results comparison (transformed from quaternions to corresponding Euler angles)

In **Figure 3-8**, both approaches exhibit similar results in terms of roll and pitch angles. However, on the yaw angle, the observed difference is larger. As depicted on the right-hand side of the 4th subplot, the 1σ value of the yaw estimation difference (UKF – EKF) is 1.1 degree.

We also plot the estimated gyro biases in **Figure 3-9**. Their differences are plotted in **Figure 3-10** (i.e., UKF - EKF). Note that the sensors have not been calibrated beforehand and that the initial gyro bias estimates are set to zero for the lack of better knowledge.

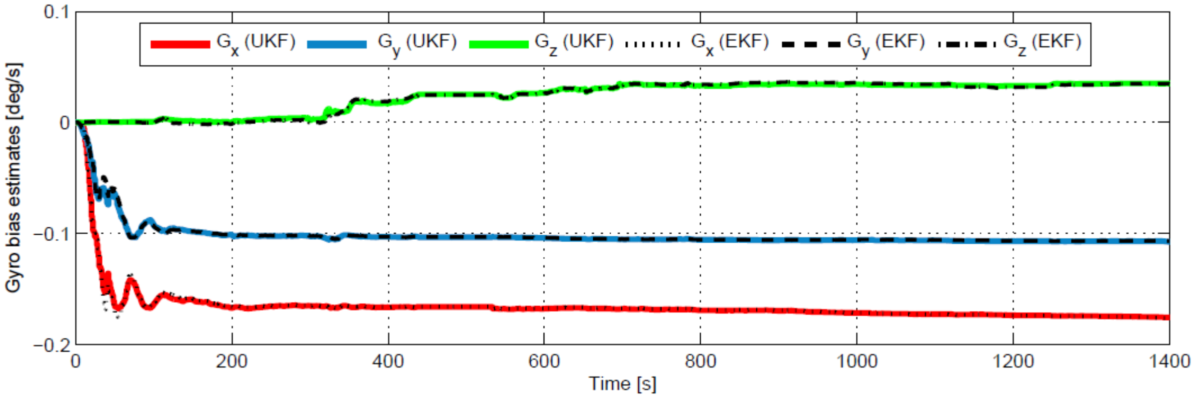


Figure 3-9. Gyro bias estimation results comparison (EKF vs. UKF)

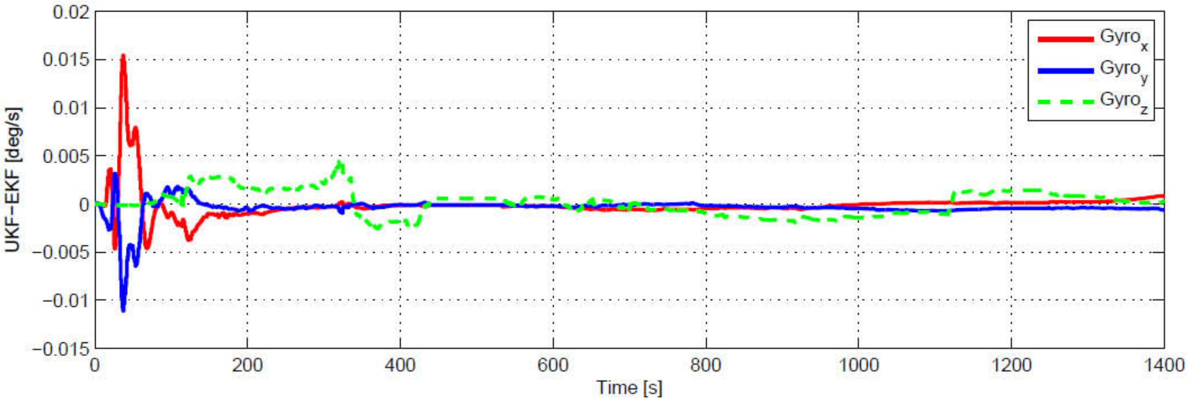


Figure 3-10. Differences in gyro bias estimation results (UKF - EKF)

As shown in **Figure 3-9** and **Figure 3-10**, both approaches yield quite similar gyro bias estimates differing mainly during transient phase of the filters. However, when looking at the unit of y axis of **Figure 3-10**, these differences are very small.

For further testing the robustness of the EKF and UKF based algorithms, an initial attitude error of 30 degrees in each axis is introduced. This error is not unrealistic for an actual application on heading, but is very large on roll and pitch for land-based navigation.

The attitude results are show in **Figure 3-11**, where the convergence process is seen at first few seconds of the roll and pitch estimates, but not at the yaw angle. This is because that, the yaw angle is the least observable state for particular vehicle motion (e.g., low dynamic).

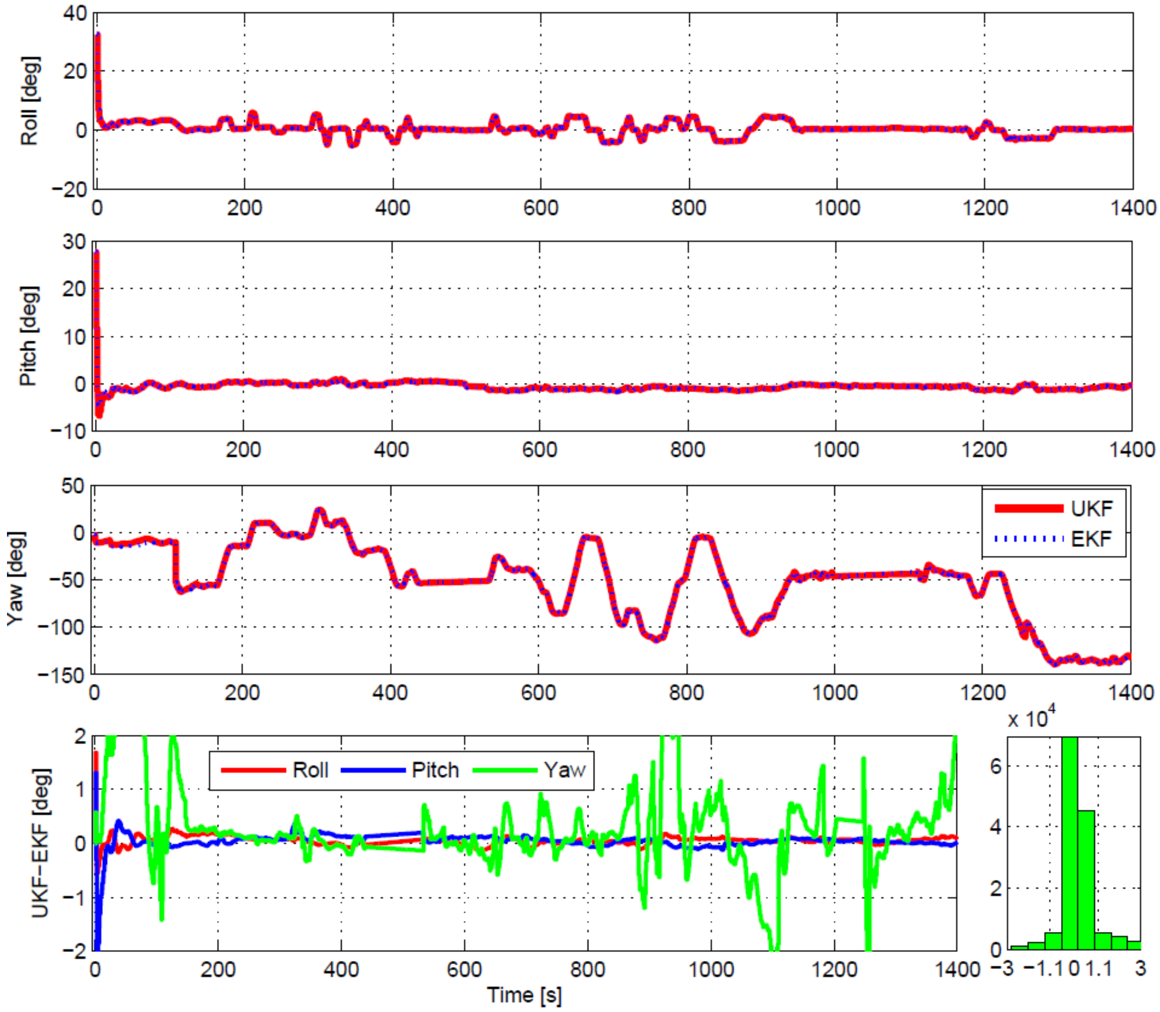


Figure 3-11. Attitude estimation results comparison with an initial attitude error of 30 degrees in each axis (transformed to Euler angles).

With the incorrect initial attitude information, the estimated gyro biases are plotted in **Figure 3-12** and their differences (UKF - EKF) are plotted in **Figure 3-13**.

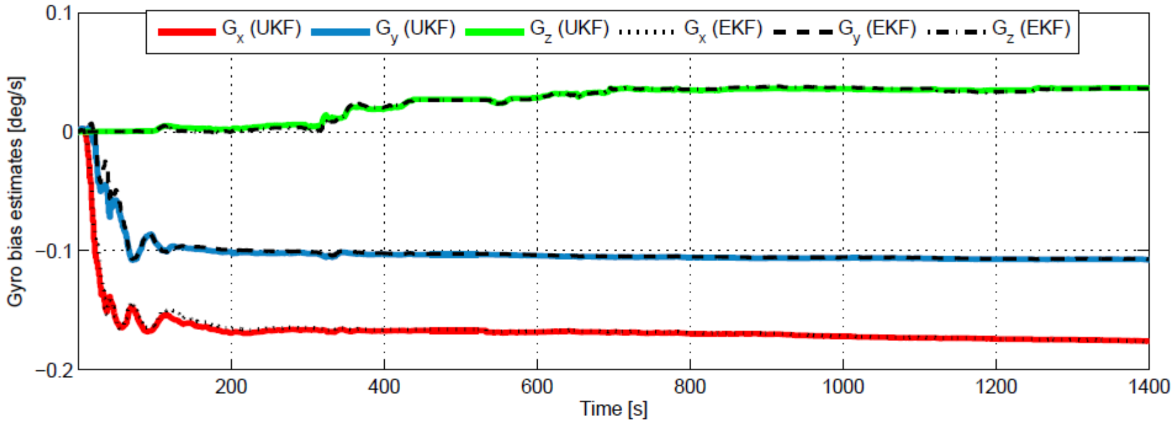


Figure 3-12. Gyro bias estimation results comparison with an initial attitude error of 30 degrees in each axis

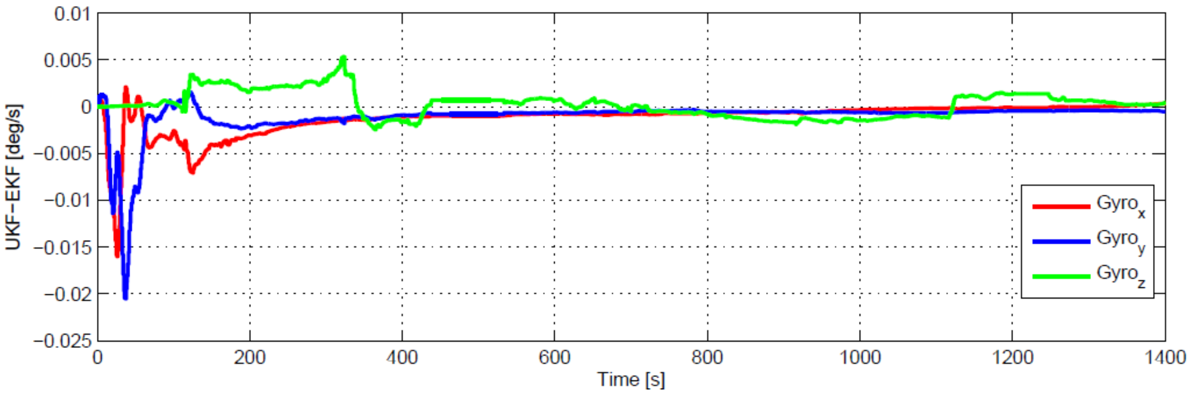


Figure 3-13. Gyro bias estimation differences between EKF and UKF with an initial attitude error of 30 degrees in each axis (UKF - EKF).

Compared with the corresponding results shown in **Figure 3-9** and **Figure 3-10**, we see slightly larger differences in the convergence processes at the first few seconds. However, when looking at the unit (y axis of the figure), these differences are still very small.

On the processing load, the EKF requires a significantly lower computational burden (about 40% of the UKF approach). This processing time is evaluated using MATLAB programming, which cannot be considered as a direct measure of the exact computational burden. For the knowledge on the exact processing load, usually, the number of multiplications and additions should be counted separately, as the way done in [57, 76, 77].

It is also worth mentioning that for any KF-based integration approach, correct tuning of the filter covariance parameters is crucial. A variation of these parameters may significantly change the system estimation performance. In order to prevent this problem, the same set of covariance parameters is applied to both methods.

3.4 Quaternion-based INS/GPS using Unscented Particle filter

3.4.1 Algorithm

For the UPF, the UKF *a posteriori* estimates (i.e., $\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$ from Equation (3.26)) are used to form the importance density distribution for generating particles.

$$\boldsymbol{\chi}_{k,i}^{+} = \hat{\mathbf{x}}_k^{+} + \Delta\hat{\mathbf{x}}_{k,i}^{+} = \begin{bmatrix} \hat{\mathbf{p}}_{n,k}^{+} + \Delta\hat{\mathbf{p}}_{n,k,i}^{+} \\ \hat{\mathbf{v}}_{n,k}^{+} + \Delta\hat{\mathbf{v}}_{n,k,i}^{+} \\ \delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k,i}^{+}) \otimes \hat{\mathbf{q}}_k^{+} \\ \hat{\mathbf{f}}_{b,k}^{bias+} + \Delta\hat{\mathbf{f}}_{b,k,i}^{bias+} \\ \hat{\boldsymbol{\omega}}_{b,k}^{bias+} + \Delta\hat{\boldsymbol{\omega}}_{b,k,i}^{bias+} \\ c\Delta\hat{t}_k^{+} + \Delta c\Delta\hat{t}_{k,i}^{+} \\ c\Delta\hat{\dot{t}}_k^{+} + \Delta c\Delta\hat{\dot{t}}_{k,i}^{+} \end{bmatrix}, \Delta\hat{\mathbf{x}}_{k,i}^{+} \sim \mathcal{N}(0, \mathbf{P}_k^{+}) \tag{3.27}$$

where $i = 1, \ldots, N$ (i.e., N is the number of particles). And $\delta\mathbf{q}(\hat{\boldsymbol{\varphi}}_{k,i}^{+}) = \begin{bmatrix} \cos(0.5\left\|\hat{\boldsymbol{\varphi}}_{k,i}^{+}\right\|) \\ \sin(0.5\left\|\hat{\boldsymbol{\varphi}}_{k,i}^{+}\right\|)\dfrac{\hat{\boldsymbol{\varphi}}_{k,i}^{+}}{\left\|\hat{\boldsymbol{\varphi}}_{k,i}^{+}\right\|} \end{bmatrix}$.

We compute the normalized importance weights as:

$$\mathrm{w}(\boldsymbol{\chi}_{k,i}^{+}) = \frac{p(\tilde{\mathbf{y}}_k \mid \boldsymbol{\chi}_{k,i}^{+})\mathcal{N}(\boldsymbol{\chi}_{k,i}^{+}; \hat{\mathbf{x}}_k^{-}, \mathbf{P}_k^{-})}{\mathcal{N}(\boldsymbol{\chi}_{k,i}^{+}; \hat{\mathbf{x}}_k^{+}, \mathbf{P}_k^{+})},\ \bar{\mathrm{w}}(\boldsymbol{\chi}_{k,i}^{+}) = \frac{\mathrm{w}(\boldsymbol{\chi}_{k,i}^{+})}{\sum_{i=1}^{N}\mathrm{w}(\boldsymbol{\chi}_{k,i}^{+})}, \text{where} \sum_{i=1}^{N}\bar{\mathrm{w}}(\boldsymbol{\chi}_{k,i}^{+}) = 1 \tag{3.28}$$

with

$$\begin{aligned} p(\tilde{\mathbf{y}}_k \mid \boldsymbol{\chi}_{k,i}^{+}) &= \frac{1}{\sqrt{(2\pi)^m \left\|\mathbf{R}_k\right\|}} \exp\left\{ -\frac{\left[\tilde{\mathbf{y}}_k - h_k(\boldsymbol{\chi}_{k,i}^{+})\right]^T \mathbf{R}^{-1} \left[\tilde{\mathbf{y}}_k - h_k(\boldsymbol{\chi}_{k,i}^{+})\right]}{2} \right\} \\ \mathcal{N}(\boldsymbol{\chi}_{k,i}^{+}; \hat{\mathbf{x}}_k^{-}, \mathbf{P}_k^{-}) &= \frac{1}{\sqrt{(2\pi)^n \left\|\mathbf{P}_k^{-}\right\|}} \exp\left\{ -\frac{\left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{-}\right]^T \left(\mathbf{P}_k^{-}\right)^{-1} \left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{-}\right]}{2} \right\} \\ \mathcal{N}(\boldsymbol{\chi}_{k,i}^{+}; \hat{\mathbf{x}}_k^{+}, \mathbf{P}_k^{+}) &= \frac{1}{\sqrt{(2\pi)^n \left\|\mathbf{P}_k^{+}\right\|}} \exp\left\{ -\frac{\left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right]^T \left(\mathbf{P}_k^{+}\right)^{-1} \left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right]}{2} \right\} \end{aligned} \tag{3.29}$$

where "m" denotes the dimension of observation vector.

The *a posteriori* mean and covariance estimates are computed as:

$$\begin{aligned} \hat{\mathbf{x}}_k^{+} &= \sum_{i=1}^{p} \bar{\mathrm{w}}(\boldsymbol{\chi}_{k,i}^{+})\boldsymbol{\chi}_{k,i}^{+} \\ \mathbf{P}_k^{+} &= \sum_{i=1}^{N} \bar{\mathrm{w}}(\boldsymbol{\chi}_{k,i}^{+})\left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right]\left[\boldsymbol{\chi}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right]^T \end{aligned} \tag{3.30}$$

where $\boldsymbol{\chi}_{k,i}^{+}-\hat{\mathbf{x}}_{k}^{+}=\begin{bmatrix}\hat{\mathbf{p}}_{n,k,i}^{+}-\hat{\mathbf{p}}_{n,k}^{+}\\ \hat{\mathbf{v}}_{n,k,i}^{+}-\hat{\mathbf{v}}_{n,k}^{+}\\ \hat{\boldsymbol{\varphi}}_{k,i}^{+}\\ \hat{\mathbf{f}}_{b,k,i}^{bias+}-\hat{\mathbf{f}}_{b,k}^{bias+}\\ \hat{\boldsymbol{\omega}}_{b,k,i}^{bias+}-\hat{\boldsymbol{\omega}}_{b,k}^{bias+}\\ c\Delta\hat{t}_{k,i}^{+}-c\Delta\hat{t}_{k}^{+}\\ c\Delta\hat{\dot{t}}_{k,i}^{+}-c\Delta\hat{\dot{t}}_{k}^{+}\end{bmatrix}$, and $\hat{\boldsymbol{\varphi}}_{k,i}^{+}$ is the rotation vector corresponding to $\hat{\mathbf{q}}_{k,i}^{+}\otimes\left(\hat{\mathbf{q}}_{k}^{+}\right)^{-1}$.

For *a posteriori* quaternion vector, renormalization must be conducted, which can be handled by replacing $\hat{\mathbf{q}}_{k}^{+}$ with $\frac{\hat{\mathbf{q}}_{k}^{+}}{\left\|\mathbf{q}_{k}^{+}\right\|}$.

3.4.2 Field experiments: analysis of system performance from quaternion-based INS/GPS using UPF

Three field experiments are conducted based on train and car rides to study the performance of a quaternion-based INS/GPS using UPF. For the train, it accelerates and decelerates very smoothly without sharp turns and has fewer vibrations. However, the trajectory involves frequent GPS outage passages (i.e., going through long tunnels). For the car, the vibrations of car's engine, road conditions, buildings and trees around the narrow streets may heavily influence the system performances.

- In the first experiment, we verify the robustness of the UPF algorithm under frequent GPS outage passages.
- In the second experiment, different levels of IMU sensors are used.
- In the third test, a very long GPS outage environment is employed, and the UPF positioning estimation performances under non-stationary, highly reflective signal environments are analyzed.

3.4.2.1 Experiment based on a train ride with frequent GPS outage passages

In this test, we use the trajectory, which was introduced in Chapter 1. The trajectory and the number of tracked satellites are given in **Figure 1-12** and **Figure 1-13**. The Landmark$^{\text{TM}}$20 eXT MEMS-based IMU is used. Its main errors are given in **Table 1-5**.

In **Figure 3-14**, the Euler angles from gyro accumulation and INS/GPS integration (UKF) are plotted.

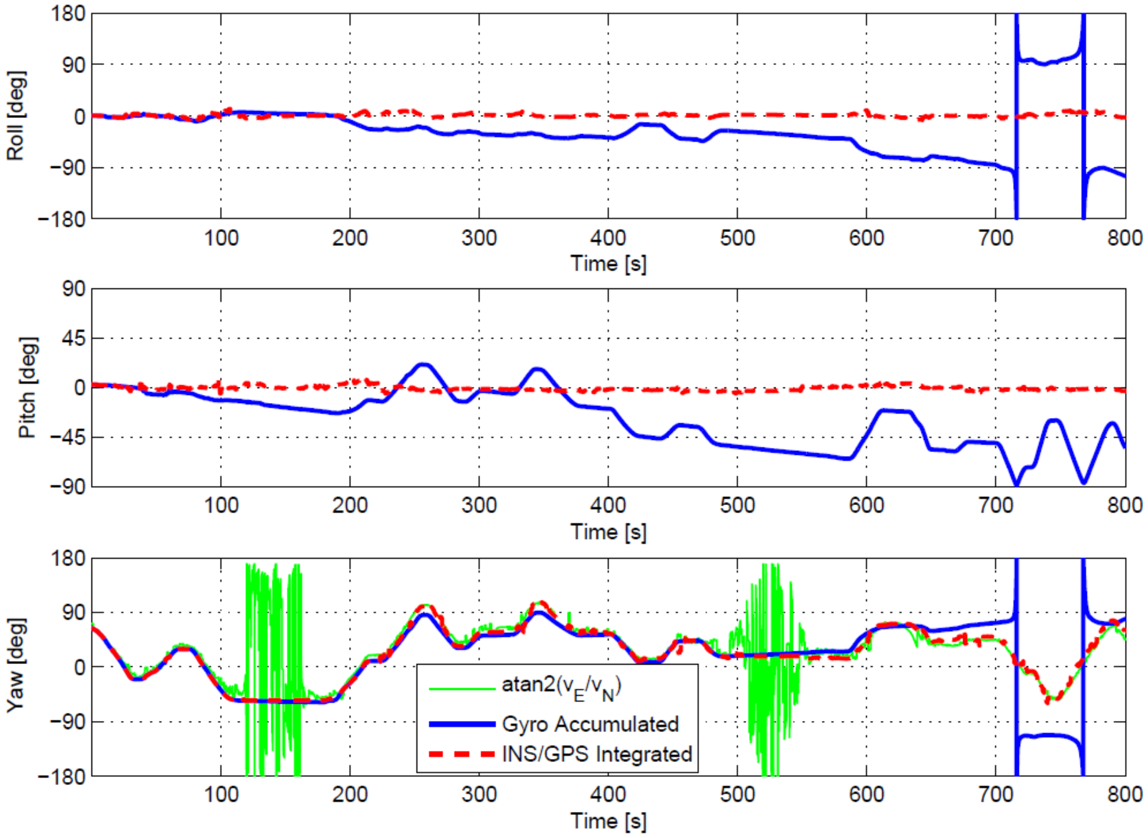


Figure 3-14. INS/GPS estimated and gyro accumulated Euler angles

For the train which can only move forwards and backwards on the track, the yaw angle can be derived from the north and east velocity estimates, i.e., $\gamma = \mathrm{atan2}(v_E / v_N)$. However, due to the tangent operation (i.e., $\tan(\gamma) = \frac{\sin(\gamma)}{\cos(\gamma)} = \frac{v_E}{v_N}$), when both velocities are approaching zero, yaw estimates will be wrong, as shown in the third subplot of **Figure 3-14**. For the standalone INS, the attitude errors are drifting unboundedly over time. When the pitch angle approaches ±90 degrees, a singularity problem appears. This means that the roll and yaw angles will be ambiguous at ±90 degrees pitch angle, if the direction cosine matrix is formed by the rotation sequence of Z-Y-X from the navigation to the body frame. The quaternion estimates in gyro accumulated case are presented in **Figure 3-15**, where no jump errors caused by the singularity problem can be observed.

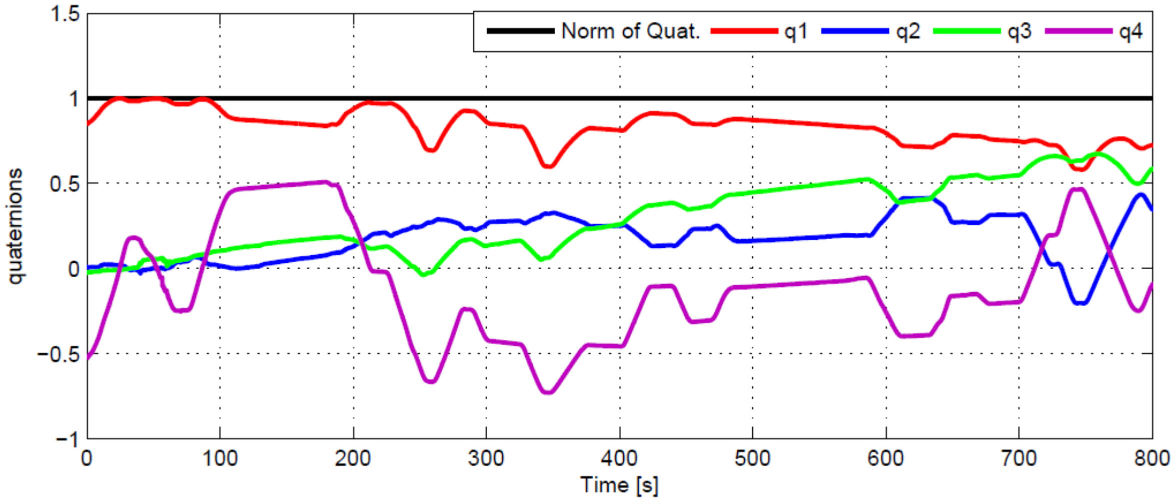


Figure 3-15. Gyro accumulated quaternion estimates

In **Figure 3-16**, the attitudes estimated by the UPF algorithms (using quaternions) are plotted. The quaternion estimates are converted to their corresponding Euler angles for showing results. For the UPF, the particles are generated from the importance density function. That is, each run of the filter returns slightly different result. Thus, the mean of attitude estimation results from 10 runs are plotted from the UPF with 50, 100, 200, 500 particles. We use GPS heading information to update the yaw estimates when the norm of the horizontal velocity is larger than 5 m/s. The readings from the magnetometers mounted inside of the Landmark^{TM}20 eXT IMU are used to initialize the attitudes. Nevertheless, they depend largely on the environment, and there is no proof to say that its attitude accuracy can be better than that of the integration solution. Therefore, we cannot use it as reference.

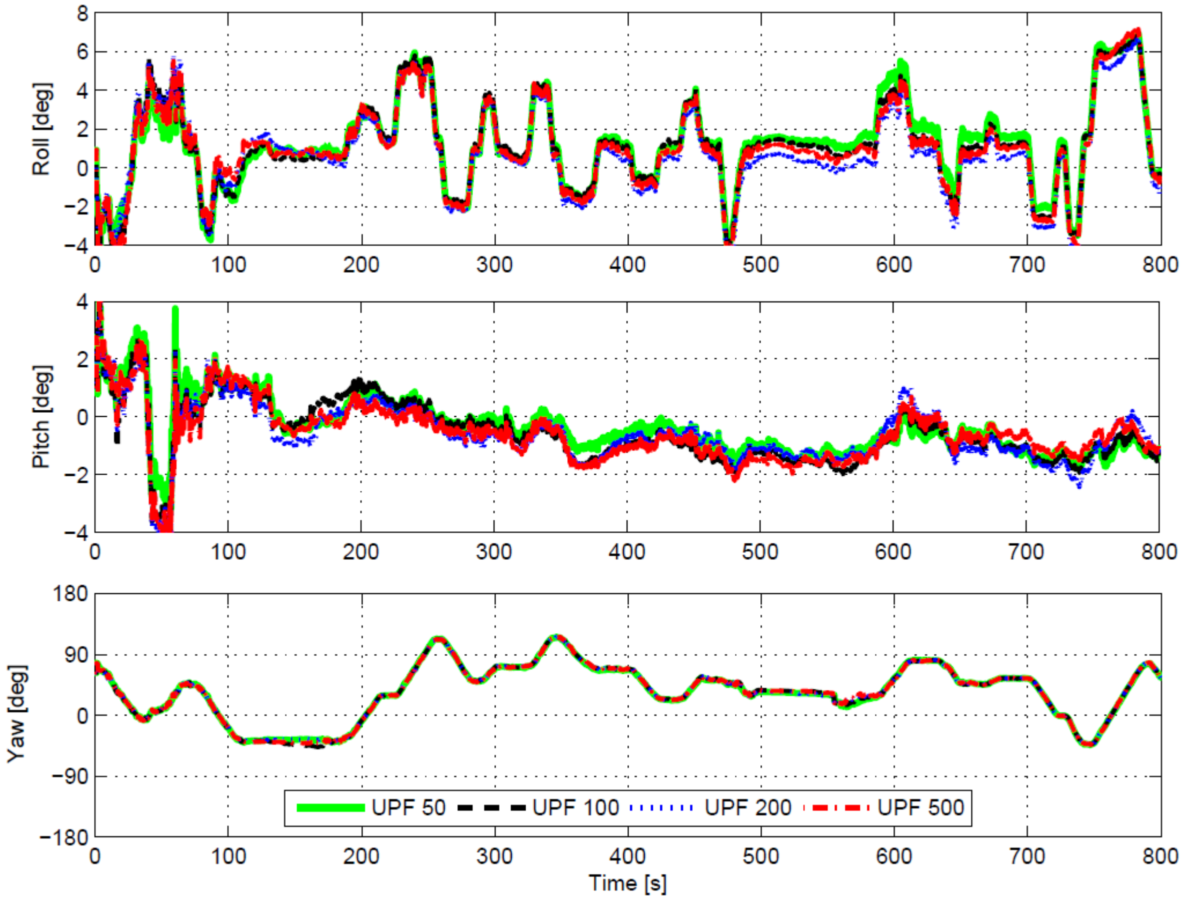


Figure 3-16. Mean of UPF attitude estimates (converted from quaternions to Euler angles) using 50, 100, 200, 500 particles from 10 runs

As shown in the figure, all filters present similar results, which is expected based on the analysis made in the simulation test in Section 2.8. That is, the UPF can present highly accurate estimation results using a small number of particles. Besides, increasing the number of particles may not significantly improve the estimation accuracy. Without reference, we cannot compute the RMS error. And simply from **Figure 3-16**, we cannot tell which one is better. However, we can partly investigate the system performance by comparing their variances at their means. For showing the repetitious performance of the UPF, we make the following computations. The results are given in **Table 3-1**.

$$V_k = \frac{\sum_{i=1}^{10} (x_{k,i} - \bar{x}_k)^2}{10}$$
$$\bar{V} = \frac{\sum_{k=1}^{N} V_k}{N} \tag{3.31}$$

where $\bar{x}_k$ is the mean of estimates from 10 runs at time instant k; $x_{k,i}$ is the estimate from the i-th run at time instant k; N is the number of time epochs in one single run; V_k

is the variance of estimates from 10 runs with respect to their mean $\bar{x}_k$ at time instant k. And $\bar{V}$ is the averaged V_k over time epochs.

Table 3-1. UPF Repetitious performance using different numbers of particles

Nonlinear Filters	$\bar{V}_{roll}$ [deg^2]	$\bar{V}_{pitch}$ [deg^2]	$\bar{V}_{yaw}$ [deg^2]	Time [s]
UPF (50)	1.1624	1.3158	3.1987	236
UPF (100)	1.0002	1.0331	2.4817	260
UPF (200)	0.9410	0.9162	2.2948	313
UPF (500)	0.9124	0.8604	1.8801	476

As shown in the table, with more particles, the filter presents less diversity in estimation results, and naturally the processing time will increase. This processing time is evaluated in MATLAB programming, which cannot be considered as a direct measure of the exact computational burden. Because in MATLAB programming tricks exist for significantly reducing processing load, which cannot be applied in C or C++ environments on a DSP platform. We use it here for a coarse comparison.

In this test, using the UKF algorithm, it takes 206 s. Thus, the increased processing time from UKF to UPF is not significant. This is true, because the measurement update happens at GPS output rate (1 Hz). Compared with the system propagation rate (at 100 Hz IMU output rate), it is not the dominant part.

3.4.2.2 Experiment based on a car ride with different levels of IMUs

Based on the former analysis, in this experiment, we use an UPF with 100 particles as a tradeoff between the precision and processing load. A car is used with different levels of sensors mounted on its roof, as shown in **Figure 3-17**. Lever-arm effects are considered. The trajectory lasts 1400 seconds. Besides the Landmark^{TM}20 eXT MEMS-based IMU, one Xsens MTi (100 Hz) is used, which has the sensor error parameters shown in **Table 3-2**.

The number of satellites in view is given in **Figure 3-18**. A DSP-based platform is developed to collect the synchronized raw data from the Xsens MTi and u-blox Antaris 4 receiver (details can be found in [78, 79]). The field collected raw data from the Landmark^{TM}20 eXT and Xsens MTi are given in **Figure 3-19**. For the integration algorithms, the quaternion-based UKF and UPF are used and compared.

Table 3-2. Xsens MTi performance specification.

	Gyroscope (Angular rates)	Accelerometer (Specific forces)
Bias stability	18000 [°/h] (1σ)	20000 [μg] (1σ)
Noise density	0.1 [°/s/√Hz] (1σ)	1000 [μg/√Hz] (1σ)
Scale Factor stability	-	0.05 %

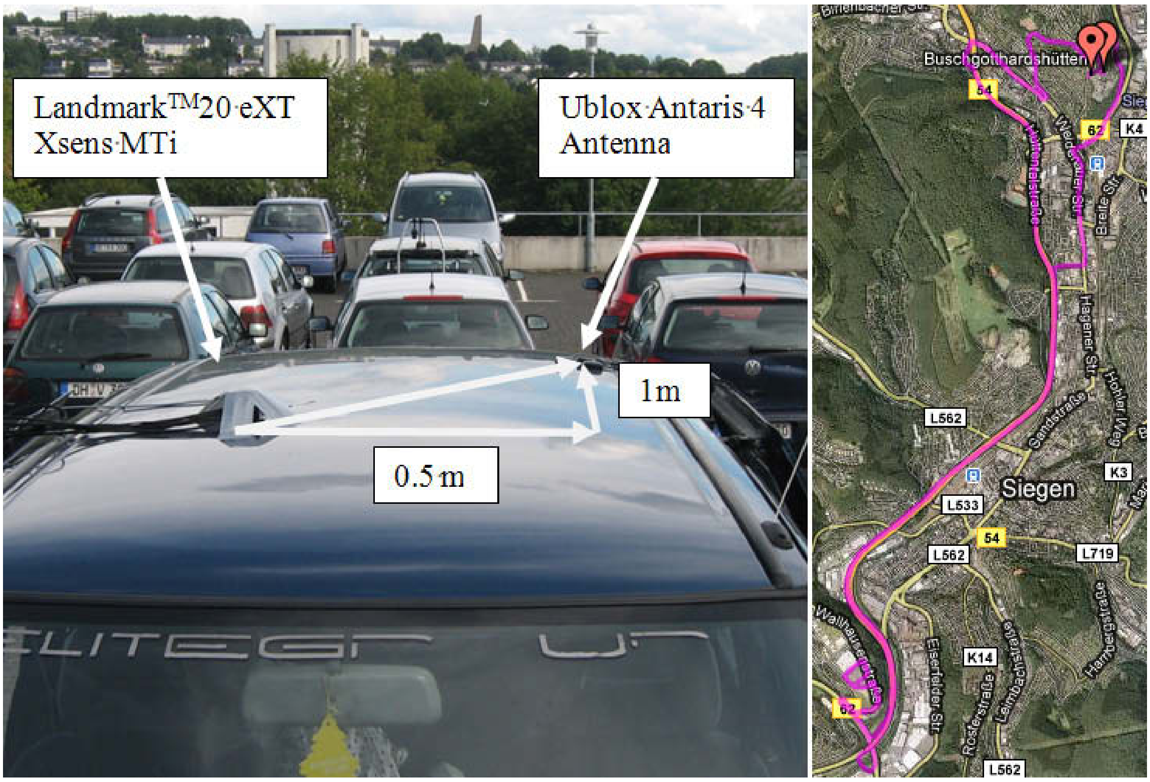


Figure 3-17. Sensor configuration and experimental trajectory (plotted in Google earth)

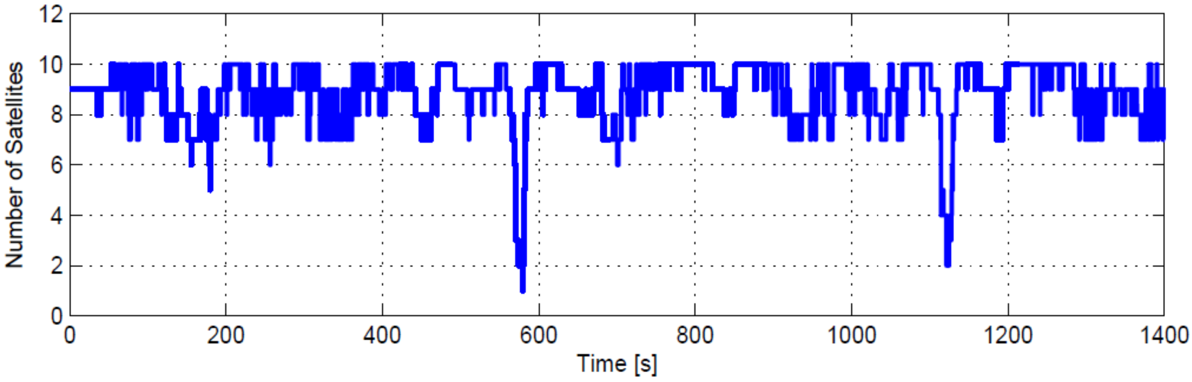


Figure 3-18. Number of satellites in view.

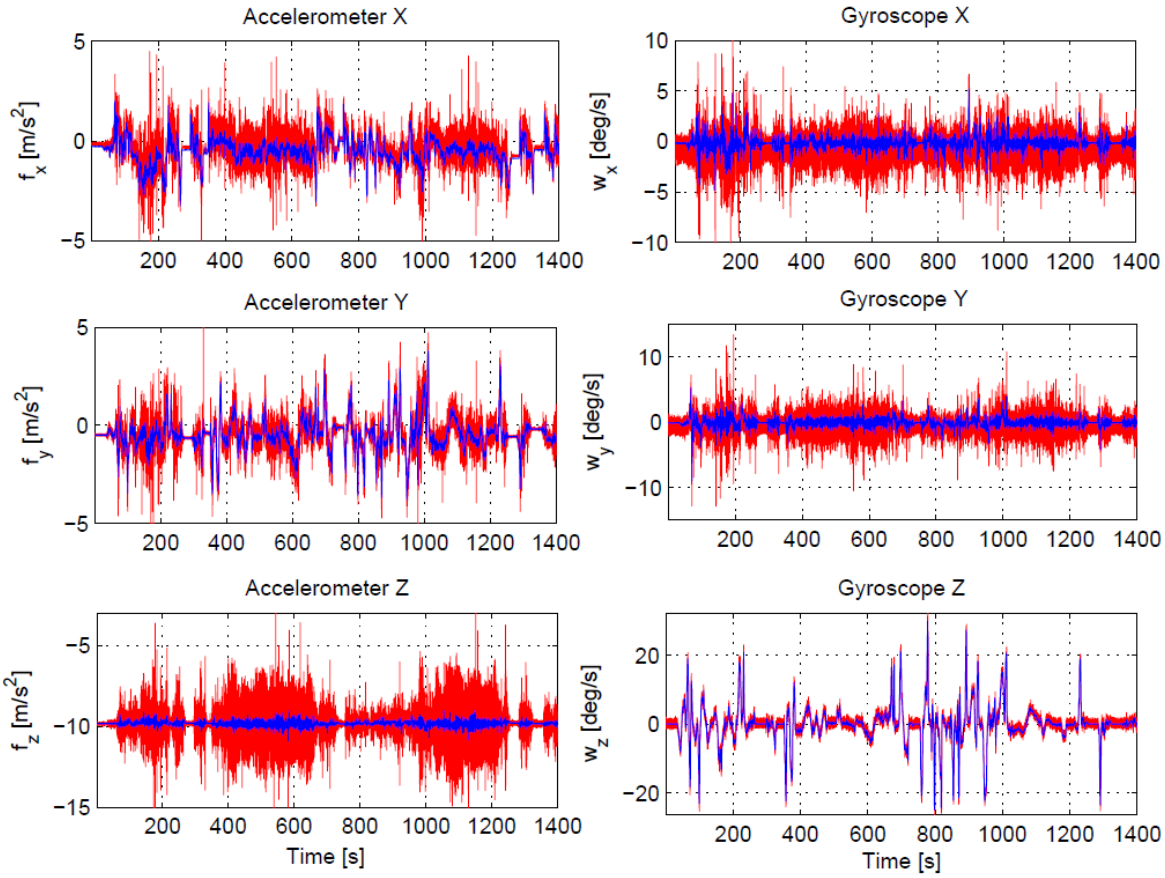


Figure 3-19. Xsens (red) and Landmark (blue) raw data output

The attitude estimation results from the integration of Landmark™20 eXT with u-blox Antaris 4, and the Xsens MTi with u-blox Antaris 4 using UKF algorithms are presented in **Figure 3-20**, where the quaternions are transformed to their corresponding Euler angles for comparison.

In the figure, for the roll angle, there is a three degrees shift from zero. This is true, because the IMU was mounted near the right side of the car where the roof is indeed tilted for small angles, as shown in **Figure 3-17**. The integrations of GPS with two levels of IMUs output similar attitude estimation results.

However, in **Figure 3-19**, the sensor noise and bias errors from Xsens are much larger than those of Landmark. In **Figure 3-20**, it shows that with GPS aiding, the lower level IMU can present comparable attitude estimation results with respect to that of using higher level IMU. This is because, firstly, gyro bias errors can be correctly estimated by the GPS aiding, which is shown in **Figure 3-21**. And secondly, if the low-cost receiver is in use, the GPS measurement errors (i.e., pseudorange and Doppler errors) are large. That is, the integral of IMU sensor errors during GPS update interval (i.e., 1 second for 1 Hz GPS update rate) is not the major error in the sensor integration.

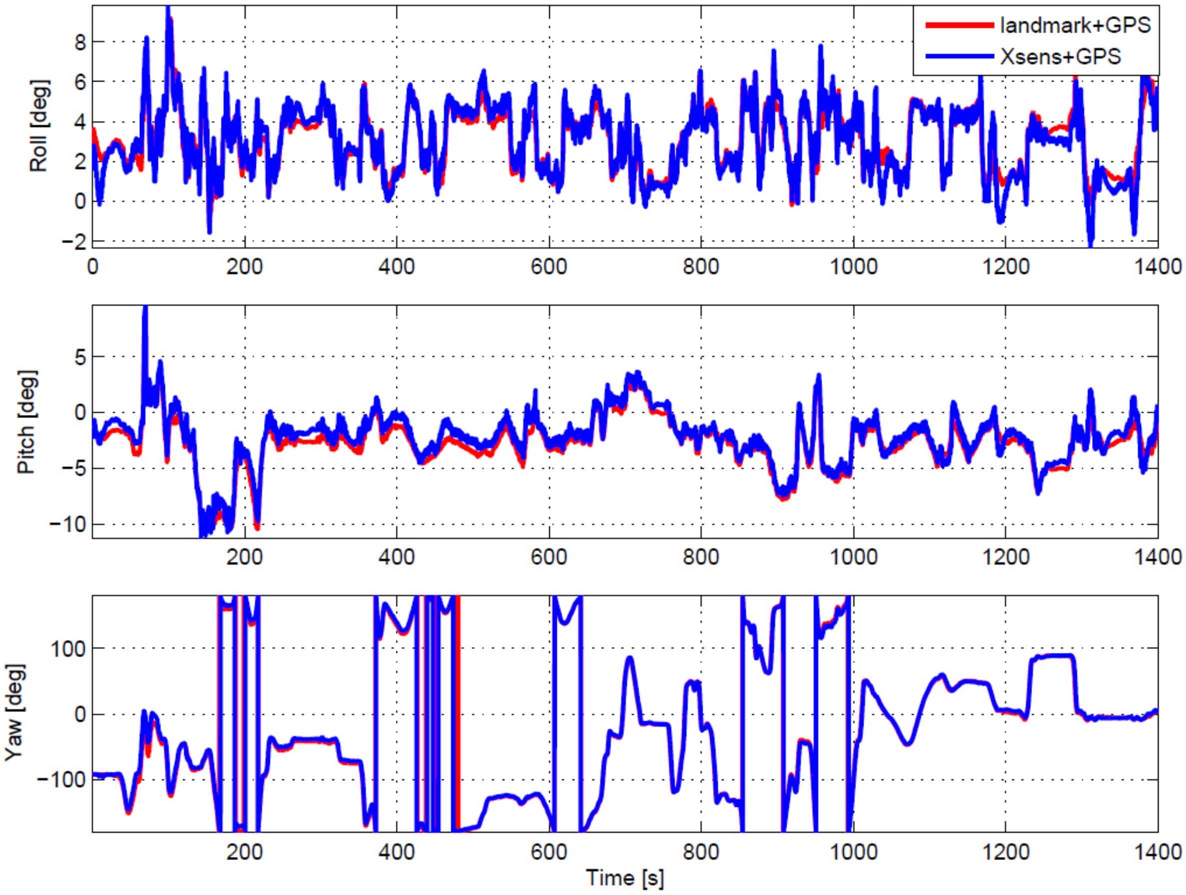


Figure 3-20. Attitude estimation results (UKF) using two levels of IMUs.

Although we have not conducted any calibration processes for detecting the IMU sensor errors before the application starts, however, in the first 30 seconds, the car is stationary. Therefore, we can average the gyroscope raw data in the first 30 seconds to detect the sensor bias errors. They will not be used in the integration, but considered as a coarse reference (at least at the first several seconds for very low-cost inertial sensors), which are plotted as dashed lines in **Figure 3-21**. Due to the sensor noises, the integral of them during run-time results in the drifts of gyro bias errors over time (not constant values), which can be observed in the figure. The gyro bias errors in z axis take more time to be estimated due to the fact that the yaw angles are the least observable states for the particular vehicle motion, for instance, when the vehicle is in low-dynamic situations [14, 47]. Performing maneuvers along the trajectory would improve the gyro bias and heading estimation [14-16, 54].

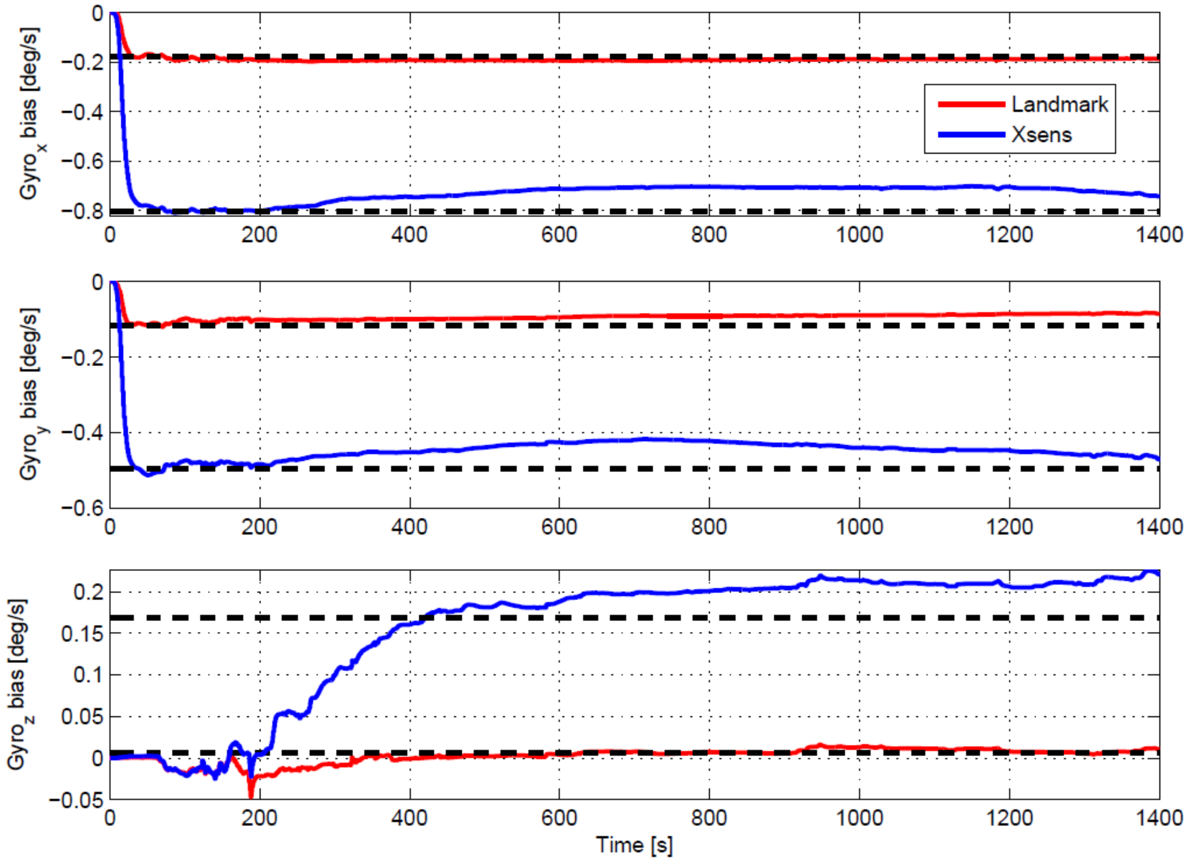


Figure 3-21. Gyro bias estimation results (UKF) using two levels of IMUs.

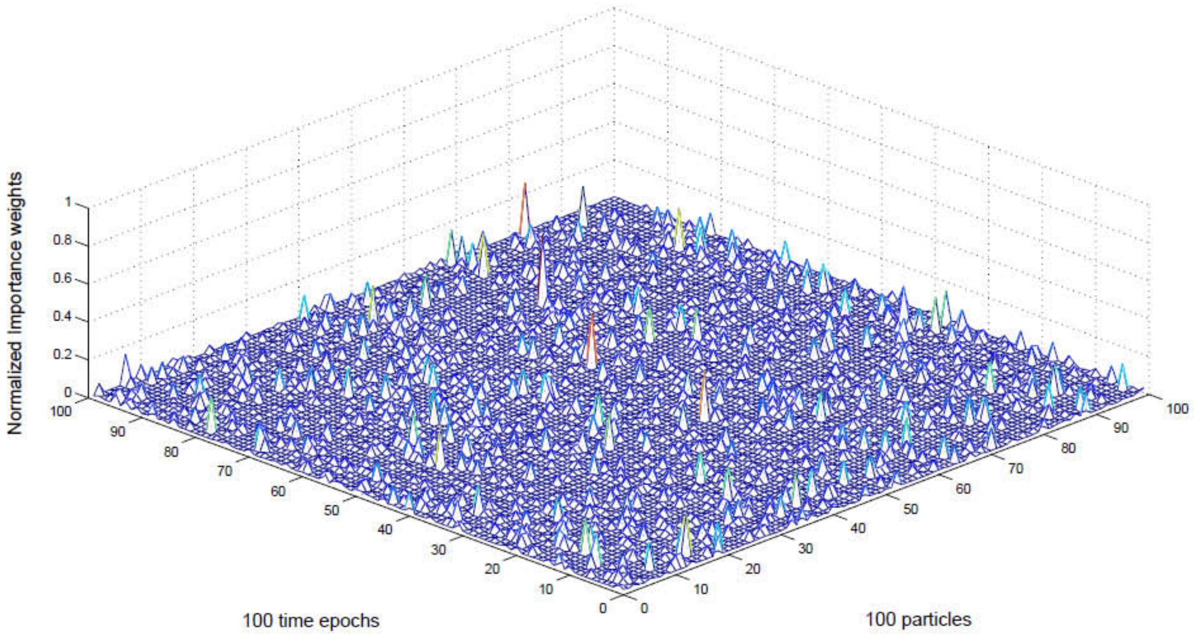


Figure 3-22. Normalized importance weights (100 particles).

We show the results from the UKF, because UKF is embedded in the UPF algorithm. The accuracy of the UKF *a posteriori* estimates is critical. That is, they form the importance density function for drawing particles. We use the dataset collected from Xsens and Ublox Antaris 4 to show the performance of UPF with respect to that of UKF. The inner working of the UPF can be exhibited by plotting the normalized importance weights (e.g., 100 particles for 100 seconds) in **Figure 3-22**.

As shown in the figure, although we only draw 100 particles, many of them are heavily weighted. The Attitude estimation results from both filters are shown in **Figure 3-23**. The gyro bias estimation results are given in **Figure 3-24**.

Figure 3-23. Attitude estimation results comparison

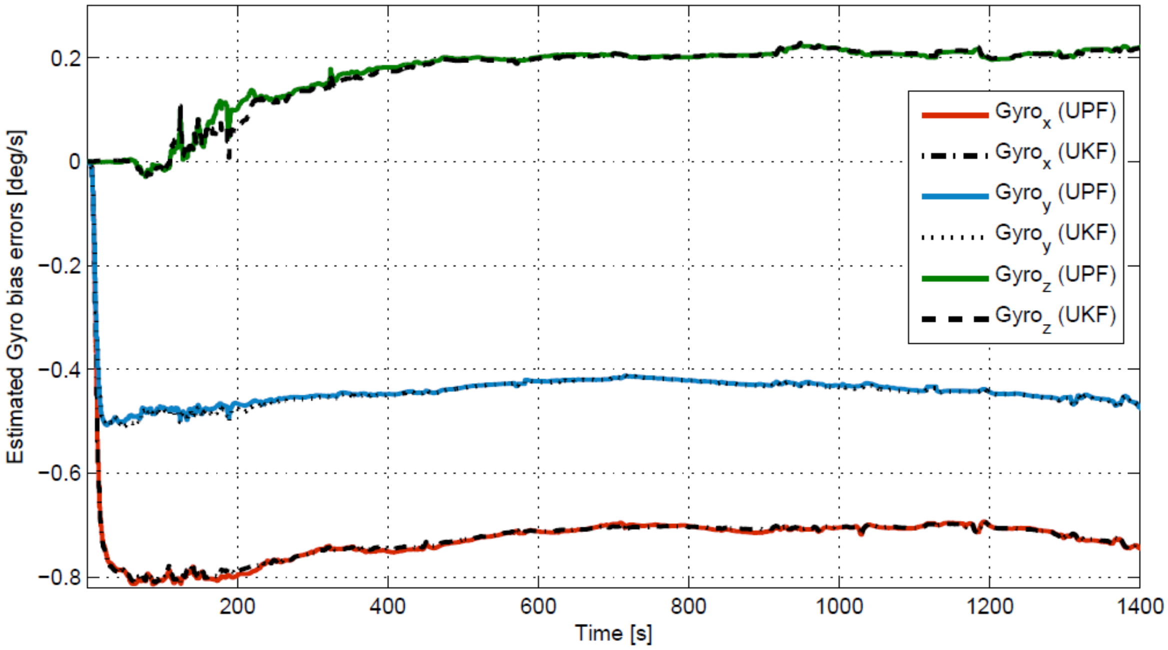


Figure 3-24. Gyro bias estimation results comparison.

In **Figure 3-23**, the attitude estimates from the UPF and UKF are roughly at the same estimation accuracy. It is worth mentioning that without redundant attitude fixes (e.g., using multi-antenna GPS system), in INS/GPS integration, the attitude information is mainly contained in the velocity estimates based on the Doppler measurements. The greater the accuracy of the velocity estimates, the greater the dampening on the attitude errors [73]. Thus, in order to understand the results depicted in **Figure 3-23** and **Figure 3-24**, we plot the Doppler innovations after transient phase of the filter (starting from the 4th epoch) in the UKF filter in **Figure 3-25**.

The innovation in the KF represents the new information brought by the incoming measurements to the system, and it is formed as the difference between the measured and predicted quantities. Due to fact that the system predicted quantities are usually smooth, thus the statistics of innovations are mainly determined by the incoming measurements (errors). In **Figure 3-25**, it shows that the Doppler innovations are zero mean, Gaussian distributed over time. Although from one single run of the test, we cannot know the ensemble statistics of innovations at each time epoch, but we believe that they are nearly Gaussian distributed, because the multipath effects on Doppler measurements are rather small. In this case, if the additive Gaussian noise assumption holds true, the UKF algorithm presents highly accurate estimation results. And it could be the reason why both filters present equivalent attitude estimation results.

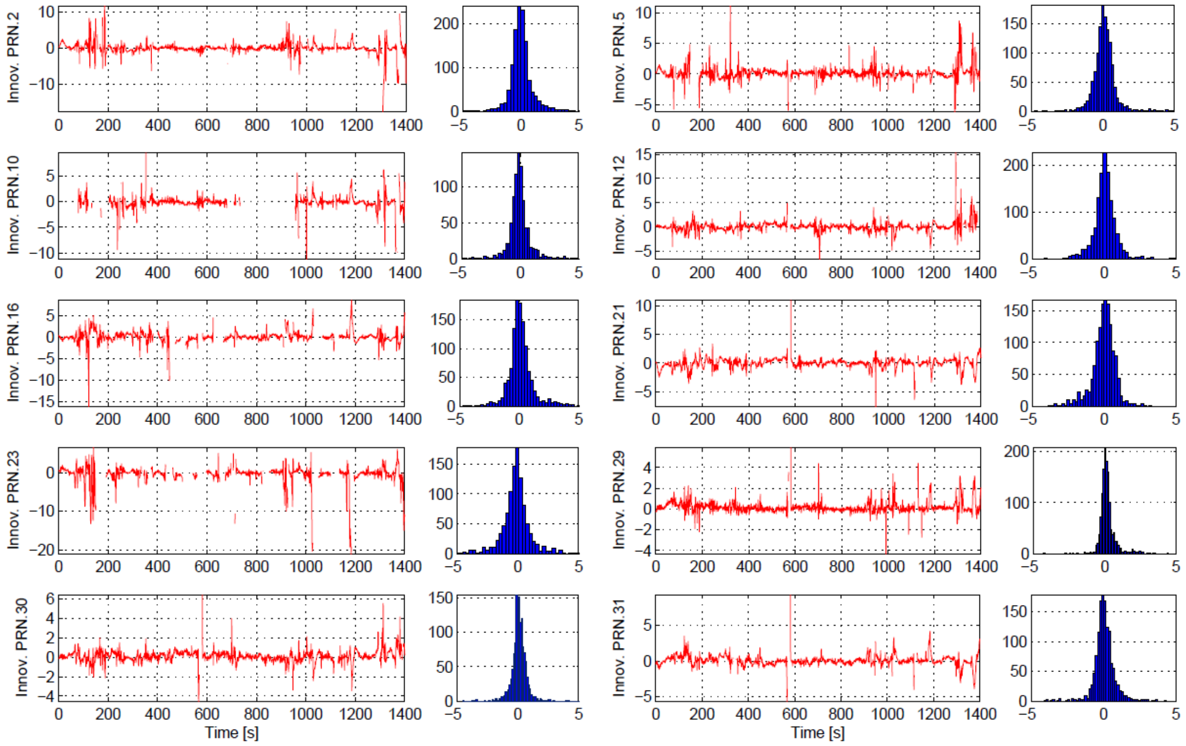


Figure 3-25. Innovation (Doppler) in the filter

We also plot the pseudorange innovations after transient phase of the filter (starting from the 4th epoch) in **Figure 3-26.**

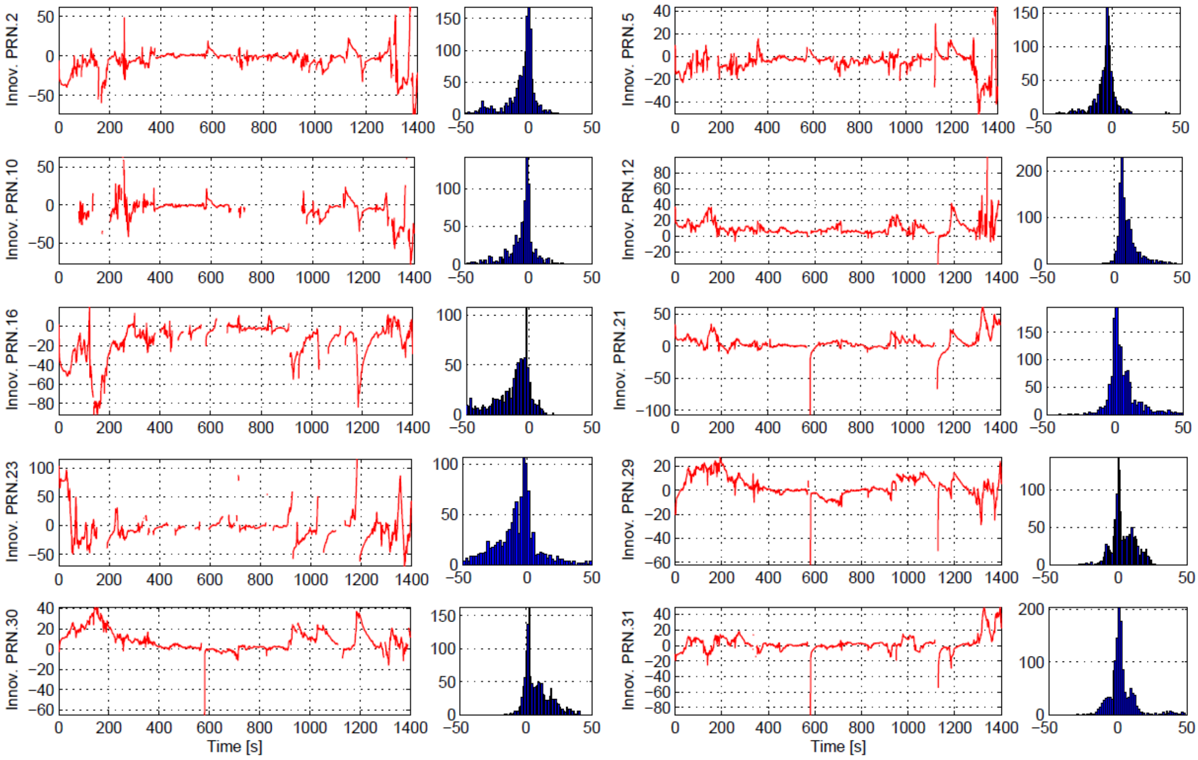


Figure 3-26. Innovation (pseudorange) in the filter.

We see that they are not exactly Gaussian distributed over time due to e.g. multipath effects. And the ensemble statistics of pseudorange innovations will also not be Gaussian distributed, especially when the vehicle is under highly reflective signal environment, (the periods when innovations are large in the figure).

In order to fully study the UPF positioning estimation performance, a trajectory with different signal environments is used, which is introduced in the next section.

3.4.2.3 Experiment based on a train ride with very long GPS outage period

We take the same trajectory as introduced in section 3.2.1. The trajectory lasts about 1400 s with two tunnels, where the first GPS outage lasts 93 s, and the second one takes 43 s. In the first tunnel, the trajectory is a straight path, and it is used to analyze the positioning drift performances from different algorithms. For the UPF algorithm, 100 particles are used and 10 runs are conducted. The means of positioning estimation results are plotted (red curve) in **Figure 3-27**. For the UKF algorithm, its results are marked as the blue curve in the figure.

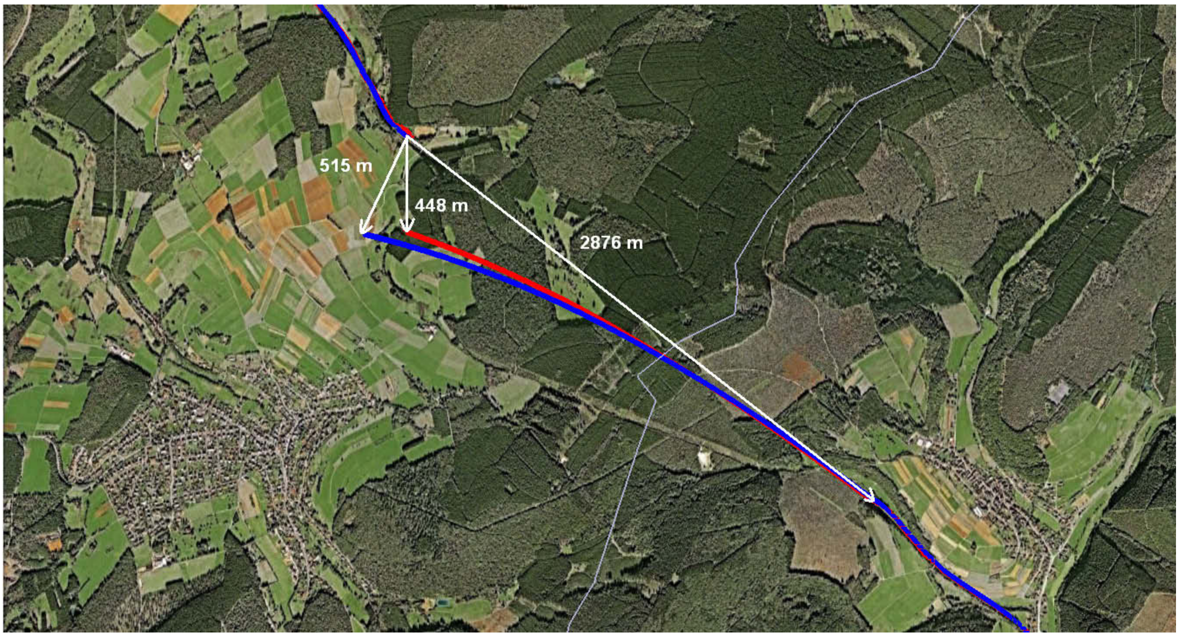


Figure 3-27. UPF (red) and UKF (blue) positioning comparison (plotted in Google earth)

As shown in the figure, the positioning drift of the UPF (mean values from 10 runs) are statistically better than those of the UKF. The tunnel is about 2876 m long. For the UKF, the positioning drift error reaches 515 m. While for the UPF, its mean positioning drift error reaches 448 m. This drift error is caused by the integral of IMU measurement errors over GPS outage period. They also depend on the accuracy of attitude and sensor bias estimates in the filter before the train enters the tunnel. That is, the tilt errors caused by inaccurate attitude estimates may project the gravity onto the horizontal plane. After integration, they yield the horizontal velocity and positioning drifts. During the GPS outage period, the latest updated sensor bias estimates are used to compensate the errors in the incoming inertial measurements. These inertial sensor errors vary slowly over time. Therefore, the positioning drift also depends on the level of inertial sensors in use, and the length of the GPS outage period.

Besides, it is important to note that the signal environment at the beginning of the time period of interest is critical for evaluating the performance of UPF algorithm. We cannot simply draw the conclusion that the UPF will perform better result than that of UKF. In this test, it bases on the signal environment, when the train enters the tunnel. If at that moment, it is an open sky environment, both approaches should principally present identical results. In **Figure 3-28**, we plot the positioning estimation results from the open sky environment, where the UPF (mean values from 10 runs) and UKF present indeed quite similar performances (i.e., differences are mainly at centimeter levels).

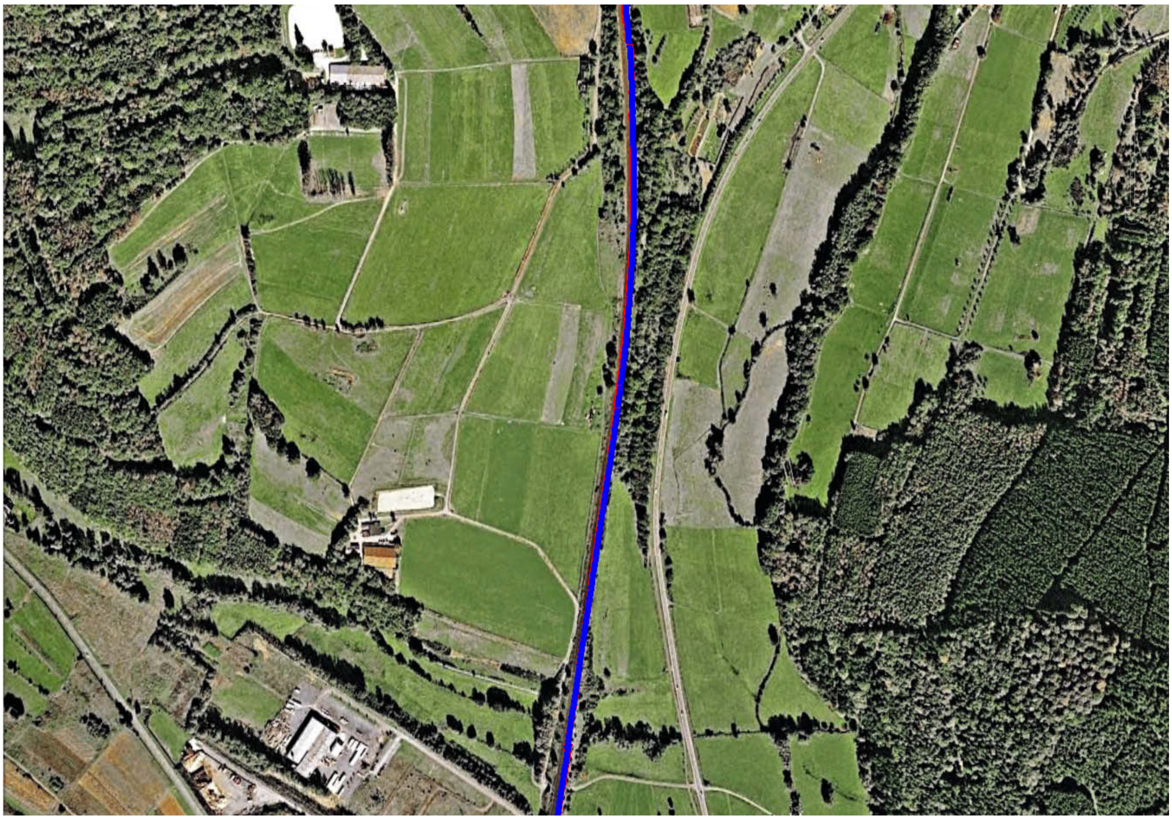


Figure 3-28. Train passes through open sky environments (plotted in Google earth)

In the former conducted experiments, the statistics of measurement errors change over time during the application, which depends on the physical signal environments. In this case, using constant **R** and **Q** parameters (treating noises as stationary) will cause a degradation of estimation accuracy. In order to achieve further improved estimation results, the prompt reflection of changes in the noise statistics caused by external influences is required [80-84]. In [59], we have proposed a residual-based adaptive estimation method for UKF and UPF algorithm to solve this problem.

3.5 Summary

In this chapter, the system performances of a tightly-coupled INS/GPS system using quaternion-based nonlinear filtering methods has been analyzed on the basis of field experiments with train and car rides.

The quaternion-based EKF algorithm was firstly compared with the Euler angle-based EKF algorithm. Numerical results show that other than the singularity issue, the quaternion-based approach is more confident on the estimation of heading related states (e.g., yaw and Gyro. z-axis bias) for land-based navigation. This may be due to the fact that the system models in the quaternion-based approach do not contain the trigonometric terms. For verifying this, the quaternion-based EKF was also compared with the quaternion-based UKF. A more accurate quaternion propagation method has been employed in the UKF algorithm. Even with incorrect initial attitude information, i.e., 30 deg errors on each axis of attitude, both approaches can still present quite similar results. However, when looking at the processing load, the EKF-based approach requires only 40% of the UKF approach.

We also applied the UPF algorithm on the quaternion-based INS/GPS. The UPF has its ability to cope with nonlinearity more accurately with relaxed statistical assumptions (allowing non-Gaussian models), which constitute a promising solution for robust and high-accuracy demanding applications. Three field experiments were made with different scenarios in land-based navigation. One experiment was conducted based on a train ride to verify the robustness of the system performance under frequent GPS outage periods. The other was based on a car ride in the city using two levels of inertial sensors. The third trajectory involved very long GPS outage periods. The work presented in this chapter has shown that the UPF algorithm with only a small number of particles can exhibit remarkable performance, especially when the vehicle is navigating under highly reflective signal environments.

4. INS/GPS Tightly-coupled Integration using Sequential Processing

4.1 Introduction

In tightly-coupled integration, the carrier-phase derived delta range measurement may also be used for velocity determination, especially when high level GPS receivers are employed. However, it is a type of integrated measurements with errors strongly related to pseudorange errors at the start and end of the integration interval. Conventional methods circumvent these errors with approximations, which may lead to large velocity estimation errors for high dynamic applications.

The weakest but often used approach is to simply consider the integral of pseudorange rate divided by time (an average quantity) as the instantaneous velocity measurement at the endpoint of the interval (e.g., [44]). It may fulfill the accuracy requirements in static or low dynamic applications. Nevertheless, if the vehicle is maneuvering under high dynamics with low GPS update rate, large velocity estimation errors will appear. These errors are strongly correlated to the accelerations and jerks involved in the trajectory [11].

In order to solve this problem, two methods are proposed in this chapter. One method is to use the delay states to “remember” the pseudorange errors at the start of the integration interval. And the other is to predict backwards the delay states by the current states considering the integrated system process noises. Sequential processing is utilized in the measurement update of both methods to avoid the time consuming matrix inversion computation in the derivation of Kalman gain. And, the topic on sequential processing of integrated measurements without making compromise on their errors is studied. The following questions will be answered:

1) How many numerical operations can be saved using sequential processing in comparison to batch processing in measurement update phrase of the KF?
2) Concerning the two methods proposed in this chapter, which one is more suitable for practical usage?
3) Can sequential processing be successfully applied in measurement update of different types of measurements (i.e., code derived pseudorange as a non-integrated

measurement; carrier-phase derived delta range as an integrated measurement) without degrading the system estimation accuracy?

4) If yes, are there special conditions, which must be met?

In the remainder of this chapter, the content is organized as follows. In Section 4.2, the often used velocity determination method based on delta range measurements is overviewed, and its drawback is explained. In Section 4.3 and 4.4, two approaches for tackling this problem are proposed. Moreover, sequential processing is utilized in both approaches. In Section 4.5, a simulation is conducted on the basis of a field collected UAV trajectory. In Section 4.6, numerical results are compared and analyzed.

4.2 Velocity determination

The delta range is formed by integrating the pseudorange rate over time. It is equal to the difference of pseudorange measured at endpoints of the integration interval. Accordingly, the errors involved in delta range are strictly related to the corresponding pseudorange errors at the start and end of the integration interval, as given in Eq (4.1).

$$\Delta \boldsymbol{\rho}_k = \int_{k-1}^{k} \delta \dot{\boldsymbol{\rho}}\, dt = \delta \boldsymbol{\rho}_k - \delta \boldsymbol{\rho}_{k-1} \tag{4.1}$$

where $\Delta \boldsymbol{\rho}_k$ represents the delta range error measured at time instance *k*; $\delta \dot{\boldsymbol{\rho}}$ denotes the pseudorange rate error; $\delta \boldsymbol{\rho}_k$ and $\delta \boldsymbol{\rho}_{k-1}$ are the pseudorange errors at two endpoints of integration interval, which can be related to their system error states (i.e., position and receiver clock bias errors) as:

$$\begin{aligned} \delta \boldsymbol{\rho}_k &= \hat{\boldsymbol{\rho}}_k - \boldsymbol{\rho}_k = \mathbf{H}_k \delta \mathbf{x}_k + \mathbf{v}_k \\ \delta \boldsymbol{\rho}_{k-1} &= \hat{\boldsymbol{\rho}}_{k-1} - \boldsymbol{\rho}_{k-1} = \mathbf{H}_{k-1} \delta \mathbf{x}_{k-1} + \mathbf{v}_{k-1} \end{aligned} \tag{4.2}$$

where error states are defined as $\delta \mathbf{x} = \hat{\mathbf{x}}_{estimated} - \mathbf{x}_{true}$; $\mathbf{H}_k$ and $\mathbf{H}_{k-1}$ are the Jacobian matrices from the first order linearization of the nonlinear pseudorange observation model at time instants of the start and end of the integration interval; $\mathbf{v}_k$ and $\mathbf{v}_{k-1}$ represent the noises.

Substituting Equation (4.2) into Equation (4.1) yields:

$$\delta \boldsymbol{\rho}_k - \delta \boldsymbol{\rho}_{k-1} = \mathbf{H}_k \delta \mathbf{x}_k - \mathbf{H}_{k-1} \delta \mathbf{x}_{k-1} + \mathbf{v}_{k,k-1} \tag{4.3}$$

For clarity purpose, we denote $\mathbf{v}_k^{\phi}$ to represent the cumulative effect of the carrier-phase derived delta range measurement noises $\mathbf{v}_{k,k-1}$ from time instants k-1 to k. We divide Equation (4.3) by time interval $\Delta t = t_k - t_{k-1}$, and assume that this time interval is small. Thus, the changes in the Jacobian matrix **H** are negligible (i.e., $\mathbf{H}_{k-1} \approx \mathbf{H}_k$). We will arrive at:

$$\begin{aligned} &\frac{\delta\boldsymbol{\rho}_k - \delta\boldsymbol{\rho}_{k-1}}{\Delta t} = \mathbf{H}_k \frac{(\delta\mathbf{x}_k - \delta\mathbf{x}_{k-1})}{\Delta t} + \mathbf{v}_k^{\phi} \\ &\Rightarrow \delta\dot{\boldsymbol{\rho}}_k = \mathbf{H}_k \delta\dot{\mathbf{x}}_k + \mathbf{v}_k^{\phi} \end{aligned} \tag{4.4}$$

where $\delta\dot{\mathbf{x}}_k$ represent the system velocity and receiver clock drift error states.

This equation is often seen for velocity determination using delta range measurements. However, it relates average information (i.e.,$(\delta\boldsymbol{\rho}_k - \delta\boldsymbol{\rho}_{k-1}) / \Delta t$) to the instantaneous velocity estimation errors at time instant k (i.e.,$\delta\dot{\mathbf{x}}_k$). If the vehicle is maneuvering under high dynamics, this approximation will be very poor.

Therefore, we lay Equation (4.4) aside. And from Equation (4.2), we have:

$$\begin{aligned} \delta\boldsymbol{\rho}_k - \delta\boldsymbol{\rho}_{k-1} &= (\hat{\boldsymbol{\rho}}_k - \boldsymbol{\rho}_k) - (\hat{\boldsymbol{\rho}}_{k-1} - \boldsymbol{\rho}_{k-1}) \\ &= (\hat{\boldsymbol{\rho}}_k - \hat{\boldsymbol{\rho}}_{k-1}) - (\boldsymbol{\rho}_k - \boldsymbol{\rho}_{k-1}). \end{aligned} \tag{4.5}$$

We denote $\hat{\mathbf{y}}_k^{\phi} = \hat{\boldsymbol{\rho}}_k - \hat{\boldsymbol{\rho}}_{k-1}$, $\mathbf{y}_k^{\phi} = \boldsymbol{\rho}_k - \boldsymbol{\rho}_{k-1}$, and $\delta\mathbf{y}_k^{\phi} = \hat{\mathbf{y}}_k^{\phi} - \mathbf{y}_k^{\phi}$. Thus, Equation (4.3) can be reformulated as:

$$\delta\mathbf{y}_k^{\phi} = \mathbf{H}_k \delta\mathbf{x}_k - \mathbf{H}_{k-1}\delta\mathbf{x}_{k-1} + \mathbf{v}_k^{\phi}. \tag{4.6}$$

Equation (4.6) relates the system current and delay error states to the delta range innovation vector. It can be used as the system observation model. However, this equation is not in the general form for applying a KF. That is, the delay state term $\mathbf{H}_{k-1}\delta\mathbf{x}_{k-1}$ appears on the right side of the equation. In order to solve this problem, modifications are required. Two approaches proposed in this chapter are summarized as follows. Details are given in Section 3 and 4 respectively.

1) Augmentation of the state vector to include delay states yields:

$$\begin{aligned}&\begin{bmatrix}\delta\mathbf{x}_k \\ \hdashline \delta\mathbf{x}_{d,k}\end{bmatrix}=\left[\begin{array}{c:c}\mathbf{\Phi}_{k,k-1} & \mathbf{O} \\ \hdashline \mathbf{O} & \mathbf{I}\end{array}\right]\begin{bmatrix}\delta\mathbf{x}_{k-1} \\ \hdashline \delta\mathbf{x}_{d,k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{w}_{k-1} \\ \hdashline \mathbf{0}\end{bmatrix} \\ &\delta\mathbf{y}_k^{\phi}=\left[\begin{array}{c:c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\begin{bmatrix}\delta\mathbf{x}_k \\ \hdashline \delta\mathbf{x}_{d,k}\end{bmatrix}+\mathbf{v}_k^{\phi} \\ &\text{with } \mathbf{w}_k \sim \mathcal{N}(0,\mathbf{Q}_k),\ \mathbf{v}_k^{\phi} \sim \mathcal{N}(0,\mathbf{R}_k^{\phi})\end{aligned} \tag{4.7}$$

where $\delta\mathbf{x}_{d,k}=\delta\mathbf{x}_{k-1}$, $\mathbf{H}_{d,k}=\mathbf{H}_{k-1}$ and the subscript "d" represents delay.

2) Backward prediction of delay states by current states leads to:

$$\begin{aligned}\delta\mathbf{x}_k &=\mathbf{\Phi}_{k,k-1}\delta\mathbf{x}_{k-1}+\mathbf{w}_{k-1} \rightarrow \delta\mathbf{x}_{k-1}=\mathbf{\Phi}_{k-1,k}\delta\mathbf{x}_k-\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1} \\ \delta\mathbf{y}_k^{\phi} &=\mathbf{H}_k\delta\mathbf{x}_k-\mathbf{H}_{k-1}(\mathbf{\Phi}_{k-1,k}\delta\mathbf{x}_k-\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1})+\mathbf{v}_k^{\phi} \\ &= (\mathbf{H}_k-\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k})\delta\mathbf{x}_k+(\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1})+\mathbf{v}_k^{\phi}\end{aligned} \tag{4.8}$$

We denote the matrices in parentheses as $\mathbf{M}_k=\mathbf{H}_k-\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}$ and $\mathbf{W}_k=\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1}$, and we have:

$$\begin{aligned}&\delta\mathbf{x}_k=\mathbf{\Phi}_{k,k-1}\delta\mathbf{x}_{k-1}+\mathbf{w}_{k-1} \\ &\delta\mathbf{y}_k^{\phi}=\mathbf{M}_k\delta\mathbf{x}_k+\mathbf{W}_k+\mathbf{v}_k^{\phi} \\ &\text{with } \mathbf{w}_k \sim \mathcal{N}(0,\mathbf{Q}_k),\ \mathbf{v}_k^{\phi} \sim \mathcal{N}(0,\mathbf{R}_k^{\phi})\end{aligned} \tag{4.9}$$

4.3 Augmentation of system state vector (1st Method)

We augment the system state vector to include delay states to "remember" the system estimation errors at the start of the integration interval. These delay states are explicitly estimated from the previous measurement update and do not evolve over time in the current measurement update interval (e.g., [85, 86]). Therefore, we model them as random constants in the current measurement update interval. The augmented system model can be formulated as:

$$\begin{bmatrix}\delta\mathbf{x}_k \\ \hdashline \delta\mathbf{x}_{d,k}\end{bmatrix}=\left[\begin{array}{c:c}\mathbf{\Phi}_{k,k-1} & \mathbf{O} \\ \hdashline \mathbf{O} & \mathbf{I}\end{array}\right]\begin{bmatrix}\delta\mathbf{x}_{k-1} \\ \hdashline \delta\mathbf{x}_{d,k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{w}_{k-1} \\ \hdashline \mathbf{0}\end{bmatrix} \tag{4.10}$$

where $\delta\mathbf{x}_{d,k}$ denotes the delay error state vector; $\mathbf{\Phi}_{k,k-1}$ is the transition matrix for current states; $\mathbf{I}$ is an identity matrix; $\mathbf{w}_{k-1}$ represents the system process noises.

The state estimation error covariance and system process noise error covariance matrices are given in their partitioned form as:

$$\mathbf{P}'_k = \left[\begin{array}{c|c} \mathbf{P}_k & \mathbf{P}_{cd,k} \\ \hline \mathbf{P}_{dc,k} & \mathbf{P}_{d,k} \end{array}\right],\ \mathbf{Q}'_k = \left[\begin{array}{c|c} \mathbf{Q}_k & \mathbf{O} \\ \hline \mathbf{O} & \mathbf{O} \end{array}\right] \tag{4.11}$$

where $\mathbf{P}_k$ is the error covariance of $\delta\mathbf{x}_k$; $\mathbf{P}_{d,k}$ is the error covariance of $\delta\mathbf{x}_{d,k}$ which is explicitly estimated from the previous measurement update; $\mathbf{P}_{cd,k}$ and $\mathbf{P}_{dc,k}$ are the cross error covariance matrices between the current error state vector $\delta\mathbf{x}_k$ and the delay error state vector $\delta\mathbf{x}_{d,k}$; $\mathbf{Q}_k$ is the process noise error covariance of $\mathbf{w}_k$; $\mathbf{O}$ is a zero matrix.

The observation model for processing delta range measurements is formulated as:

$$\delta\mathbf{y}_k^{\phi} = \left[\begin{array}{c|c} \mathbf{H}_k & -\mathbf{H}_{d,k} \end{array}\right] \left[\begin{array}{c} \delta\mathbf{x}_k \\ \hline \delta\mathbf{x}_{d,k} \end{array}\right] + \mathbf{v}_k^{\phi} \tag{4.12}$$

where $\mathbf{H}_k$ and $\mathbf{H}_{d,k}$ are the Jacobian matrices related to the time instants of the start and end of the integration interval; $\delta\mathbf{y}_k^{\phi}$ is the delta range innovation vector; $\mathbf{v}_k^{\phi}$ represents the measurement noise.

As compared with Equation (4.6), in this arrangement, we actually consider the "delay states" as a part of the current states. Thus, all the terms are assumed to be related with the current time instant k.

For the code derived pseudorange measurements, the observation model is:

$$\hat{\boldsymbol{\rho}}_k - \boldsymbol{\rho}_k = \mathbf{H}_k \delta\mathbf{x}_k + \mathbf{v}_k^{\rho} \tag{4.13}$$

Substituting $\delta\mathbf{y}_k^{\rho} = \hat{\boldsymbol{\rho}}_k - \boldsymbol{\rho}_k$ into Equation(4.13), the overall system observation model can be derived in its partitioned form as:

$$\left[\begin{array}{c} \delta\mathbf{y}_k^{\rho} \\ \hline \delta\mathbf{y}_k^{\phi} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{H}_k & \mathbf{O} \\ \hline \mathbf{H}_k & -\mathbf{H}_{d,k} \end{array}\right] \left[\begin{array}{c} \delta\mathbf{x}_k \\ \hline \delta\mathbf{x}_{d,k} \end{array}\right] + \left[\begin{array}{c} \mathbf{v}_k^{\rho} \\ \hline \mathbf{v}_k^{\phi} \end{array}\right], \text{with } \mathbf{R}'_k = \left[\begin{array}{c|c} \mathbf{R}_k^{\rho} & \mathbf{O} \\ \hline \mathbf{O} & \mathbf{R}_k^{\phi} \end{array}\right] \tag{4.14}$$

where $\mathbf{R}_k^{\rho}$ and $\mathbf{R}_k^{\phi}$ are the pseudorange and delta range measurement error covariance matrices, which are uncorrelated with each other.

Equation (4.10) and (4.14) comprise the new state space system and observation models.

4.3.1 Measurement updates

We process the pseudorange and delta range measurements as two batches of data, and the dimension of observation vector will be half of the case when we process all the data in one batch.

4.3.1.1 Batch measurement update of delta range measurement

The batch measurement update of delta range measurements is performed as follows:

$$
\begin{aligned}
&\left[\begin{array}{c}\mathbf{K}_k\\ \hline \mathbf{K}_{d,k}\end{array}\right]=\left[\begin{array}{c|c}\mathbf{P}_k^- & \mathbf{P}_{cd,k}^-\\ \hline \mathbf{P}_{dc,k}^- & \mathbf{P}_{d,k}^-\end{array}\right]\left[\begin{array}{c}\mathbf{H}_k^T\\ \hline -\mathbf{H}_{d,k}^T\end{array}\right]\left[\left[\begin{array}{c|c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\left[\begin{array}{c|c}\mathbf{P}_k^- & \mathbf{P}_{cd,k}^-\\ \hline \mathbf{P}_{dc,k}^- & \mathbf{P}_{d,k}^-\end{array}\right]\left[\begin{array}{c}\mathbf{H}_k^T\\ \hline -\mathbf{H}_{d,k}^T\end{array}\right]+\mathbf{R}_k^{\phi}\right]^{-1}\\
&\left[\begin{array}{c}\delta\hat{\mathbf{x}}_k^+\\ \hline \delta\hat{\mathbf{x}}_{d,k}^+\end{array}\right]=\left[\begin{array}{c}\delta\hat{\mathbf{x}}_k^-\\ \hline \delta\hat{\mathbf{x}}_{d,k}^-\end{array}\right]+\left[\begin{array}{c}\mathbf{K}_k\\ \hline \mathbf{K}_{d,k}\end{array}\right]\left[\delta\mathbf{y}_k^{\phi}-\left[\begin{array}{c|c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\left[\begin{array}{c}\delta\hat{\mathbf{x}}_k^-\\ \hline \delta\hat{\mathbf{x}}_{d,k}^-\end{array}\right]\right]\\
&\left[\begin{array}{c|c}\mathbf{P}_k^+ & \mathbf{P}_{cd,k}^+\\ \hline \mathbf{P}_{dc,k}^+ & \mathbf{P}_{d,k}^+\end{array}\right]=\left[\left[\begin{array}{c|c}\mathbf{I} & \mathbf{O}\\ \hline \mathbf{O} & \mathbf{I}\end{array}\right]-\left[\begin{array}{c}\mathbf{K}_k\\ \hline \mathbf{K}_{d,k}\end{array}\right]\left[\begin{array}{c|c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\right]\left[\begin{array}{c|c}\mathbf{P}_k^- & \mathbf{P}_{cd,k}^-\\ \hline \mathbf{P}_{dc,k}^- & \mathbf{P}_{d,k}^-\end{array}\right],
\end{aligned}
\tag{4.15}
$$

where $\mathbf{K}_k$ is the Kalman gain for the current error state vector $\delta\mathbf{x}_k$, while $\mathbf{K}_{d,k}$ is the Kalman gain for the delay error state vector $\delta\mathbf{x}_{d,k}$.

As stated before, the delay error states will not be updated from current measurement update. Thus, we can set $\mathbf{K}_{d,k}$ to be a zero matrix. Hereby, Equation (4.15) can be simplified as:

$$
\begin{aligned}
&\mathbf{K}_k=\left[\mathbf{P}_k^-\mathbf{H}_k^T-\mathbf{P}_{cd,k}^-\mathbf{H}_{d,k}^T\right]\left[\begin{array}{l}\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T-\mathbf{H}_{d,k}\mathbf{P}_{dc,k}^-\mathbf{H}_k^T\\ -\mathbf{H}_k\mathbf{P}_{cd,k}^-\mathbf{H}_{d,k}^T+\mathbf{H}_{d,k}\mathbf{P}_{d,k}^-\mathbf{H}_{d,k}^T+\mathbf{R}_k^{\phi}\end{array}\right]^{-1}\\
&\delta\hat{\mathbf{x}}_k^+=\delta\hat{\mathbf{x}}_k^-+\mathbf{K}_k[\delta\mathbf{y}_k^{\phi}-\mathbf{H}_k\delta\hat{\mathbf{x}}_k^-+\mathbf{H}_{d,k}\delta\hat{\mathbf{x}}_{d,k}^-]\\
&\mathbf{P}_k^+=(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-+\mathbf{K}_k\mathbf{H}_{d,k}\mathbf{P}_{dc,k}^-\\
&\mathbf{P}_{cd,k}^+=(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{cd,k}^-+\mathbf{K}_k\mathbf{H}_{d,k}\mathbf{P}_{d,k}^-\\
&\mathbf{P}_{dc,k}^+=(\mathbf{P}_{cd,k}^+)^T
\end{aligned}
\tag{4.16}
$$

It is worth mentioning that, in Equation (4.16), we have not considered the measurement update of $\delta\hat{\mathbf{x}}_{d,k}$, but the uncertainty of $\delta\hat{\mathbf{x}}_{d,k}$ (i.e., $\mathbf{P}_{d,k}^{-}$, $\mathbf{P}_{dc,k}^{-}$, $\mathbf{P}_{cd,k}^{-}$) is used in the derivation of Kalman gain $\mathbf{K}_k$.

4.3.1.2 Sequential measurement updates of delta range measurement

For sequential processing, when measurement errors are uncorrelated, they can be processed one after another with zero-width time interval until all measurements are sequentially updated. Otherwise, decoupling of the correlated measurement errors must be conducted, and linear combinations of the measurements should be made to yield a new set of measurements whose errors are uncorrelated. After sequentially updating the measurements, the final state estimates and error covariance matrix should be the same as if the measurements are processed in one batch. Thus, the time consuming calculation of the matrix inversion can be prevented (e.g., [45]). The exact number of numerical operations involved in the matrix inversion is calculated in the Appendix B using the Gauss-Jordan Elimination method (e.g., [87]).

In sequential measurement update, the following parameters need to be firstly initialized. We denote $\mathbf{H}_k(i)$ and $\mathbf{H}_{d,k}(i)$ as the *i*-th row of $\mathbf{H}_k$ and $\mathbf{H}_{d,k}$, and use $\mathbf{R}_k^{\phi}(i,i)$ to represent the *i*-th main diagonal element of $\mathbf{R}_k^{\phi}$.

$$\mathbf{P}_k(0)=\mathbf{P}_k^{-},\ \mathbf{P}_{cd,k}(0)=\mathbf{P}_{cd,k}^{-},\ \mathbf{P}_{dc,k}(0)=\mathbf{P}_{dc,k}^{-},\ \delta\hat{\mathbf{x}}_k(0)=\delta\hat{\mathbf{x}}_k^{-} \tag{4.17}$$
$$\mathbf{H}_k=\begin{bmatrix}\mathbf{H}_k(1)\\ \vdots\\ \mathbf{H}_k(m)\end{bmatrix},\ \mathbf{H}_{d,k}=\begin{bmatrix}\mathbf{H}_{d,k}(1)\\ \vdots\\ \mathbf{H}_{d,k}(m)\end{bmatrix},\ \mathbf{R}_k^{\phi}=\begin{bmatrix}\mathbf{R}_k^{\phi}(1,1) & \dots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \mathbf{R}_k^{\phi}(m,m)\end{bmatrix}.$$

From *i* =1 to *m*, the measurements are updated sequentially as follows:

$$\begin{aligned}\mathbf{K}_k(i)=&\left[\mathbf{P}_k(i-1)\mathbf{H}_k(i)^T-\mathbf{P}_{cd,k}(i-1)\mathbf{H}_{d,k}(i)^T\right]\\ &\times[\mathbf{H}_k(i)\mathbf{P}_k(i-1)\mathbf{H}_k(i)^T-\mathbf{H}_{d,k}(i)\mathbf{P}_{dc,k}(i-1)\mathbf{H}_k(i)^T\\ &-\mathbf{H}_k(i)\mathbf{P}_{cd,k}(i-1)\mathbf{H}_{d,k}(i)^T+\mathbf{H}_{d,k}(i)\mathbf{P}_{d,k}\mathbf{H}_{d,k}(i)^T+\mathbf{R}_k^{\phi}(i,i)]^{-1}\\ \delta\hat{\mathbf{x}}_k(i)=&\,\delta\hat{\mathbf{x}}_k(i-1)+\mathbf{K}_k(i)[\delta\mathbf{y}_k^{\phi}(i)-\mathbf{H}_k(i)\delta\hat{\mathbf{x}}_k(i-1)+\mathbf{H}_{d,k}(i)\delta\hat{\mathbf{x}}_{d,k}]\\ \mathbf{P}_k(i)=&\,[1-\mathbf{K}_k(i)\mathbf{H}_k(i)]\mathbf{P}_k(i-1)+\mathbf{K}_k(i)\mathbf{H}_{d,k}(i)\mathbf{P}_{dc,k}(i-1)\\ \mathbf{P}_{cd,k}(i)=&\,[1-\mathbf{K}_k(i)\mathbf{H}_k(i)]\mathbf{P}_{cd,k}(i-1)+\mathbf{K}_k(i)\mathbf{H}_{d,k}(i)\mathbf{P}_{d,k}\\ \mathbf{P}_{dc,k}(i)=&\,\mathbf{P}_{cd,k}(i)^T.\end{aligned} \tag{4.18}$$

4.3.1.3 Measurement update of pseudorange measurements

The batch processing of pseudorange measurements is conducted after we process the delta range measurements. The *a posteriori* estimates from delta range measurement update (i.e., $\delta\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$ in Equation (4.18)) are used here as *a priori* estimates (i.e., $\delta\hat{\mathbf{x}}_k^-$ and $\mathbf{P}_k^-$) in Equation (4.19).

$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T+\mathbf{R}_k^\rho)^{-1}\\ \delta\hat{\mathbf{x}}_k^+ &= \delta\hat{\mathbf{x}}_k^- + \mathbf{K}_k[\delta\mathbf{y}_k^\rho - \mathbf{H}_k\delta\hat{\mathbf{x}}_k^-]\\ \mathbf{P}_k^+ &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-.\end{aligned} \tag{4.19}$$

For sequential processing, we define:

$$\mathbf{H}_k = \begin{bmatrix}\mathbf{H}_k(1)\\ \vdots\\ \mathbf{H}_k(m)\end{bmatrix}, \mathbf{R}_k^\rho = \begin{bmatrix}\mathbf{R}_k^\rho(1,1) & \dots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \mathbf{R}_k^\rho(m,m)\end{bmatrix}. \tag{4.20}$$

From i =1 to m, the measurements are sequentially updated as:

$$\begin{aligned}&\mathbf{K}_k(m+i) = \mathbf{P}_k(m+i-1)\mathbf{H}_k(i)^T[\mathbf{H}_k(i)\mathbf{P}_k(m+i-1)\mathbf{H}_k(i)^T+\mathbf{R}_k^\rho(i,i)]^{-1}\\ &\delta\hat{\mathbf{x}}_k(m+i) = \delta\hat{\mathbf{x}}_k(m+i-1)+\mathbf{K}_k(m+i)[\delta\mathbf{y}_k^\rho(i)-\mathbf{H}_k(i)\delta\hat{\mathbf{x}}_k(m+i-1)]\\ &\mathbf{P}_k(m+i) = [1-\mathbf{K}_k(m+i)\mathbf{H}_k(i)]\mathbf{P}_k(m+i-1).\end{aligned} \tag{4.21}$$

After all measurements are updated, the *a posteriori* estimates are calculated as:

$$\delta\hat{\mathbf{x}}_k^+ = \delta\hat{\mathbf{x}}_k(m+m), \mathbf{P}_k^+ = \mathbf{P}_k(m+m). \tag{4.22}$$

They are used as the delay states and covariance parameters in the next recursion:

$$\delta\hat{\mathbf{x}}_{d,k+1} = \delta\hat{\mathbf{x}}_k(m+m), \mathbf{P}_{d,k+1} = \mathbf{P}_k(m+m). \tag{4.23}$$

The whole process repeats at the next epoch.

4.3.2 Computational burden analysis

It is known that the sequential processing can save computational time. But how much can be exactly saved? In order to answer this question, the number of numerical operations involved in the additions and multiplications should be counted separately. The method utilized in [48] has been employed in this chapter, where subtraction is counted as addition and the division over scalar is considered as multiplication by the inverse of the scalar. The "*n*" represents the dimension of state vector and "*m*" is the dimension of the observation vector. We denote the "*nn*" as an *n*-by-*n* square matrix, and "$nn \times nn$" stands for an *n*-by-*n* matrix multiplies another *n*-by-*n* matrix. "*n1*" means an *n*-dimensional vector, "*1*" means a scalar value. The numbers in the column of "Num." represent the amount of operations involved in the measurement update of both pseudorange and delta range measurements, while the numbers in the parentheses denote the operations encountered in the delta range measurement update only.

For measurement update in one batch (i.e., Equation (4.16) and (4.19)), the involved numerical operations are given in the left side of **Table 4-1**. For measurement update in sequence (i.e., Equation (4.18) and (4.21)), they are given in the right side of **Table 4-1** summaries the overall conducted numerical operations involved in sequential and batch measurement updates.

Table 4-1. Numerical operations involved in sequential and batch measurement updates.

Batch processing				Sequential processing			
Operation	Num.	Add.	Multi.	Operation	Num.	Add.	Multi.
$nn \times nm$	3(2)	$(n-1)nm$	mn^2	$nn \times n1$	3(2)×m	$n(n-1)$	n^2
$nm \times mn$	5(4)	$(m-1)n^2$	mn^2	$n1+n1$	3(2)×m	n	–
$nm+nm$	1(1)	nm	–	$1n \times nn$	10(8)×m	$n(n-1)$	n^2
$nm \times mm$	2(1)	$(m-1)mn$	nm^2	$1n \times n1$	8(6)×m	$n-1$	n
$mn \times nn$	10(8)	$mn(n-1)$	mn^2	$1+1$	8(6)×m	1	–
$mn \times nm$	5(4)	$(n-1)m^2$	nm^2	1×1	2(1)×m	–	1
$mm+mm$	5(4)	m^2	–	$n1 \times 1$	4(2)×m	–	n
$nn+nn$	5(4)	n^2	–	$n1 \times 1n$	5(4)×m	–	n^2
$mn \times n1$	3(2)	$m(n-1)$	mn	$nn+nn$	5(4)×m	n^2	–
$m1+m1$	3(2)	m	–				
$nm \times m1$	2(1)	$(m-1)n$	nm				
$n1+n1$	2(1)	n	–				
$\text{inv}(mm)$	2(1)	$\frac{3}{2}m^3-2m^2+\frac{m}{2}$	$\frac{3}{2}m^3-\frac{1}{2}m$				

Table 4-2. Computational burden comparison.

Operations	Additions	Operations	Multiplications
Batch	$3m^3+(7n-4)m^2+(18n^2-9n+1)m$	Batch	$3m^3+7nm^2+(18n^2+5n-1)m$
Sequential	$(18n^2-2n)m$	Sequential	$(18n^2+12n+2)m$

The correctness of calculations in **Table 4-2** can be easily verified when we consider the case that there is only one satellite in view (m=1). In this case, there should be no difference between the sequential and batch processing. We can prove this by assigning m to be 1. The batch processing returns $18n^2 - 2n$ for additions, and $18n^2 + 12n + 2$ for multiplications, which equals to the required operations from sequential processing.

Based on the results shown in **Table 4-2**, the relationships between the number of numerical operations and the dimension of the observation vector are illustrated in **Figure 4-1** and **Figure 4-2**, where the dimension of the state vector is 17 (i.e., n=17).

Figure 4-1 shows that if more measurements are processed, more computational time can be saved by sequential processing. However, if the number of measurements is smaller than 4, no big difference can be observed. In this chapter, we update the delta range and pseudorange as two batches of data. And hence, the dimension of observation vector is always smaller than 11 (i.e., $m \le 11$), which saves a large amount of numerical operations.

Figure 4-2 illustrates that the measurement update of the delta range requires much more numerical operations than those from the pseudorange measurement update (i.e., increased computational complexity in Equation (4.18) as compared with Equation (4.21)). However, when compared with batch processing (i.e., red curves in **Figure 4-2**), the advantage of applying sequential measurement update is obvious.

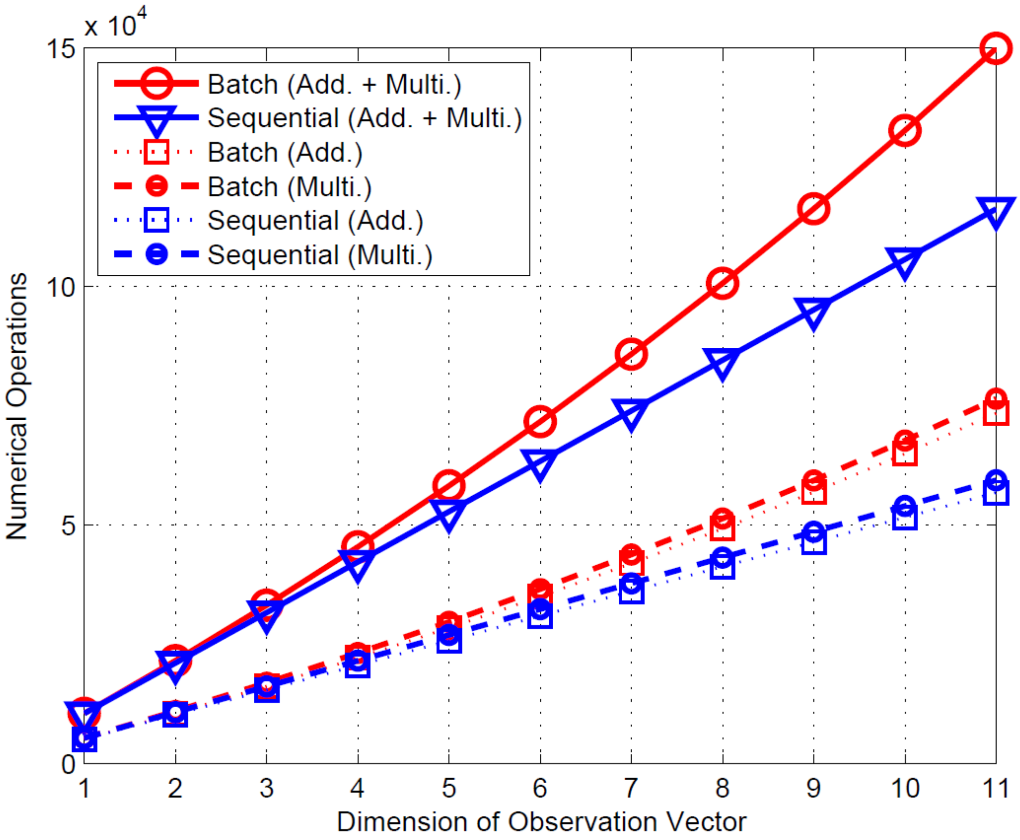


Figure 4-1. Computational burden analysis (addition versus multiplication).

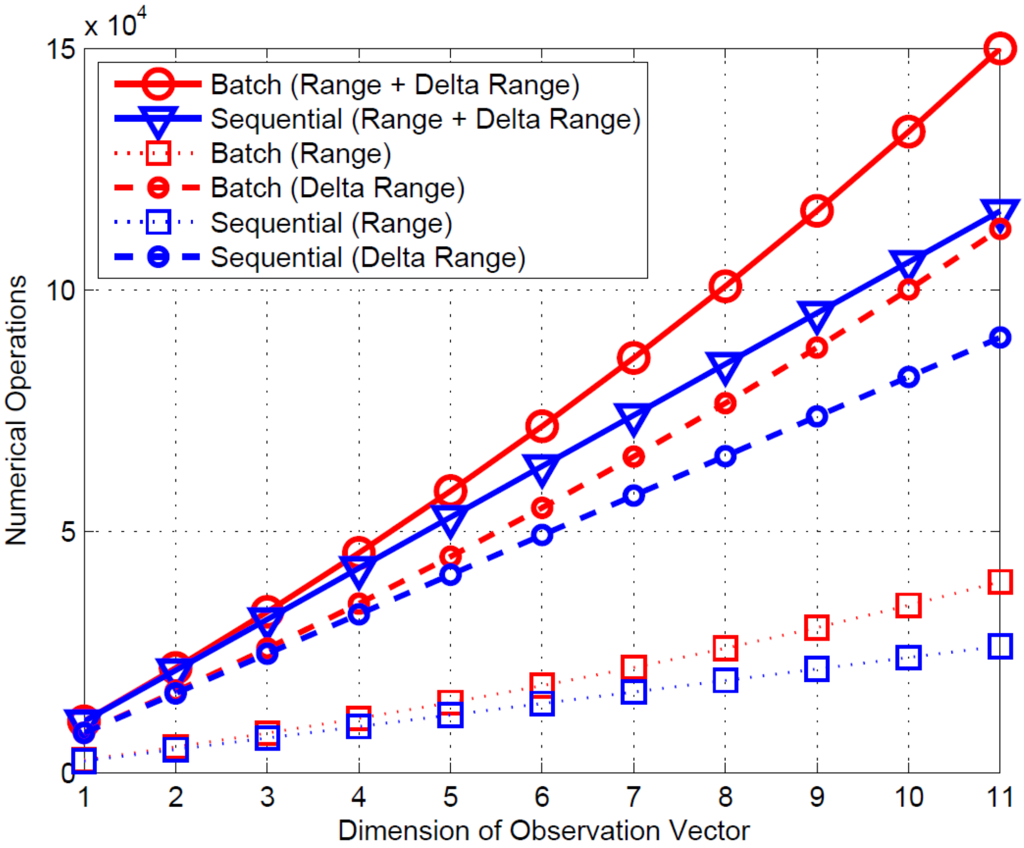


Figure 4-2. Computational burden analysis (pseudorange versus delta range).

4.4 Backward prediction of delay states by current states (2nd method)

In the second method, we predict the delay states backwards by current states considering the integrated system process noises. Equation (4.6) is repeated here for convenience

$$\delta\mathbf{y}_k^{\phi} = \mathbf{H}_k \delta\mathbf{x}_k - \mathbf{H}_{k-1} \delta\mathbf{x}_{k-1} + \mathbf{v}_k^{\phi}.$$

After the backward prediction, the delay states are represented by current states as:

$$\delta\mathbf{x}_k = \mathbf{\Phi}_{k,k-1} \delta\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \rightarrow \delta\mathbf{x}_{k-1} = \mathbf{\Phi}_{k-1,k} \delta\mathbf{x}_k - \mathbf{\Phi}_{k-1,k} \mathbf{w}_{k-1}. \tag{4.24}$$

Substituting $\delta\mathbf{x}_{k-1}$ from Equation (4.24) into Equation (4.6) yields:

$$\begin{aligned} \delta\mathbf{y}_k^{\phi} &= \mathbf{H}_k \delta\mathbf{x}_k - \mathbf{H}_{k-1} (\mathbf{\Phi}_{k-1,k} \delta\mathbf{x}_k - \mathbf{\Phi}_{k-1,k} \mathbf{w}_{k-1}) + \mathbf{v}_k^{\phi} \\ &= (\mathbf{H}_k - \mathbf{H}_{k-1} \mathbf{\Phi}_{k-1,k}) \delta\mathbf{x}_k + (\mathbf{H}_{k-1} \mathbf{\Phi}_{k-1,k} \mathbf{w}_{k-1}) + \mathbf{v}_k^{\phi}. \end{aligned} \tag{4.25}$$

We denote the matrices in parentheses as:

$$\mathbf{M}_k = \mathbf{H}_k - \mathbf{H}_{k-1} \mathbf{\Phi}_{k-1,k} \text{ and } \mathbf{W}_k = \mathbf{H}_{k-1} \mathbf{\Phi}_{k-1,k} \mathbf{w}_{k-1}. \tag{4.26}$$

System and observation models can be re-formulated as:

$$\begin{aligned} &\delta\mathbf{x}_k = \mathbf{\Phi}_{k,k-1} \delta\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \\ &\delta\mathbf{y}_k^{\phi} = \mathbf{M}_k \delta\mathbf{x}_k + \mathbf{W}_k + \mathbf{v}_k^{\phi} \\ &\text{with } \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k),\ \mathbf{v}_k^{\phi} \sim \mathcal{N}(0, \mathbf{R}_k^{\phi}). \end{aligned} \tag{4.27}$$

In Equation (4.27), the newly formulated measurement noise $(\mathbf{W}_k + \mathbf{v}_k^{\phi})$ contains the system noise $\mathbf{w}_{k-1}$. Hence, the cross-correlation between system process noise and measurement noise must be considered. The covariance of the correlated noises is calculated as:

$$\begin{aligned} \mathbf{C}_{k,k-1} &= E[\mathbf{w}_{k-1} (\mathbf{W}_k + \mathbf{v}_k^{\phi})^T] \\ &= E[\mathbf{w}_{k-1} (\mathbf{H}_{k-1} \mathbf{\Phi}_{k-1,k} \mathbf{w}_{k-1} + \mathbf{v}_k^{\phi})^T] \\ &= \mathbf{Q}_{k-1} (\mathbf{\Phi}_{k-1,k})^T \mathbf{H}_{k-1}^T. \end{aligned} \tag{4.28}$$

The measurement error covariance matrix for the new formulated system observation model (i.e., in Equation (4.27)) is computed as:

$$\begin{aligned}\mathbf{R}_{k,k-1}^{cor} &= E[(\mathbf{W}_k+\mathbf{v}_k^{\phi})(\mathbf{W}_k+\mathbf{v}_k^{\phi})^T] \\ &= E[(\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1}+\mathbf{v}_k^{\phi})(\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{w}_{k-1}+\mathbf{v}_k^{\phi})^T] \\ &= \mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{Q}_{k-1}(\mathbf{\Phi}_{k-1,k})^T\mathbf{H}_{k-1}^T+\mathbf{R}_k^{\phi}\end{aligned} \tag{4.29}$$

where $\mathbf{Q}_{k-1}=E[\mathbf{w}_{k-1}\mathbf{w}_{k-1}^T]$, and $\mathbf{R}_k^{\phi}=E[\mathbf{v}_k^{\phi}(\mathbf{v}_k^{\phi})^T]$.

Thus, the KF measurement update can be summarized as

$$\begin{aligned}&\mathbf{K}_k=(\mathbf{P}_k^-\mathbf{M}_k^T+\mathbf{C}_{k,k-1})[\mathbf{M}_k\mathbf{P}_k^-\mathbf{M}_k^T+\mathbf{R}_{k,k-1}^{cor}+\mathbf{M}_k\mathbf{C}_{k,k-1}+\mathbf{C}_{k,k-1}^T\mathbf{M}_k^T]^{-1} \\ &\delta\hat{\mathbf{x}}_k^+=\delta\hat{\mathbf{x}}_k^-+\mathbf{K}_k(\delta\mathbf{y}_k^{\phi}-\mathbf{M}_k\delta\hat{\mathbf{x}}_k^-) \\ &\mathbf{P}_k^+=\mathbf{P}_k^--\mathbf{K}_k[\mathbf{M}_k\mathbf{P}_k^-\mathbf{M}_k^T+\mathbf{R}_{k,k-1}^{cor}+\mathbf{M}_k\mathbf{C}_{k,k-1}+\mathbf{C}_{k,k-1}^T\mathbf{M}_k^T]\mathbf{K}_k^T.\end{aligned} \tag{4.30}$$

The detail derivation of Equation (4.30) can be also found in the reference (e.g., [44]), which is omitted here.

4.4.1 Decoupling of correlated measurement errors

In Equation (4.30), the measurement error covariance matrix $\mathbf{R}_{k,k-1}^{cor}$ is full. Thus, in order to apply sequential measurement update, decoupling has to be conducted [87, 88]. For symmetric positive definite matrix $\mathbf{R}_{k,k-1}^{cor}$, the Jordan form decomposition is always possible (e.g., [22]), as shown in Equation (4.31).

$$\begin{aligned}\mathbf{R}_{k,k-1}^{cor} &= E[(\mathbf{W}_k+\mathbf{v}_k^{\phi})(\mathbf{W}_k+\mathbf{v}_k^{\phi})^T] \\ &= \mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}\mathbf{Q}_{k-1}(\mathbf{\Phi}_{k-1,k})^T\mathbf{H}_{k-1}^T+\mathbf{R}_k^{\phi}=\mathbf{UDU}^T\end{aligned} \tag{4.31}$$

where $\mathbf{U}$ is an orthonormal matrix (i.e., $\mathbf{U}^T\mathbf{U}=\mathbf{I}$), which contains the eigenvectors of $\mathbf{R}_{k,k-1}^{cor}$, and $\mathbf{D}$ is the diagonal matrix containing the eigenvalues of $\mathbf{R}_{k,k-1}^{cor}$.

The equivalent measurement vector $\delta\mathbf{y}_k^s$, diagonal matrix $\mathbf{R}_{k,k-1}^s$ and the transformed observation matrix $\mathbf{M}_k^s$ are computed as:

$$\begin{aligned}&\delta\mathbf{y}_k^s=\mathbf{U}^T\delta\mathbf{y}_k^{\phi} \\ &\mathbf{R}_{k,k-1}^s=E[\mathbf{U}^T(\mathbf{W}_k+\mathbf{v}_k^{\phi})(\mathbf{W}_k+\mathbf{v}_k^{\phi})^T\mathbf{U}]=\mathbf{U}^T\mathbf{UDU}^T\mathbf{U}=\mathbf{D} \\ &\mathbf{M}_k^s=\mathbf{U}^T\mathbf{M}_k=\mathbf{U}^T\mathbf{H}_k-\mathbf{U}^T\mathbf{H}_{k-1}\mathbf{\Phi}_{k-1,k}=\mathbf{H}_k^s-\mathbf{H}_{k-1}^s\mathbf{\Phi}_{k-1,k}.\end{aligned} \tag{4.32}$$

4.4.2 Computational burden analysis

After the decoupling of correlated measurement errors, sequential processing can be applied. The newly formulated observation model and error covariance matrix are used, and the following parameters are initialized.

$$\mathbf{P}_k(0) = \mathbf{P}_k^-, \delta\hat{\mathbf{x}}_k(0) = \delta\hat{\mathbf{x}}_k^-, \mathbf{C}_{k,k-1}(i) = \mathbf{Q}_{k-1}(\mathbf{\Phi}_{k-1,k})^T \mathbf{H}_{k-1}^s(i)^T \tag{4.33}$$
$$\mathbf{M}_k^s = \begin{bmatrix} \mathbf{M}_k^s(1) \\ \vdots \\ \mathbf{M}_k^s(m) \end{bmatrix}, \mathbf{R}_{k,k-1}^s = \begin{bmatrix} \mathbf{R}_{k,k-1}^s(1,1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{R}_{k,k-1}^s(m,m) \end{bmatrix}$$

where $\mathbf{M}_k^s(i)$ denotes the i-th row of $\mathbf{M}_k^s$, and $\mathbf{R}_{k,k-1}^s(i,i)$ represents the i-th diagonal element of $\mathbf{R}_{k,k-1}^s$.

From i =1 to m, the delta ranges are sequentially updated as:

$$\begin{aligned} \mathbf{K}_k(i) &= [\mathbf{P}_k(i-1)\mathbf{M}_k^s(i)^T + \mathbf{C}_{k,k-1}(i)]\{\mathbf{M}_k^s(i)\mathbf{P}_k(i-1)\mathbf{M}_k^s(i)^T \\ &\quad + \mathbf{R}_{k,k-1}^s(i,i) + \mathbf{M}_k^s(i)\mathbf{C}_{k,k-1}(i) + \mathbf{C}_{k,k-1}(i)^T \mathbf{M}_k^s(i)^T\}^{-1} \\ \delta\hat{\mathbf{x}}_k(i) &= \delta\hat{\mathbf{x}}_k(i-1) + \mathbf{K}_k(i)[\delta\mathbf{y}_k^s(i) - \mathbf{M}_k^s(i)\delta\hat{\mathbf{x}}(i-1)] \\ \mathbf{P}_k(i) &= \mathbf{P}_k(i-1) - \mathbf{K}_k(i)\{\mathbf{M}_k^s(i)\mathbf{P}_k(i-1)\mathbf{M}_k^s(i)^T + \mathbf{R}_{k,k-1}^s(i,i) \\ &\quad + \mathbf{M}_k^s(i)\mathbf{C}_{k,k-1}(i) + \mathbf{C}_{k,k-1}(i)^T \mathbf{M}_k^s(i)^T\}\mathbf{K}_k(i)^T. \end{aligned} \tag{4.34}$$

The sequential processing of pseudorange measurements is given in Equation (4.19) and (4.21). After sequentially updating all measurements, the *a posteriori* states and error covariance matrix are updated as:

$$\delta\hat{\mathbf{x}}_k^+ = \delta\hat{\mathbf{x}}(m+m), \mathbf{P}_k^+ = \mathbf{P}_k(m+m). \tag{4.35}$$

Table 4-3 shows the computational burden involved in the sequential and batch measurement updates in this method. The relationship between the number of numerical operations and the dimension of observation vector is illustrated in **Figure 4-3.**

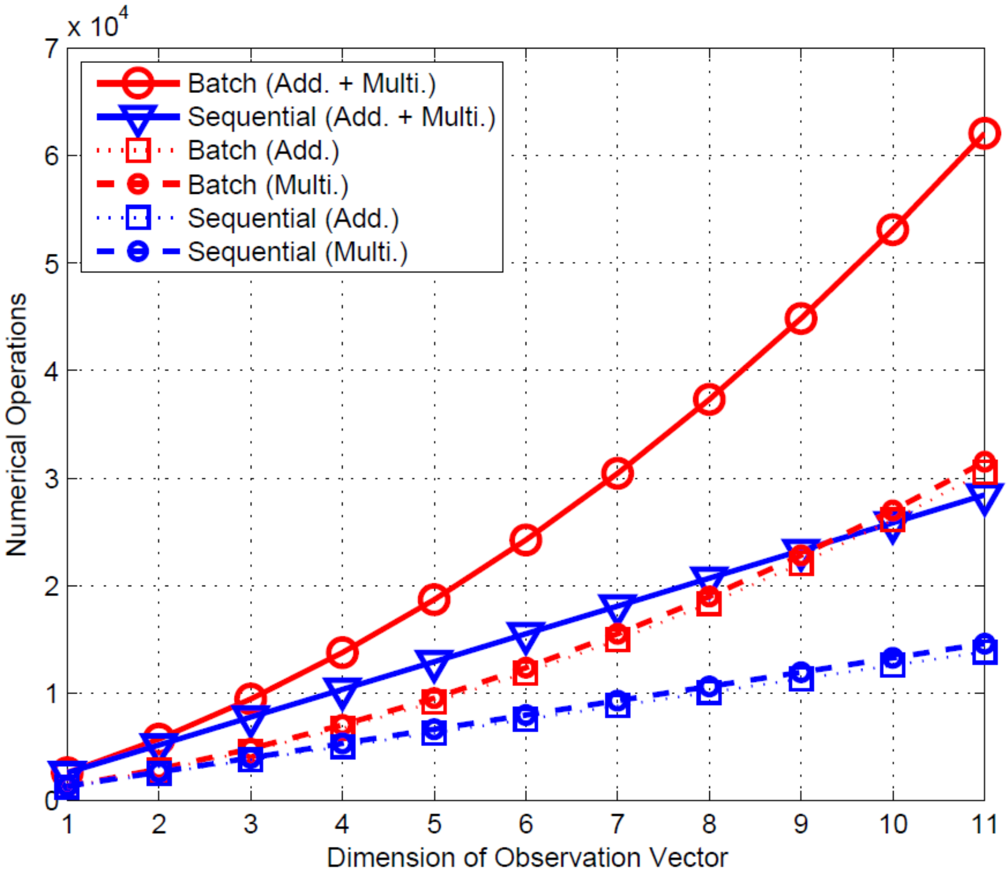


Figure 4-3. Numerical operations involved in sequential and batch measurement updates.

Table 4-3. Computational burden comparison.

Operations	Additions	Operations	Multiplications
Batch	$1.5m^3+(8n-2)m^2+(4n^2-2n+0.5)m$	Batch	$1.5m^3+8nm^2+(4n^2+2n-0.5)m$
Sequential	$(4n^2+6n)m$	Sequential	$(4n^2+10n+1)m$

Figure 4-3 shows that the computational burden of sequential measurement update is smaller than that from batch processing, especially when the dimension of observation vector grows high.

4.5 Comparisons of two approaches

From the above analysis, the latter approach requires less computational burden. Nevertheless, its drawbacks are severe. Firstly, in the recursive algorithm, the inverse of transition matrix (i.e., $\mathbf{\Phi}_{k-1,k}=\left(\mathbf{\Phi}_{k,k\text{-}1}\right)^{-1}$) needs to be computed. Although this matrix inversion can be circumvented with appropriate algebraic substitutions in Equation

(4.30) (e.g., [44]), the resultant equations for computing the Kalman gain and covariance update are still very complicated. Secondly, in the system state space models, the system noise and measurement noise are correlated with each other. This cross-correlation will significantly increase the complexity of the KF algorithm, which makes the filter difficult to be tuned. Thirdly, the $\mathbf{R}_{k,k-1}^{cor}$ is full and time-varying during the application. Therefore, the measurement decoupling step needs to be made at each step of the measurement update, which introduces additional computational burden. A concise comparison is made in **Table 4-4**.

Table 4-4. Comparison of methods.

Problems:	Method 1	Method 2
Augmented system state vector	YES	No
Correlated system process and measurement noises	No	YES
Decoupling of correlated measurements at measurement update rate	No	YES
Inversion of transition matrix $\mathbf{\Phi}_{k-1,k} = \left(\mathbf{\Phi}_{k,k\text{-}1}\right)^{-1}$	No	YES

In **Table 4-4**, the advantages of the first method are quite clear. Although we augment the system state vector, we never perform the computation directly on the augmented system and observation models. Instead, we use the “sub-matrices” in the system and observation model (i.e., Equation (4.10) and (4.14)) to calculate the partitioned Kalman gain and update the state vectors with covariance matrices. In following sections, the first method is employed in the INS/GPS tightly-coupled integration. The remaining questions will be answered.

1. Can sequential processing successfully handle the measurement update of different types of measurements without degrading the system estimation accuracy?
2. If yes, are there any conditions, which must be met?

4.6 Simulation setup

A field collected UAV trajectory is used in the simulation. The position and velocity dynamic profiles are depicted in **Figure 4-4** and **Figure 4-5**. For verifying the algorithm, the simulation is simplified. That is, the GPS receiver is assumed to be running at DGPS mode. Therefore, the majority of GPS measurement errors are corrected or minimized to small values, i.e., the ionospheric and tropospheric delays, satellite clock errors. And hence, they are neglected in this simulation.

For the INS/GPS integration, we assume that a high level GPS receiver is used (i.e., NovAtel DL-4plus GPS receiver). The parameters shown in **Table 4-5** are used to simulate the receiver-related measurement errors. For the instantaneous Doppler shift measurements, we approximate them by the first order central difference of 50 Hz carrier phase data (i.e., $d_k^{(i)} \approx (\phi_{k+1}^{(i)} - \phi_{k-1}^{(i)}) / (2 \cdot \Delta t)$) with thermal noise and multipath errors given in **Table 4-5**. The multipath errors are generated using the SATNAV toolbox. The basic idea is to form the zero-elevation angle equivalent pseudorange multipath errors using a linear autoregressive model (i.e., "filter" function in MATLAB) and scale these errors by the cosine of their elevation angles before they are applied to the range measurements.

Regarding the IMU, a LandMark^{TM}20 eXT MEMS-IMU from Gladiator Technologies Inc. is simulated. The main sensor errors are generated according to its specification as shown in **Table 4-6**. The IMU raw data in body frame are plotted in **Figure 4-6**. For the KF, the system time-update happens at the IMU measurements update rate, which is 50 Hz, while the measurement-update happens at the GPS measurement update rate, which is 5 Hz. The simulation is conducted with 8 satellites in view.

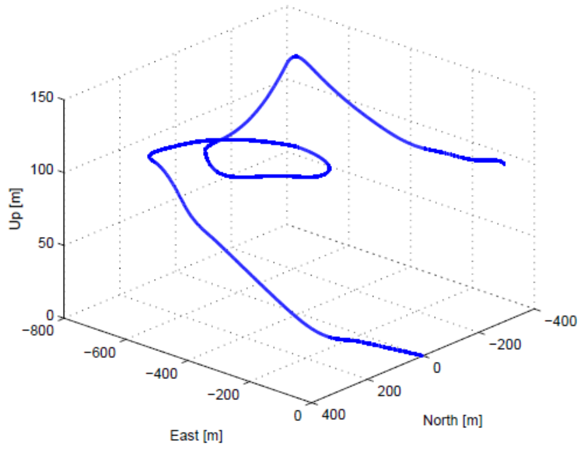


Figure 4-4. UAV trajectory.

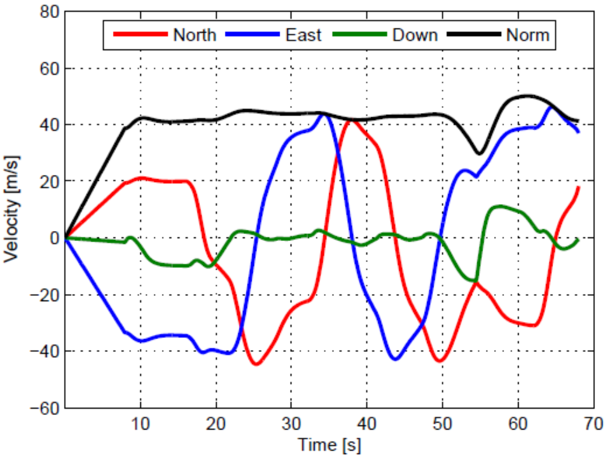


Figure 4-5. Dynamic profiles.

Table 4-5. Parameters for receiver-related measurement errors (1 sigma).

	Receiver thermal noise	Multipath errors
Pseudorange (DLL)	0.5 m	0.4 m
Doppler (FLL)	0.05 m/s	0.04 m/s
Carrier phase (PLL)	0.001 m	0.004 m

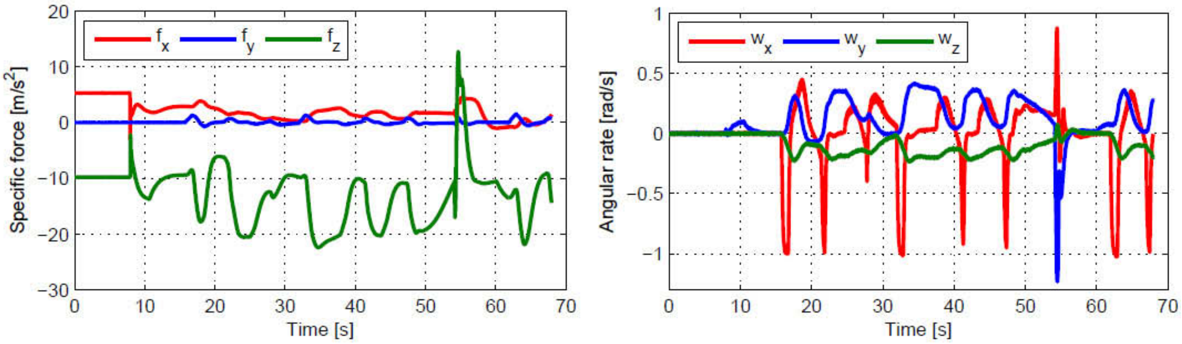


Figure 4-6. Simulated LandMark™ 20 eXT MEMS-IMU raw data.

Table 4-6. Landmark™20 eXT MEMS-based IMU performance specification.

Gyroscope (Angular rates)	Bias in-run stability 20 [°/h] (1σ)	Noise (ARW) 0.035[°/s/√Hz] (1σ)	Scale Factor Error ≤1000 [ppm]
Accelerometer (Specific forces)	Bias in-run stability 20 [μg] (1σ)	Noise (VRW) 40 [μg/√Hz] (1σ)	Scale Factor Error ≤1000 [ppm]

4.7 Numerical result

4.7.1 System performance comparison using different approaches

This test is to show the advantage of correctly handling the errors in the delta range measurements (i.e., integrated measurements). We initialize the filter to be 1 m away to the north, east and down directions respectively from their true values. The errors analyzed here represent the norm of position, velocity and attitude errors. For instance, the norm of position errors will be calculated as $\|\Delta\mathbf{x}\| = \sqrt{\Delta x_n^2 + \Delta x_e^2 + \Delta x_d^2}$.

The comparisons are made between the conditions listed in **Table 4-7**, and the results are depicted in **Figure 4-7** and **Figure 4-8**. The "augmented" method denotes the first method introduced in this chapter (i.e., Equations (4.10) and (4.14)). The standard approach represents the method we introduced in chapter one (i.e., pure linearization based approach from Equations (1.43) and (1.45)). Joseph error covariance update formula is used in the KF to prevent the numerical instability, which will be introduced in Appendix C.

Table 4-7. Conditions for comparison.

Red	"Augmented" method using pseudorange and delta range measurements
Green	"Augmented" method using delta range measurements only
Blue	"Standard" method using pseudorange for position and delta range for velocity
Black	"Standard" method using pseudorange for position and Doppler shift for velocity

For the "augmented" approach (red curves), both code derived pseudorange and carrier-phase derived delta range measurements are related with position estimation through the system observation model (i.e., Equation (4.14)). The pseudorange measurements provide the absolute positioning information, while the delta range measurements provide the relative displacement of system position from one time instant to the next. In this test, the filter is initialized from a position, which is one meter away to the north, east and down directions respectively from their true values. In such case, when the filter gives more confidence on GPS pseudorange measurements, the position estimation errors will decrease rapidly, and the filter shows noisier performance. On the other hand, when more confidence is given on the delta range measurements, the positioning estimation errors decrease more slowly. However, the system will present much improved estimation accuracy after the transient phrase of the filter. By giving correct weights on the pseudorange and delta range measurements based on parameters shown in **Table 4-5** the system performance plotted in red is obtained. In **Figure 4-7**, the third subplot is the "zoom-in" version of the second subplot, where the red curve represents the velocity estimation error. It shows that with correctly handling the errors involved in the delta range measurement (i.e., integrated measurement), the system can correctly track the accelerations and jerks involved in the trajectory.

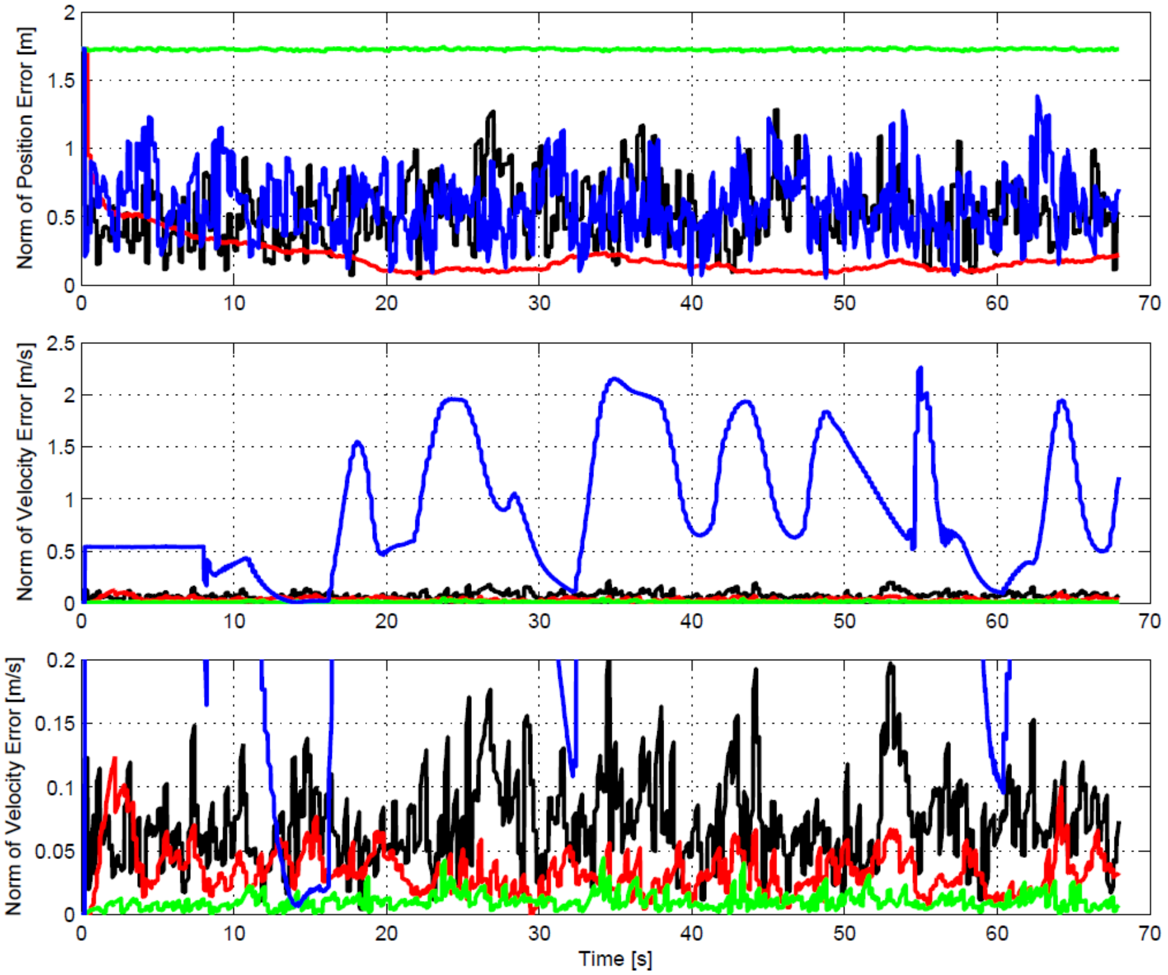


Figure 4-7. Comparison of position and velocity estimation results

The green curves in **Figure 4-7** represent the system performances obtained from an extreme case. That is, if no pseudorange measurements (absolute positioning information) are available in the filter (i.e., having only the delta range measurements), the system presents the best velocity estimation results. However, for position, incorrect position initialization errors remain through the whole trajectory, as shown in the first subplot of **Figure 4-7**.

The blue curves in **Figure 4-7** represent the estimation results using the "standard" approach. Since the dynamics encountered in the trajectory is severe, large velocity estimation errors occur which are subject to the accelerations and jerks involved in the trajectory. The same problem can also be observed in the attitude estimation results.

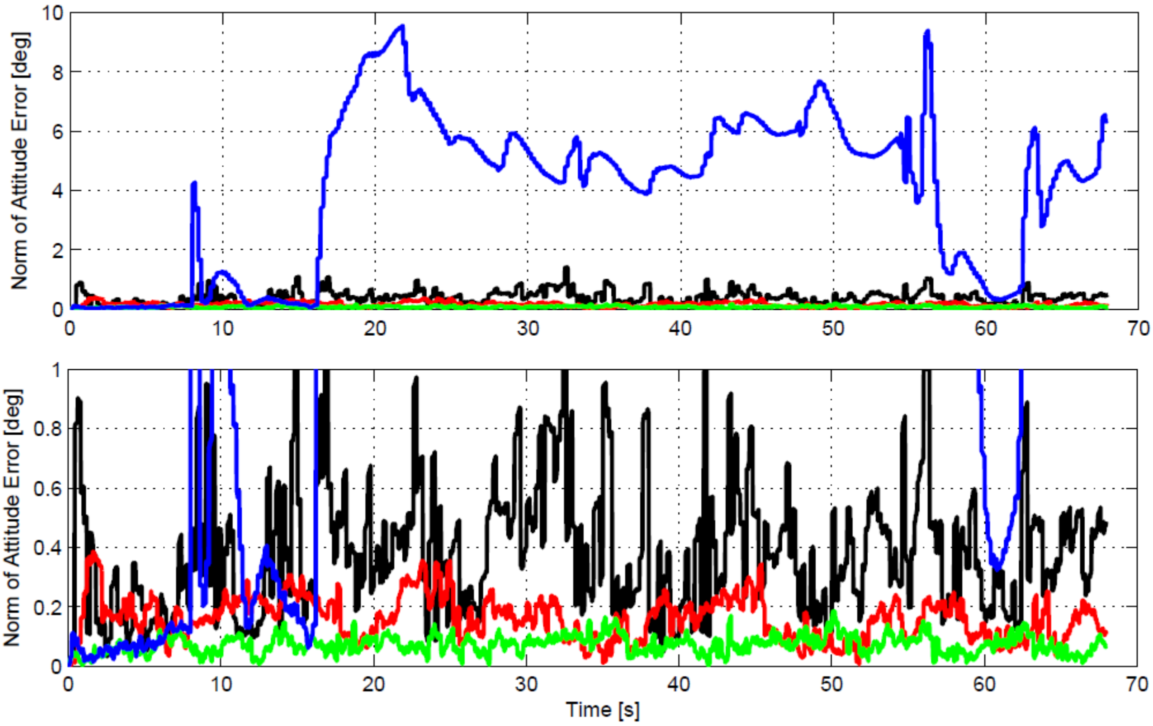


Figure 4-8. Comparison of attitude estimation results.

In **Figure 4-8**, the second subplot is the "zoom-in" version of the first subplot. In this chapter, we integrate the measurements obtained from a single GPS receiver antenna with measurements from an INS without redundant attitude information (e.g., from magnetometers or a multi-antenna GPS system), therefore no direct attitude fixes are available as measurements. In such case, the attitude errors are mainly related to the velocity estimation errors through the off-diagonal parameters in their error covariance matrices. Hence, the more accurate velocity estimates, the greater the dampening will be on the attitude errors.

As an alternate approach, the Doppler measurements are typically used for velocity determination (black curves in **Figure 4-7** and **Figure 4-8**). However, they are the raw measurements from the receiver frequency lock loop (FLL), which is known to be noisier than the carrier phase derived measurements.

4.7.2 System performance comparison using sequential and batch measurement updates

This test is to prove that the sequential measurement update of different types of measurements can present identical system performance with respect to that of batch measurement update. Test setup remains unchanged as the aforementioned one. **Figure 4-9** plots the position, velocity and attitude estimation errors from 50 Monte Carlo runs using the "augmented" system with sequential and batch measurement updates.

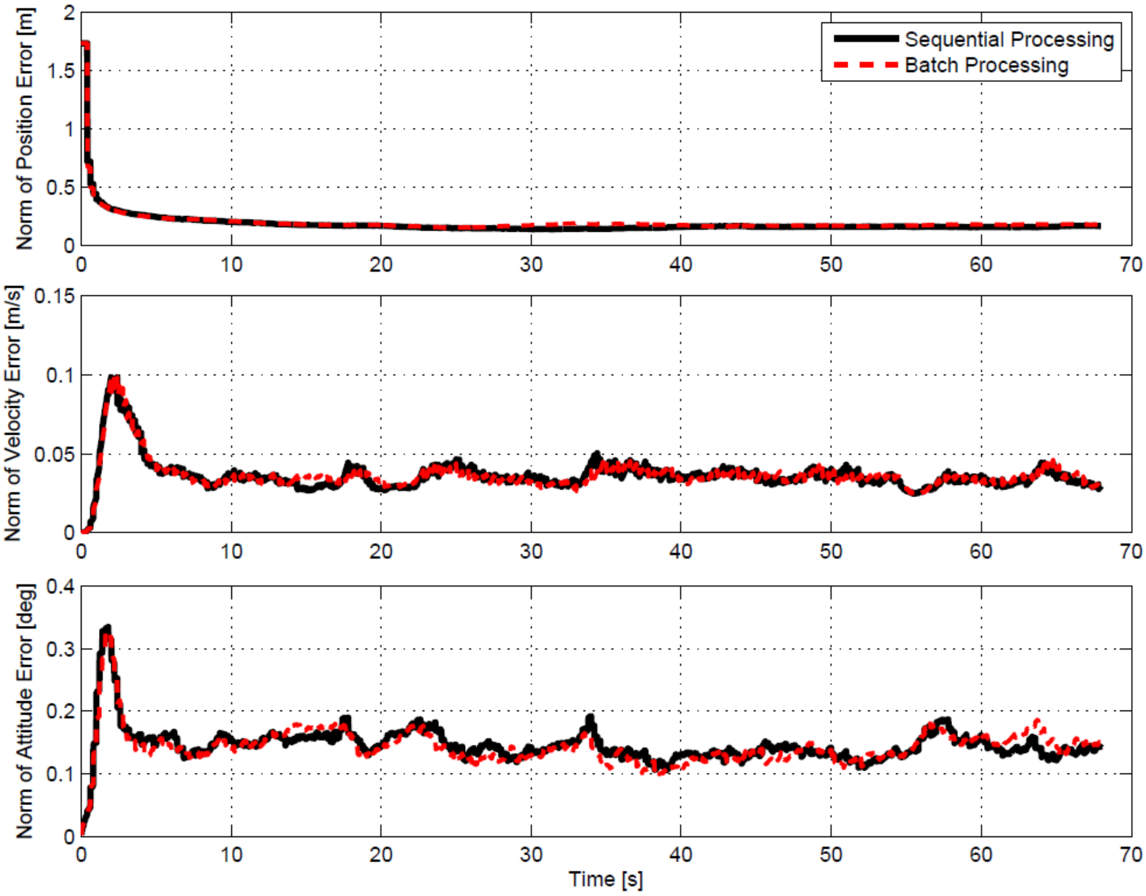


Figure 4-9. System performance comparison between sequential and batch measurement updates.

As depicted in the figure, both approaches present statistically comparable estimation results. Nevertheless, two rules must be obeyed.

Firstly, when we process a group of measurements, which involve both integrated and non-integrated measurements, we must always process the integrated measurements (delta range measurements) before the non-integrated measurements (pseudorange measurements) to guarantee that there is no measurement update before we process the integrated measurements.

Secondly, the cross covariance matrices (i.e., $\mathbf{P}_{cd,k}$ and $\mathbf{P}_{dc,k}$) need to be carefully treated. That is, in sequential processing, among the measurement scalar updates, the cross covariance matrices, i.e., $\mathbf{P}_{cd,k}$ and $\mathbf{P}_{dc,k}$, are updated from one measurement to the other with zero-width time interval, as shown in Equation (4.18).

Without fulfilling these two requirements, we will not obtain comparable estimation results from the sequential and batch measurement updates in updating different types of measurements.

4.8 Summary

The errors involved in the delta range measurement are related to the pseudorange errors at the start and end of the integration interval. The correct handling of these errors is essential in cases that the vehicle is maneuvering under high dynamics with low GPS update rate. After comparing the two approaches proposed in this chapter, the method with the "augmented" system state vector can be considered as more suitable for practical usage. And hence, it is employed in the INS/GPS tightly-coupled integration. Simulation was conducted using a field collected UAV trajectory. Numerical results have shown that the filter can correctly track the accelerations and jerks involved in the application. Moreover, sequential processing was applied for measurement update in the KF to prevent the matrix inverse computation. 50 Monte Carlo runs have been conducted, which verified that the sequential processing can update different types of measurements (i.e., integrated and non-integrated) without degrading the system estimation accuracy, in case certain rules are obeyed.

5. Summary and Conclusions

5.1 Summary

This thesis is on low-cost INS/GPS integrated navigation system with focuses on the improvements of system performance (i.e., robustness and accuracy) by using more advanced sensor fusion algorithms. The aim of this work is to take advantage of the recent progress made in the field of estimation theory to handle nonlinear and non-Gaussian issues more accurately, which makes the usage of low-cost sensors in the integrated navigation system feasible. Moreover, an algorithm for high dynamic applications using delta phase measurement has been proposed in this thesis. By using additional delay states in the Kalman filter to remember the estimation errors in the start point of delta phase integration interval, the system can accurately estimate the instantaneous dynamics of a vehicle.

This thesis started with the introduction of basic knowledge on low-cost L1 GPS data processing, INS principles and the integrated navigation system. The merits of an INS/GPS integrated system with respect to a standalone GPS device have been shown using a field experiment. After that, the nonlinear filtering estimation methods were discussed, with a special focus on particle filtering. The discussion started from the introduction of the general recursive Bayesian filter. In order to use it practically, assumptions must be made, which leads to the unscented Kalman filter and particle filter. As one highlight of this thesis, for overcoming the problems of the particle filter (e.g., processing load), an unscented particle filter has been proposed with the concept that we can improve the importance density function in generating particles more intelligently rather than increasing the number of particles based on a poor quality importance density function. By using the *a posteriori* estimates from the unscented Kalman filter to specify the importance density, the outperformance of the unscented particle filter has been observed with a dramatically reduced number of particles. In order to prove this idea, a simulation test has been made assuming a nonlinear non-Gaussian problem. Besides the simulation, field experiments were also made to compare and analyze the integration performance. Quaternions have been used as the representation of attitude in this work. Two approaches in quaternion propagation over time have been investigated and compared. From the analysis, the following conclusions can be made: quaternion-derived system is more confident on the estimation of heading related states for land-based navigation; in an integration system, using a lower level IMU can provide similar navigation performance with respect to using a higher quality IMU, if the sensor errors can be correctly estimated; the unscented particle filter works fine in field test with a dramatically reduced number of particles, and its outperformance depends on the signal environment. If it is open sky environment, the unscented particle filter shows basically the same behavior as using the unscented Kalman filter.

The work presented in this thesis is not only on land-based navigation system, but also on high dynamic UAV applications. In order to have accurate velocity estimates of a high dynamic platform, the carrier delta phase measurement can be used. However, the delta phase is a type of integrated measurements with errors strongly related to the range errors at the start and end of the integration interval. Normal methods circumvent these errors with approximations, which leads to large velocity estimation errors. For solving this problem, delay states have been used here to “remember” the range errors at the start of the integration interval. Furthermore, sequential processing has been utilized in the measurement update to avoid the time consuming matrix inversion computation in the derivation of the Kalman gain. The highlight of this work is that during sequential measurement update of integrated (delta phase) and non-integrated (range) measurements, no compromise has been made on their errors.

5.2 Conclusions

The contributions of this thesis can be summarized as:

1. A new nonlinear non-Gaussian filter (unscented particle filter) has been proposed for low-cost INS/GPS integration. Both simulation test and field experiments have been used to verify its outperformance with respect to other nonlinear filters. Numerical results have shown that it works well with a dramatically reduced number of particles, and it presents advantages when system encounters non-Gaussian problem.
2. In this thesis, quaternions have been used in the INS/GPS system as the representation of attitude, which do not exhibit singularity issue. Besides, it is more confident on the estimation of heading related states with respect to the Euler angle-based approach for land-based navigation. This thesis has provided two approaches

in the propagation of quaternions over time. Both are suitable to be used in a low-cost INS/GPS integration system.

3. For high dynamic applications, delay states have been introduced in order to accurately capture the delta phase measurement errors, so that the accurate instantaneous velocity can be estimated.
4. Sequential processing has been successfully applied on updating different types of measurements (integrated and non-integrated) without compromise made on their errors.

Appendix A: Basics on Quaternions

Quaternions are represented as a complex number with four bases (i.e., 1, i, j, k):

$$\mathbf{q} = q_1 + q_2 i + q_3 j + q_4 k \tag{A-1}$$

In this thesis, we denote $\otimes$ as quaternion product. The quaternion products between its bases show the following properties:

$$\begin{aligned} &i \otimes i = -1,\ i \otimes j = k,\ i \otimes k = -j, \\ &j \otimes j = -1, j \otimes k = i,\ j \otimes i = -k, \\ &k \otimes k = -1, k \otimes i = j,\ k \otimes i = -i, \end{aligned} \tag{A-2}$$

With above properties, the product of quaternions $\mathbf{q}$ and $\mathbf{a}$ is computed as:

$$\begin{aligned} \mathbf{q} \otimes \mathbf{a} &= \left(q_1 a_1 - q_2 a_2 - q_3 a_3 - q_4 a_4\right) + \left(q_1 a_2 + q_2 a_1 + q_3 a_4 - q_4 a_3\right) i \\ &\quad + \left(q_1 a_3 - q_2 a_4 + q_3 a_2 - q_4 a_1\right) j + \left(q_1 a_4 + q_2 a_3 - q_3 a_2 + q_4 a_1\right) k \\ &= \begin{bmatrix} q_1 & -q_2 & -q_3 & -q_4 \\ q_2 & q_1 & -q_4 & q_3 \\ q_3 & q_4 & q_1 & -q_2 \\ q_4 & -q_3 & q_2 & q_1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} a_1 & -a_2 & -a_3 & -a_4 \\ a_2 & a_1 & a_4 & -a_3 \\ a_3 & -a_4 & a_1 & a_2 \\ a_4 & a_3 & -a_2 & a_1 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \end{aligned} \tag{A-3}$$

As shown in equation A-3, we can easily find that $\mathbf{q} \otimes \mathbf{a}$ does not equal to $\mathbf{a} \otimes \mathbf{q}$. Therefore, the quaternion multiplication is not commutative.

The conjugate of quaternions $\mathbf{q}$ is presented as:

$$\bar{\mathbf{q}} = q_1 - q_2 i - q_3 j - q_4 k \tag{A-4}$$

The inverse of quaternions $\mathbf{q}$ is computed as:

$$\mathbf{q}^{-1} = \frac{\bar{\mathbf{q}}}{\sqrt{q_1^2 + q_2^2 + q_3^2 + q_4^2}} = \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|} \tag{A-5}$$

Quaternions are often represented in vector form as:

$$\mathbf{q} = \begin{bmatrix} q_1 \\ \vec{\mathbf{q}} \end{bmatrix} \tag{A-6}$$

where $\vec{\mathbf{q}} = [q_2, q_3, q_4]^T$.

Quaternions are used to compute the rotational transformation from one coordinate system to the other. Based on Euler's theorem, given two right-handed orthogonal coordinates, one coordinate system axes can be aligned onto another by successive single rotations along fixed axes. The corresponding rotation vector is defined as that one coordinate axes can be aligned with the second coordinate axes by rotating $\|\boldsymbol{\varphi}\|$ radian (rotation angle magnitude) about an invariant axis (rotation axis) [8]. This invariant axis is referred to as the Euler axis. Given a unit vector $\mathbf{u}$ along the Euler axis, the quaternion vector $\mathbf{q}$ that represents the same rotational transformation is formulated as:

$$\mathbf{q} = \begin{bmatrix} \cos(0.5\|\boldsymbol{\varphi}\|) \\ \sin(0.5\|\boldsymbol{\varphi}\|)\mathbf{u} \end{bmatrix}, \text{with } \mathbf{u} = \begin{bmatrix} u_x, u_y, u_z \end{bmatrix}^T \tag{A-7}$$

Apparently, $\mathbf{q}$ has the normality property (i.e., $\|\mathbf{q}\| = 1$). That is, although the quaternion vector $\mathbf{q}$ has four parameters, it has only three degrees of freedom.

Based on Equation A-7, the computation of quaternion vector given a rotation vector $\boldsymbol{\varphi} = [\varphi_1, \varphi_2, \varphi_3]^T$ is handled as follows:

$$\|\boldsymbol{\varphi}\| = \sqrt{\varphi_1^2 + \varphi_2^2 + \varphi_3^2}, \mathbf{u} = \frac{\boldsymbol{\varphi}}{\|\boldsymbol{\varphi}\|}, \mathbf{q} = \begin{bmatrix} \cos(0.5\|\boldsymbol{\varphi}\|) \\ \sin(0.5\|\boldsymbol{\varphi}\|)\dfrac{\boldsymbol{\varphi}}{\|\boldsymbol{\varphi}\|} \end{bmatrix} \tag{A-8}$$

Given a quaternion vector $\mathbf{q} = [q_1, q_2, q_3, q_4]^T$, we can also compute its corresponding rotation vector as:

$$\begin{aligned} &\cos(0.5\|\boldsymbol{\varphi}\|) = q_1, \sin(0.5\|\boldsymbol{\varphi}\|) = \|\vec{\mathbf{q}}\| = \sqrt{q_2^2 + q_3^2 + q_4^2} \\ &\|\boldsymbol{\varphi}\| = 2 \cdot \operatorname{atan2}(\|\vec{\mathbf{q}}\|/q_1), \mathbf{u} = \left[q_2/\|\vec{\mathbf{q}}\|, q_3/\|\vec{\mathbf{q}}\|, q_4/\|\vec{\mathbf{q}}\|\right]^T \end{aligned} \tag{A-9}$$

The rotation vector is formulated as $\boldsymbol{\varphi} = \|\boldsymbol{\varphi}\| \cdot \mathbf{u}$.

If the quaternion vector $\mathbf{q}$ represents the rotation from navigation frame to body frame, the rotation matrix $\mathbf{R}_n^b$ can be formulated using the quaternion parameters as:

$$\mathbf{R}_n^b = \begin{bmatrix} q_1^2+q_2^2-q_3^2-q_4^2 & 2(q_2q_3-q_1q_4) & 2(q_1q_3+q_2q_4) \\ 2(q_2q_3+q_1q_4) & q_1^2-q_2^2+q_3^2-q_4^2 & 2(q_3q_4-q_1q_2) \\ 2(q_2q_4-q_1q_3) & 2(q_1q_2+q_3q_4) & q_1^2-q_2^2-q_3^2+q_4^2 \end{bmatrix} \tag{A-10}$$

An advantage of using quaternions is that the $\mathbf{R}_n^b$ formed in this way only involves quadratic terms and does not have trigonometric functions, as shown in Equation (1.37). Thus, if a rotation matrix $\mathbf{R}_n^b$ is known, we can compute quaternion parameters as:

$$\mathbf{q} = \begin{bmatrix} 0.5\sqrt{1+\mathbf{R}_n^b[1,1]+\mathbf{R}_n^b[2,2]+\mathbf{R}_n^b[3,3]} \\ (\mathbf{R}_n^b[3,2]-\mathbf{R}_n^b[2,3])/4q_1 \\ (\mathbf{R}_n^b[1,3]-\mathbf{R}_n^b[3,1])/4q_1 \\ (\mathbf{R}_n^b[2,1]-\mathbf{R}_n^b[1,2])/4q_1 \end{bmatrix} \tag{A-11}$$

Based on Equation (1.37), the rotation matrix $\mathbf{R}_n^b$ represented in Euler angles is formulated as:

$$\mathbf{R}_n^b = \left(\mathbf{R}_b^n\right)^T = \begin{bmatrix} C\gamma C\beta & C\gamma S\beta S\alpha - S\gamma C\alpha & C\gamma S\beta C\alpha + S\gamma S\alpha \\ S\gamma C\beta & S\gamma S\beta S\alpha + C\gamma C\alpha & S\gamma S\beta C\alpha - C\gamma S\alpha \\ -S\beta & C\beta S\alpha & C\beta C\alpha \end{bmatrix}^T \tag{A-12}$$

From an element to element comparison in matrix from Equation A-10 and A-12, we can compute Euler angle in terms of quaternion parameters:

$$\begin{aligned} &\alpha = \mathrm{atan2}(2(q_3q_4-q_1q_2), 1-2(q_2^2+q_3^2)) \\ &\beta = \mathrm{asin}(-2(q_2q_4+q_1q_3)) \\ &\gamma = \mathrm{atan2}(2(q_2q_3-q_1q_4), 1-2(q_3^2+q_4^2)) \end{aligned} \tag{A-13}$$

where atan2 denotes the four quadrant inverse tangent function.

Appendix B: Transformation of Quaternion Covariance to Euler Angle Covariance

Due to the quaternion normalization constraint, the degree of freedom of a quaternion vector is three rather than four. Thus, if we use quaternion vector elements as states for representing attitude, the dimension of the state vector is 4×1, but the dimension of state error covariance matrix is 3×3. In this section, the propagation of quaternion covariance to Euler angle covariance is derived.

In Appendix A, Equation A-13 shows the nonlinear relationship between the quaternion vector elements and the Euler angle elements. Based on this relationship, the propagation of quaternion covariance to Euler angle covariance can be accomplished using the covariance law as:

$$\mathbf{P}_{3\times3}^{euler} = \mathbf{A}_{3\times4}\mathbf{P}_{4\times4}^{quat}\left(\mathbf{A}_{3\times4}\right)^{T} \tag{B-1}$$

where $\mathbf{P}_{3\times3}^{euler}$ is the covariance matrix of Euler angle vector; $\mathbf{P}_{4\times4}^{quat}$ is the covariance matrix of quaternion vector; $\mathbf{A}_{3\times4}$ is the Jacobian matrix containing the partial derivatives of the Euler angle elements with respect to the quaternion elements.

The Jacobian matrix $\mathbf{A}_{3\times4}$ is compuated as:

$$\mathbf{A}_{3\times4} = \begin{bmatrix} \frac{\partial\alpha}{\partial q_1} & \frac{\partial\alpha}{\partial q_2} & \frac{\partial\alpha}{\partial q_3} & \frac{\partial\alpha}{\partial q_4} \\ \frac{\partial\beta}{\partial q_1} & \frac{\partial\beta}{\partial q_2} & \frac{\partial\beta}{\partial q_3} & \frac{\partial\beta}{\partial q_4} \\ \frac{\partial\gamma}{\partial q_1} & \frac{\partial\gamma}{\partial q_2} & \frac{\partial\gamma}{\partial q_3} & \frac{\partial\gamma}{\partial q_4} \end{bmatrix} \tag{B-2}$$

where

$$\frac{\partial\alpha}{\partial q_1} = \frac{2\cdot q_2\cdot\left(2\cdot q_2^2+2\cdot q_3^2-1\right)}{\left(2\cdot q_2^2+2\cdot q_3^2-1\right)^2+\left(2\cdot q_1\cdot q_2-2\cdot q_3\cdot q_4\right)^2}$$

$$\frac{\partial\alpha}{\partial q_2} = \frac{2\cdot q_1\cdot\left(2\cdot q_2^2+2\cdot q_3^2-1\right)-4\cdot q_2\cdot\left(2\cdot q_1\cdot q_2-2\cdot q_3\cdot q_4\right)}{\left(2\cdot q_2^2+2\cdot q_3^2-1\right)^2+\left(2\cdot q_1\cdot q_2-2\cdot q_3\cdot q_4\right)^2}$$

$$\frac{\partial \alpha}{\partial q_3} = -\frac{2 \cdot q_4 \cdot \left(2 \cdot q_2^2 + 2 \cdot q_3^2 - 1\right) + 4 \cdot q_3 \cdot \left(2 \cdot q_1 \cdot q_2 - 2 \cdot q_3 \cdot q_4\right)}{\left(2 \cdot q_2^2 + 2 \cdot q_3^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_2 - 2 \cdot q_3 \cdot q_4\right)^2}$$

$$\frac{\partial \alpha}{\partial q_4} = -\frac{2 \cdot q_3 \cdot \left(2 \cdot q_2^2 + 2 \cdot q_3^2 - 1\right)}{\left(2 \cdot q_2^2 + 2 \cdot q_3^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_2 - 2 \cdot q_3 \cdot q_4\right)^2}$$

$$\frac{\partial \beta}{\partial q_1} = -\frac{2 \cdot q_3}{\sqrt{1 - \left(2 \cdot q_1 \cdot q_3 + 2 \cdot q_2 \cdot q_4\right)^2}}$$

$$\frac{\partial \beta}{\partial q_2} = -\frac{2 \cdot q_4}{\sqrt{1 - \left(2 \cdot q_1 \cdot q_3 + 2 \cdot q_2 \cdot q_4\right)^2}}$$

$$\frac{\partial \beta}{\partial q_3} = -\frac{2 \cdot q_1}{\sqrt{1 - \left(2 \cdot q_1 \cdot q_3 + 2 \cdot q_2 \cdot q_4\right)^2}}$$

$$\frac{\partial \beta}{\partial q_4} = -\frac{2 \cdot q_2}{\sqrt{1 - \left(2 \cdot q_1 \cdot q_3 + 2 \cdot q_2 \cdot q_4\right)^2}}$$

$$\frac{\partial \gamma}{\partial q_1} = \frac{2 \cdot q_4 \cdot \left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)}{\left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)^2}$$

$$\frac{\partial \gamma}{\partial q_2} = -\frac{2 \cdot q_3 \cdot \left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)}{\left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)^2}$$

$$\frac{\partial \gamma}{\partial q_3} = -\frac{2 \cdot q_2 \cdot \left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right) + 4 \cdot q_3 \cdot \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)}{\left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)^2}$$

$$\frac{\partial \gamma}{\partial q_4} = \frac{2 \cdot q_1 \cdot \left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right) - 4 \cdot q_4 \cdot \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)}{\left(2 \cdot q_3^2 + 2 \cdot q_4^2 - 1\right)^2 + \left(2 \cdot q_1 \cdot q_4 - 2 \cdot q_2 \cdot q_3\right)^2}$$

Appendix C: Calculation of Matrix Inversion using Gauss-Jordan Elimination Method

The Gauss-Jordan Elimination method is one of the methods, which are widely used to calculate the inverse of a square matrix (e.g., [87]). It is done by augmenting the square matrix with an identity matrix of the same dimension, and passing through the following matrix operations:

$$\text{Multiply}\left[\begin{array}{cccc|cccc} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & 0 & 0 & \cdots & 1 \end{array}\right] \text{ by } \mathbf{A}^{-1} \text{ to have } \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & b_{11} & b_{12} & \cdots & b_{1n} \\ 0 & 1 & \cdots & 0 & b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & b_{n1} & b_{n2} & \cdots & b_{nn} \end{array}\right] \tag{C-1}$$

$$\mathbf{A} \qquad\qquad\qquad\qquad \mathbf{A}^{-1}$$

After the Gauss-Jordan elimination, the right side of the augmented matrix is the inverse of the original square matrix. In this contribution, we firstly transform the left side of the augmented matrix (original square matrix) to be a diagonal matrix and divide the augmented matrix by its diagonal elements on each row to have an identity matrix on the left side and the inverse of original matrix on the right side. The computation is shown as follows.

In the first step, we use the elementary row operations to introduce zeros in the first column beginning from the second row, which involves $2n$ multiplications and $2n$-1 additions.

$$\left[\begin{array}{cccc|cccc} a_{11} & a_{12} & \cdots & a_{1n} & b_{11} & b_{12} & \cdots & b_{1n} \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_{21}^{(1)} & b_{22}^{(1)} & \cdots & b_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} & b_{n1}^{(1)} & b_{n2}^{(1)} & \cdots & b_{nn}^{(1)} \end{array}\right]_{n\times 2n} \tag{C-2}$$

After the first step, the elements in the rows below the first will change. We indicate this by denoting the elements with a superscript, which stands for the number of arithmetic operations involved.

In the second step, we introduce zeros in the second column except the second row, which involves 2*n*-1 multiplications and 2*n*-2 additions.

In the *n*-th step, we introduce zeros in the *n*-th column except the *n*-th row, which involves *n*+1 multiplications and *n* additions.

$$\left[\begin{array}{cccc|cccc} a_{11} & 0 & \cdots & 0 & b_{11}^{(n-1)} & b_{12}^{(n-1)} & \cdots & b_{1n}^{(n-1)} \\ 0 & a_{22}^{(1)} & \cdots & 0 & b_{21}^{(n-1)} & b_{22}^{(n-1)} & \cdots & b_{2n}^{(n-1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn}^{(n-1)} & b_{n1}^{(n-1)} & b_{n2}^{(n-1)} & \cdots & b_{nn}^{(n-1)} \end{array}\right]_{n\times 2n} \qquad \text{C-3}$$

In the last step, we divide the augmented matrix by its diagonal elements on each row to have the matrix as shown in Equation C-1. The involved numerical operations are given in **Table C-1**.

Table C-1. Computational burden of matrix inversion.

Steps	Left side Multi.	Add.	Right side Multi.	Add.	Rows processed
1	*n*	*n*-1	*n*	*n*	*n*-1
2	*n*-1	*n*-2	*n*	*n*	*n*-1
⋮	⋮	⋮	⋮	⋮	⋮
n-1	2	1	*n*	*n*	*n*-1
n	1	0	*n*	*n*	*n*-1
Division	0	0	*n*	0	*n*

Using the summation formula:

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \qquad \text{C-4}$$

According to **Table C-1**, the number of multiplications involved in Gauss-Jordan Elimination is

$$(n-1)\sum_{i=1}^{n} i + n^2\left(n-1\right) + n^2 = (n-1)\frac{n(n+1)}{2} + n^2\left(n-1\right) + n^2 = \frac{3}{2}n^3 - \frac{1}{2}n \qquad \text{C-5}$$

And the required number of additions is computed as

$$(n-1)\sum_{i=1}^{n-1} i + n^2(n-1) = \frac{3}{2}n^3 - 2n^2 + \frac{n}{2} \qquad \text{C-6}$$

We can verify our computation by inverting a scalar number as an extreme example for square matrix inversion (i.e., n=1). The arithmetic operations involved contain obviously only one multiplication (division), which is the same as the results computed from Equation C-5 and C-6.

Appendix D: Sequential Measurement Update using Joseph Covariance Update Formula

In Equation (4.15) and (4.19), we use the simplified formula to derive the state error covariance matrices. However, such a simplified equation is known to be numerically unstable. In practice, the symmetric Joseph formula is widely used.

Using the Joseph formula, for the delta range measurement, the state covariance matrix will be updated as:

$$\begin{aligned}\left[\begin{array}{c|c}\mathbf{P}_k^+ & \mathbf{P}_{cd,k}^+ \\ \hline \mathbf{P}_{dc,k}^+ & \mathbf{P}_{d,k}^+\end{array}\right] &= \left[\left[\begin{array}{c|c}\mathbf{I} & \mathbf{O} \\ \hline \mathbf{O} & \mathbf{I}\end{array}\right] - \left[\begin{array}{c}\mathbf{K}_k \\ \hline \mathbf{K}_{d,k}\end{array}\right]\left[\begin{array}{c|c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\right]\left[\begin{array}{c|c}\mathbf{P}_k^- & \mathbf{P}_{cd,k}^- \\ \hline \mathbf{P}_{dc,k}^- & \mathbf{P}_{d,k}^-\end{array}\right] \\ &\times \left[\left[\begin{array}{c|c}\mathbf{I} & \mathbf{O} \\ \hline \mathbf{O} & \mathbf{I}\end{array}\right] - \left[\begin{array}{c}\mathbf{K}_k \\ \hline \mathbf{K}_{d,k}\end{array}\right]\left[\begin{array}{c|c}\mathbf{H}_k & -\mathbf{H}_{d,k}\end{array}\right]\right]^T + \left[\begin{array}{c}\mathbf{K}_k \\ \hline \mathbf{K}_{d,k}\end{array}\right]\mathbf{R}_k^{\phi}\left[\begin{array}{c|c}\mathbf{K}_k^T & \mathbf{K}_{d,k}^T\end{array}\right]\end{aligned} \quad \text{D-1}$$

By setting $\mathbf{K}_{d,k}$ to be a zero matrix, the *a posteriori* error covariance matrix $\mathbf{P}_k^+$ in Equation (4.16) will be replaced by:

$$\begin{aligned}\mathbf{P}_k^+ &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{H}_{d,k}\mathbf{P}_{dc,k}^-(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^T \\ &+ (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{cd,k}^-\mathbf{H}_{d,k}^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{H}_{d,k}\mathbf{P}_{d,k}^-\mathbf{H}_{d,k}^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{R}_k^{\phi}\mathbf{K}_k^T\end{aligned} \quad \text{D-2}$$

And accordingly, for sequential processing, from *i*=1 to *m*, it is updated as:

$$\begin{aligned}\mathbf{P}_k(i) &= [1-\mathbf{K}_k(i)\mathbf{H}_k(i)]\mathbf{P}_k(i-1)[1-\mathbf{K}_k(i)\mathbf{H}_k(i)]^T \\ &+ \mathbf{K}_k(i)\mathbf{H}_{d,k}(i)\mathbf{P}_{dc,k}(i-1)[1-\mathbf{K}_k(i)\mathbf{H}_k(i)]^T \\ &+ [1-\mathbf{K}_k(i)\mathbf{H}_k(i)]\mathbf{P}_{cd,k}(i-1)\mathbf{H}_{d,k}(i)^T\mathbf{K}_k(i)^T \\ &+ \mathbf{K}_k(i)\mathbf{H}_{d,k}(i)\mathbf{P}_{d,k}\mathbf{H}_{d,k}(i)^T\mathbf{K}_k(i)^T + \mathbf{K}_k(i)\mathbf{R}_k^{\phi}(i,i)\mathbf{K}_k(i)^T\end{aligned} \quad \text{D-3}$$

For batch processing of pseudorange measurements, we have

$$\mathbf{P}_k^+ = (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k^{\rho}\mathbf{K}_k^T \quad \text{D-4}$$

In sequential processing, from *i*=1 to *m*, we replace the $\mathbf{P}_k(m+i)$ in Equation (4.21) by

$$\begin{aligned}\mathbf{P}_k(m+i) &= [1-\mathbf{K}_k(m+i)\mathbf{H}_k(i)]\mathbf{P}_k(m+i-1)[1-\mathbf{K}_k(m+i)\mathbf{H}_k(i)]^T \\ &+ \mathbf{K}_k(m+i)\mathbf{R}_k^{\rho}(i,i)\mathbf{K}_k(m+i)^T\end{aligned} \qquad \text{D-5}$$

After the above substitutions, a set of new KF recursive equations using Joseph covariance update formula are derived. The involved computational burden from the sequential and batch measurement updates is given in **Table D-1**. The results are plotted in **Figure D-1**.

Table D-1. Computational burden comparison using Joseph covariance updates formula.

Operations	Additions	Operations	Multiplications
Batch	$3m^3+(19n-9)m^2+(36n^2-24n+1)m$	Batch	$3m^3+19nm^2+(36n^2+5n-1)m$
Sequential	$(36n^2-5n-5)m$	Sequential	$(36n^2+24n+2)m$

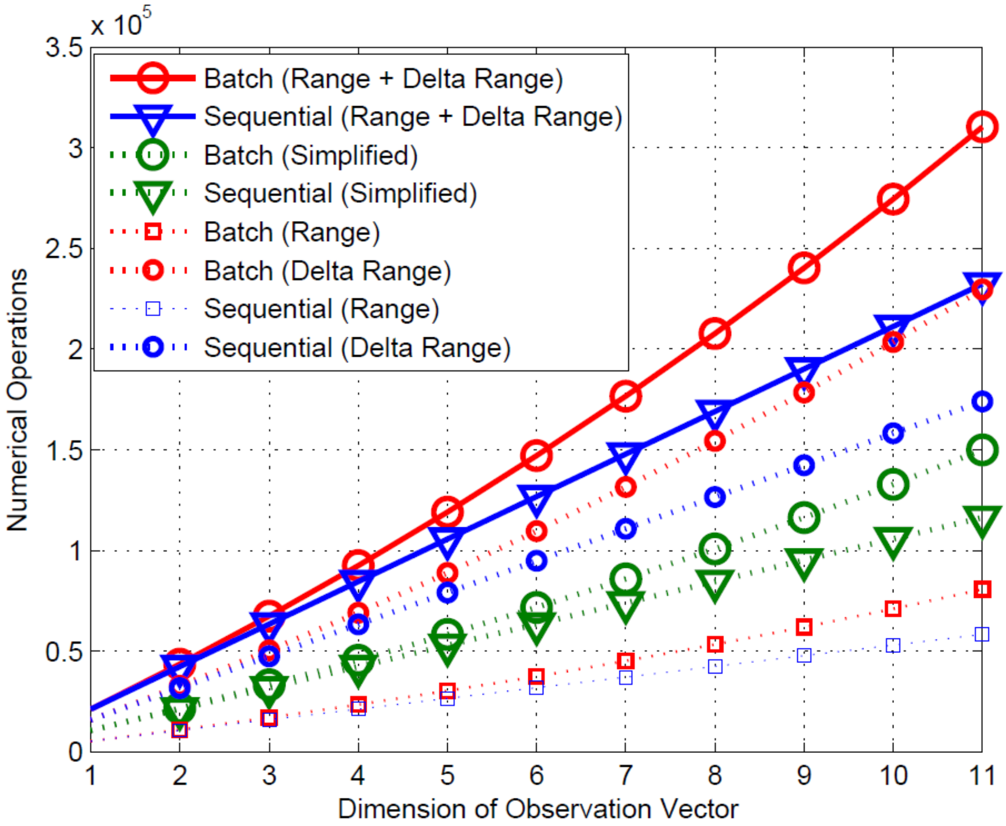


Figure D-1. Numerical operation comparisons using Joseph covariance update formulas.

The green curves are numerical operations involved in sequential and batch measurement updates using simplified covariance update formula, as given in **Figure 4-1** and **Figure 4-2**. The other curves are derived using the symmetric Joseph covariance update formula in the algorithm. The additional computational burden comes from the increased computational complexity in the update of the state error covariance matrices (i.e., Equation D-1 and Equation D-4) with respect to the formulas in Equations (4.15) and (4.19). Though Joseph formula introduces extra computational load, it guarantees the stability of mathematical computation.

Bibliography

[1] E. D. Kaplan and C. J. Hegarty, *Understanding GPS Principles and Applications*, 2nd ed.: Artech House, 2006.

[2] P. D. Groves, *Principles of GNSS, Inertial and multisensor integrated navigation systems*: Artech House Publishers 2008.

[3] N. El-sheimy and X. Niu. (2007) The Promise of MEMS to the Navigation Community. *Inside GNSS*.

[4] H. Carvalho, P. Del Moral, A. Monin, and G. Salut, "Optimal nonlinear filtering in GPS/INS integration," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 33, pp. 835 - 850, 1997.

[5] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing,* vol. 50, pp. 425 - 437, 2002.

[6] W. R. Hamilton, *Elements of Quaternions*. London, England: Longmans, green and Co., 1866.

[7] U. S. C. G. N. Center, "Global Positioning System Interface Control Document (ICD-GPS-200C including IRN-200c-001 to 004)," 2000.

[8] M. S. Grewal, L. R. Weill, and A. P. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*, 2nd ed. New Jersey: Wiley-Interscience, 2007.

[9] B. Hofmann-Wellenhof, K. Legat, and M. Wieser, *Navigation principles of positioning and guidance*. Wien: Springer-verlag, 2003.

[10] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System. Theory and Practice*. Wien: Springer-Verlag, 1997.

[11] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed.: Ganga-Jamuna Press, 2006.

[12] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*: American Institute of Aeronautics and Astronautics, 1997.

[13] J. A. Farrell, *Aided Navigation: GPS with High Rate Sensors*. New York: Mcgraw-Hill Professional, 2008.

[14] S. Hong, M. H. Lee, H.-H. Chun, S.-H. Kwon, and J. L. Speyer, "Observability of error States in GPS/INS integration," *IEEE Trans. on Vehicular Technology,* vol. 54, pp. 731 - 743, 2005.

[15] J. T. W. Ryley, P. M. G. Silson, A. Tsourdos, and S. Jordan, "Observability of a stationary GPS aided INS," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, 2010.

[16] I. Rhee, M. F. Abdel-Hafez, and J. L. Speyer, "Observability of an integrated GPS/INS during maneuvers," *IEEE Trans. on Aerospace and Electronic Systems,* vol. 40, pp. 526 - 535, 2004.

[17] S. Knedlik, "Introduction to satellite navigation, inertial navigation, and GNSS/INS integration," Habilitation, University of Siegen, Siegen, 2011.

[18] A. J. Haug, "A Tutorial on Bayesian Estimation and Tracking Techniques Applicable to Nonlinear and Non-Gaussian Processes," Technical report2005.

[19] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel Approach to Nonlinear and Non-Gaussian Baysian State Estimation," *IEE Radar and Signal Processing,* vol. 140, pp. 107 - 113, 1993.

[20] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. on Automatic Control,* vol. 45, pp. 477 - 482, 2000.

[21] M. S. Arulampalam, S. Maskell, G. N., and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Trans. on Signal Processing,* vol. 50, pp. 174 - 188, 2002.

[22] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*, 1 ed. New Jersey: John Wiley & Sons, 2006.

[23] S. J. Julier and J. K. Uhlmann, "A Consistent, Debiased Method for Converting Between Polar and Cartesian Coordinate Systems," in *AeroSense: Acquisition, Tracking and Pointing XI*, Orlando FL, 1997, pp. 110–121.

[24] S. J. Julier and J. K. Uhlmann, "Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations," in *American Control Conference*, Anchorage AK, 2002, pp. 887 - 892.

[25] S. J. Julier, "The scaled unscented transformation," in *American Control Conference*, Anchorage, Alaska, USA, 2002, pp. 4555 - 4559.

[26] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *IEEE Review,* vol. 92, pp. 401 - 422, 2004.

[27] R. van der Merwe and E. A. Wan, "Sigma-Point Kalman Filters for Integrated Navigation," in *The 60th Annual Meeting of The Institute of Navigation*, Dayton, OH, 2004, pp. 641-654.

[28] R. van der Merwe, A. Doucet, N. D. Freitas, and E. Wan, "The Unscented Particle Filter," Cambridge University Engineering Department, Technical Report2000.

[29] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing,* vol. 10, pp. 197-208, 2000.

[30] J. H. Kotecha and P. M. Djuric, "Gaussian particle filtering," *IEEE Trans. on Signal Processing,* vol. 51, pp. 2592 - 2601, 2003.

[31] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, *Sequential Monte Carlo Methods in Practice (Statistics for Engineering and Information Science)*: Springer-Verlag, 2001.

[32] A. Doucet, "On Sequential Simulation-Based Methods for Bayesian Filtering," Technical Report, Signal Processing Group, Department of Engineering, University of Cambridge, Cambridge, Technical Report1998.

[33] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Norwell, Massachusetts: Artech House, 2004.

[34] J. MacCormick and A. Blake, "A probabilistic Exclusion Principle for Tracking Multiple Objects," in *International Conference on Computer Vision*, Kerkyra , Greece, 1999, pp. 572-578.

[35] D. Crisan, P. D. Moral, and T. Lyons, "Non-linear Filtering Using Branching and Interacting Particle Systems," *Markov Processes and Related Fields,* vol. 5, pp. 293-319, 1999.

[36] K. Kanazawa, D. Koller, and S. Russell, "Stochastic simulation algorithms for dynamic probabilistic networks," in *The 11th annual conference on uncertainty in AI*, 1995, pp. 346-351.

[37] A. Smith and A. Gelfand, "Bayesian statistics without tears: A sampling-resampling perspective," *American Statistician,* vol. 46, pp. 84-88, 1992.

[38] J. Liu and R. Chen, "Sequential Monte Carlo Methods for Dynamic Systems," *Journal of the American Statistical Association,* vol. 93, pp. 1032-1044, 1998.

[39] N. Bergman, "Recursive Bayesian Estimation: navigation and Tracking Applications," Linköping University, Linköping, Dissertation1999.

[40] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Radar, Sonar & Navigation,* vol. 146, pp. 2-7, 1999.

[41] B. Ripley, *Stochastic Simulation*. New York: John Wiley & Sons, 1987.

[42] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics,* vol. 5, pp. 1-25, 1996.

[43] P. Closas, C. Fernandez-Prades, J. A. Fernandez-Rubio, and D. Bernal, "Particle Filtering Strategies for Efficient Multipath Mitigation," in *ION GNSS 2008*, Savannah, GA, 2008, pp. 644 - 651.

[44] R. G. Brown and P. Y. C. Hwang, *Introduction to Random signals and Applied Kalman Filtering*, 3 ed. New York: John Wiley & Sons, 1997.

[45] P. S. Maybeck, *Stochastic Models, Estimation and Control* vol. 1. New York: Academic Press Inc., 1979.

[46] D. Lerro and Y. Bar-Shalom, "Tracking with debiased consistent converted measurements versus EKF," *IEEE Trans. on Aerospace and Electronic Systems,* vol. 29, pp. 1015 - 1022, 1993.

[47] J. L. Crassidis, "Sigma-Point Kalman Filtering for Integrated GPS and Inertial Navigation," in *AIAA Guidance, Navigation, and Control Conference*, San Francisco, California, 2005.

[48] Y. Li, C. Rizos, J. L. Wang, P. Mumford, and W. D. Ding, "Sigma-Point Kalman Filtering for Tightly Coupled GPS/INS Integration," *NAVIGATION,* vol. 55, pp. 167 - 177, 2008.

[49] Y. Li, J. Wang, and C. Rizos, "Comparison of the Extended and Sigma-point Kalman Filters on Inertial Sensor Bias Estimation Through Tight Integration of GPS and INS," in *ION GNSS 2006*, Fort Worth, TX, 2006, pp. 1625-1634.

[50] Y. Li, J. Wang, C. Rizos, P. Mumford, and W. Ding, "Low-cost Tightly Coupled GPS/INS Integration Based on a Nonlinear Kalman Filtering Design," in *ION NTM 2006*, Monterey, CA, 2006, pp. 958-966.

[51] E.-H. Shin and N. El-Sheimy, "Unscented Kalman Filter and Attitude Errors of Low-cost Inertial Navigation Systems," *NAVIGATION,* vol. 54, pp. 1-9, 2007.

[52] E.-H. Shin, X. Niu, and N. El-Sheimy, "Performance Comparison of the Extended and the Unscented Kalman Filter for Integrated GPS and MEMS-Based Inertial Systems," in *ION NTM 2005*, San Diego, CA, 2005, pp. 961-969.

[53] J. Wendel, J. Metzger, R. Moenikes, A. Maier, and G. F. Trommer, "A performance comparision of tightly coupled GPS/INS Navigation systems based on extended and Sigma point Kalman filter," *NAVIGATION,* vol. 53, pp. 21-31, 2006.

[54] Y. Yi, "On improving the accuracy and reliability of GPS/INS-based direct sensor georeferencing," Ph.D Dissertation, Ohio State University, Geodetic Science and Surveying, 2007.

[55] Y. Yi and D. A. Grejner-Brzezinska, "Performance Comparison of the Nonlinear Bayesian Filters Supporting GPS/INS Integration," in *ION NTM 2006*, Monterey, CA, 2006, pp. 977 - 983.

[56] D.-J. Jwo and C.-N. Lai, "Unscented Kalman filter with nonlinear dynamic process modeling for GPS navigation," *GPS Solutions,* vol. 12, pp. 249-260, 2008.

[57] Y. Yang, J. Zhou, and O. Loffeld, "Quaternion-based Kalman filtering on INS/GPS," in *15th International Conference on Information Fusion (FUSION 2012)*, Singapore, 2012.

[58] J. Zhou, E. Edwan, S. Knedlik, and O. Loffeld, "Low-cost INS/GPS with nonlinear filtering methods," in *13th International Conference on Information Fusion (FUSION 2010)*, Edinburgh, 2010.

[59] J. Zhou, S. Knedlik, and O. Loffeld, "INS/GPS tightly-coupled integration using adaptive unscented particle filter," *Journal of Navigation,* vol. 63, pp. 491-513, 2010.

[60] J. Zhou, Y. Yang, J. Zhang, and E. Edwan, "Applying Quaternion-based Unscented Particle Filter on INS/GPS with Field Experiments," in *ION GNSS 2011 (Student paper award)*, Portland, Oregon, 2011.

[61] J. Zhou, Y. Yang, J. Zhang, E. Edwan, O. Loffeld, and S. Knedlik, "Tightly-coupled INS/GPS using Quaternion-based Unscented Kalman filter," in *AIAA Guidance, Navigation and Control 2011*, Portland, Oregon, 2011.

[62] H. L. Dyckman, S. Sloat, and B. Pettus, "Particle Filtering to Improve GPS/INS Integration," in *ION GNSS 2004*, Long Beach, CA, 2004, pp. 1619-1626.

[63] F. Caron, M. Davy, E. Duflos, and P. Vanheeghe, "Particle Filtering for Multisensor Data Fusion With Switching Observation Models: Application to Land Vehicle Positioning," *IEEE Trans. on Signal Processing,* vol. 55, pp. 2703 - 2719, 2007.

[64] A. Giremus, A. Doucety, A.-C. Escher, and J.-Y. Tourneret, "Nonlinear filtering approaches for INS/GPS Integration," in *12th European Signal Processing Conference*, Vienna, Austria, 2004, pp. 873-876.

[65] A. Giremus, J.-Y. Tourneret, and P. M. Djuric, "An improved regularized particle filter for GPS/INS integration," in *2005 IEEE 6th Workshop on Signal Processing Advances in Wireless Communications*, New York, NY, 2005, pp. 1013 - 1017.

[66] J. Georgy, U. Iqbal, M. Bayoumi, and A. Noureldin, "Reduced Inertial Sensor System (RISS)/GPS Integration using Particle Filtering for Land Vehicles," in *ION GNSS 2008*, Savannah, GA, 2008, pp. 30-37.

[67] Y. Yi and D. A. Grejner-Brzezinska, "Tightly-coupled GPS/INS Integration using Unscented Kalman filter and Particle filter," in *ION GNSS 2006*, Fort Worth, TX, 2006, pp. 2182-2191.

[68] Y. Yi and D. A. Grejner-Brzezinska, "Nonlinear Bayesian Filter: Alternative to the Extended Kalman Filter in the GPS/INS Fusion Systems," in *ION GNSS 2005*, Long Beach, CA, 2005, pp. 1391-1400.

[69] P. Aggarwal and N. El-Sheimy, "Hybrid Extended Particle Filter (HEPF) for INS/GPS Integrated System," in *ION GNSS 2009*, Savannah, GA, 2009, pp. 1858 - 1866.

[70] E.-H. Shin, "A Quaternion-Based Unscented Kalman Filter for the Integration of GPS and MEMS INS," in *ION GNSS 2004*, Long Beach, CA, 2004, pp. 1060-1068.

[71] W. Khoder, B. Fassinut-Mombot, and M. Benjelloun, "Quaternion Unscented Kalman Filtering for integrated Inertial Navigation and GPS," in *11th International Conference on Information Fusion, 2008* Cologne, Germany, 2008.

[72] E. J. Lefferts, F. L. Markley, and M. D. Schuster, "Kalman Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics,* vol. 5, pp. 417-429, 1982.

[73] S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte, "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle Applications," *IEEE Trans. on Robotics and Automation,* vol. 15, pp. 572-578, 1999.

[74] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology*, 2 ed.: The institution of Electrical Engineers, 2004.

[75] S. Knedlik, J. Zhou, and O. Loffeld, "On position and attitude estimation for remote sensing with bistatic SAR," in *Geoscience and remote sensing*, P.-G. P. Ho, Ed., ed Vukovar: In-Tech, 2009, pp. 75-90.

[76] J. Zhou, S. Knedlik, and O. Loffeld, "Sequential Processing of Integrated Measurements in Tightly-coupled INS/GPS Integrated Navigation System," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario Canada, 2010.

[77] J. Zhou, S. Knedlik, and O. Loffeld, "INS/GPS for High-Dynamic UAV-based Applications," *International Journal of Navigation and Observation,* vol. 2012, p. 11, 2012.

[78] J. Zhang, E. Edwan, J. Zhou, and O. Loffeld, "Development and Investigation of Real-time Hybrid Navigation System Using a DCM Based Integration Method," in *ION GNSS 2010*, Portland, Oregon, 2010, pp. 2986 - 2994.

[79] J. Zhang, S. Knedlik, and O. Loffeld, "Performance Investigation of Real-time MEMS-IMU/GNSS Integrated System," in *ION GNSS 2009*, Savannah, GA, 2009, pp. 978-986.

[80] C. Hide, T. Moore, and M. Smith, "Adaptive Kalman Filtering for Low-cost INS/GPS," *Journal of Navigation,* vol. 56, pp. 143-152, 27 Jan 2003 2003.

[81] A. H. Mohamed and K. P. Schwarz, "Adaptive Kalman Filtering for INS/GPS," *Journal of Geodesy,* vol. 73, pp. 193-203, 1999.

[82] R. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Transactions on Automatic Control,* vol. 15, pp. 175 - 184, 1970.

[83] R. Mehra, "Approaches to adaptive filtering," *IEEE Trans. on Automatic Control,* vol. 17, pp. 693 - 698, 1972

[84] C. Hide, T. Moore, and M. Smith, "Adaptive Kalman filtering algorithms for integrating GPS and low cost INS," in *Position Location and Navigation Symposium*, Monterey, USA, 2004, pp. 227 - 233.

[85] D. F. Hartman, "Sequential Processing of Integrated Measurements in a Generalized Kalman Filter," in *ION NTM 2002*, San Diego, CA, 2002, pp. 632-638.

[86] D. F. Hartman and D. B. Tyler, "New Kalman Filter Formulation for GPS Delta Pseudo Range Processing," in *ION GPS 1998*, Nashville, TN, 1998, pp. 1395-1400.

[87] G. Strang and K. Borre, *Linear Algebra, Geodesy, and GPS*: Wellesley College, 1997.

[88] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*: Dover Publications, 2006.