Master Thesis

Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Radioelectronics

Robust Navigation Solution using Extended Kalman Filter

Bc. Michal Šimek

Supervisor: Assoc. prof. Ing. Jan Roháč, Ph.D
May 2020

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Šimek Michal**	Personal ID number:	**457249**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Radioelectronics**		
Study program:	**Open Electronic Systems**		
Branch of study:	**Communications and Signal Processing**		

II. Master’s thesis details

Master’s thesis title in English:

Robust Navigation Solution using Extended Kalman Filter

Master’s thesis title in Czech:

Robustní navigační řešení s využitím rozšířeného Kalmanova filtru

Guidelines:

Design and develop a navigation algorithm using data from a single/dual-antenna GNSS receiver, inertial measurement unit, absolute pressure sensor, magnetometer, and ultrasonic sensor as inputs in order to estimate the position, speed and orientation of a navigated object (NO) in 3D. The solution should be based on data fusion realized using an extended Kalman filter and an error model. It should also allow evaluation and compensation of the individual measuring parts’ location from the NO center of gravity and deterministic errors (alignment, sensor sensitivity, traffic delays, etc.) as well as taking into account the NO dynamic characteristics. Analyze the navigation solution in terms of the nature of movement and environment, and their effects on accuracy. Verify the algorithm in Matlab on experimental data and determine its accuracy. Convert the algorithm into C/C++ and implement it into a microcontroller. Verify a final solution by comparing the results.

Bibliography / sources:

[1] Soták M., Sopata M., Bréda R, Roháč J., Váci L. (ed.): Integrácia navigačných systémov. 1. vyd. Košice: Bréda Róbert, 2006. 344 s. ISBN 80-969619-9-3.
[2] S. Grewal / P. Andrews: Kalman Filtering – Theory and Practice using Matlab ® 3rd edition 2008, Wiley.
[3] J. A. Farrell / M. Barth: The global positioning system & inertial navigation 2nd edition 1999, Mc Graw Hill.

Name and workplace of master’s thesis supervisor:

doc. Ing. Jan Roháč, Ph.D., Department of Measurement, FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **10.02.2020** Deadline for master's thesis submission: **14.08.2020**

Assignment valid until: **30.09.2021**

doc. Ing. Jan Roháč, Ph.D.	doc. Ing. Josef Dobeš, CSc.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

.

Date of assignment receipt Student’s signature

Acknowledgements

I would like to thank my supervisor Assoc. prof. Ing. Jan Roháč, Ph.D., for helping me and giving me the opportunity to explore the basic principals of inertial navigation systems. I would also like to thank Ing. Martin Šipoš, Ph.D., for his help and time in laboratory with testing the navigation unit. Last but not least, I would like to thank my friends and family for their support.

Declaration

I declare that I have prepared the submitted work independently and that I have listed all sources used in accordance with the Methodological Instruction on Compliance ethical principles in the preparation of university final theses.

Prague, May 16, 2020

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 16. května 2020

Abstract

This diploma thesis deals with designing a navigation solution based on error modeling and Kalman filtering. The aim is to describe and develop an algorithm to estimate position, velocity and orientation of a vehicle taking data from inertial measurement unit (IMU) consisting of inertial sensors, GNSS receiver, magnetometer, absolute pressure sensor, and ultrasonic sensor. The design of a dynamic detector is discussed and examined for the purposes of Kalman filtering. The thesis further describes implementation and simulation of the navigation algorithm on experimental data to analyze its accuracy and confirm results. The navigation solution using IMU measurements and GNSS data is implemented to a navigation unit and tested in a real environment.

Keywords: Inertial navigation, Error state Kalman filter, MEMS sensors, Dynamic detection

Supervisor: Assoc. prof. Ing. Jan Roháč, Ph.D

Abstrakt

Tato diplomová práce se zabývá návrhem navigačního řešení, které je založeno na odchylkovém modelu a Kalmanově filtraci. Hlavním cílem je vyvinout algoritmus využívající data ze senzorů inericální jednotky, GNSS příjimače, magnetometru, absolutního tlakového senzoru a ultrazvukového senzoru který estimuje pozici, rychlost a orientaci navigovaného prostředku. Je diskutován a vyzkoušen detektoru dynamiky pro účely použití v Kalmanově filtraci. Součástí práce je implementace a simulace navrženého navigačního algoritmu na experimentálních datech pro ověření funčnosti a přesnosti algoritmu. Navigační řešení využívající GNSS data a IMU měření je pak implementováno do navigační jednotky a otestováno v reálném prostředí.

Klíčová slova: Inerciální navigace , odchylkový Kalmánův filter, MEMS senzory, Detekce dynamiky

Překlad názvu: Robustní navigační řešení s využitím rozšířeného Kalmanova filtru

Contents

Figures

Tables

Chapter 1
Introduction

Navigation systems have been evolving for centuries and have become an indispensable part of many applications. The need for position and direction determination is crucial in a broad area of human activity. The applications of interest include marine navigation, aviation, army systems or the space industry, to name a few. One of the oldest methods is known to be celestial navigation that makes use of space objects in the sky. Modern techniques exploit knowledge of science and physics and the development of electronics. Today, Global Navigation Satellite Systems (GNSS) and radar systems have become very popular due to its availability and accuracy.
On the other hand, to ensure high accuracy, the GNSS receiver requires a clear line of sight with the satellites. Besides, signal dropouts can lead to loss of position information, which can be fatal or unwanted in many applications. This is one of the reasons why inertial systems are commonly deployed, for instance, in the aviation industry. The self-sufficiency of the inertial navigation system, however, depends on the utilized sensors. Technological progress has improved the availability of the inertial navigation systems due to cost-effective sensors using MEMS (Micro-Electro-Mechanical Systems). Usage of cost-effective sensors, on the other hand, can suffer from worse noise characteristics and, thus, the development of a robust navigation system has become more challenging [1]. Different aiding sensors are deployed to improve navigation. Typically, the estimation can further be improved by making use of magnetometers or compasses measuring Earth's magnetic field or exploiting sonar or radar systems [2].

The thesis is divided into eight chapters. The introduction represents the first chapter. Chapter 2 describes the necessary knowledge and theoretical principles of navigation. It also gives a brief description of the sensors used in navigation and a Kalman filter algorithm. In the third chapter, standard reference frames and the transformations between them are presented in order to get acquainted with basic equations. Moreover, it derives mechanization equations for inertial navigation systems using the Inertial Meaurement Unit (IMU). Chapter 4 expands the navigation solution by including the aiding sensors and the state-space model's design and measurement model. It also introduces the initialization process of the algorithm, such as the alignment of the reference axis. Chapter 5 is devoted to the design of a dynamic detector. The sixth chapter verifies the algorithm on experimental data in Matlab. Finally, the last chapter is dedicated to real-time testing. An experiment is carried out and the obtained results are shown and compared to the simulations.

Chapter 2
Principles of Navigation

This chapter provides a basic description and explanation of systems and algorithms typically applied in navigation. Particular attention is paid to Inertial Navigation Systems (INS). These systems use the IMU unit made up of inertial sensors. The inertial sensors consist of three orthogonal accelerometers and three orthogonal angular rates sensors to estimate the position, velocity and attitude of the navigated object. It is necessary to define initial conditions in terms of starting position and velocity. The computational complexity and accuracy of the navigation solution are influenced by many factors depending on the application, sensors and data fusion algorithms used. To develop long-term and high-accurate INS, the usage of expensive navigation grade sensors is necessary. Such INSs are intended for applications where no other navigation system is available. To give an example, GNSS signals cannot penetrate water to a depth suitable for submarines [3]. Therefore, submarine navigation is dependent on INS systems. By adding additional sensors to inertial systems, the position and attitude determination may be further improved. If the sensors are rigidly attached to the vehicle, it is referred to as *strapdown systems*. Strapdown systems tend to be smaller and mechanically less complex than stabilized platform systems, which were used more often in the past [4].

As it was mentioned earlier, the MEMS sensors can suffer from higher noise components. Hence, sensor calibration of IMU unit is required and it is described in many articles including [5], [6] or [7]. To describe and determine the sensor's noise parameters, the Allan variance analysis and sensor modeling are introduced.

The basic principle of GNSS systems is also presented. Inertial navigation may use the GNSS receivers to determine the initial conditions and to perform corrections.

Since the inertial data errors are integrated, it can lead to uncontrolled error output. Therefore, it is necessary to compensate for the errors with another source of data such as GNSS receivers or an absolute pressure sensor. Data fusion algorithms, such as Extended Kalman filter, Unscented Kalman filter or Particle filters, combine data from different sensors providing the same information to improve the estimation. More information can be found in [8] or [9]. In terms of simplicity and computational cost, the Kalman filter has

proven its efficiency and it is often used in navigation systems.
Finally, algorithms used for dynamic detection are presented.

2.1 Sensors

This section provides a brief introduction to sensors and sensor modeling. The accuracy of the inertial navigation systems depends on the quality of inertial sensors. Different aiding sensors are often integrated to improve the position, velocity and attitude estimation of the navigated object. Such sensors include magnetometers to correct the heading estimation, absolute air pressure sensors or ultrasonic sensors to provide another source of altitude measurements.

2.1.1 Inertial sensors

Accelerometers

An accelerometer is a device that measures specific force or acceleration. Accelerometers can be divided into various groups depending on the designed mechanism such as [4]

- Mechanical
- Solid state
- MEMS

Mechanical accelerometers contain a mass suspended by springs [4]. To determine the acceleration, a displacement of the mass is measured, leading to a signal proportional to the acceleration [4]. Solid-state accelerometers are divided into other sub-groups such as vibration, silicone or acoustic wave accelerometers [4]. Finally, the accelerometers can also be constructed in MEMS technology, making use of the same principles as mechanical or solid-state devices. Very frequently, they consist of a seismic mass and the movement of the mass is measured by the change of capacity as depicted in Figure 2.1.

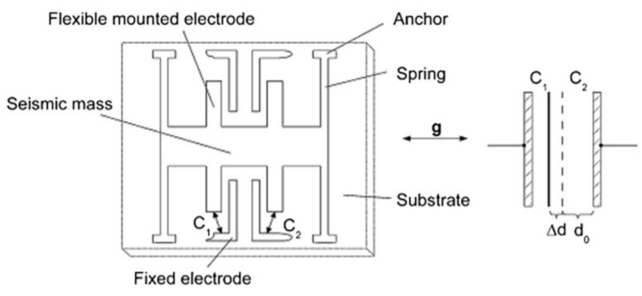


Figure 2.1: Capacitive accelerometers [10]

Gyroscopes

Sensors measuring angular rate are called gyroscopes. Various techniques and types of gyroscopes apply different mechanisms to obtain angular velocity such as [4]

- Mechanical
- Optical
- MEMS

Optical gyroscopes include Fibre Optics Gyro (FOG) and Ring Laser Gyro (RLG). RLG gyroscopes tend to be the most accurate but also the most expensive. This is in contrast with MEMS gyroscopes that are relatively cheap to manufacture [4]. To measure the angular rate, MEMS gyroscopes consist of a vibrating element and they exploit the Coriolis effect which can be written mathematically as follows

$$\boldsymbol{F} = -2m(\boldsymbol{\omega} \times \boldsymbol{v}). \tag{2.1}$$

If an external angular velocity $\boldsymbol{\omega}$ in the perpendicular direction begins to act on the vibrating element of a moving mass m, force $\boldsymbol{F}$ will appear. The force then induces a deflection that can be measured and transformed to obtain the angular velocity as shown in Figure 2.2 [11]. MEMS gyroscopes are relatively cheap, but they can suffer from higher noise and lower precision. However, the progress in technology has made the MEMS sensors sufficient for many applications and the popularity of MEMS sensors increases due to the small size, low weight and low power consumption [4].

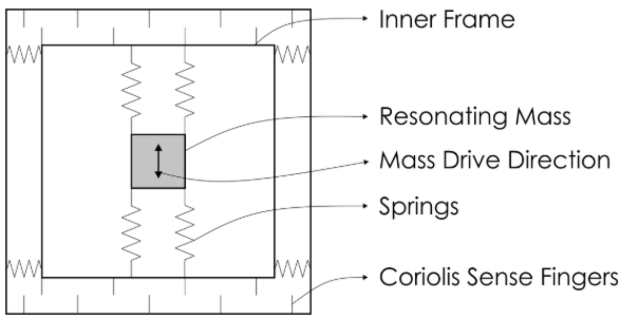


Figure 2.2: MEMS gyroscope using Coriolis effect [12]

2.1.2 Allan Variance

Allan Variance analysis (AVAR) is a known time-domain method to identify noise types for modeling purposes in inertial sensors. The method was invented by David Allan. Originally, he wanted to study the frequency stability of oscillators. The AVAR analysis was later extended and applied to inertial sensors. Five basic noises have been introduced [17]

- Quantization noise
- Angle/Velocity random walk
- Bias instability
- Rate random walk
- Drift rate ramp

The angle random walk is modeled as white noise in a rate domain. Time τ_0 when the bias instability occurs is an important parameter for INS application. It determines the maximal time when an estimated bias can be considered valid and when the noise in the sensors can be modeled as white.
AVAR works with a variance of averaged data in a cluster of a defined extending interval τ . The total number of clusters in a data set used is calculated as follows

$$M = \left\lfloor \frac{N}{m} \right\rfloor, \tag{2.2}$$

where $\lfloor . \rfloor$ denotes floor function, N denotes the size of the dataset and m is the number of samples in the cluster.

Based on the cluster shifts in the dataset, several variants of AVAR can be defined [18] . The standard non-overlapped AVAR is frequently used and it is defined as

$$\sigma_y^2(\tau) = \frac{1}{2(M-1)} \sum_{i=1}^{M-1} (\overline{y}_{i+1} - \overline{y}_y)^2, \tag{2.3}$$

where $\overline{y}_{i+1}$ and $\overline{y}_{i+1}$ are mean values of clusters i and $i+1$.

Figure 2.3 shows the typical plot of Allan variance. More information can be found in [19].

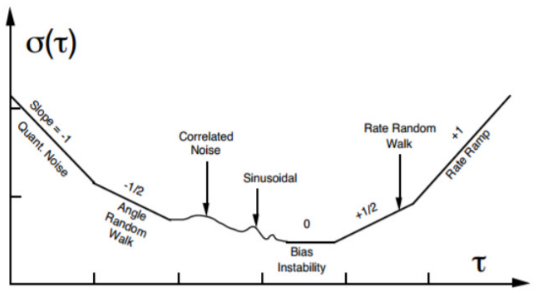


Figure 2.3: Typical Allan variance plot [19]

2.1.3 Aiding sensors

Magnetometers

Magnetometers are sensors measuring the presence of a magnetic field. They are often added to inertial systems as aiding sensors to provide information

about the vehicle's heading. Magnetometers allow to determine the location of the magnetic north [4]. However, the magnetic north differs from the true north needed for yaw estimation. An angle between the true north and the magnetic north is known as the declination δ. Since the Earth's magnetic field lines are not parallel to the Earth's surface, there is another error in magnetic measurement occurs. The angle between the horizontal plane and the Earth's magnetic field lines is called magnetic inclination ζ.
There are two main technologies for measuring the magnetic field. Fluxgate magnetometers can achieve high accuracy and the range is sufficient for measuring the Earth's magnetic field. AMR magnetometers use a different method to measure the magnetic field. A material changing its resistance due to a magnetic field's presence is deployed in the sensor, allowing to measure the strength of the magnetic field [20]. Magnetometers can be affected by local disturbances, such as the presence of ferromagnetic material, which can lead to low accuracy. Thus, compensation techniques have to be present to eliminate unwanted effects.

- Hard iron effect

  Permanent magnets and magnetic hysteresis cause the 3-axis magnetometer output to be shifted by a constant bias $\boldsymbol{b}_m$.

- Soft iron effect

  Ferromagnetic materials induce magnetism when an external field is present. Soft iron effect is more diffucult to compansate since it changes the intesity as well as the direction of the sensed field [13]. To eliminate the soft iron effect, a transformation matrix $\mathbf{A}$ needs to be estimated.

The compensated magnetic vector $\tilde{\boldsymbol{m}}$ is then given by [13]

$$\tilde{\boldsymbol{m}} = \mathbf{A}\boldsymbol{m} + \boldsymbol{b}_m, \tag{2.4}$$

where $\boldsymbol{m}$ is the vector of magnetic 3D components measured.

If the magnetometer is calibrated, a full 360 heading rotation should result in a circle output centered at the origin.

Absolute Air Pressure Sensors

Another aiding sensor commonly used in inertial navigation is an absolute air pressure sensor to improve estimation of altitude. The target pressure is measured relative to a known pressure of the absolute vacuum. The measurements are dependent on temperature. Various methods of measuring atmospheric pressure can be deployed. Widespread technology is a piezoresistive strain gauge. The piezoresistive strain gauge method makes use of the piezoresistive effect of bonded or formed strain gauges. The applied pressure deforms

the material and increases or decreases the resistance [14]. The change of resistance is then converted to a voltage.
The pressure measurements can be converted to estimate height above the ground as follows [15]

$$h_p = 44330 \left[1 - \left(\frac{p_h}{p_0} \right)^{c_1} \right], \tag{2.5}$$

where p_h is the measured pressure, $c_1 = \frac{1}{5.2559}$ and p_0 is the standard atmospheric pressure.

Due to the atmospheric conditions, the reference pressure p_0 can vary. If the altitude is known at the beginning of the measurements, it is convenient to calculate p_0. From the equation (2.5), the p_0 can be expressed as

$$p_0 = \frac{p_h}{(1 - (\frac{1}{44330} h)^{5.2559}}. \tag{2.6}$$

Ultrasonic Sensor

The ultrasonic sensors have found usage in many applications. They are used to detect objects or measure distances. In indoor navigation, the ultrasonic sensors are deployed to provide another source of relative positioning information.
The devices typically consist of a transmitter and a receiver. The transmitter emits high-frequency sound pulses at certain time intervals. The pulses propagate at the velocity of sound. If an obstacle gets in the way, the pulse is reflected back to the receiver. The sensor measures the time difference between the transmitted pulse and the received one and calculates the distance from the obstacle. The typical scheme of an ultrasonic sensor is depicted in Figure 2.4.

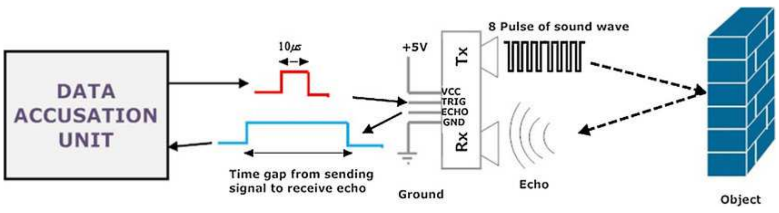


Figure 2.4: Ultrasonic sensor scheme [16]

2.1.4 Sensor Modeling

In an ideal case, the sensor's output would give the exact and precise value of the measured quantity. However, due to manufactural imperfections, thermal vibration and other factors, the measured values are noisy. Therefore, a need for sensor modeling arises in order to estimate the real value and minimize the stochastic component. The typical sensor model is given by

$$y = y_r + b_y + u_y, \tag{2.7}$$

where y is the measured value, y_r is the true value , b_y is a sensor bias and u_y is zero mean Gaussian noise .

From the error modeling point of view, the bias error $\dot{b}_y$ is typically described by first-order Gauss-Markov process, also known as Gaussian stacionary process. Let $\boldsymbol{w}(t)$ be a Gaussian random process. Then, equation (2.8) represents a Gaussian-Markov process [29].

$$\dot{\boldsymbol{x}}(t) = \mathbf{F}(t)\boldsymbol{x}(t) + \mathbf{G}(t)\boldsymbol{w}(t) \tag{2.8}$$

In a special case of error modeling, the Gauss-Markov process reduces to simpler equation as follows [11]

$$\dot{b}_y = -\frac{1}{\tau_y} b_y + w_y, \tag{2.9}$$

where τ_y is a time constant of the sensor and w_y is zero mean Gaussian noise.

Advanced models may also take into account the scale factors and misalignment of the 3-axis sensors.

2.2 Global Navigation Satellite System

GNSS stands for Global Navigation Satellite Systems and it refers to a constellation of satellites transmitting signals to a GNSS receiver to determine position and velocity. There are currently four major GNSS systems, namely, GPS, GLONASS, GALILEO and BeiDou. GPS was developed by the United States Department of Defense, Russians developed GLONASS, GALILEO is a project of the European Union and China invented BeiDou. GNSS architecture consists of three segments [21].

- Space segment
- Control segment
- User segment

The satellite constellation forms the space segment. It provides carrier phase signals and broadcast the navigation message. The Control segment maintains and monitors the system status and updates navigation message and the user segment is formed by GNSS receivers providing information to the user [21]. The positioning systems exploit the idea of triangulation. In general, however, at least four satellites are required to determine the position of a receiver on Earth and to set the time offset on the receiver clock. The satellites transmit signals containing navigation message in which the information of

the satellite position and a timestamp is included [22]. The GNSS receiver calculates the propagation time from the satellite to the receiver antenna. Based on the knowledge of the satellite's position and the propagation time, the receiver position can be determined. Several factors can negatively affect the calculations, such as signal propagation delay caused by atmospheric layers or multipath propagation [22]. Even small timing errors can imply significant position errors. Several methods have been introduced such as differential GPS or Real-Time Kinematics (RTK) to make the position measurements more precise [21].
The GNSS systems are also able to determine the speed of the GNSS receiver. There are three techniques that can be implemented.

1. The first method is trivial and easy to calculate, however, it suffers from high inaccuracy. It is based on the traveled distance of the GNSS receiver δs and the time t it took to travel the distance. The speed is calculated as follows [23]
$$v = \frac{\delta s}{t}. \tag{2.10}$$

2. Second approach makes use of Doppler frequency shift. Due to the relative motion of the GNSS satellite and the GNSS receiver, frequency difference will occur. To determine the speed, at least, four sattelites are required. Signal multipath or signal delays in atmosphere can effect the computation, nevertheless, the Doppler shift method is known to be more accurate than the method based on distance difference [23].

3. The last approach exploits time difference of successive carrier phases to the same satellite [23]. It is called time-difference carrier phase method and it provides the best estimation of the navigated object speed.

2.3 Kalman Filter

The Kalman filter is a recursive filter that estimates unknown state variables based on measured data and a dynamic state-space model. It is necessary to design a dynamic mathematical model that is based on physical laws. The time-invariant dynamic system in continuous time with an additive noise model is defined as follows [29]

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{w}(t), \tag{2.11}$$
$$\boldsymbol{y}(t) = h(\boldsymbol{x}(t)) + \boldsymbol{v}(t), \tag{2.12}$$

where
$\boldsymbol{x}(t)$ is the state of the system,
$\boldsymbol{u}(t)$ stands for the input signal,
$\boldsymbol{y}(t)$ is the output signal
$\boldsymbol{w}(t)$ and $\boldsymbol{v}(t)$ are noise components and
$f(\cdot)$ and $h(\cdot)$ are non-linear state space function and measurement function,

respectively.

If the system is linear, the state space model can be expressed as follows [29]

$$\dot{\boldsymbol{x}}(t) = \mathbf{F}\boldsymbol{x}(t) + \mathbf{G}\boldsymbol{u}(t) + \boldsymbol{w}(t), \tag{2.13}$$
$$\boldsymbol{y}(t) = \mathbf{H}\boldsymbol{x}(t) + \boldsymbol{v}(t), \tag{2.14}$$

where $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{H}$ are system matrices.

Similar equations can be written for discrete-time domain as

$$\boldsymbol{x}_n = \boldsymbol{\Phi}\boldsymbol{x}_{n-1} + \boldsymbol{\Gamma}\boldsymbol{u}_{n-1} + \boldsymbol{w}_n, \tag{2.15}$$
$$\boldsymbol{y}_n = \mathbf{H}\boldsymbol{x}_{n-1} + \boldsymbol{v}_n, \tag{2.16}$$

where, again, $\boldsymbol{\Phi}$, $\boldsymbol{\Gamma}$ and $\mathbf{H}$ are system matrices.

In case of Kalman filtering, there are also a few prerequisites related to noise modeling that need to be assumed. Both noise are supposed to be white Gaussian with zero mean

$$\boldsymbol{w}_n \sim N(0, \mathbf{Q}_n), \tag{2.17}$$
$$\boldsymbol{v}_n \sim N(0, \mathbf{R}_n), \tag{2.18}$$

where $\mathbf{Q}_n$ and $\mathbf{R}_n$ are covariance matrices of process noise and measurement noise and it holds [24]

$$\mathrm{E}[\boldsymbol{w}_n\boldsymbol{w}_k^T] = \delta_{nk}\mathbf{Q}_n, \tag{2.19}$$
$$\mathrm{E}[\boldsymbol{v}_n\boldsymbol{v}_k^T] = \delta_{nk}\mathbf{R}_n, \tag{2.20}$$
$$\mathrm{E}[\boldsymbol{w}_n\boldsymbol{v}_n^T] = 0, \tag{2.21}$$

where δ_{nk} is the dirac delta function that is equal to 1 only if $n = k$, otherwise it is 0.

The equation (2.21) indicates that there is no correlation between measurement and process noise. Given the linear discrete model described in eqs. (2.15)-(2.16), the Kalman filter consists of two steps. The first step is called the time update and it predicts the state vector $\hat{\boldsymbol{x}}_{n,n-1}$ and the covariance matrix $\hat{\mathbf{P}}_{n,n-1}$ based on previous filter output $\hat{\boldsymbol{x}}_{n-1,n-1}$ and $\hat{\mathbf{P}}_{n-1,n-1}$. To clarify the marking of the variables, the nominal (true) state of the system is depicted as $\boldsymbol{x}_n$ and the estimated state by Kalman filter is denoted as $\hat{\boldsymbol{x}}_{n,n}$. In addition, to distinguish the estimated state in both steps, there are two sub-indexes. The first one determines the update in the prediction step and the second one refers to the update in the second step of Kalman filter, which is called the measurement update. Considering $\boldsymbol{\Gamma} = \mathbf{0}$ the time update of the state vector is computed as follows [26]

$$\hat{\boldsymbol{x}}_{n,n-1} = \Phi\hat{\boldsymbol{x}}_{n-1,n-1} \tag{2.22}$$

The covariance matrix $\hat{\mathbf{P}}_{n,n-1}$ in the time update is given by [26]

$$\begin{aligned}\hat{\mathbf{P}}_{n,n-1} &= \mathrm{E}[(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_{n,n-1})(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_{n,n-1})^T] \\ &= \mathbf{\Phi}\mathrm{E}[(\boldsymbol{x}_{n-1} - \hat{\boldsymbol{x}}_{n-1,n-1})(\boldsymbol{x}_{n-1} - \hat{\boldsymbol{x}}_{n-1,n-1})^T]\mathbf{\Phi} + \mathrm{E}[\boldsymbol{w}_{n-1}\boldsymbol{w}_{n-1}^T] \\ &= \mathbf{\Phi}\hat{\mathbf{P}}_{n-1,n-1}\mathbf{\Phi}^T + \mathbf{Q}_n. \end{aligned} \tag{2.23}$$

Next, when the measurements $\boldsymbol{z}_n$ are obtained, the measurement update step is performed. It is divided into three sub-steps. First of all, matrix known as Kalman gain $\mathbf{K}_n$ is computed. It basically gives optimal weights for each measurements [25]. Then, the state vector is updated and finally, new covariance matrix is calculated. The measurement update step is summerized in eqs (2.24)-(2.26).

$$\mathbf{K}_n = \hat{\mathbf{P}}_{n,n-1}\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}_{n,n-1}\mathbf{H}^T + \mathbf{R}_n)^{-1} \tag{2.24}$$

$$\hat{\boldsymbol{x}}_{n,n} = \hat{\boldsymbol{x}}_{n,n-1} + \mathbf{K}_n(\boldsymbol{z}_n - \mathbf{H}\hat{\boldsymbol{x}}_{n,n-1}) \tag{2.25}$$

$$\hat{\mathbf{P}}_{n,n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H})\hat{\mathbf{P}}_{n,n-1} \tag{2.26}$$

The algorithm is repeated all over again. One of the significant advantages is that the algorithm is easily applied in real-time systems, making it very powerful. It is essential to ensure that the positive-semidefinite property and symmetry of the covariance matrix $\mathbf{P}$ are fulfilled. The development of the covariance matrix $\mathbf{P}$ indicates the functionality of the algorithm.
Due to the implementation, the model has to be transformed into a discrete-time equivalent model to get a matrix $\mathbf{\Phi}_n$. Under various assumptions such that the $\mathbf{F}$ matrix is constant over a small period t the matrix $\mathbf{\Phi}_n$ can be computed as [29]

$$\mathbf{\Phi} = \mathrm{e}^{\mathbf{F}t} \approx \mathbf{I} + \mathbf{F}t + \frac{1}{2}\mathbf{F}t^2. \tag{2.27}$$

The same applies for matrix $\mathbf{Q}$ that is usually aproximated by matrix $\mathbf{Q_d}$ [29]

$$\mathbf{Q_d} = \frac{1}{2}t(\mathbf{\Phi}\mathbf{G}\mathbf{Q}\mathbf{G}^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T\mathbf{\Phi}^T). \tag{2.28}$$

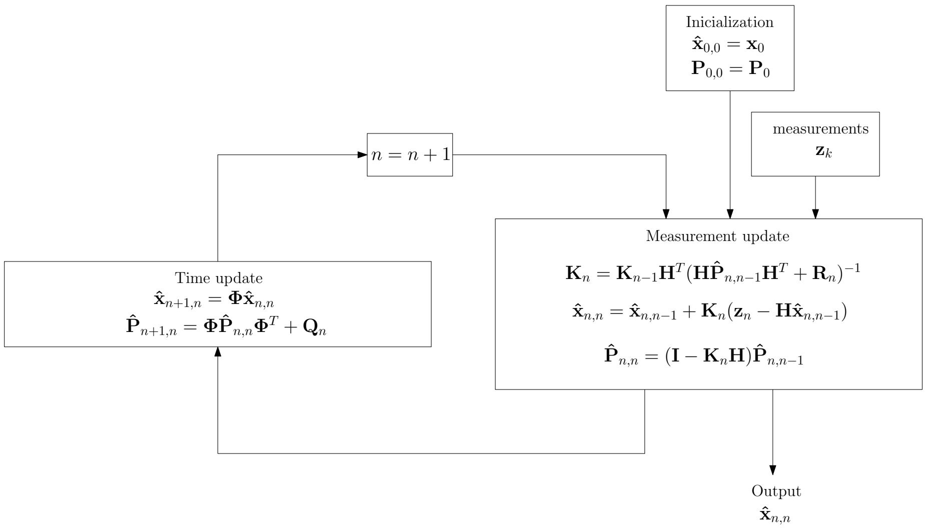
The whole Kalman filtering algorithm is depicted in Figure 2.5 .

2.3.1 Error State Kalman Filter

Equations describing dynamic systems can be non-linear. The Error State Kalman Filter takes the non-linear function and linearizes it around an estimate of the previous state. A time-variant non-linear system with white noise $\boldsymbol{u}(t)$ and output signal $\boldsymbol{y}(t)$ is given by

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{2.29}$$

$$\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t)). \tag{2.30}$$



Inicialization
$\hat{\mathbf{x}}_{0,0} = \mathbf{x}_0$
$\mathbf{P}_{0,0} = \mathbf{P}_0$
measurements
$\mathbf{z}_k$
$n = n+1$
Measurement update
$\mathbf{K}_n = \mathbf{K}_{n-1}\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}_{n,n-1}\mathbf{H}^T + \mathbf{R}_n)^{-1}$
$\hat{\mathbf{x}}_{n,n} = \hat{\mathbf{x}}_{n,n-1} + \mathbf{K}_n(\mathbf{z}_n - \mathbf{H}\hat{\mathbf{x}}_{n,n-1})$
$\hat{\mathbf{P}}_{n,n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H})\hat{\mathbf{P}}_{n,n-1}$
Time update
$\hat{\mathbf{x}}_{n+1,n} = \mathbf{\Phi}\hat{\mathbf{x}}_{n,n}$
$\hat{\mathbf{P}}_{n+1,n} = \mathbf{\Phi}\hat{\mathbf{P}}_{n,n}\mathbf{\Phi}^T + \mathbf{Q}_n$
Output
$\hat{\mathbf{x}}_{n,n}$

Figure 2.5: KF algorithm

Let $\delta \boldsymbol{x}$ be an error vector defined as a deviation of real state vector $\boldsymbol{x}$ and estimated state vector $\hat{\boldsymbol{x}}$ and $\delta \boldsymbol{y}$ is a deviation of real output vector $\boldsymbol{y}$ and estimated output vector $\hat{\boldsymbol{y}}$

$$\begin{aligned} \delta \boldsymbol{x} &= \boldsymbol{x} - \hat{\boldsymbol{x}}, && (2.31)\\ \delta \boldsymbol{y} &= \boldsymbol{y} - \hat{\boldsymbol{y}}. && (2.32) \end{aligned}$$

The functions $\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))$ and $\boldsymbol{h}(\boldsymbol{x}(t))$ can be approximated and linearized by using Taylor series expanssion around the estimated state $\hat{\boldsymbol{x}}$ and a given input $\hat{\boldsymbol{u}}$ and taking only the first derivatives as follows [11]

$$\begin{aligned} \dot{\boldsymbol{x}} &\approx \boldsymbol{f}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}},\hat{\boldsymbol{u}}} \delta \boldsymbol{x} + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{u}}\right|_{\hat{\boldsymbol{x}},\hat{\boldsymbol{u}}} \delta \boldsymbol{u} && (2.33)\\ &\approx \boldsymbol{f}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) + \mathbf{F}(t)\delta \boldsymbol{x} + \mathbf{G}(t)\delta \boldsymbol{u}, && (2.34)\\ \boldsymbol{y}(t) &\approx \boldsymbol{h}(\hat{\boldsymbol{x}}) + \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}(t))}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}} \delta \boldsymbol{x} && (2.35)\\ &\approx \boldsymbol{h}(\hat{\boldsymbol{x}}) + \mathbf{H}(t)\delta \boldsymbol{x} && (2.36) \end{aligned}$$

where $\mathbf{F}(t)$, $\mathbf{G}(t)$ and $\mathbf{H}(t)$ are Jacobians matrices of vector function $\boldsymbol{f}$ and $\boldsymbol{h}$ respectively and are defined as follows

$$\mathbf{F}(t) = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix} \tag{2.37}$$

$$\mathbf{G}(t) = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))}{\partial \boldsymbol{u}}\right|_{\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}} = \begin{pmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_m} \end{pmatrix} \tag{2.38}$$

$$\mathbf{H}(t) = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}(t))}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}} = \begin{pmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots & \frac{\partial h_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \frac{\partial h_n}{\partial x_2} & \cdots & \frac{\partial h_n}{\partial x_n} \end{pmatrix} \tag{2.39}$$

By using equations (2.31) and (2.32), the linearized error state model is given by [11]

$$\delta\dot{\boldsymbol{x}} = \dot{\boldsymbol{x}} - \boldsymbol{f}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) = \mathbf{F}(t)\delta\boldsymbol{x} + \mathbf{G}(t)\delta\boldsymbol{u} \tag{2.40}$$

$$\delta\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{h}(\hat{\boldsymbol{x}}) = \mathbf{H}(t)\delta\boldsymbol{x}. \tag{2.41}$$

The Figure 2.6 shows the algortihm of error state Kalman Filter. One of the

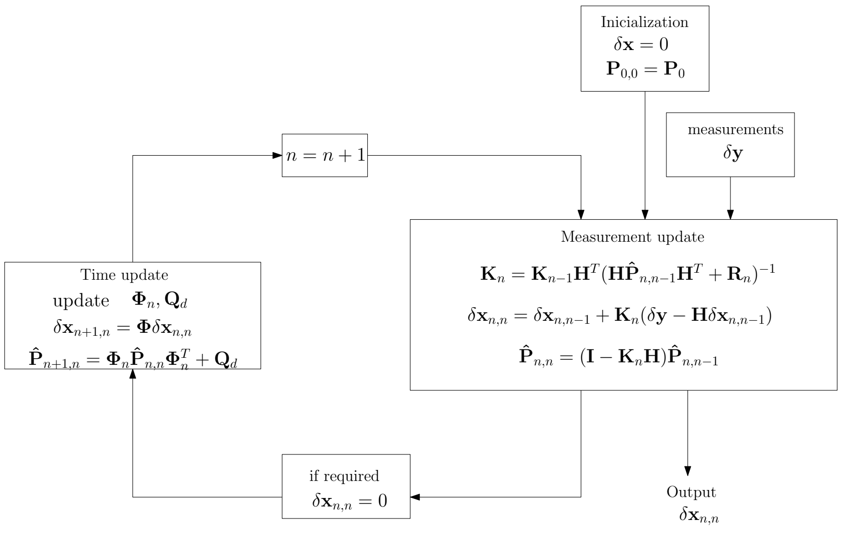

Inicialization
$\delta\mathbf{x} = 0$
$\mathbf{P}_{0,0} = \mathbf{P}_0$
measurements
$\delta\mathbf{y}$
$n = n + 1$
Measurement update
$\mathbf{K}_n = \mathbf{K}_{n-1}\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}_{n,n-1}\mathbf{H}^T + \mathbf{R}_n)^{-1}$
$\delta\mathbf{x}_{n,n} = \delta\mathbf{x}_{n,n-1} + \mathbf{K}_n(\delta\mathbf{y} - \mathbf{H}\delta\mathbf{x}_{n,n-1})$
$\hat{\mathbf{P}}_{n,n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H})\hat{\mathbf{P}}_{n,n-1}$
Time update
update $\boldsymbol{\Phi}_n, \mathbf{Q}_d$
$\delta\mathbf{x}_{n+1,n} = \boldsymbol{\Phi}\delta\mathbf{x}_{n,n}$
$\hat{\mathbf{P}}_{n+1,n} = \boldsymbol{\Phi}_n\hat{\mathbf{P}}_{n,n}\boldsymbol{\Phi}_n^T + \mathbf{Q}_d$
if required
$\delta\mathbf{x}_{n,n} = 0$
Output
$\delta\mathbf{x}_{n,n}$

Figure 2.6: Error state Kalman filter algorithm

advantages of the error state Kalman filter is that the filter is known to be more stable than classical Kalman filter estimating the absolute values.

2.4 Dynamic Detection

The Kalman filter requires to design a dynamic mathematical model. If the model is optimal, no dynamic detection may be needed. However, optimal models can become very difficult to describe and may be unobservable. Therefore suboptimal models are used instead. The essential issue in Kalman filtering using suboptimal models is to perform dynamic detection within a navigated object's motion. Reduction of bias influencing and protection of the model states against erroneous behavior can, then, be made by adjusting covariance matrices.
The dynamic detection is based on accelerometer and angular velocity measurements. Absolute magnitudes of the measured accelerations and angular rates at time n are computed according to equations (2.42) and (2.43),

$$||\boldsymbol{a}_n|| = \sqrt{a_x^2[n] + a_y^2[n] + a_z^2[n]}, \tag{2.42}$$

$$||\boldsymbol{\omega}_n|| = \sqrt{\omega_x^2[n] + \omega_y^2[n] + \omega_z^2[n]}, \tag{2.43}$$

where $a_i[n]$ and $\omega_i[n]$, $i \in \{x, y, z\}$ signify measurements in individual i-axis.

In the case of accelerometers measurements, when the vehicle is not accelerating, the absolute value should be equal to 1g. To eliminate the gravitational acceleration, the accelerometer absolute value is compensated as follows

$$\tilde{a}_n = 1 - ||\boldsymbol{a}_n||. \tag{2.44}$$

Adaptive CUSUM. In many applications, the stochastic properties of the data are difficult to describe or time consuming to identify or they are not known at all. Therefore, there are various detection algorithms that does not require any stochastic knowledge about the dataset.
The adaptive CUSUM algorithm combines the idea of CUSUM algorithm together with adaptive filtering [28]. The parameter of interest x_n is filtered according to (2.45).

$$\theta_x[n] = \alpha\theta_x[n-1] + (1-\alpha)x_n, \tag{2.45}$$

where α is a filtering constant.

Then, an error $s[n]$ is calculated as

$$s[n] = x_n - \theta_x[n]. \tag{2.46}$$

Test statistic function $G[n]$ is updated according to (2.47). If a given threshold h is exceeded, the signal has changed. The CUSUM algorithm is summarized in Alg. 1.

$$G[n] = \max(G[n-1] + s[n], 0). \tag{2.47}$$

```
initialization;
$G[0] = 0$;
$n = 1$;
set $h, \alpha$;
while Data is available do
    take current sample $x_n$;
    calculate;
    CUSUM $\theta_x[n] = \alpha\theta_x[n-1] + (1-\alpha)x_n$;
    $s = \boldsymbol{x} - \theta_x[n]$;
    $G[n] = \max(G[n-1] + s, 0)$;
    if $G[n] > h$ then
        $d_x[n] = 1$;
    else
        $d_x[n] = 0$;
    end
    $n = n + 1$;
end
```

Algorithm 1: Adaptive CUSUM algorithm [28]

Chapter 3
Navigation Solution

The third chapter provides derivation and design of the basic inertial navigation solution consisting of IMU measurements to compute position, velocity and orientation of the vehicle. To introduce the chapter, different reference frames and transformations between them are described. Then, the derivation of the mechanization equation in the navigation frame is presented.

3.1 Reference Frames

Many applications require properly designed reference frames and coordinate systems in order to solve and calculate mathematical equations. Various reference frames are typically considered in navigation systems. The ability to convert between individual frames is necessary. Unless otherwise stated, the coordinate system defined in all reference frames uses a right-handed Cartesian coordinate system consisting of three orthogonal axes $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ with an origin fixed within the reference system. The orthogonal and right-handed properties allow the conversion between individual frames-of-reference. The most common reference frames in navigation systems are Earth-centred inertial (ECI), Earth-centered-Earth-fixed (ECEF), Local tangent plane, Body frame and Latitude-Longitude-Altitude (LLA) frame that is the only reference frame that does not utilize Cartesian coordinates.

3.1.1 Earth-Centred Inertial

An inertial reference frame is defined as a reference frame, where Newton's laws of motion can be applied [29]. It means that an isolated object with no force applied to it remains at rest or moves with constant velocity, as stated by the First law of motion. The ECI frame has its origin at the center of mass of Earth and it is fixed with respect to the stars. The fact that the ECI frame does not rotate with Earth makes the system inertial. The equatorial plane of Earth corresponds to the x-y plane and the x-axis points toward the vernal equinox. The z-axis coincides with Earth's axis of rotation that runs through the North pole [29]. Finally, the y axis completes the right-handed, orthogonal coordinate system, as shown in Figure 3.1. The ECI reference

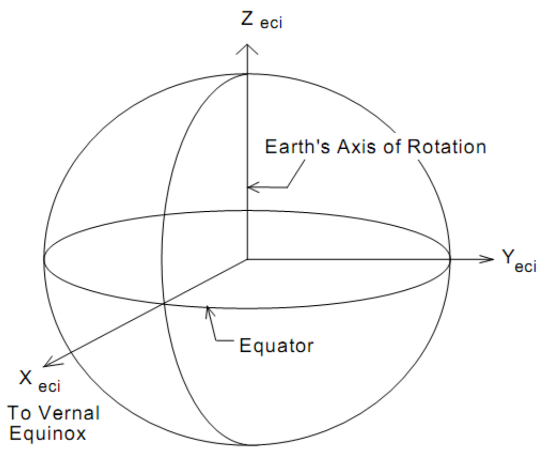


Figure 3.1: ECI reference frame [31]

frame is widely used in astronomy to describe objects in space since the motion equations are simpler and easier to solve in an inertial reference frame.

3.1.2 Earth-Centered-Earth-Fixed

ECEF reference frame is defined similarly as the ECI reference frame. The origin is located in the center of mass of Earth and the z-axis coincides with Earth's axis of rotation as shown in Figure 3.2. However, the x-axis intersects the Greenwich Meridian and y axis completes the right-handed orthogonal coordinate system. [33] Both x and y axes are fixed with respect to Earth's surface which makes the coordinate system not inertial as it rotates with Earth. It is convenient to define an angular rate ω_{ie} that represents the rotation of ECEF frame with respect to ECI frame. The angular rate ω_{ie} can be estimated as follows [29]

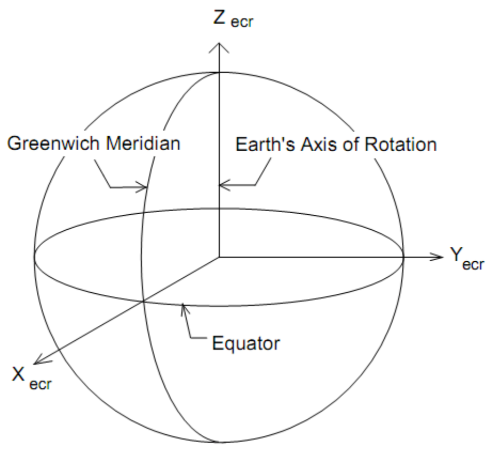


Figure 3.2: ECEF reference frame [31]

$$\omega_{ie} \approx \frac{1 + 365.25 \text{ cycle}}{(365.25)(24 \text{ hour})} \frac{2\pi \text{ rad/cycle}}{3600 \text{ sec/hour}} = 7.292115 \text{ x } 1\text{e}^{-5} \frac{\text{rad}}{\text{s}}. \tag{3.1}$$

The Earth rotational angular rate vector in ECEF frame is then defined as

$$\boldsymbol{\omega}_{ie}^{e} = \begin{pmatrix} 0 & 0 & \omega_{ie} \end{pmatrix}^{T} . \tag{3.2}$$

World Geodetic System

World Geodetic System (WGS) is a set of standards and definitions to describe a mathematical model of Earth. The progress of space science in the last century as well as the need for global maps in the navigation or aviation industry led to the first WGS system developed in the late 1950' by the US Department of Defense [35]. The latest WGS system was produced in 1984. Thus, it is referred to as WGS-84. The coordinate system used in the standards is the same as the coordinate system in the ECEF reference frame. The WGS-84 approximates the planet Earth by a reference ellipsoid with the origin lying in the Earth's center of mass and its parameters defined in Table 3.1.

To describe the position of an object on the reference ellipsoid, geographic

Semi-major axis	a	6378137 m
Flattening Factor	1/f	298.257223563
Angular rate	ω_{ie}	7.292115 x $1e^{-5}\frac{\text{rad}}{\text{s}}$
Geocentric Gravitational Constant	GM	3.986004418 x $1e^{14}\frac{\text{m}^3}{\text{s}^2}$

Table 3.1: Reference ellipsoid parameters [36]

coordinates called longitude and latitude are used. Latitude ϕ is defined as an angle between the equatorial plane and straight line that begins at the origin of the coordinate system and goes through a point on the ellipsoid [29]. Longitude λ is an angle in the equatorial plane from the Greenwich Meridian to the projection of the point on the reference ellipsoid onto the equatorial plane [29]. Both longitude and latitude are depicted in Figure 3.3 . The last coordinate describes the height h above the reference ellipsoid. A better approximation of Earth, known as the geoid, can be computed based on the influence of Earth's gravity and rotation. It is important to distinguish between the height h above the reference ellipsoid, the height H above the geoid and the true height N above the surface of Earth. The WGS-84 is currently being used in the Global Positioning System (GPS).

Let P be a point on the reference ellipsoid. A vertical circle on the celestial sphere is defined as a circle that passes through a center of the sphere and it is perpendicular to the horizontal of point P [33]. The meridian plane of P is a verticle circle that passes through the north and the south horizon points. By intersecting the meridian plane with the Earth ellipsoid, the meridian is obtained [29] .The Radius of Curvature of Meridian R_M is an important

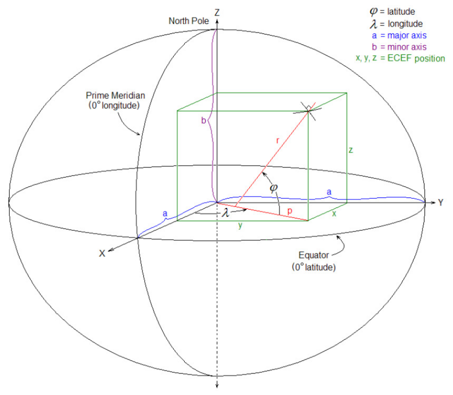


Figure 3.3: Longitude and latitude definition [37]

parameter and it can be computed as follows [33]

$$R_M(\phi) = \frac{a(1-e^2)}{(1-e^2\sin^2(\phi))^{\frac{3}{2}}}, \tag{3.3}$$

where ϕ stands for the latitude of point P and e is the excentricity of the reference ellipsoid.

Another important parameter is the Radius of Curvature in Prime Vertical R_N. Prime Vertical is a vertical circle that passes though the east and west horizon points [29]. If the prime vertical plane and the Earth ellipsoid are intresected, the resulting plane has a radius reffered to as the normal radius defined by the following equation [33]

$$R_N(\phi) = \frac{a}{(1-e^2\sin^2(\phi))^{\frac{1}{2}}} \tag{3.4}$$

The conversion of geographical position $\boldsymbol{r}^{LLA} = (\phi,\ \lambda,\ h)$ and ECEF frame position $\boldsymbol{r}^{e} = (x,\ y,\ z)$ is given by [29]

$$\begin{aligned} x &= (R_N + h)\cos\phi\cos\lambda && (3.5)\\ y &= (R_N + h)\cos\phi\sin\lambda && (3.6)\\ z &= (R_N(1-e^2) + h)\sin\phi && (3.7) \end{aligned}$$

The inverse transform is solved by applying iterative numerical methods since closed form expression of the transformation does not exist.

3.1.3 Local Tangent Plane

The local tangent plane reference frame is designed by fitting a tangent plane to a certain fixed point on the geodetic reference ellipse [29]. The fixed point where the tangent plane is constructed becomes the origin of the reference frame. The north axis is denoted by x-axis and it points to the true north. z - axis can either point down to the Earth's ground or point up perpendicular to the tangent plane. Finally, the y - axis completes the Cartesian coordinate system and points to the East. Depending on the direction of the $z-$ axis, the local tangent plane reference frame can also be referred to as East-North-Up (ENU) or North, East and Down (NED) navigation reference frame. This thesis uses the NED coordinate system. It is often used in the aircraft industry for local navigation but many other engineering fields have found NED frame usage. The tangent plane reference frame is depicted in Figure 3.4

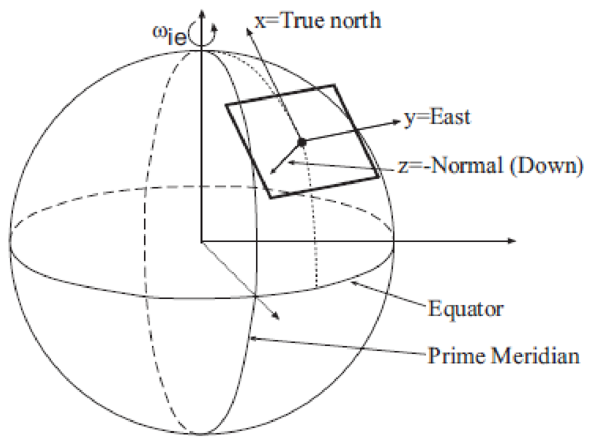


Figure 3.4: Tangent plane reference frame [32]

3.1.4 Body Frame

Strapdown systems have sensors attached to the body of the vehicle. For this reason, a body frame is defined to describe measurements of individual sensors correctly. The origin is usually fixed in the center of gravity of the vehicle. The x-axis points to the forward direction as the vehicle moves. The z axis points to the bottom of the vehicle and again, the y axis is defined as it completes the right-handed orthogonal system.

3.1.5 Transformation between Reference Frames

Since various navigation systems compute the position and the velocity in different frames, it is essential to find conversions between these frames. Given the fact that the Cartesian coordinate system describes most of the reference frames, the transformation in terms of rotation and translation can be easily defined by a transformation matrix $\mathbf{C}_a^b$ from reference frame a to b. In the case of LLA reference frames where points are often described by angles, the situation can become more difficult. The use of a numerical method is then inevitable.
The matrix $\mathbf{C}_a^b$ is called a direction cosine matrix and it has nine elements. However, there are three orthogonality and three normality constraints meaning only three angles are enough to describe the matrix in general [29]. Since

the matrix $\mathbf{C}_a^b$ represents the rotation transformation, it holds

$$\det \mathbf{C}_a^b = 1. \tag{3.8}$$

In addition, due to the orthogonal property, the matrix inversion of rotation can be easily computed as follows

$$\mathbf{C}_b^a = (\mathbf{C}_a^b)^{-1} = (\mathbf{C}_a^b)^T. \tag{3.9}$$

3.1.6 Euler angles

Every rotation in three-dimensional Euclidean space can be achieved by rotating around the coordinate system's three axes. An orientation of a rigid body with respect to a fixed coordinate system can be described by three angles denoted as Euler angles - roll φ, pitch θ and yaw ψ [29]. Euler angles play a significant role in aircraft navigation and are defined according to Figure 3.5. To give an example, plane rotation matrices about axes $\boldsymbol{x}$, $\boldsymbol{y}$ and

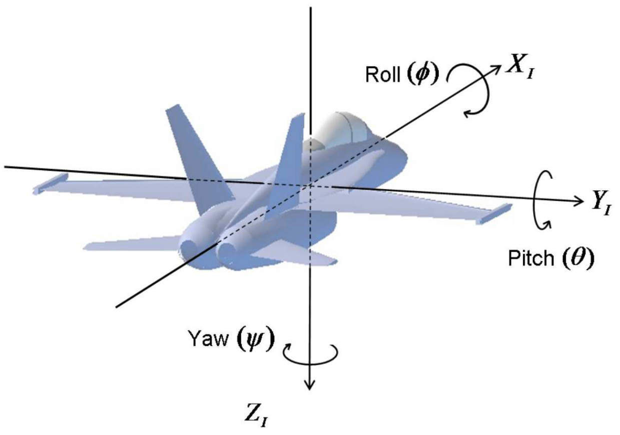


Figure 3.5: Body frame and Euler angles [34]

$\boldsymbol{z}$ in a counterclockwise direction are shown in eq. (3.10), eq. (3.11) and eq. (3.12), respectively .

$$\mathbf{C}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x \\ 0 & -\sin\theta_x & \cos\theta_x \end{pmatrix}, \tag{3.10}$$

$$\mathbf{C}_y = \begin{pmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{pmatrix}, \tag{3.11}$$

$$\mathbf{C}_z = \begin{pmatrix} \cos\theta_z & \sin\theta_z & 0 \\ -\sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.12}$$

where θ_i, $i \in \{x, y, z\}$ is an angle of rotation about the axis of interest.

ECEF to Tangent Plane

The convertion between ECEF and NED frame can be easily computed by applying two plane rotations. Let P be a point on the reference ellipsoid expressed by ECEF coordinates $\boldsymbol{r}_{\text{ecef}}$. To describe the point P in NED frame, a plane rotation about the z-axis of ECEF coordinate system is performed to align the rotated y-axis with the east axis of NED frame [29]. The angle of rotation coincides with the longitude λ of point P. The rotation matrix $\mathbf{C}_z$ looks as follows

$$\mathbf{C}_z = \begin{pmatrix} \cos\lambda & \sin\lambda & 0 \\ -\sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.13}$$

Then, the new coordinate system is rotated by angle $\frac{\pi}{2} + \phi$ to merge the $z-axis$ with tangent plane inward pointing vector [29], where ϕ is the latitude of point P. The second rotation matrix is defined as [29]

$$\mathbf{C}_y = \begin{pmatrix} \cos\frac{\pi}{2}+\phi & 0 & \sin\frac{\pi}{2}+\phi \\ 0 & 1 & 0 \\ -\sin\frac{\pi}{2}+\phi & 0 & \cos\frac{\pi}{2}+\phi \end{pmatrix} = \begin{pmatrix} -\sin\phi & 0 & \cos\phi \\ 0 & 1 & 0 \\ -\cos\phi & 0 & -\sin\phi \end{pmatrix} \tag{3.14}$$

Finally, the transformation matrix $\mathbf{C}_e^n$ from ECEF to NED frame is computed by multiplying the rotation matrices in the same order as it was described.

$$\mathbf{C}_e^n = \mathbf{C}_y\mathbf{C}_z = \begin{pmatrix} -\sin\phi\cos\lambda & -\sin\phi\sin\lambda & \cos\phi \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\phi\cos\lambda & -\sin\lambda\cos\phi & -\sin\phi \end{pmatrix} \tag{3.15}$$

The inverse transformation is easily done according to eq. 3.9. It is often convinient to perform the transformation on the Earth rotational angular rate vector $\boldsymbol{\omega}_{ie}^e$. As a result, it yields

$$\boldsymbol{\omega}_{ie}^n = \mathbf{C}_e^n\boldsymbol{\omega}_{ie}^e = \begin{pmatrix} \omega_{ie}\cos\phi \\ 0 \\ -\omega_{ie}\sin\phi \end{pmatrix}, \tag{3.16}$$

where ϕ denotes longitude.

Body to Tangent Plane

In order to investigate the transformation matrix $\mathbf{C}_b^n$ from body to NED frame, the Euler angles need to be involved. From the geometry point of view, it can be shown that the matrix can be computed by performing three plane rotations about all three axes in body frame [29]. Since the matrix multiplication is not commutative, the order in which the matrices are multiplied is important. First, a rotation about z-axis is carried out by an angle ψ. Then, another plane rotation is performed about the y-axis by an angle θ. Finally, the y-z plane is rotated by an angle φ. Using the plane rotation matrices defined in eq.(3.10 - 3.12), the NED to body frame transformation matrix $\mathbf{C}_n^b$ reads

$$\begin{aligned}
\mathbf{C}_n^b &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \mathrm{c}\varphi & \mathrm{s}\varphi \\ 0 & -\mathrm{s}\varphi & \mathrm{c}\varphi \end{pmatrix} \begin{pmatrix} \mathrm{c}\theta & 0 & -\mathrm{s}\theta \\ 0 & 1 & 0 \\ \mathrm{s}\theta & 0 & \mathrm{c}\theta \end{pmatrix} \begin{pmatrix} \mathrm{c}\psi & \mathrm{s}\psi & 0 \\ -\mathrm{s}\psi & \mathrm{c}\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \\
\mathbf{C}_n^b &= \begin{pmatrix} \mathrm{c}\theta\mathrm{c}\psi & \mathrm{c}\theta\mathrm{s}\psi & -\mathrm{s}\theta \\ \mathrm{s}\varphi\mathrm{s}\theta\mathrm{c}\psi - \mathrm{c}\theta\mathrm{s}\psi & \mathrm{s}\theta\mathrm{s}\psi\mathrm{s}\varphi - \mathrm{c}\theta\mathrm{c}\psi & \mathrm{s}\theta\mathrm{s}\varphi \\ \mathrm{c}\varphi\mathrm{s}\theta\mathrm{c}\psi + \mathrm{s}\varphi\mathrm{s}\psi & \mathrm{c}\varphi\mathrm{s}\theta\mathrm{s}\psi - \mathrm{s}\varphi\mathrm{c}\psi & \mathrm{c}\varphi\mathrm{c}\theta \end{pmatrix},
\end{aligned}$$

where $\mathrm{c}\alpha$ and $\mathrm{s}\alpha$ signify cosine and sine of an angle $\alpha \in \{\varphi,\ \theta,\ \psi\}$, respectively.

Given the matrix $\mathbf{C}_b^n$, it may be simply verified that the Eular angles can be computed as follows [29]

$$\theta = -\mathrm{asin}(\mathbf{C}_b^n[3,1]), \tag{3.17}$$

$$\varphi = \mathrm{atan2}(\mathbf{C}_b^n[3,2], \mathbf{C}_b^n[3,3]), \tag{3.18}$$

$$\psi = \mathrm{atan2}(\mathbf{C}_b^n[2,1], \mathbf{C}_b^n[1,1]). \tag{3.19}$$

Rotating Frames Transformation

This section gives a brief description of relations between individual frames of reference when the frames are rotating. The derivation of the expressions and more information can be found in [11] or [29]. Later on, when deriving the error model, it is necessary to describe a time change of position in body frame $\dot{\boldsymbol{r}}^b = \boldsymbol{v}^b = (v_x^b,\ v_y^b,\ v_z^b)$ with respect to the time change of position in LLA frame $\dot{\boldsymbol{r}}^{LLA} = (\dot{\phi},\ \dot{\lambda},\ \dot{h})$. It is also required to find their mutal angular velocity vector $\boldsymbol{\omega}_{en}^n$. It can be shown that the relation between $\boldsymbol{v}^n = (v_n^n,\ v_e^n,\ v_d^n)$ representing velocity in navigation frame and $\dot{\boldsymbol{r}}^{LLA}$ is as follows [11]

$$\begin{pmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{pmatrix} = \begin{pmatrix} \frac{v_n^n}{R_M+h} \\ \frac{v_e^n}{(R_N+h)\cos(\phi)} \\ -v_d^n \end{pmatrix} \tag{3.20}$$

Exploiting the geometry of the problem, the mutal anglura velocity $\boldsymbol{\omega}_{en}^n$ can be derived by applying two transformations and it reads [11]

$$\boldsymbol{\omega}_{en}^n = \begin{pmatrix} \dot{\lambda}\cos\phi \\ -\dot{\phi} \\ -\dot{\lambda}\sin(\phi) \end{pmatrix} = \begin{pmatrix} \frac{v_e^n}{R_N+h} \\ -\frac{v_n^n}{R_M+h} \\ -\frac{v_e^n}{R_N+h}\tan\phi \end{pmatrix} \tag{3.21}$$

A mutal angular velocity $\boldsymbol{\omega}_{in}^n$ between inertial and navigation frame expressed in naviagtion frame will aslo be needed. It holds [11]

$$\boldsymbol{\omega}_{in}^n = \boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n. \tag{3.22}$$

Using the eq. 3.16 and 3.21, it yields

$$\boldsymbol{\omega}_{in}^{n} = \begin{pmatrix} (\omega_{ie} + \dot{\lambda})\cos\phi \\ -\dot{\phi} \\ -(\omega_{ie} + \dot{\lambda})\sin(\phi) \end{pmatrix} = \begin{pmatrix} \omega_{ie}\cos\phi + \frac{v_e^n}{R_N+h} \\ -\frac{v_n^n}{R_M+h} \\ -\omega_{ie}\sin\phi - \frac{v_e^n}{R_N+h}\tan\phi \end{pmatrix} \tag{3.23}$$

It is convinient to define a skew matrix that will be used often in the derivations.
Let $\boldsymbol{\omega} = (\omega_1 \;\; \omega_2 \;\; \omega_3)^T$ be a 3×1 vectror. The skew-symmetric matrix $\boldsymbol{\Omega}$ of vector $\boldsymbol{\omega}$ is defined as follows

$$(\boldsymbol{\omega}\times) = \boldsymbol{\Omega} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}. \tag{3.24}$$

3.1.7 Rotation of Reference Frame

It is obvious that the direction cosine matrix is not constant due to the fact that the reference frames can rotate. It is therefore necessary to describe the time change of the DCM matrix $\dot{\mathbf{C}}_a^b$ from reference frame a to b. Let $\dot{\mathbf{C}}_a^b$ express by the definition of time derivative [29]

$$\dot{\mathbf{C}}_a^b = \lim_{\delta t \longrightarrow 0} \frac{\mathbf{C}_a^b(t+\delta t) - \mathbf{C}_a^b(t)}{\delta t}. \tag{3.25}$$

Since the term δt goes to zero, the rotation matrix $\mathbf{C}_a^b(t+\delta t)$ can be decomposed to two rotations as follows [29]

$$\mathbf{C}_a^b(t+\delta t) = \mathbf{C}_{b(t)}^{b(t+\delta t)}\mathbf{C}_a^b(t) \tag{3.26}$$

The term $\mathbf{C}_{b(t)}^{b(t+\delta t)}$ represents the small angle rotation which can be replaced by [29]

$$\mathbf{C}_{b(t)}^{b(t+\delta t)} = \mathbf{I} - \boldsymbol{\Omega}_{ab}^b \delta t, \tag{3.27}$$

where $\boldsymbol{\Omega}_{ab}^b = (\boldsymbol{\omega}_{ab}^b \times)$.

Finally, using the presented equations, the time change of transformation matrix $\dot{\mathbf{C}}_a^b$ can be computed as

$$\dot{\mathbf{C}}_a^b = \lim_{\delta t \longrightarrow 0} \frac{(\mathbf{I} - \boldsymbol{\Omega}_{ab}^b \delta t)\mathbf{C}_a^b - \mathbf{C}_a^b}{\delta t} \tag{3.28}$$

$$= -\boldsymbol{\Omega}_{ab}^b \mathbf{C}_a^b \tag{3.29}$$

$$= \mathbf{C}_a^b \boldsymbol{\Omega}_{ba}^a \tag{3.30}$$

3.2 Mechanization Equations of the Navigation Solution

The mechanization equations provide the mathematical description of vehicle kinematics. It is a set of differential equations determining the change in position, velocity and angular position with respect to acceleration and angular velocity. It is very important to choose a reference frame and solve the equations in it. Depending on the reference frame used, the equations have different forms. In this section, the derivation of mechanization equations is done in navigation frame. Unless otherwise stated, by navigation frame, it is referred to as LLA frame in position and NED reference frame in velocity.

3.2.1 Mechanization equations in Navigation frame

In navigation, the machanization equations in navigation frame are the main point of interest. The derivation of the mechanization equations in navigation frame is done by applying the transformation matrix $\mathbf{C}_i^n$ to velocity in inertial frame $\boldsymbol{v}^i$ as follows [11]

$$\boldsymbol{v}^n = \mathbf{C}_i^n \boldsymbol{v}^i \tag{3.31}$$

To find the equation for acceleration, derivative with respect to time is computed which gives

$$\dot{\boldsymbol{v}}^n = \dot{\mathbf{C}}_i^n \boldsymbol{v}^i + \mathbf{C}_i^n \dot{\boldsymbol{v}}^i. \tag{3.32}$$

Exploiting the eq. 3.30, the time derivative of transformation matrix $\dot{\mathbf{C}}_i^n$ can be written as

$$\dot{\mathbf{C}}_i^n = \mathbf{C}_i^n \boldsymbol{\Omega}_{ni}^i. \tag{3.33}$$

The acceleration in inertial frame can be expressed as follows [11]

$$\dot{\boldsymbol{v}}^{\boldsymbol{i}} = \mathbf{C}_b^i \boldsymbol{f}^b + \boldsymbol{g}^i - \boldsymbol{\Omega}_{ie}^i \boldsymbol{v}^i, \tag{3.34}$$

where $\boldsymbol{g}^i$ is a gravitational acceleration, $-\boldsymbol{\Omega}_{ie}^i \boldsymbol{v}^i$ refers to Coriolis acceleration.

Plugging in eq. 3.33 to eq. 3.32, substituting the inertial velocity $\boldsymbol{v}^i$ by eq. 3.34 and performing simple mathematical manipulations, the acceleration in navigation frame $\dot{\boldsymbol{v}}^n$ is given by

$$\dot{\boldsymbol{v}}^n = \left(\boldsymbol{\Omega}_{ni}^n - \boldsymbol{\Omega}_{ie}^n\right) \boldsymbol{v}^n + \mathbf{C}_b^n \boldsymbol{f}^b + \boldsymbol{g}^n. \tag{3.35}$$

It holds [11]

$$\begin{aligned} \boldsymbol{\omega}_{ni}^n &= \boldsymbol{\omega}_{ne}^n + \boldsymbol{\omega}_{ei}^n, & (3.36)\\ \boldsymbol{\omega}_{ei}^n &= -\boldsymbol{\omega}_{ie}^n. & (3.37) \end{aligned}$$

The final expression looks as follows

$$\dot{\boldsymbol{v}}^n = \mathbf{C}_b^n \boldsymbol{f}^b - (\boldsymbol{\Omega}_{en}^n + 2\boldsymbol{\Omega}_{ie}^n)\, \boldsymbol{v}^n + \boldsymbol{g}^n, \tag{3.38}$$

where

$$\begin{aligned} \boldsymbol{\Omega}_{en}^n &= (\boldsymbol{\omega}_{en}^n \times) & (3.39)\\ \boldsymbol{\Omega}_{ie}^n &= (\boldsymbol{\omega}_{ie}^n \times). & (3.40) \end{aligned}$$

The term $\boldsymbol{\omega}_{en}^n$ represents the angular rate of navigation reference frame with respect to the Earth rotation. The navigated object is also influenced by centripetal acceleration $\boldsymbol{a}_c$ given by

$$\boldsymbol{a}_c = \boldsymbol{\omega}_{ib}^b \times \boldsymbol{v}^b. \tag{3.41}$$

The centripetal acceleration is added to the navigation solution.
To conclude the section, the fig 3.6 shows the scheme of inertial navigation system. The set of navigation equations is given by

$$\begin{aligned} \dot{\boldsymbol{p}}^{\text{LLA}} &= \mathbf{D}\boldsymbol{v}^n & (3.42)\\ \dot{\boldsymbol{v}}^n &= \mathbf{C}_b^n(\boldsymbol{f}^b - \boldsymbol{a}_c) - (\boldsymbol{\Omega}_{en}^n + 2\boldsymbol{\Omega}_{ie}^n)\, \boldsymbol{v}^n + \boldsymbol{g}^n & (3.43)\\ \dot{\mathbf{C}}_b^n &= \mathbf{C}_b^n \boldsymbol{\Omega}_{nb}^b, & (3.44) \end{aligned}$$

where $\mathbf{D}$ is a transforantion matrix from NED to LLA frame given by

$$\mathbf{D} = \begin{pmatrix} \frac{1}{R_M+h} & 0 & 0 \\ 0 & \frac{1}{(R_N+h)\cos(\phi)} & 0 \\ 0 & 0 & -1 \end{pmatrix}. \tag{3.45}$$

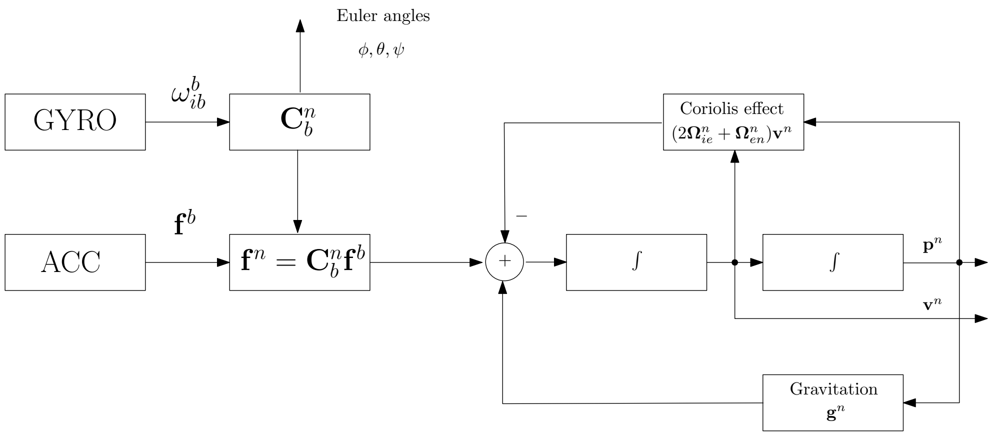


Figure 3.6: INS scheme

Chapter 4
Aided Navigation solution

The basic navigation solution based on IMU measurements was described in the previous chapter. The goal of the aided navigation solution is to add GNSS receiver, magnetometer, absolute air pressure sensor and an ultrasonic sensor to the inertial navigation system to compensate for the errors caused by integrating inertial sensor offsets and noise components.
First, the proposed navigation algorithm is described and explained. An error state dynamic model, as well as a measurement model, are derived and used in the error state Kalman filter. Finally, a description of the course alignment carried out at the initial stage of the algorithm is given.

4.1 Designed Navigation algorithm

The designed navigation solution consists of three main parts. The mechanization equations form the basic block, which takes IMU measurements as input and calculates the position, velocity and attitude of a navigated vehicle, as shown in Figure 3.6. When data from aiding sensors are available, the difference between the calculated values and measured one enters the Kalman filter. GNSS receiver provides information about position and velocity. The absolute air pressure sensor and ultrasonic sensor help estimate the altitude and magnetometer improve the heading's estimation. The Kalman filter is based on an error model and it fuses the data from the aiding sensors to predict an error $\delta \boldsymbol{x}$. The estimated error then compensates for the values provided by mechanization equations. After the compensation is done, the state error vector is reset to zero. As it was explained earlier, the need for dynamic detection arises due to the sub-optimality of the dynamic model. The dynamic detector estimates the dynamic state of the navigated vehicle and based on the estimated dynamic mode, parameters of the Kalman filter algorithm are modified. A more detailed description of the dynamic detector and the parameter modification is given in Chapter 5. To delete higher frequencies in the accelerometer and gyroscope measurements induced mainly by engine vibrations, a simple low-pass FIR filter was added. The GNSS data can be a little delayed. Thus, the filtration delay is not an issue as long as the order of the filter is relatively small to the sensor's sampling frequency. The filter design is carried out in Chapter 6. The whole proposed navigation algorithm is depicted in Figure 4.1.

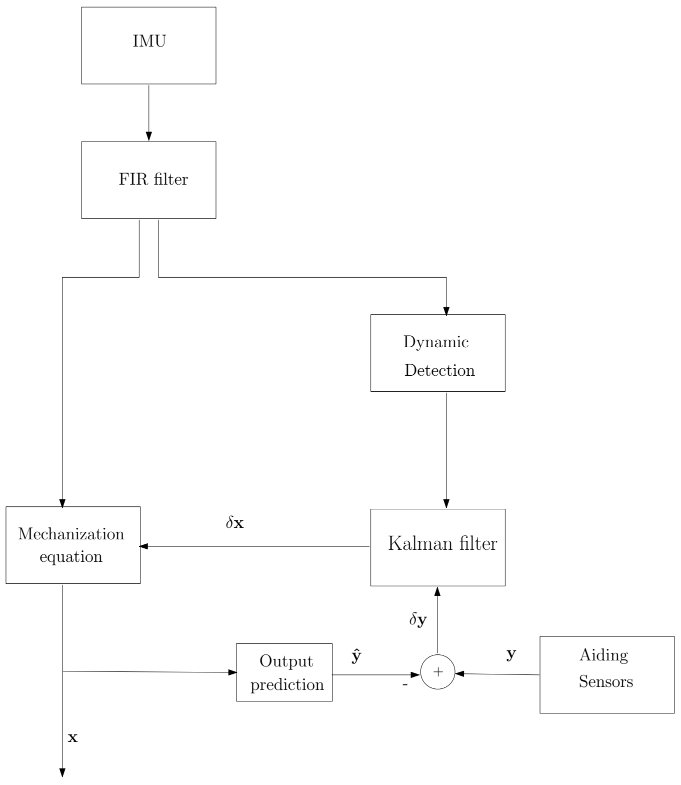


Figure 4.1: Developed INS system

4.2 Model Derivation

This section provides derivation and explenation of error state dynamic model and measurement model needed for error state Kalman filter algorithm.

4.2.1 Error state dynamic model

The error state dynamic model for position error, velocity error and small angle rotation (attitude) error and an augmented model including accelereometer and bias error estimation are well-known and they are described in many books and articles including [29], [11] or [30]. This section gives a brief derivation of the known model. Based on the sensor models and augmented error state model, a dyanmic error state vector is proposed.

Position Error Model

Let position error vector $\delta\boldsymbol{p}$ be composed of latitude error $\delta\phi$, longitude error $\delta\lambda$ and altitude error δh above the reference ellipsoid

$$\delta\boldsymbol{p} = \begin{pmatrix}\delta\phi & \delta\lambda & \delta h\end{pmatrix}^T . \tag{4.1}$$

The derivation of position error model is based on eq. (3.20). The time change of position vector can generally be written as

$$\dot{\boldsymbol{p}} = \boldsymbol{f_p}(\boldsymbol{p}, \boldsymbol{v}), \tag{4.2}$$

where

$$\boldsymbol{f_p}(\boldsymbol{p}, \boldsymbol{v}) = \begin{pmatrix} \frac{v_n^n}{R_M+h} \\ \frac{v_e^n}{(R_N+h)\cos(\phi)} \\ -v_d^n \end{pmatrix} \tag{4.3}$$

As shown before, the eq.4.2 can be linearized by using Taylor series expansion. Moreover, applying the error vector definition, the linearized postion error model is derived to the following form

$$\begin{aligned} \delta\dot{\boldsymbol{p}} &= \frac{\partial \boldsymbol{f_p}}{\partial \boldsymbol{p}}\delta\boldsymbol{p} + \frac{\partial \boldsymbol{f}_p}{\partial \boldsymbol{v}}\delta\boldsymbol{v} & (4.4) \\ &= \mathbf{F}_{pp}\delta\boldsymbol{p} + \mathbf{F}_{pv}\delta\boldsymbol{v}, & (4.5) \end{aligned}$$

where $\mathbf{F}_{pp}$ and $\mathbf{F}_{pv}$ are the Jacobian matrices of function $\boldsymbol{f_p}$ with explicit form

$$\begin{aligned} \mathbf{F}_{pp} &= \begin{pmatrix} 0 & 0 & -\frac{v_n^n}{(R_M+h)^2} \\ \frac{v_e^n\sin(\phi)}{(R_N+h)\cos^2(\phi)} & 0 & \frac{-v_e^n}{(R_N+h)^2\cos(\phi)} \\ 0 & 0 & 0 \end{pmatrix}, & (4.6) \\ \mathbf{F}_{pv} &= \begin{pmatrix} \frac{1}{(R_M+h)} & 0 & 0 \\ 0 & \frac{1}{(R_n+h)\cos(\phi)} & 0 \\ 0 & 0 & -1 \end{pmatrix}. & (4.7) \end{aligned}$$

Velocity Error Model

In this section, the derivation of velocity error model is explained. The velocity error vector $\delta\dot{\boldsymbol{v}}^n$ in navigation frame is defined as

$$\delta\dot{\boldsymbol{v}}^n = \dot{\boldsymbol{v}}^n - \dot{\hat{\boldsymbol{v}}}^n, \tag{4.8}$$

where $\dot{\boldsymbol{v}}^n$ is the true velocity and $\dot{\hat{\boldsymbol{v}}}^n$ is the estimated one.
Making use of results in section 3.2.1 , the true velocity $\dot{\boldsymbol{v}}^n$ and the estimated one $\dot{\hat{\boldsymbol{v}}}^n$ are substituted by eq. 3.38 to yield

$$\delta\dot{\boldsymbol{v}}^n = (\mathbf{C}_b^n\boldsymbol{f}^b - \hat{\mathbf{C}}_b^n\hat{\boldsymbol{f}}^b) + ((2\hat{\boldsymbol{\Omega}}_{ie}^n + \hat{\boldsymbol{\Omega}}_{en}^n)\hat{\boldsymbol{v}}^n) - ((2\boldsymbol{\Omega}_{ie}^n + \boldsymbol{\Omega}_{en}^n)\boldsymbol{v}^n) + \boldsymbol{g}^n - \hat{\boldsymbol{g}}^n \tag{4.9}$$

Performing a gravity model analysis, the error gravity vector $\delta\boldsymbol{g}^n$ can be defined as follows [11]

$$\delta\boldsymbol{g}^n = \boldsymbol{g}^n - \hat{\boldsymbol{g}}^n = \begin{pmatrix} 0 \\ 0 \\ -2\frac{g}{R+h}\delta h \end{pmatrix}. \tag{4.10}$$

It can be shown that the following equality holds [29]

$$(\mathbf{C}_b^n\boldsymbol{f}^b - \hat{\mathbf{C}}_b^n\hat{\boldsymbol{f}}^b) = -[(\boldsymbol{f}^n\times)\boldsymbol{\rho}^n + \mathbf{C}_b^n\delta\boldsymbol{f}^b]. \tag{4.11}$$

Using eqs. (4.10) and (4.11), the error velocity vector $\delta\dot{\boldsymbol{v}}^n$ is re-written as

$$\delta\dot{\boldsymbol{v}}^n = -[(\boldsymbol{f}^n\times)\boldsymbol{\rho}^n + \mathbf{C}_b^n\delta\boldsymbol{f}^b] + ((2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n)\times\delta\boldsymbol{v}^n) - ((2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)\times\boldsymbol{v}^n) + \delta\boldsymbol{g}^n \tag{4.12}$$

The term $(2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n)$ is explicitly known as it was mentioned in section 3.1.6. For the sake of clarity, the equation is expressed again

$$(2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) = \begin{pmatrix} 2\omega_{ie}^n\cos\phi + \frac{v_e^n}{R_N+h} \\ -\frac{v_n^n}{R_M+h} \\ -2\omega_{ie}^n\sin\phi - \frac{v_e^n\tan\phi}{R_N+h} \end{pmatrix}. \tag{4.13}$$

The attention is paid to the expression $(2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)$. Once again, it can be expanded by Taylor series to the first order to get

$$\begin{aligned} (2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n) &= \frac{\partial(2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n)}{\partial\boldsymbol{p}^n}\delta\boldsymbol{p}^n + \frac{\partial(2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n)}{\partial\boldsymbol{v}^n}\delta\boldsymbol{v}^n && (4.14) \\ &= \boldsymbol{\Omega}_p\delta\boldsymbol{p}^n + \boldsymbol{\Omega}_v\delta\boldsymbol{v}^n, && (4.15) \end{aligned}$$

where $\boldsymbol{\Omega}_p$ and $\boldsymbol{\Omega}_v$ are Jacobian matrices given by

$$\begin{aligned} \boldsymbol{\Omega}_p = \frac{\partial(2\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)}{\partial\boldsymbol{p}^n} &= \begin{pmatrix} -2\omega_{ie}\sin(\phi) & 0 & \frac{-v_e^n}{(R_N+h)^2} \\ 0 & 0 & \frac{v_n^n}{(R_M+h)^2} \\ -2\omega_{ie}\cos(\phi) - \frac{-v_e^n}{\cos(\phi)(R_N+h)^2} & 0 & \frac{v_e^n\tan(\phi)}{(R_N+h)^2} \end{pmatrix}, \\ \boldsymbol{\Omega}_v = \frac{\partial(2\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)}{\partial\boldsymbol{v}^n} &= \begin{pmatrix} 0 & \frac{1}{R_N+h} & 0 \\ \frac{-1}{R_M+h} & 0 & 0 \\ 0 & \frac{-\tan\phi}{R_N+h} & 0 \end{pmatrix}. \end{aligned}$$

Next step is to perform the vector product $(2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)\times\ \boldsymbol{v}^n$. Exploiting a mathematical formula for vector product computation and using eq. (4.15), the result is

$$(2\delta\boldsymbol{\omega}_{ie}^n + \delta\boldsymbol{\omega}_{en}^n)\times\ \boldsymbol{v}^n = (\boldsymbol{v}^n\times)\boldsymbol{\Omega}_p\delta\boldsymbol{p}^n + (\boldsymbol{v}^n\times)\boldsymbol{\Omega}_v\delta\boldsymbol{v}^n. \tag{4.16}$$

Finally, taking into account the derivations above, it holds

$$\begin{aligned}\delta\dot{\boldsymbol{v}}^n &= \boldsymbol{\Omega}_p\delta\boldsymbol{p}^n + \delta\boldsymbol{g}^n + ((2\hat{\boldsymbol{\Omega}}^n_{ie} + \hat{\boldsymbol{\Omega}}^n_{en}) - (\boldsymbol{v}^n\times)\boldsymbol{\Omega}_v)\delta\boldsymbol{v}^n - (\boldsymbol{f}^n\times)\boldsymbol{\rho}^n - \mathbf{C}^n_b\delta\boldsymbol{f}^b \\ &= \mathbf{F}_{vp}\delta\boldsymbol{p}^n + \mathbf{F}_{vv}\delta\boldsymbol{v}^n + \mathbf{F}_{v\rho}\delta\boldsymbol{\rho}^n - \mathbf{C}^n_b\delta\boldsymbol{f}^b \qquad (4.17)\end{aligned}$$

where

$$\mathbf{F}_{vp} = \begin{pmatrix} -2\omega_{ie}\cos(\phi)v^n_e - \dfrac{(v^n_e)^2}{R_N\cos^2(\phi)} & 0 & -\dfrac{v^n_n v^n_d}{R^2_M} - \dfrac{(v^n_e)^2\tan(\phi)}{R^2_N} \\ 2\omega_{ie}(v^n_n\cos(\phi) - v^n_d\sin(\phi)) + \dfrac{(v^n_e)^2}{R_N\cos^2(\phi)} & 0 & \dfrac{v^n_n v^n_e\tan(\phi) - v^n_d v^n_e}{R_M R_N} \\ 2v^n_e\omega_{ie}\sin(\phi) & 0 & \dfrac{(v^n_e)^2}{R^2_M} + \dfrac{(v^n_n)^2}{R^2_N} - 2\dfrac{g}{R_M} \end{pmatrix}$$

$$\mathbf{F}_{vv} =$$

$$\begin{pmatrix} \dfrac{v_d}{R_M} & -2\left(\omega_{ie}\sin(\phi) + \dfrac{-v^n_e\tan(\phi)}{R_N}\right) & \dfrac{v^n_n}{R_M} \\ -2\omega_{ie}\sin(\phi) + \dfrac{-v^n_e\tan(\phi)}{R_N} & \dfrac{v^n_d + v^n_n\tan(\phi)}{R_M} & 2\omega_{ie}\cos(\phi) + \dfrac{v^n_e}{R_N} \\ -2\dfrac{v^n_n}{R_M} & -2\left(\omega_{ie}\cos(\phi) + \dfrac{v^n_e}{R_N}\right) & 0 \end{pmatrix}$$

$$\mathbf{F}_{v\rho} = -(\boldsymbol{f}^n\times).$$

Attitude Error Model

The vector $\boldsymbol{\rho}$ consists of angle errors between the real navigation frame and the computed one. It is convenient for correcting the body to navigation transformation matrix $\mathbf{C}^n_b$ as follows [29]

$$\mathbf{C}^n_b = (\mathbf{I} + \boldsymbol{\Sigma})\hat{\mathbf{C}}^n_b, \qquad (4.18)$$
$$\mathbf{C}^b_n = \hat{\mathbf{C}}^b_n(\mathbf{I} - \boldsymbol{\Sigma}), \qquad (4.19)$$

where $\hat{\mathbf{C}}^n_b$ is the old estimated transformation matrix and $\boldsymbol{\Sigma} = (\boldsymbol{\rho}\times)$ and $\mathbf{C}^n_b$ is the new estimate of the true transformation matrix.
In order to derive the attitude error model, the eq. 4.20 is assumed to be true. Further explenation of the equality can be found in [29].

$$\dot{\boldsymbol{\rho}}^n = \hat{\mathbf{C}}^n_b(\delta\boldsymbol{\omega}^b_{ib} - \delta\boldsymbol{\omega}^b_{in}) \qquad (4.20)$$

The variable $\delta\boldsymbol{\omega}^b_{ib}$ stands for the error of the body frame inertial relative angular rate in gyro measurements. Using eq. (4.19), it is straightforward to show that

$$\hat{\boldsymbol{\omega}}^b_{in} + \delta\boldsymbol{\omega}^b_{in} = \hat{\mathbf{C}}^b_n(\mathbf{I} - \boldsymbol{\Sigma})(\hat{\boldsymbol{\omega}}^n_{in} - \delta\boldsymbol{\omega}^n_{in}), \qquad (4.21)$$
$$\delta\boldsymbol{\omega}^b_{in} = \hat{\mathbf{C}}^b_n(\delta\boldsymbol{\omega}^n_{in} - \boldsymbol{\Sigma}\hat{\boldsymbol{\omega}}^n_{in}). \qquad (4.22)$$

As a result the following equation is obtained

$$\dot{\boldsymbol{\rho}} + \hat{\boldsymbol{\Omega}}^n_{in}\boldsymbol{\rho}^n = -\delta\boldsymbol{\omega}^n_{in} + \hat{\mathbf{C}}^n_b\delta\boldsymbol{\omega}^b_{ib}. \tag{4.23}$$

Now, the term $\delta\boldsymbol{\omega}^n_{in}$ is the point of interest since it is a variable that does not appear in error state vector. Yet again, the error angular rate $\delta\boldsymbol{\omega}^n_{in}$ is approximated by Taylor series expansion to first order to yield

$$\delta\boldsymbol{\omega}^n_{in} = \frac{\partial\boldsymbol{\omega}^n_{in}}{\partial\hat{\boldsymbol{p}}}\delta\boldsymbol{p}^n + \frac{\partial\boldsymbol{\omega}^n_{in}}{\partial\hat{\boldsymbol{v}}}\delta\boldsymbol{v}^n = \mathbf{F}_{\rho p}\delta\boldsymbol{p}^n + \mathbf{F}_{\rho v}\delta\boldsymbol{v}^n. \tag{4.24}$$

Finally, the linearized attitude error model looks as follows

$$\dot{\boldsymbol{\rho}} = \mathbf{F}_{\rho p}\delta\boldsymbol{p}^n + \mathbf{F}_{\rho v}\delta\boldsymbol{v}^n + \mathbf{F}_{\rho\rho}\boldsymbol{\rho}^n + \hat{\mathbf{C}}^n_b\delta\boldsymbol{\omega}^b_{ib} \tag{4.25}$$

where

$$\mathbf{F}_{\rho p} = \begin{pmatrix} \omega_{ie}\sin(\phi) & 0 & \dfrac{v^n_e}{(R_N+h)^2} \\ 0 & 0 & \dfrac{-v^n_n}{(R_M+h)^2} \\ \omega_{ie}\cos(\phi) + \dfrac{v^n_e}{(R_N+h)\cos^2(\phi)} & 0 & \dfrac{-v^n_e\tan(\phi)}{(R_N+h)^2} \end{pmatrix} \tag{4.26}$$

$$\mathbf{F}_{\rho v} = \begin{pmatrix} 0 & \dfrac{-1}{(R_N+h)} & 0 \\ \dfrac{-1}{(R_M+h)} & 0 & 0 \\ 0 & \dfrac{\tan(\phi)}{(R_N+h)} & 0 \end{pmatrix} \tag{4.27}$$

$$\mathbf{F}_{\rho\rho} = -(\boldsymbol{\omega}^n_{in}\times) \tag{4.28}$$

Augmented error state model

The inertial sensor's errors and biases can have a significant impact on the accuracy of the navigation solution. As a result, it is convenient to estimate the inertial sensors' biases and correct the measured values. Since the Kalman filter works with error terms, the dynamic error model is augmented to estimate the bias estimation errors. Both the accelerometer bias error vector and gyroscope bias error vector are modeled as a first-order Gauss-Markov process as defined in equation 2.9.

The augmented error state dynamic model is then given as follows [29]

$$\begin{pmatrix} \delta\dot{\boldsymbol{p}} \\ \delta\dot{\boldsymbol{v}} \\ \dot{\boldsymbol{\rho}} \\ \delta\dot{\boldsymbol{b}}_a \\ \delta\dot{\boldsymbol{b}}_\omega \end{pmatrix} = \begin{pmatrix} \mathbf{F}_{pp} & \mathbf{F}_{pv} & \mathbf{F}_{p\rho} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{F}_{vp} & \mathbf{F}_{vv} & \mathbf{F}_{v\rho} & -\mathbf{C}_b^n & \mathbf{0}_{3\times3} \\ \mathbf{F}_{\rho p} & \mathbf{F}_{\rho v} & \mathbf{F}_{\rho\rho} & \mathbf{0}_{3\times3} & \mathbf{C}_b^n \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathrm{dg}(\frac{1}{\tau_a}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathrm{dg}(\frac{1}{\tau_\omega}) \end{pmatrix} \begin{pmatrix} \delta\boldsymbol{p} \\ \delta\boldsymbol{v} \\ \boldsymbol{\rho} \\ \delta\boldsymbol{b}_a \\ \delta\boldsymbol{b}_\omega \end{pmatrix}$$
$$+ \begin{pmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ -\mathbf{C}_b^n & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{C}_b^n & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{pmatrix} \begin{pmatrix} \boldsymbol{v_a} \\ \boldsymbol{v}_\omega \\ \boldsymbol{w}_a \\ \boldsymbol{w}_\omega \end{pmatrix}, \tag{4.29}$$

where τ_a and τ_ω signify sensor time constants for accelerometer and gyroscope, respectively, dg stands for 3×3 diagonal matrix, $\boldsymbol{w}_a$, $\boldsymbol{w}_\omega$, $\boldsymbol{v_a}$ and $\boldsymbol{v}_\omega$ are white Gaussian noises.

Designed error state vector

Since the aiding sensors can have deflections leading to noisy or shifted outputs, it is necessary to compensate for these errors. Therefore, the error state space model is extended to additional parameters as the reference air pressure error δp_0, magnetometer offset error δb_m^n and ultrasonic sensor offset δh_0. These parameters are described as random constant leading to equations as follows

$$\delta\dot{p}_0 = 0, \tag{4.30}$$
$$\delta\dot{b}_m^n = 0, \tag{4.31}$$
$$\delta\dot{h}_0 = 0. \tag{4.32}$$

To sum up, in the presented model, the state error vector $\delta\boldsymbol{x}$ is given by

$$\delta\boldsymbol{x} = \begin{pmatrix} \delta\boldsymbol{p}^{LLA} & \delta\boldsymbol{v}^n & \boldsymbol{\rho} & \delta\boldsymbol{b}_a & \delta\boldsymbol{b}_\omega & \delta p_0 & \delta b_m^n & \delta h_0 \end{pmatrix}^T, \tag{4.33}$$

where $\delta\boldsymbol{p}^{LLA}$ is position error vector in LLA frame , $\delta\boldsymbol{v}^n$ denotes velocity error vector in NED frame , $\boldsymbol{\rho}$ represents small-angle rotation error vector, $\delta\boldsymbol{b}_a$ and $\delta\boldsymbol{b}_\omega$ are accelerometer bias error vector and gyro bias error vector, respectively, δp_0 represents reference air pressure error, δb_m^n denotes magnetometer offset error and, finally, δh_0 signifies ultrasonic sensor offset.

4.2.2 Measurement Model

The measurement error vector $\delta\boldsymbol{y}$ contains position error $\delta\boldsymbol{p}^{LLA}$ and velocity error $\delta\boldsymbol{v}^n$, magnetic field error δm^n, air pressure sensor measurement error δp and altitude error δh_u

$$\delta\boldsymbol{y} = \begin{pmatrix} \delta\boldsymbol{p}^{LLA} & \delta\boldsymbol{v}^n & \delta m^n & \delta p & \delta h_u \end{pmatrix}^{\mathrm{T}}. \tag{4.34}$$

If the measurements are available, the measurement error vector $\delta\boldsymbol{y}$ is computed as follows

$$\delta\boldsymbol{y} = \boldsymbol{y} - \hat{\boldsymbol{y}}, \tag{4.35}$$

where $\hat{\boldsymbol{y}}$ denotes a vector computed from navigation equations and sensor models and $\boldsymbol{y}$ represents a vector containing values measured by aiding sensors.

Magnetic Measurement Model

Concerning the magnetic measurement model, let m be total magnetic field strength, δ denotes declination angle in radians and ζ is a magnetic inclination. The vector $\boldsymbol{m}^n$ is referred to as reference magnetic vector, it points the true north and it is given by

$$\boldsymbol{m}^n = \left(m^n \quad 0 \quad 0\right)^T, \tag{4.36}$$

where $m^n = m\cos(\zeta)\cos(\delta)$.

The error measurement model for magnetic measurements looks as follows [29]

$$\begin{aligned}
\delta\boldsymbol{m}^n &= \boldsymbol{m}^n - \hat{\boldsymbol{m}}^n & (4.37)\\
&= \boldsymbol{m}^n - (\mathbf{I} - \boldsymbol{\Sigma})\mathbf{C}_b^n(\boldsymbol{m}^b + \boldsymbol{b}_m - \hat{\boldsymbol{b}}_m + \boldsymbol{w}_m) & (4.38)\\
&= \boldsymbol{\Sigma}\boldsymbol{m}^n - \mathbf{C}_b^n(\delta\boldsymbol{b}_m + \boldsymbol{w}_m) + \boldsymbol{\Sigma}\mathbf{C}_b^n(\delta\boldsymbol{b}_m + \boldsymbol{w}_m). & (4.39)
\end{aligned}$$

Because the models are linearized, the term $\boldsymbol{\Sigma}\mathbf{C}_b^n(\delta\boldsymbol{b}_m + \boldsymbol{w}_m)$ is neglected since it represents the second order. Using the fact that $\boldsymbol{\Sigma}\boldsymbol{m}^n = -(\boldsymbol{m}^n\times)\boldsymbol{\rho}$ and deleting the vertical measurements of the magnetic filed because ϵ_E calibration can be inaccurate[29], the model reduces to 1-dimensional eqaution as follows

$$\delta m^n = \mathbf{H_m}\delta\boldsymbol{x}, \tag{4.40}$$

where $\mathbf{H}_m = \Big(\mathbf{0}_{1\times3} \quad \mathbf{0}_{1\times3} \quad [0\ 0\ m^n] \quad \mathbf{0}_{1\times3} \quad \mathbf{0}_{1\times3} \quad 0 \quad 1 \quad 0\Big)$.

Pressure Measurement Model

The measurement model for air pressure sensor is given by

$$p(p_0, h) = p_0(1 - k_1 h)^{k_2}, \tag{4.41}$$

where $k_1 = 2.2558e^{-5}$, $k_2 = 5.2559$ and p_0 is the reference pressure .

To derive the error measurement model, the equation 4.41 is expanded to the first order by Taylor series expansion as follows

$$p(p_0, h) = \hat{p} + \frac{\partial p}{\partial h}\delta h + \frac{\partial p}{\partial p_0}\delta p_0. \tag{4.42}$$

Performing the partial derivatives and making use of the error definition it yields

$$\delta p = p - \hat{p} = -k_1 k_2 p_0 (1 - k_1 h)^{k_2 - 1}\delta h + (1 - k_1 h)^{k_2}\delta p_0. \tag{4.43}$$

The final model for air pressure measurements is

$$\delta p = \left(\boldsymbol{q}\ \mathbf{0}_{1\times3}\ \mathbf{0}_{1\times3}\ \mathbf{0}_{1\times3}\ \mathbf{0}_{1\times3}\ \ (1 - k_1 h)^{k_2}\ 0\ 0\right)\delta\boldsymbol{x}, \tag{4.44}$$

where $\boldsymbol{q} = [0\ 0\ \ -k_1 k_2 p_0 (1 - k_1 h)^{k_2 - 1}]$.

Ultrasonic Measurement Model

The ultrasonic sensor outputs height h_{um} above the ground. It is assumed that the navigated object is on the ground during the inicialization process. The measurement model is given by

$$h_u = h_{um} + h_g + h_0 = h + h_0, \tag{4.45}$$

where h_u is the computed height, h_g is the initial height above the reference ellipsoid and h_0 is an offset of the sensor . The error measurement is then developed as follows

$$\begin{aligned} \delta h_u &= \frac{\partial h_u}{\partial h}\delta h + \frac{\partial h_u}{\partial h_0}\delta h_0, & (4.46)\\ &= \delta h + \delta h_0. & (4.47) \end{aligned}$$

$$\delta h_u = \left([0,\ \ 0,\ \ 1]\ \ \mathbf{0}_{1\times3}\ \ \mathbf{0}_{1\times3}\ \ \mathbf{0}_{1\times3}\ \ \mathbf{0}_{1\times3}\ \ 0\ \ 0\ \ 1]\right)\delta\boldsymbol{x}. \tag{4.48}$$

4.3 Proposed Full Error Model

The full model is summarized as follows

$$\begin{pmatrix} \delta\dot{\boldsymbol{p}} \\ \delta\dot{\boldsymbol{v}} \\ \dot{\boldsymbol{\rho}} \\ \delta\dot{\boldsymbol{b}}_a \\ \delta\dot{\boldsymbol{b}}_\omega \\ \delta\dot{p}_0 \\ \delta\dot{b}^n_m \\ \delta\dot{h}_0 \end{pmatrix} = \begin{pmatrix} \mathbf{F}_{pp} & \mathbf{F}_{pv} & \mathbf{F}_{p\rho} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{F}_{vp} & \mathbf{F}_{vv} & \mathbf{F}_{v\rho} & -\mathbf{C}^n_b & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{F}_{\rho p} & \mathbf{F}_{\rho v} & \mathbf{F}_{\rho\rho} & \mathbf{0}_{3\times3} & \mathbf{C}^n_b & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathrm{dg}(\frac{1}{\tau_a}) & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathrm{dg}(\frac{1}{\tau_\omega}) & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} \end{pmatrix} \begin{pmatrix} \delta\boldsymbol{p} \\ \delta\boldsymbol{v} \\ \boldsymbol{\rho} \\ \delta\boldsymbol{b}_a \\ \delta\boldsymbol{b}_\omega \\ \delta p_0 \\ \delta b^n_m \\ \delta h_0 \end{pmatrix}$$

$$+ \begin{pmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ -\mathbf{C}^n_b & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{C}^n_b & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{I}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1 & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & 1 & \mathbf{0}_{1\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{v_a} \\ \boldsymbol{v}_\omega \\ \boldsymbol{w}_a \\ \boldsymbol{w}_\omega \\ w_p \\ w_m \\ w_h \end{pmatrix}, \tag{4.49}$$

Based on the derivation of individual measurement models, the full mesurement model is given by

$$\boldsymbol{z} = \mathbf{H}\delta\boldsymbol{x}, \tag{4.50}$$

where

$$\mathbf{H} = \begin{pmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & [0 \;\; 0 \;\; m\cos(\zeta)\sin(\delta)] & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & 1 & 0 \\ \boldsymbol{q} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & (1-k_1h)^k_2 & 0 & \mathbf{0}_{1\times1} \\ [0\;0\;1] & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & 1 \end{pmatrix}, \tag{4.51}$$

4.4 Lever Arm Compensation

Sensors and receivers are often attached to a different locations on the navigated object rather than the center of gravity leading to additional errors. Nevertheless, the errors can be easily corrected by simple calculations.

4.4.1 GNSS Lever Arm Compensation

The GNSS systems determine the position and velocity of the GNSS receiver. Let $\delta\boldsymbol{p}^b_{\mathrm{GNSS}}$ be a vector from the center of gravity of a vehicle to GNSS receiver in body frame. The position of GNSS receiver is denoted as $\boldsymbol{p}^{LLA}$ and $\mathbf{D}$ is the transformation matrix from NED to LLA frame given by equation 3.45.

Then, the position of the navigated object $\tilde{\boldsymbol{p}}^{LLA}$ expressed in LLA frame is given by

$$\tilde{\boldsymbol{p}}^{LLA} = \boldsymbol{p}^{LLA} - \mathbf{D}\mathbf{C}_b^n \delta\boldsymbol{p}_{\text{GNSS}}^b, \tag{4.52}$$

where $\delta\boldsymbol{p}_{\text{GNSS}}^b$ is a deviation vector from vehicle's center of gravity to GNSS receiver antenna.
The velocity of a vehicle measured by GNSS receiver is not influenced by the lever arm effect.

4.4.2 IMU Lever Arm Compensation

Since the IMU is usually placed outside of the vehicle center of gravity, lever arm correction can also be applied to IMU measurements. The equation for specific force correction is calculated as follows [38]

$$\tilde{\boldsymbol{f}}^b = \boldsymbol{f}^b - \boldsymbol{\Omega}_{nb}^b \boldsymbol{\Omega}_{ib}^b \delta\boldsymbol{p}_{\text{IMU}}^b, \tag{4.53}$$

where $\delta\boldsymbol{p}_{IMU}^b$ is a vector from the center of gravity to IMU.

Similar correction is done for angular rate measurements given by the following equation [38]

$$\tilde{\boldsymbol{\omega}}^b = \boldsymbol{\omega}^b - \frac{\delta\boldsymbol{p}_{\text{IMU}}^b \times \delta\boldsymbol{v}^b}{|\delta\boldsymbol{p}_{\text{IMU}}^b|^2}, \tag{4.54}$$

where $\delta\boldsymbol{v}^b$ is the increment of vehicle velocity.

4.5 Inicialization process

Before the algorithm is started, the inicialization process have to be run. First of all, the inertial systems require initial conditions such as initial position and initial velocity. During the inicialization process, the navigated object is considered to be stationary to apply following steps. The velocity is assumed to be zero. In addition, the IMU is aligned to the body frame of the navigated object.

4.5.1 Initial Alignment

The process of alignment is performed to determine the orientation of the inertial system platform and the reference navigation frame. There are several self-alignment techniques and methods. Based on the knowledge of the gravity vector in navigation frame $\boldsymbol{g}^n$ and given the accelerometer measurements $\boldsymbol{f}^b$, it is possible to compute the roll φ and pitch θ angle. It holds [11]

$$\boldsymbol{f}^b = -\mathbf{C}_n^b \boldsymbol{g}^n, \tag{4.55}$$

where $\boldsymbol{g}_n = [0,\, 0,\, g\ ]$ and g is the gravity acceleration.

Given the set of equations 4.55, it is straightforward to show that the angles can be computed as follows

$$\begin{aligned}
\varphi &= \text{atan2}(-f_y^b, -f_z^b) && (4.56)\\
\theta &= \text{atan2}\left(f_x^b, \sqrt{(f_y^b)^2 + (f_z^b)^2}\right) && (4.57)
\end{aligned}$$

The yaw ψ angle is estimated using the magnetometer measurements and the computed pitch and roll angles. First, the measured values are transformed to horizontal plane as follows [29]

$$\begin{pmatrix} m_x^h \\ m_y^h \\ m_z^h \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta)\sin(\varphi) & \sin(\theta)\cos(\varphi) \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ -\sin(\theta) & \cos(\theta)\sin(\varphi) & \cos(\theta)\cos(\varphi) \end{pmatrix} \begin{pmatrix} m_x^b \\ m_y^b \\ m_z^b \end{pmatrix}. \tag{4.58}$$

Second, the yaw ψ angle is given by

$$\psi = \text{atan2}\left(-m_y^h, m_x^h\right) \pm \delta, \tag{4.59}$$

where δ is the declination angle in radians.

Chapter 5
Design of a Dynamic Detector

The goal of the dynamic detector is to detect the dynamics of a vehicle based on the angular rate and accelerometer measurements. The data from MEMS sensors are noisy causing the classical thresholding approach on raw data without any further processing failing. Three dynamic modes are defined and they are categorized as follows

- $d[n] = 0$
  This mode signifies low dynamics meaning the vehicle is not accelerating or it is not moving at all. If inertial sensors used in the naviagtion system are not capable of measuring the Earth rotation and the vehicle is stand-still, the bias estimation is typically more accurate than in other modes. Concerning accelerometer bias estimation, the use of perfect velocity measurments also needs to be deployed. As a result, if the low dynamic mode is deteted, bias covariance values in $\mathbf{Q}$ and $\mathbf{P}$ matrices are slightly increased.

- $d[n] = 1$
  If mode 1 is detected, it signifies that the vehicle is in motion but the dynamics do not change very quickly.

- $d[n] = 2$
  Finally, this mode indicates occurrence of high dynamic mode. If the high dynamic mode is present, it is more convinient to rely only on IMU measurements since the GNSS measurements can be behindhand. Thus, the measurement matrix for position measurements is set to zeros $\mathbf{H}_p = \mathbf{0}_{3\times 3}$ and the values in $\mathbf{R}$ matrix are increased for velocity measurements.

The adaptive CUSUM algorithm (as described in Chapter 2) was chosen to detect the change of parameters μ_a and μ_ω defined as

$$\mu_a = |1 - ||\boldsymbol{a}|||, \tag{5.1}$$

for accelerometer measurements and

$$\mu_\omega = ||\boldsymbol{\omega}||, \tag{5.2}$$

for gyroscope measurements.

The threshold h was set experimentally. If the decision function G exceeds the threshold h, an alarm occurs and dynamic is detected. To distinguish the dynamic modes, an algorithm that keeps track of the maximal value of the accelerometer or gyro data based on absolute mean values is implemented. The idea is that if the high dynamic mode is present, the mean value will increase. In contrast, if no severe dynamic manoeuvres occur, the mean value will be low. Since the following algorithm is the same for both accelerometer and gyroscope data and to simplify the notation, the variable $x \in \{\mathrm{ACC},\ \mathrm{GYRO}\}$ is introduced to represent both accelerometer (ACC) and gyroscope (GYRO) measurements. The algorithm computes maximal $x^{\max}$ value and adaptively updates it. First, $x^{\max}$ is assigned as

$$x^{\max} \quad = \quad \max(\theta_x, x^{\max}), \tag{5.3}$$

where θ_x is a filtered paramter μ_x given by

$$\theta_x = \alpha\theta_x + (1-\alpha)\mu_x, \tag{5.4}$$

where α is a forgetting factor.
Finally, new value of $x^{\max}$ is computed as

$$x^{\max} \quad = \quad qx^{\max} + (1-q)\theta_x \tag{5.5}$$

where q is another forgetting factor.

If the CUSUM algorithm detectes dynamics, the maximal value algorithm then decides the dynamic mode. If the condition $x_n^{\max} > \psi_x$ is fulfilled then the high dynamic mode is present. The final decision is made according to the following conditions

```
if  $(d_a[n] == 0)$ && $(d_\omega[n] == 0)$ || $[(d_a[n] == 1)$ && $(d_\omega[n] == 0)]$
 then
 |  $d[n] = 0$
else
 |  if $[(d_a[n] == 0)$ && $(d_\omega[n] == 1)]$ || $[(d_a[n] == 1)$ &&
 |   $(d_\omega[n] == 1)]$ then
 |   |  $d[n] = 1$
 |  else
 |   |  $d[n] = 2$
 |  end
end
```

Algorithm 2: Decision conditions

The Cusum algorithm with maximal value tracking algorithm is shown in Algorithm 3. The whole detector scheme is depicted in Fig. 5.1

```
initialization;
$G[0] = 0$;
$n = 1$;
set $h, \alpha$, $\hat{\mu}$, $\psi$,$q$;
while Data is available do
    calculate;
    $\hat{\sigma}_x$;
    maximal value tracking;
    $x^{\max} = \max(\hat{\mu}_x, x^{\max})$;
    $\theta[n] = \alpha\theta[n-1] + (1-\alpha)\hat{\mu}_x$;
    $x^{\max} = qx^{\max} + (1-q)\hat{\mu}_x$;
    CUSUM RLS;
    $s = \hat{\mu}_x - \theta[n]$;
    $G[n] = \max(G[n-1] + s, 0)$;
    if $G[n] > h$ then
        if $x_n^{\max} > \psi_x$ then
            $d_x[n] = 2$;
        else
            $d_x[n] = 1$;
        end
    else
        $d_x[n] = 0$;
    end
    $n = n + 1$;
end
```

Algorithm 3: Modified CUSUM algorithm

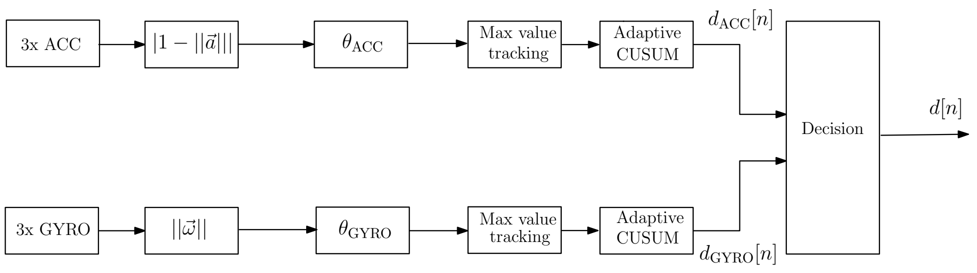


Figure 5.1: Proposed detection scheme

Chapter 6

Experimental verification

The verification of the proposed navigation algorithm was carried out on experimental data consisting of IMU measurements, GPS measurements, magnetometer and air pressure measurements. ADXRS453 sensor made by Analog devices measured the gyroscope data. Acceleration and magnetic data were obtained from the FXOS8700 sensor developed by NXP semiconductors. The IMU unit is sampled at 400 Hz and the GPS has a sampling frequency of 10 Hz. The experimental data were obtained from a 30-minute flight.
As described in chapter 4, a simple FIR filter is added to the navigation solution. The filter design for the accelerometer and gyroscope data is described in this chapter. Before the whole algorithm is evaluated, the performance of the dynamic detector is studied. The navigation algorithm is then tested and analyzed.

6.1 FIR filter design

To design a FIR filter suitable for the IMU sensors, a frequncy domain analysis was carried out. The accelerometer x-axis data were chosen to represent the influence of higher frequency noise components. Spectrogram of the accelerometer x-axis data was plotted. As can be seen in Figure 6.1, after around 2 minutes, higher frequencies in the spectrogram appears. This frequencies are probably induced by engine vibration since the plane engine was started after approximately 2 minutes. It is desirable to eliminate these frequencies since it can affect the performance of the inertial navigation system.

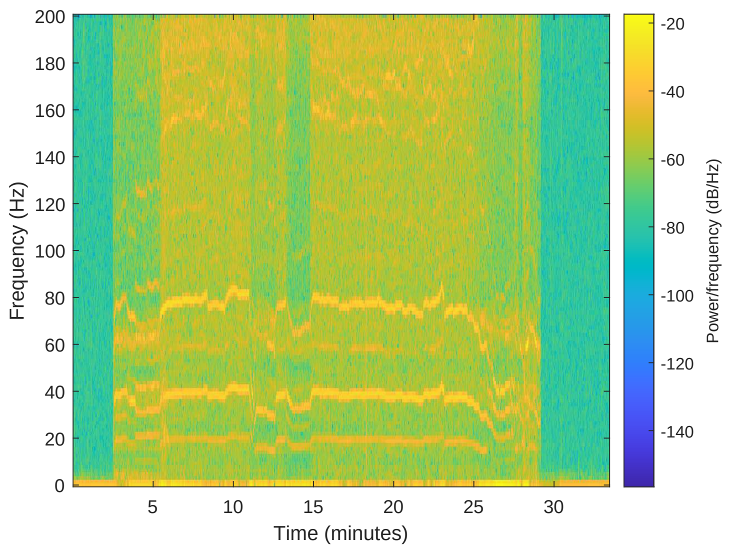


Figure 6.1: Spectrogram - x-axis accelerometer - before filtering

To filter out these higher frequencies, a simple low-pass FIR filter was designed. The filter has an order of 40 and the cut-off frequency was set to 7 Hz. The magnitude response of the filter is depicted in Figure 6.2

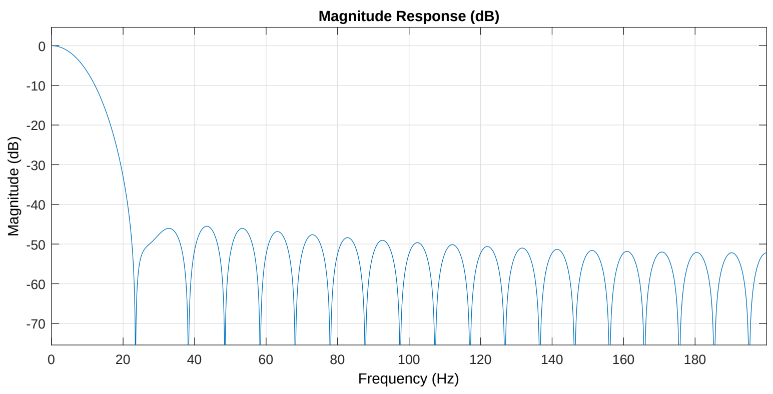


Figure 6.2: Magnitude response

After the filtration is applied, the spectrogram of the filtered data is depicted in Figure 6.3. As it can be seen, the unwanted frequencies were suppressed.

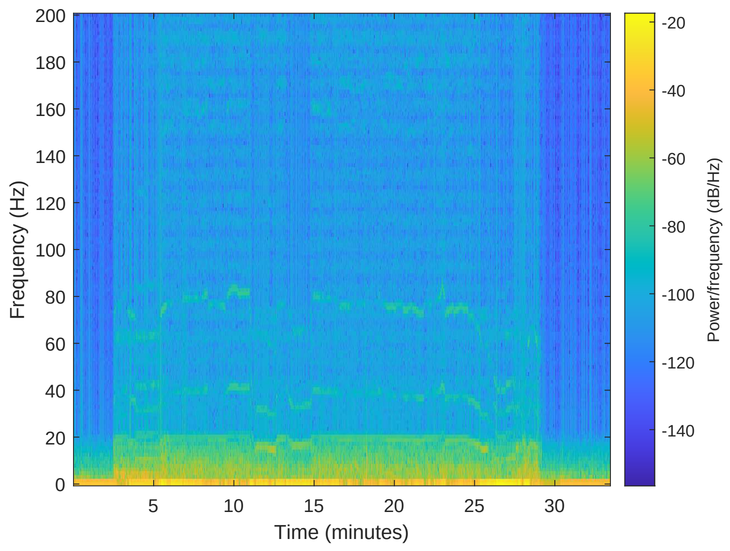


Figure 6.3: Spectrogram - x-axis accelerometer - after filtering

6.2 Dynamic detector performance

The performance of the dynamic detector was tested on the experimental data. The detector has five designing parameters that have to be set. First off, the attention is paid to the maximal value tracking algorithm, where the parameter setting is essential. The parameter q corresponds to the adaptation speed of the maximal value tracking algorithm. The adaptation should be neither too slow nor too fast. If the q parameter value is set to high, the adaptation is slow and the dynamics is not detected in time.
In contrast, if the adaptation constant q is small, the algorithm reacts very fast to signal changes and it cannot correctly keep track of the maximal value. The incorrect setting of the parameter ψ_x can lead to misclassification between medium and high dynamic mode. The forgetting parameter α is calculated as follows

$$\alpha = \frac{D-1}{D}, \tag{6.1}$$

where D indicates the length of a sliding window.
Finally, the parameter h determines the presence of dynamics. For instance, high values of h can lead to not detecting any dynamics at all. A detection analysis was carried out to get an idea about the parameter values. Figure 6.4 shows the detected occurrence of dynamic mode (meaning mode 1 was detected) depending on the threshold h_x for both accelerometer and gyroscope data. The Figure 6.4 also illustrates the idea of not detecting any dynamic if the h_x value is set to high. After further inspection, the h_x threshold was set to 10 for both accelerometer and gyroscope since the performance (based on subjective evaluation) seemed to be the best.

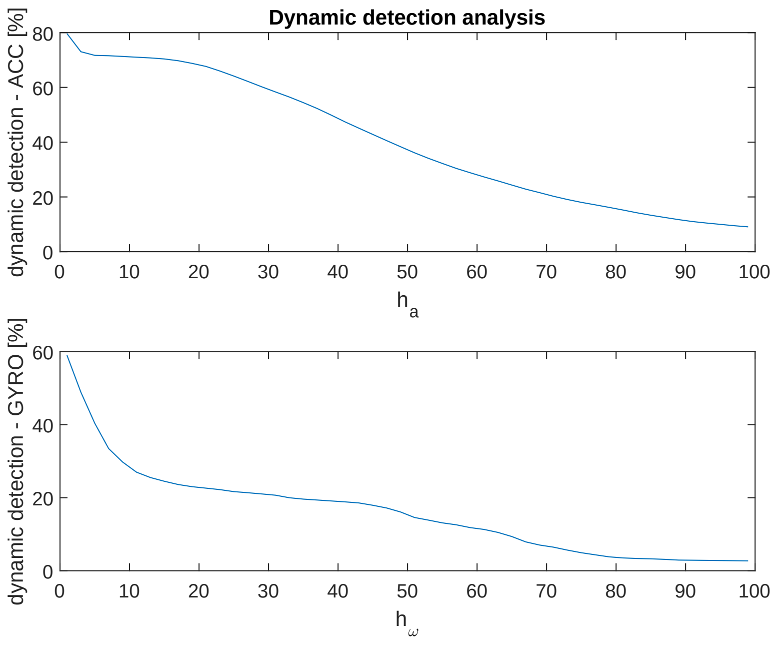


Figure 6.4: Detection occurrence analysis with respect to threshold h

After the h_x threshold was set , second analysis, depicted in Figure 6.5, was performed to determine the parameter ψ_x. The analysis shows the occurance of high dynamic mode with respect to the threshold ψ_x.

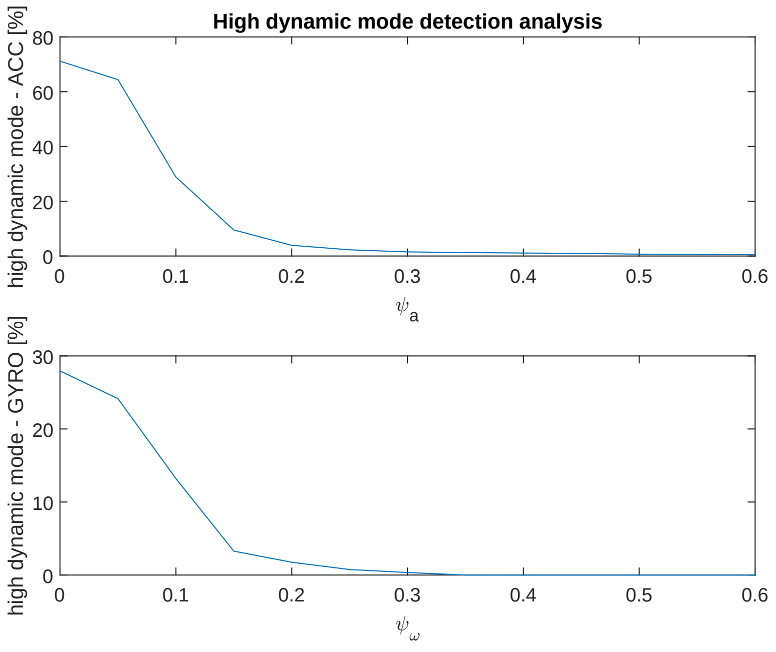


Figure 6.5: High dynamic mode occurrence analysis with respect to threshold ψ

Finally, the constants were set according to Table 6.1.

	q	D	α	h	ψ
ACC	0.999	800	0.99875	10	0.17
GYRO	0.999	800	0.99875	10	0.14

Table 6.1: Dynamic detector - parameter settings

Figure 6.6 shows the zoomed-in detail on accelerometer data and the decision function when dynamic is present. As it can be seen from Figure 6.6 and 6.7, the decision function G intensifies parts where the acceleration or angular velocity values are higher and it suppresses the parts when only noise appears.

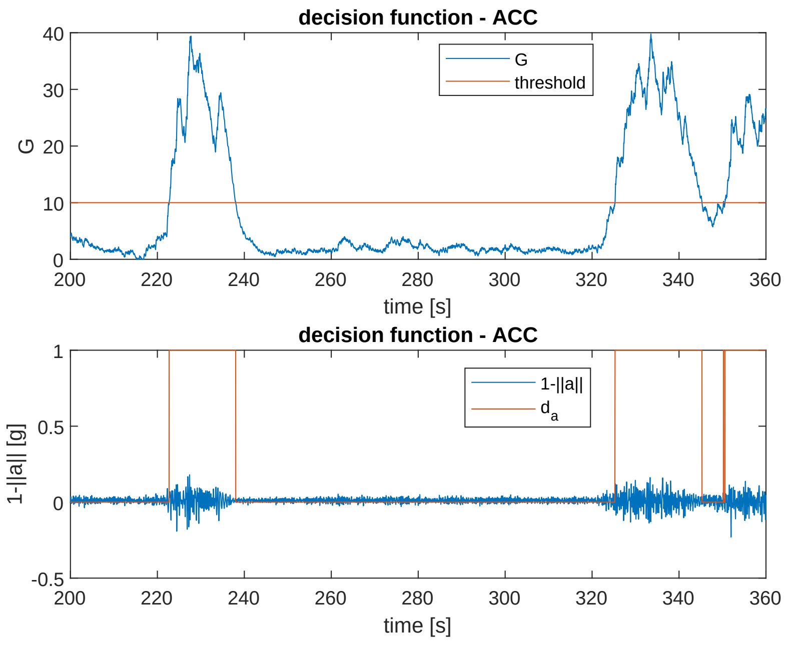


Figure 6.6: Decision function G - acc data

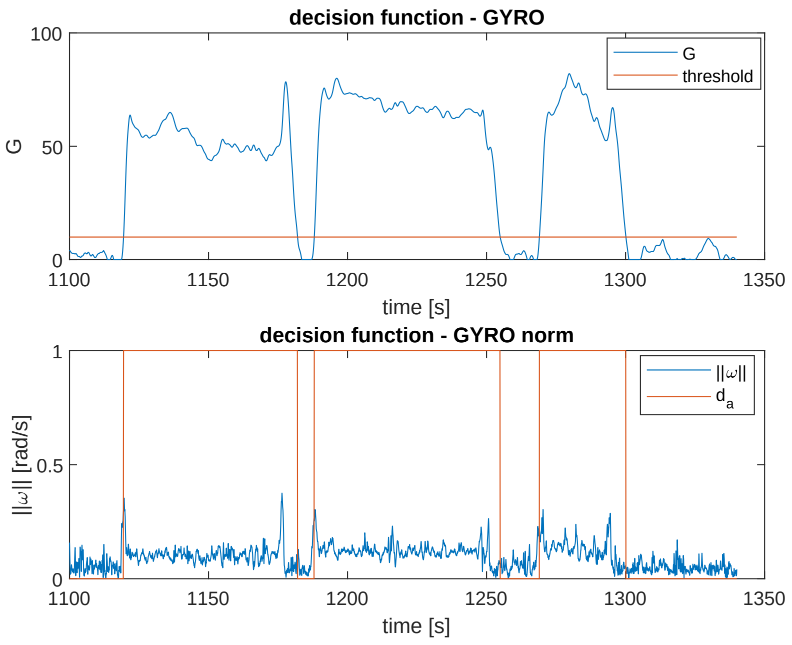


Figure 6.7: Decision function G - gyro data

The maximal value tracking function of accelerometer is depicted in Figure 6.8.

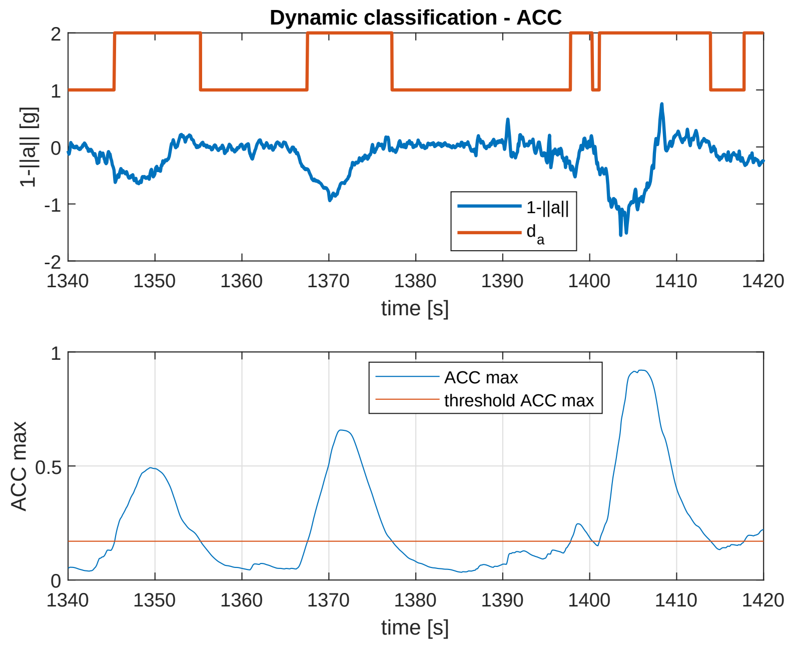


Figure 6.8: Maximal value tracking - acc data

The overall detection performance is compared to GPS velocity measurements and evaluated for two different flights with the same sensors and parameters used. The results can be seen in Figure 6.9 and 6.10. Parts of the flight where GPS velocity significantly changes in a short time period have mostly been marked as high dynamic mode whereas if the GPS velocity is close to zero or parts where velocity is constant, the low-dynamic mode was detected.

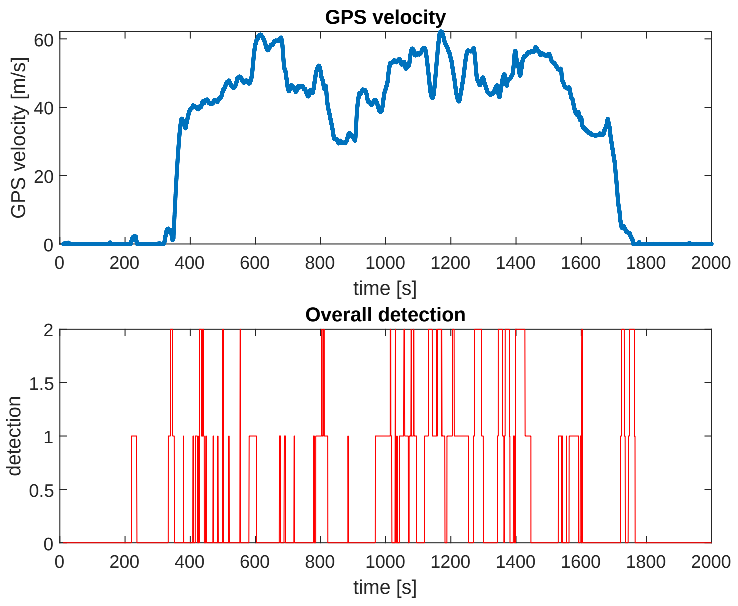


Figure 6.9: Dynamic detection - flight 1

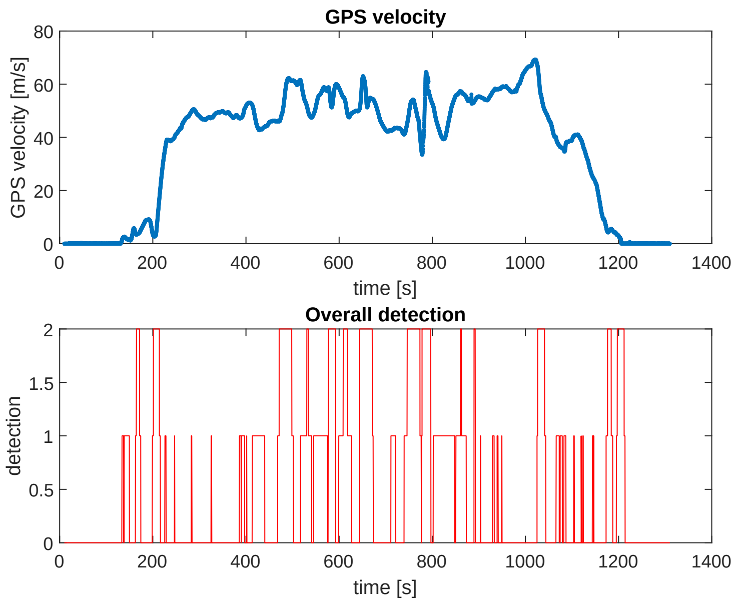


Figure 6.10: Dynamic detection - flight 2

The detector correctly labels parts where the high dynamic is present. For instance, the occurrence of high dynamic mode was correctly labeled during the second flight around 800 seconds. In general, the performance is satisfactory.

6.3 Navigation algorithm testing

In this section, the navigation algorithm (NA) is tested on experimental data obtained from real flight measurements. The data contain measurements from IMU unit, GPS receiver, magnetometer and absolute air pressure sensors. The covariance matrix $\mathbf{R}$ was set as follows

$$\mathbf{R} = \operatorname{diag}\begin{pmatrix}0.00009^2 & 0.00009^2 & 0.9^2 & 0.01^2 & 0.01^2 & 0.007^2 & 10^2 & 15^2\end{pmatrix} \tag{6.2}$$

The correct computation of yaw angle from magnetometer depends on the proper hard and soft effect compansation as described in chapter 2. The compansation was done using matlab function *magcal()*. Following parameters were obtained

$$\mathbf{A} = \begin{pmatrix}0.795002576548202 & 0 & 0\\ 0 & 1.034732800172445 & 0\\ 0 & 0 & 1.215635145970239\end{pmatrix}, \tag{6.3}$$

$$\boldsymbol{b}_m = \begin{pmatrix}75.879835788037 & -22.082233975800 & 25.877416874555\end{pmatrix}\mu\mathrm{T}. \tag{6.4}$$

The magnetic data before and after correction are depicted in Figure 6.14.

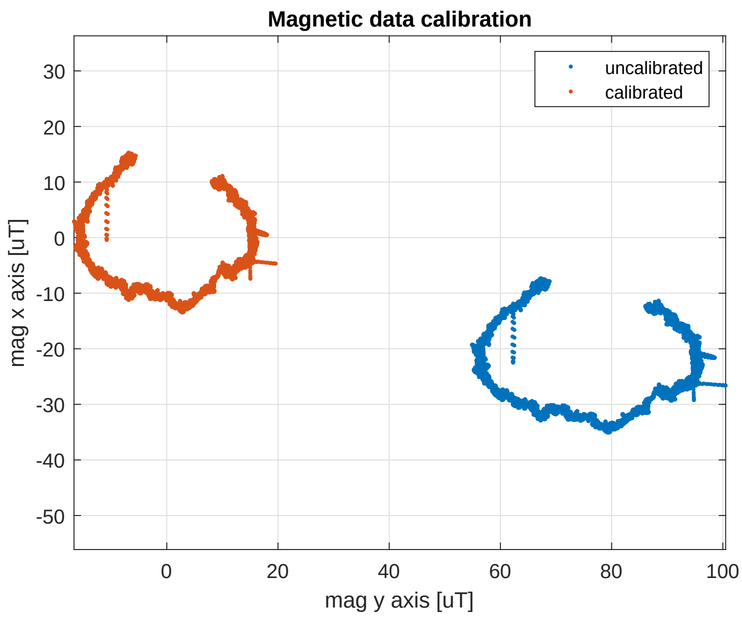


Figure 6.11: Magnetic data calibration

Figure 6.12 shows the position estimation in NED coordinate system and it is compared to measured GNSS position. The velocity estimation is depicted in Figure 6.13 and the comparison with GPS velocity measurements is shown as well.

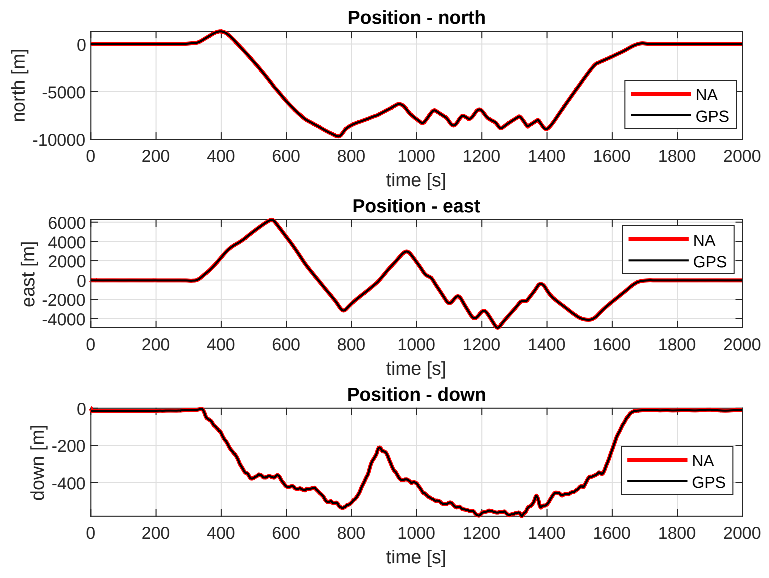


Figure 6.12: Position estimation
NA - Navigation algorithm

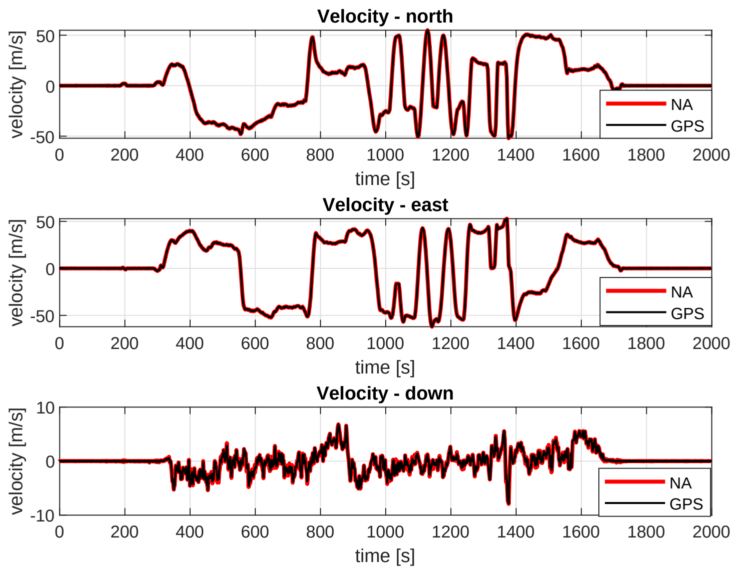


Figure 6.13: Velocity estimation
NA - Navigation algorithm

The estimates of Euler angles - roll, pitch and yaw, are displayed in Figure 6.14. To compare the results, the Euler angles are also calculated only from angular rate measurements. In addition, the roll and pitch can be estimated from accelerometer measurements. It is essential to mention that the computation of roll and pitch from accelerometer data may be inaccurate during high dynamic modes due to external influences. However, estimation of Euler angles by integrating angular rates can lead to an uncontrolled error output, as can be seen in Figure 6.14 at the end of the flight. The yaw angle is also calculated from magnetometer measurements.

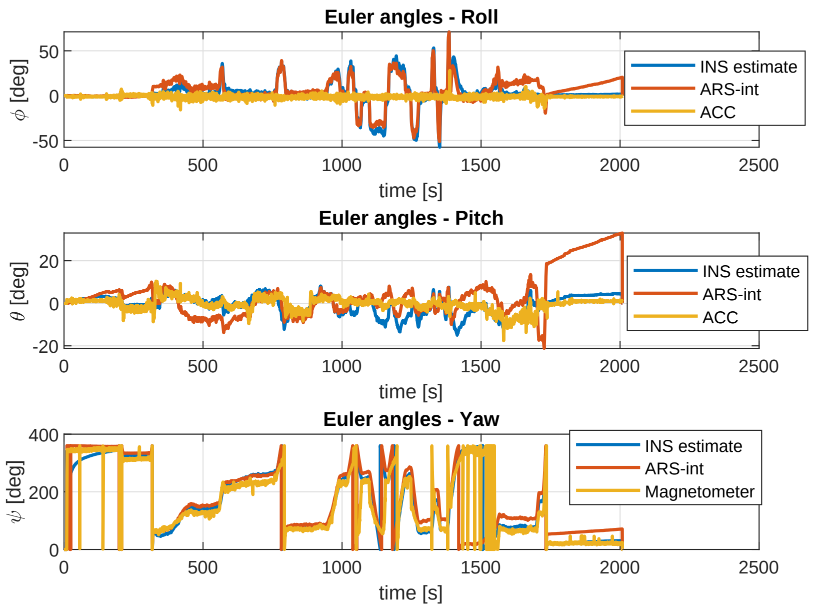
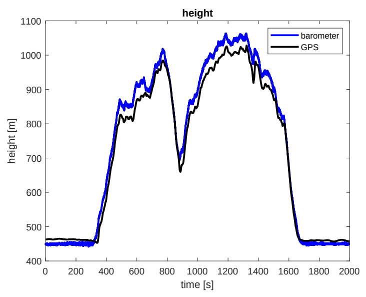
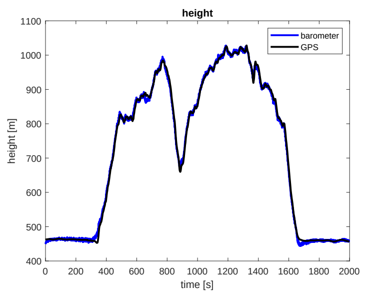


Figure 6.14: Euler angles estimation

The reference pressure p_0 is updated and estimated. Figure 6.15 depicts the barometric altitude estimates with and without the reference pressure correction. The Figure 6.15 also shows the importance of p_0 estimation. In the case of setting the reference pressure p_0 as a constatnt at the beginning, the difference between the true altitude and the barometric altitude changes during the flight.



(a) Altitude without p_0 estimation



(b) Altitude with p_0 estimation

Figure 6.15: Barometric altitude estimation

Accelerometer and gyroscope bias estimation is shown in Figure 6.16 and 6.17. The estimation is most likely influenced mainly by high dynamics.

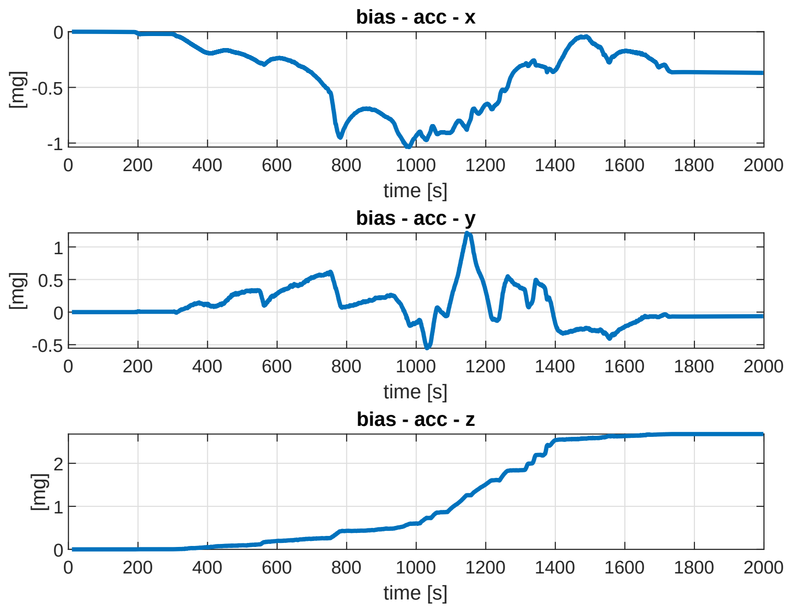


Figure 6.16: Accelerometer bias estimates

To determine the functionality of the algorithm, the diagonal elements of the Kalman gain matrix for position and velocity are depicted. The elements cannot exceed the value of 1. If the measurements are trusted, then the Kalman gain value approaches to 1. On the contrary, the measurements are not taken into account when the value is close to 0. The algorithm sets the Kalman gain to zero in position and it lowers the Kalman gain in velocity when the high dynamic is detected as it can be seen in Figure 6.18 and 6.19. Also, the standard deviation of position and velocity representing uncertainty increases during the high dynamic mode, as depicted in Figure 6.20 and 6.21. That is expected behavior since the GNSS measurements are not trusted, so

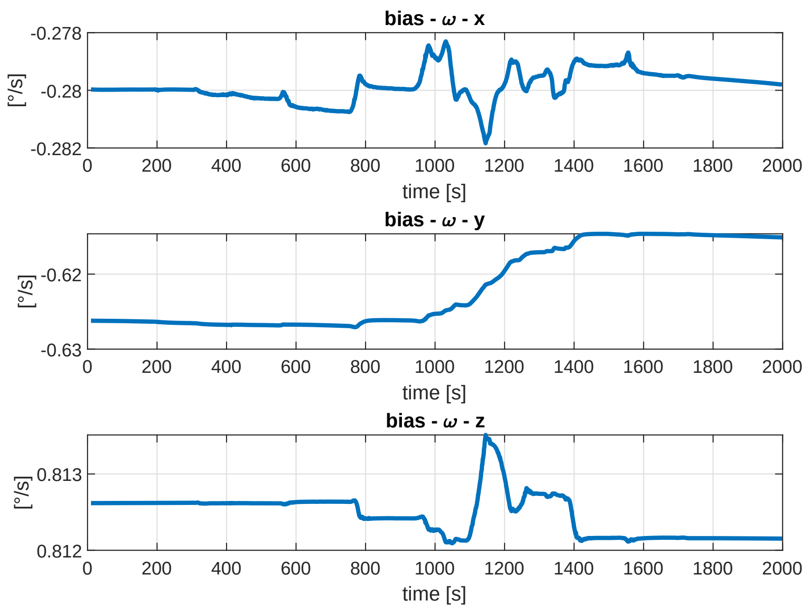


Figure 6.17: Gyro bias estimates

the Kalman gain should be smaller, however, the uncertainty increases.

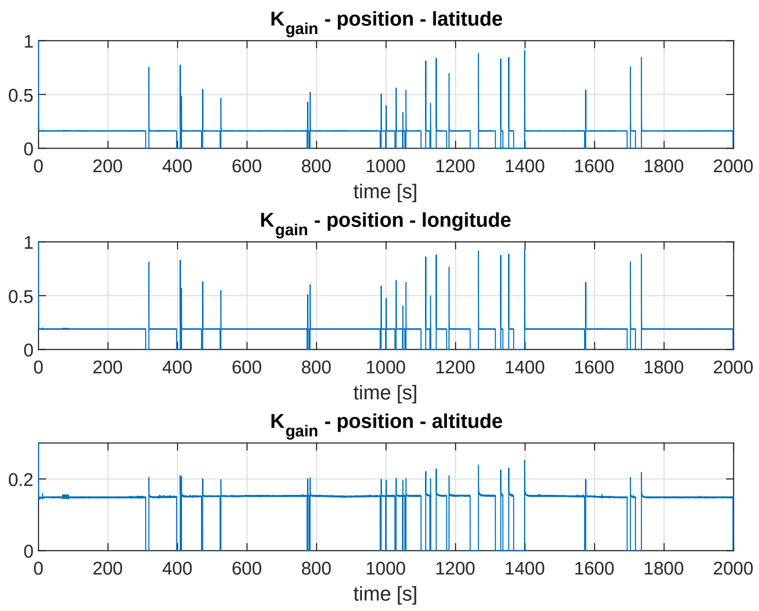


Figure 6.18: Kalman gain - position

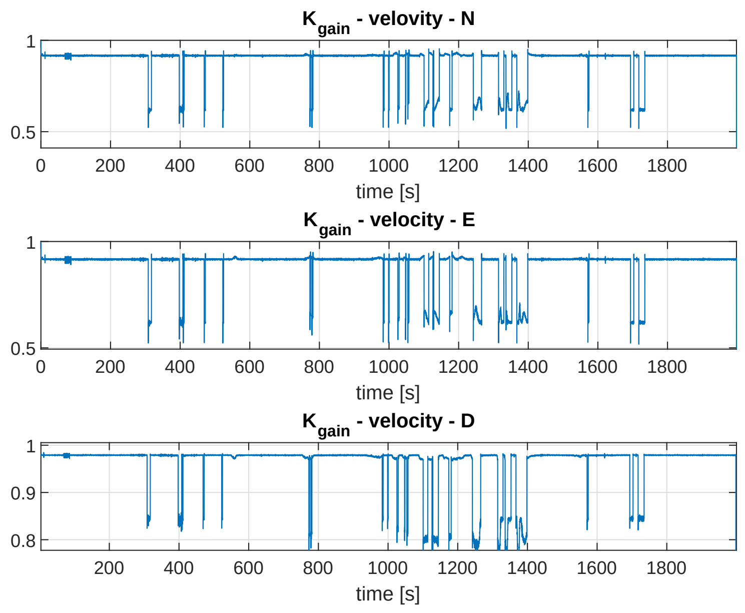


Figure 6.19: Kalman gain - velocity

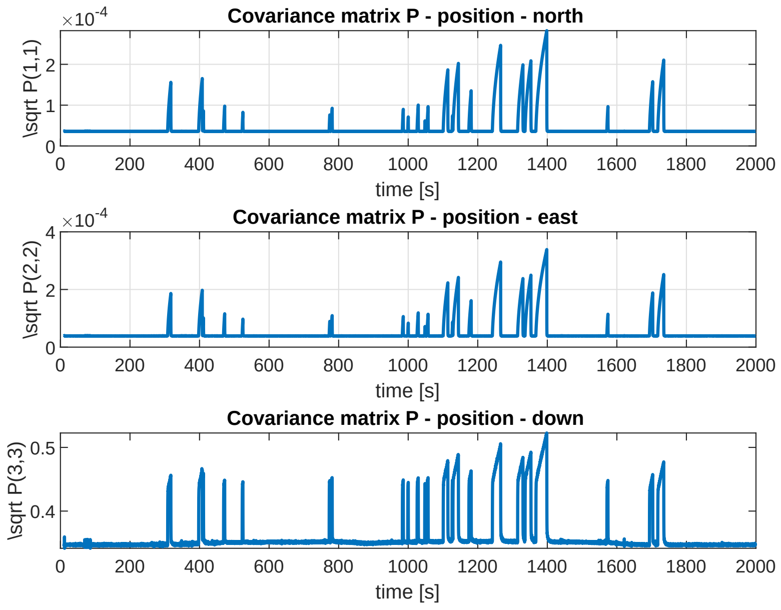


Figure 6.20: Standard deviation - position

To test that the algorithm is capable of estimating the position for a while, even without the GNSS aiding, twenty-five second GNSS outage was artificially added twice to the experimental data. The result is depicted in Figure 6.22.

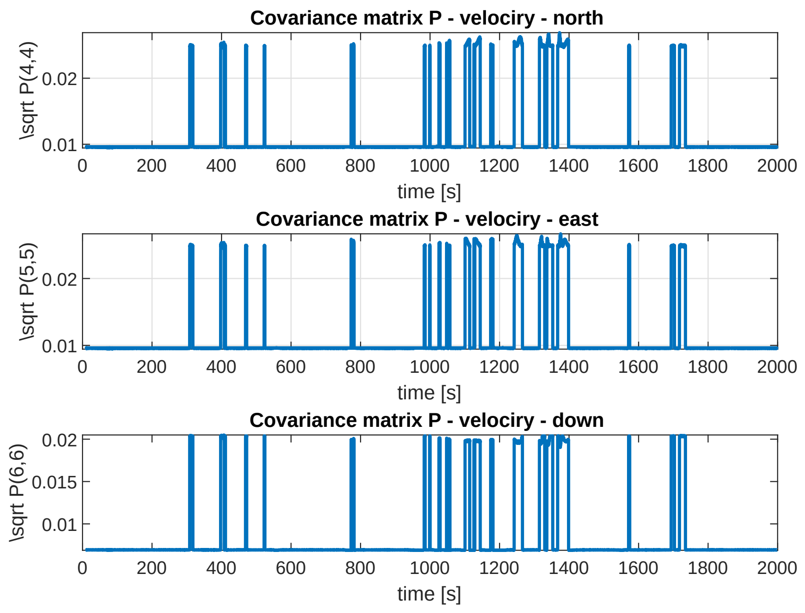


Figure 6.21: Standard deviation - velocity

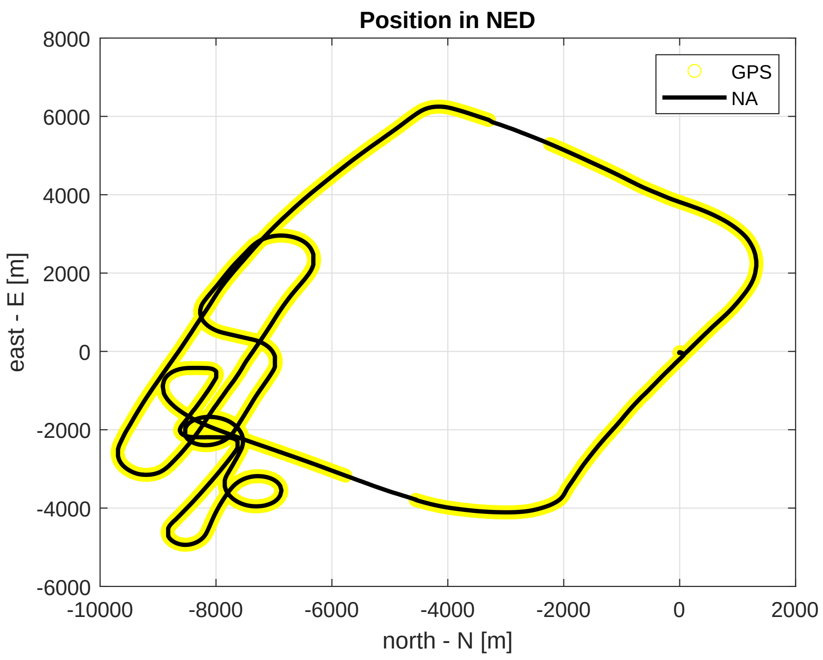


Figure 6.22: GNSS outage

The error in position caused by the GNSS outages is depicted in Figure 6.23.

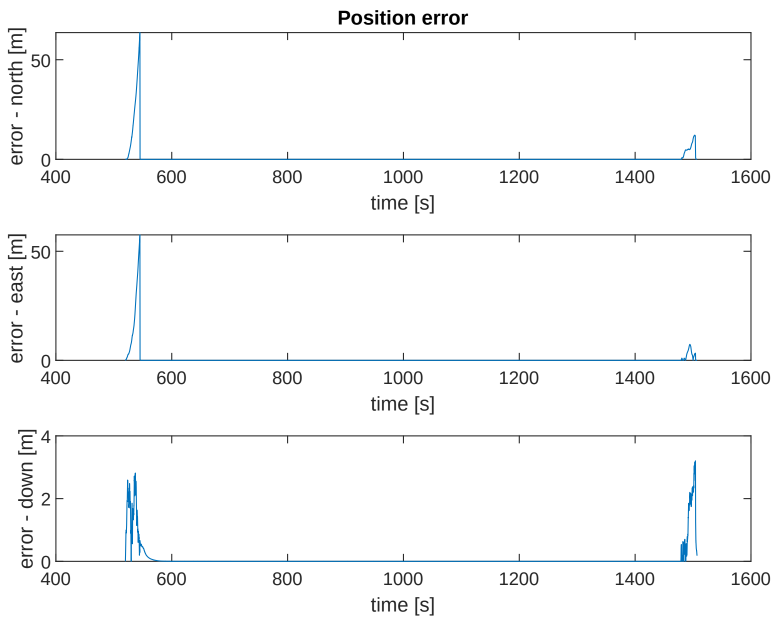


Figure 6.23: GNSS outage - error

The down position axis representing altitude is not significantly influenced by the GNSS outage since the pressure sensor helps to estimate the altitude as well. The first GNSS outage resulted in a quite significant error reaching above 50 meters both in north and east axes. On the other hand, The navigation solution performed well during the second GNSS outage
The analysis has showed that the navigation algorithm can estimate the position, velocity and Euler angles. The aiding sensors help the algorithm to adapt and correct the errors caused by the noise in IMU measurements. However, when the algorithm is not aided by GNSS, the performance of the position estimation decreases. It can probably be caused by the worse bias estimation for accelerometer sensor and higher noise components in the sensor outputs.

Chapter 7

Real-time testing

The Navigation unit 11-800 was used to test the algorithm in a real-time environment. It consists of low-cost MEMS inertial measurement unit DMU11 providing accelerations and angular rates in three axes. The navigation unit also contains GPS receiver MB800 and the main processing is done by STM32F745Z6 microcontroller running up to 216 MHz and made by STMicroelectronics [41]. The IMU measurements were sampled at 100 Hz and the GPS measurements were obtained at a sampling frequency of 10 Hz. The data were saved to SSD logger through Controller Area Network (CAN) based communication protoco l[41]. The measurement unit is shown in Figure 7.1.

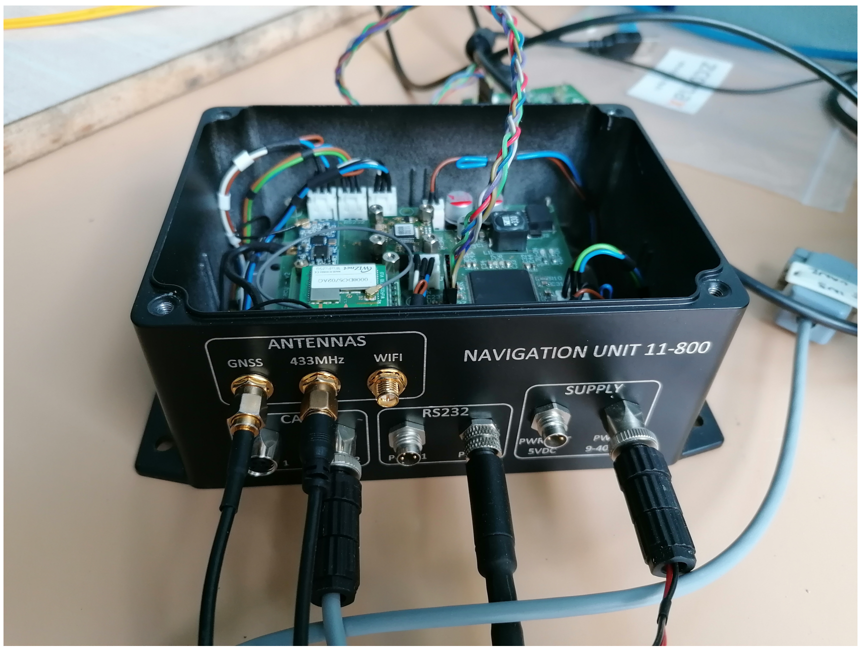


Figure 7.1: Navigation Unit 11-800

After the navigation solution was analyzed in Matlab on experimental data, the algorithm was converted to C language and implemented into a navigation unit. Before the real-time test was carried out, the C code was verified on an off-line simulation. The navigation unit was placed on the roof of a car to verify the implemented algorithm, as depicted in Figure 7.2. The experiment was repeated twice. During the first attempt, data were collected and parameters were adjusted. Results from the second experiment are shown in this section. Before the navigation algorithm itself is started, inicialization process takes place to determine the initial position and calculate the DCM matrix. The ride took about twenty minutes.

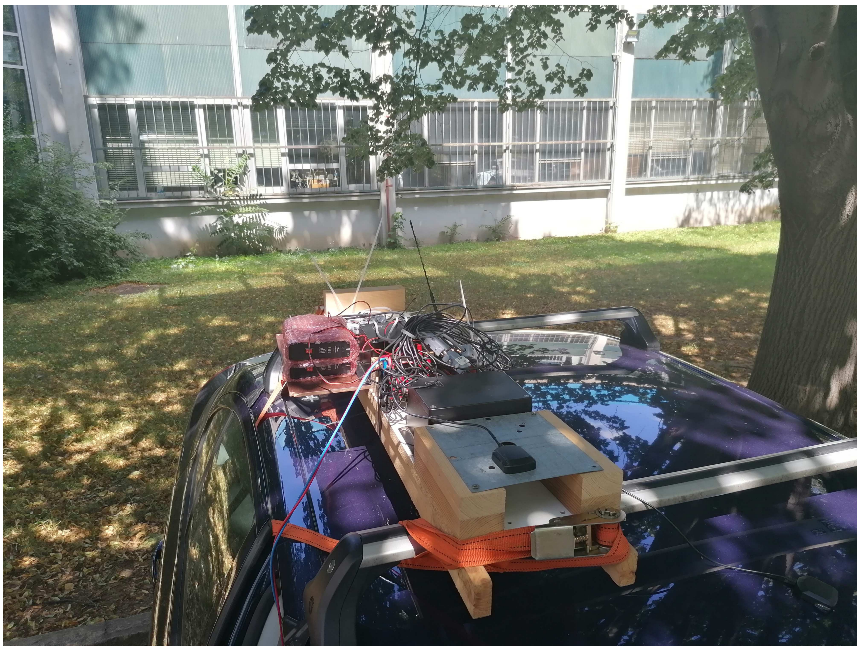


Figure 7.2: Navigation unit attachment

7.1 Allan Variance

Allan variance analysis was carried out on the IMU sensors to determine their noise characteristics. The data were collected during twenty-hour measurement, where the navigation unit was stabilized and unmoved. The Allan analysis of the gyroscope sensor indicates satisfactory properties as specified in the datasheet. However, the accelerometer analysis revealed poor characteristics in the x and z-axis. The bias instability occurs in less than 1 second, which is inadequate for INS applications. The data were collected once more during ten-hour measurements to confirm the results, but the outcome was very similar. The Allan deviation for gyroscope and accelerometer can be seen in Figure 7.3 and 7.4, respectively.

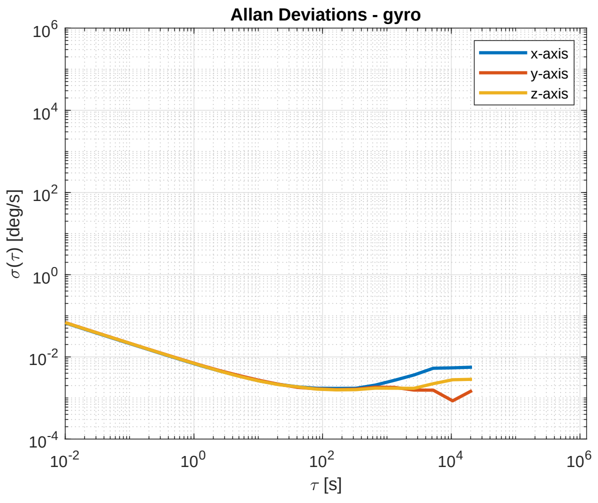


Figure 7.3: Allan deviation - gyroscope

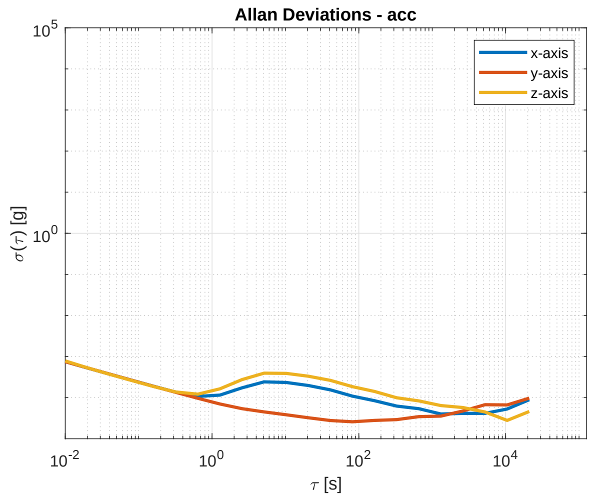


Figure 7.4: Allan deviation - accelerometer

7.2 FIR filter design

Since the IMU sampling frequency in the navigation unit differs from the dataset used for the navigation algorithm testing in Matlab, a new simple low-pass filter was designed. The order of the filter was set to 15 and the cut-off frequency is 5 Hz. The magnitude response of the filter is depicted in Figure 7.5

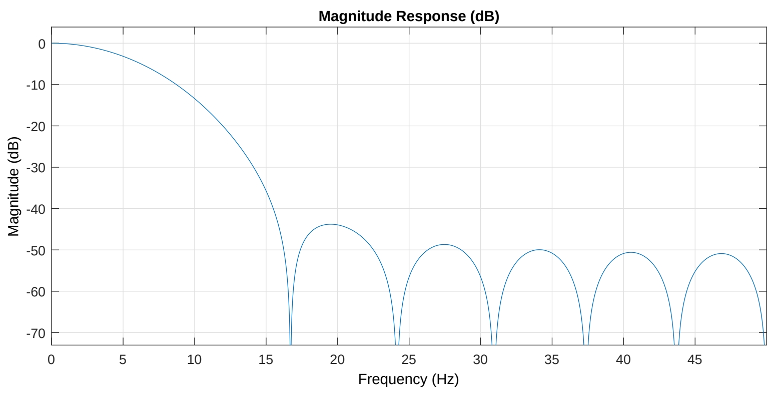


Figure 7.5: Magnitude responce

7.3 Results

The traveled route calculated by Matlab, Navigation unit (NU) and GPS measurements is shown in Figure 7.6.

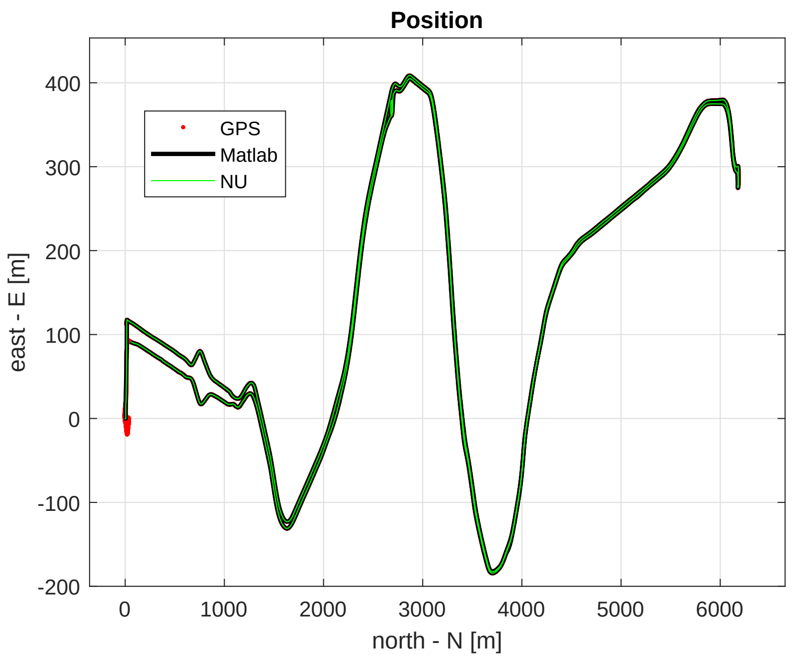


Figure 7.6: Route traveled

Estimation of Euler angles by navigation unit as well as by Matlab processing is compared in Figure 7.7. The results shows that the Euler angles computed by Matlab as well as by navigation unit are essentially the same.

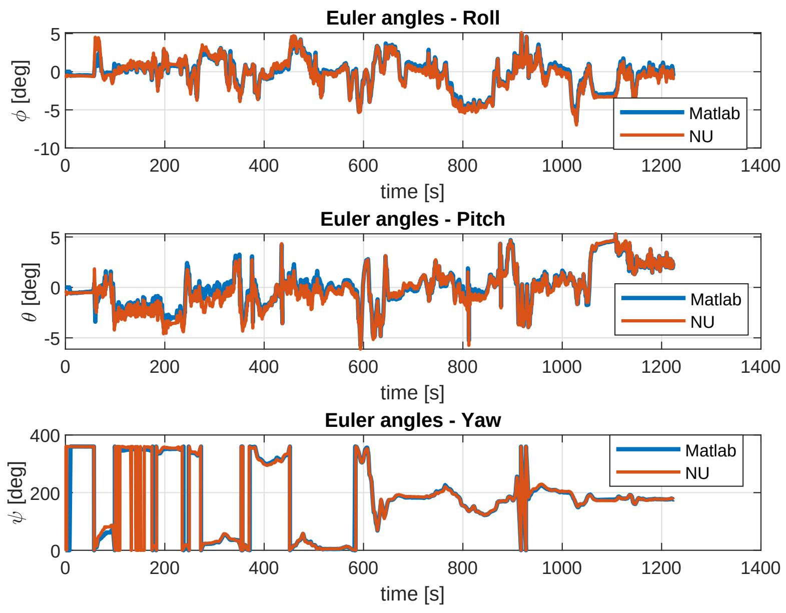


Figure 7.7: Euler angles

The route contained a traffic roundabout. To test the yaw estimation, the roundabout was circled twice. As it can be seen in Figure 7.8, the yaw angle indicates the car turning of 360° twice in a row.

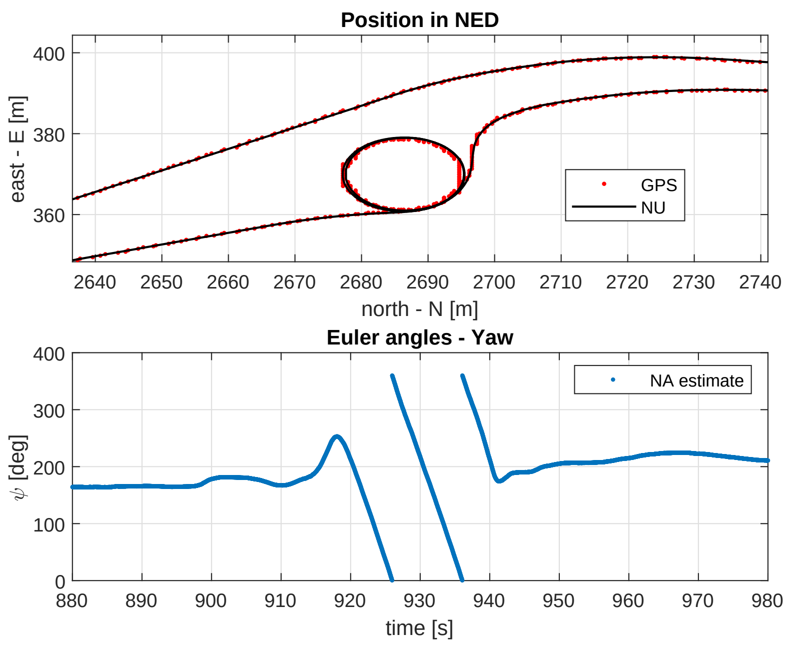


Figure 7.8: Yaw estimation - roundabout

The innovations must follow the characteristic of white noise. It provides good feedback on correct parameter settings. The innovations for position and velocity are depicted in Figure 7.9 and Figure 7.12. As can be seen, the white noise property is mostly fulfilled.

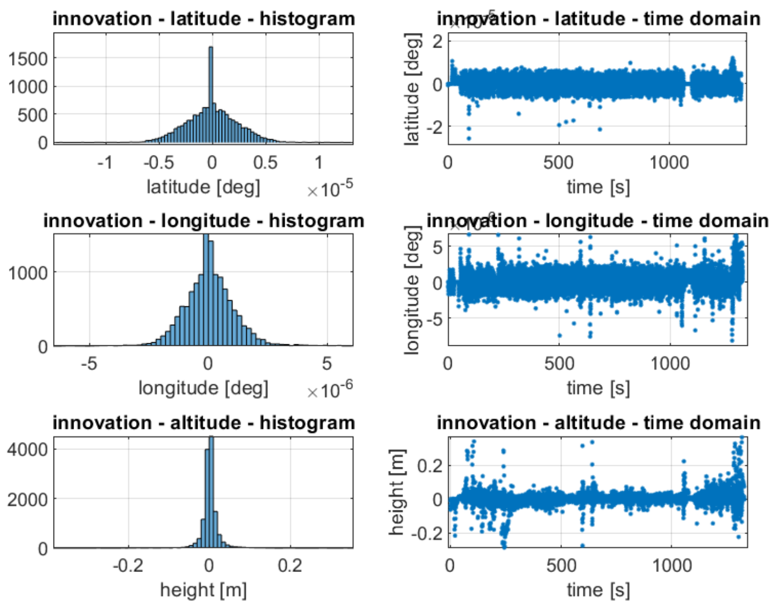


Figure 7.9: Innovation - position

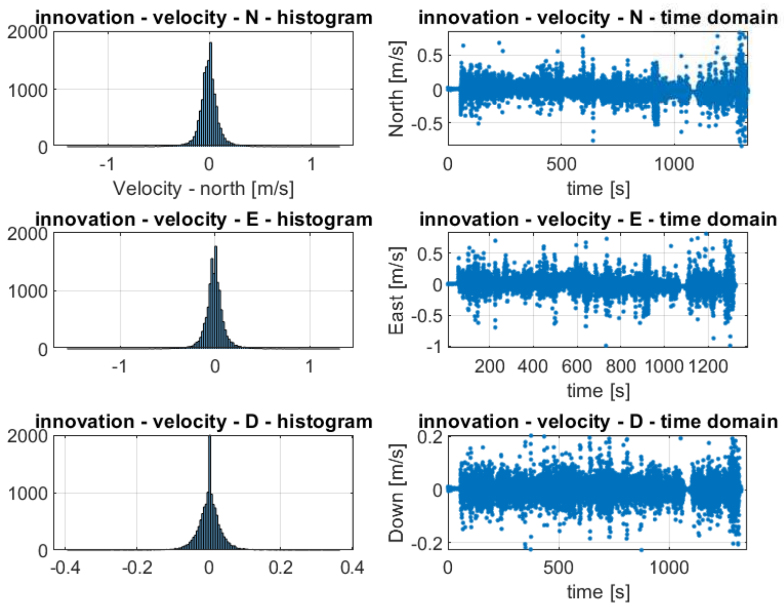


Figure 7.10: Innovation - velocity

An error between the results from Matlab processing and Navigation unit calculations was computed to compare and verify the obtained results. The position error and the velocity error in NED frame can be seen in Figure 7.11 and 7.12, respectively .

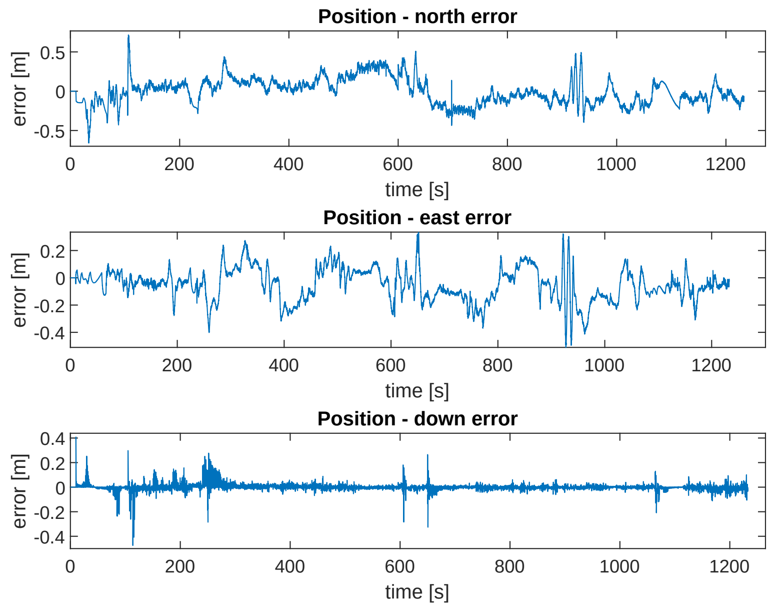


Figure 7.11: Position error - Matlab and navigation unit (NU) comparison

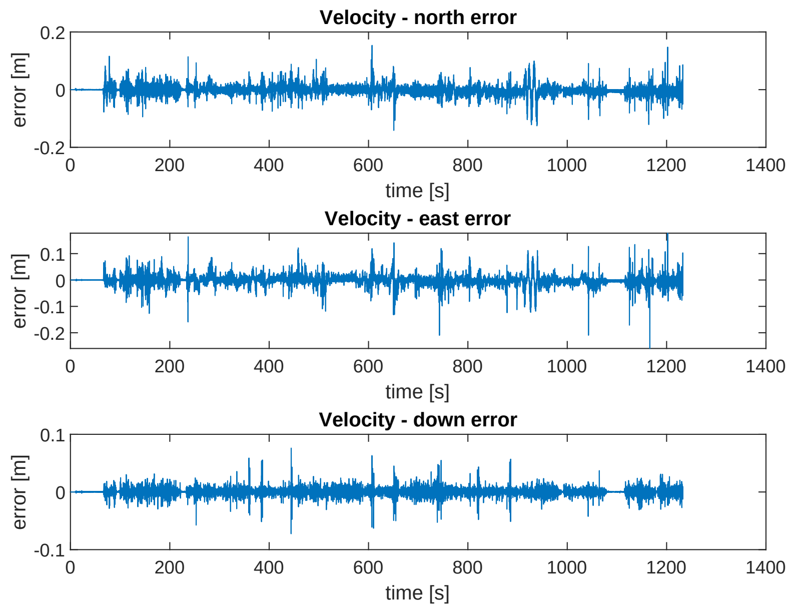


Figure 7.12: Velocity error - Matlab and navigation unit (NU) comparison

The small deviations between the Matlab processing and Naviagtion unit results can be caused by incorrect setting of initial GPS position in Matlab processing since the algorihtm works in LLA frame and the initial position setting plays a significant role.
In general, the algorithm works as expected.
Once more, the GPS outage was added to the data for 12 seconds to determine the self-sufficiency of the inertial solution.

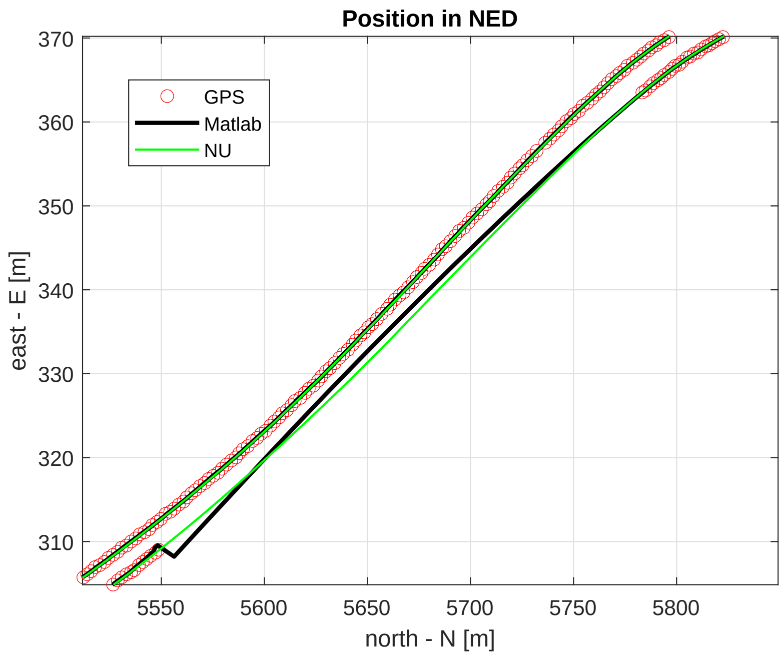


Figure 7.13: GNSS outage simulation

The position error was calculated and it is depicted in Figure 7.14. The GNSS outage occurs at 675 second. As it can be seen, the algorithm is able to estimate the position for about 5 second before it starts to diverge from the true position mainly in the north axis.

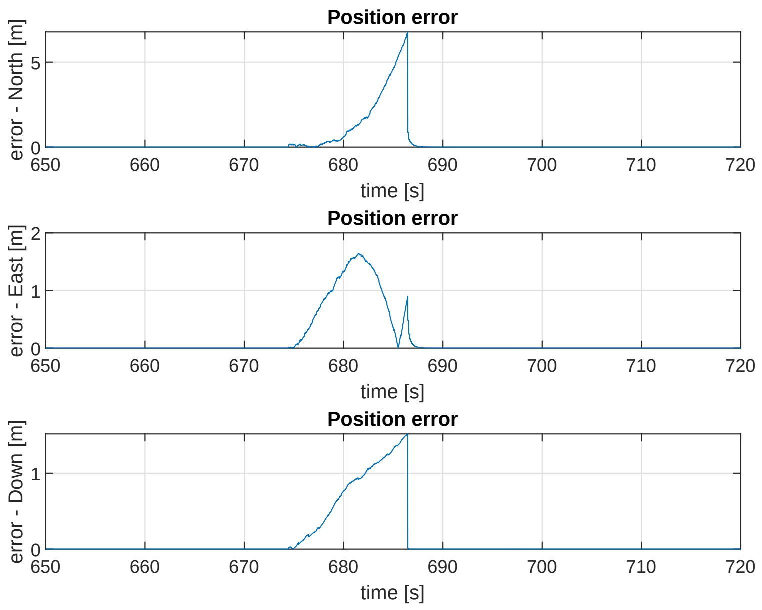


Figure 7.14: GNSS outage - error

The estimation error without GPS aiding is quite satisfactory. There are several reasons why the algorithm starts to fail after a few seconds. It can be caused by worse accelerometer noise charachteristic revealed by the Allan variance analysis. Moreover, the inertial sensors have not been calibrated before the experiment and the sensor error models such as non-orthogonality of the 3-axes sensor have not been taken into account. These factors can worsen the position estimation of the inertial navigation algorithm. However, when the GNSS is available for correction purposes, the algorithm works correctly.

Chapter 8
Conclusion

The goal of the thesis was to design a navigation solution to estimate the position, velocity and orientation of a vehicle. The proposed algorithm applies error modeling and Kalman filtering taking data from IMU measurements, GNSS receiver, magnetometer, absolute pressure sensor and ultrasonic sensor. In the beginning, the introduction to navigation principles was given. Sensors used in the algorithm were described as well as the basic theoretical knowledge of the Kalman filter. Then, various reference frames were described to introduce the navigation algorithm and mechanization equations were derived. The basic navigation solution was extended to utilize also aiding sensors as GNSS receiver, absolute pressure sensor, magnetometer and ultrasonic sensor. The error state-space model was described and the navigation algorithm was proposed, including an initial alignment process and lever arm compensation. Then, the dynamic detector was designed based on the adaptive CUSUM filtering algorithm.

To analyze the navigation solution's performance, simulations were carried out on experimental data that included measurements from IMU, GNSS receiver, absolute air pressure sensor, and magnetometer. The simulations confirmed that the algorithm estimates the position, velocity and vehicle orientation. It was also verified that if a GNSS outage occurs, the navigation solution is able to keep track of the position for a few seconds. However, the performance gets worse if longer GNSS outage is present. The dynamic detector correctly detects low dynamic modes where the vehicle is not moving. It also satisfactorily recognizes quick maneuvers and labels them as high dynamic modes. The dynamic detector design was introduced in a conference paper in a student conference Pegasus with the 4th place out of 32.

The algorithm was then converted to C language and implemented into a navigation unit. The communication and control functions were already implemented. The main part was to add the navigation algorithm. To investigate the noise parameters of the IMU unit, Allan variance analysis was carried out. The accelerometer showed worse noise characteristics, not matching the datasheet parameters. Final verification was done in a car drive where the navigation unit was attached to the car roof. The comparison of results from the navigation unit and from the off-line Matlab simulation shows similar results. The minor differences may probably be caused by the incorrect setting of the initial GPS position in MATLAB calculations. However, the overall performance of the algorithm works as expected.

Further work could focus on transforming the navigation solution to quaternion representation. The main problem of the direction cosine matrix approach is that the solution fails near the poles since the calculations contain a tangent function of latitude. The attention could also be paid to the dynamic detector to develop an algorithm for automated parameter setting.

Bibliography

[1] A. K. Brown. GPS/INS uses low-cost MEMS IMU. *IEEE Aerospace and Electronic Systems Magazine*, 2005

[2] Barbour, Neil M. Inertial Navigation Sensors. 2010

[3] Wallischeck, Eric. GPS Vulnerabilities in the Transportation Sector, 2016

[4] Oliver J. Woodman. An introduction to inertial navigation. *Technical report*, 2007

[5] P. Aggarwal, Z. Syed, X. Niu and N. El-Sheimy. A Standard Testing and Calibration Procedure for Low Cost MEMS Inertial Sensors and Units. *University of Calgary*

[6] Isaac Skog, Peter Handel. CALIBRATION OF A MEMS INERTIAL MEASUREMENT UNIT.

[7] Alison K. Brown. TEST RESULTS OF A GPS/INERTIAL NAVIGATION SYSTEM USING A LOW COST MEMS IMU *NAVSYS Corporation*, 2006

[8] EH Shin. Estimation techniques for low-cost inertial navigation. *UCGE report*,2005

[9] EH Shin. Unscented Kalman filter and attitude errors of low-cost inertial navigation systems, 2014

[10] F. Kulau, U. Gietzelt, M. and Wolf, L. Comparison and Validation of Capacitive Accelerometers for Health Care Applications. *Comput. Methods Programs Biomed.*,2012

[11] Miloš Soták and Milan Sopata and Róbert Bréda and Jan Roháč and Luboš Váci. Integrácia Navigačných sytémov. *Monografia*,2006

[12] Inertial Sensors - Autonomous Robots Lab. `https://www.autonomousrobotslab.com/inertial-sensors.html`, Accessed: 2020-04-26

[13] V. Renaudin and M. Haris Afzal and G. Lachapelle. Complete Triaxis Magnetometer Calibration in the Magnetic Domain. *Position Location and Navigation (PLAN) Group, Schulich School of Engineering, University of Calgary, 2500 University Drive N.W., Calgary, AB, Canada T2N 1N4*, 2010

[14] S. Ham and K. Kim and J. Kim and N. Min and W. Choi and C. Park. Design, fabrication, and characterization of piezoresisitve strain gage-based pressure sensors for mechatronic systems. *2015 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*, 2015

[15] W. Zhu and Y. Dong and G. Wang and Z. Qiao and F. Gao. High-precision barometric altitude measurement method and technology. *2013 IEEE International Conference on Information and Automation (ICIA)*, 2013

[16] Panda, Kirtan and Agrawal, Deepak and Nshimiyimana, Arcade and Hossain, Ashraf. Effects of Environment on Accuracy of Ultrasonic Sensor Operates in Millimeter Range. *Perspectives in Science*, 2016

[17] Naser El Sheimy, Haiying Hou and Xiaoji Niu. Analysis and Modeling of Inertial Sensors Using Allan Variance. *0018-9456*, 2008

[18] Riley, W.J. (2007) Handbook of Frequency Stability Analysis, Hamilton Technical Services.

[19] A. A. Hussen, I. N. Jleta. Low-Cost Inertial Sensors Modeling Using Allan Variance. *Engineering and Technology International Journal of Electrical and Computer Engineering Vol:9, No:5*, 2015

[20] D. Robbes, Highly sensitive magnetometers—a review, *CNRS - Groupe de Recherche en Informatique, Image, Automatique et Instrumentation de Caen*, January 2006

[21] ESA - GNSS DATA PROCESSING. *Volume I: Fundamentals and Algorithms*, 2013

[22] GNSS principles and comparison. Safaa Dawoud, 2012

[23] Keskin, Muharrem and Akkamiş, Mustafa and Sekerli, Yunus. An Overview of GNSS and GPS based Velocity Measurement in Comparison to Other Techniques. 2018

[24] Vince Kurtz and Hai Lin. Kalman Filtering with Gaussian Processes Measurement Noise, arXiv:1909.10582, 2019

[25] Montella, Corey. The Kalman Filter and Related Algorithms: A Literature Review, 2011

[26] Welch, Greg, and Gary Bishop. "An introduction to the Kalman filter." (1995).

[27] Pierre Granjon, The CuSum algorithm - a small review", hal-00914697, 2013

[28] Fredrich Gustafsson, Adaptice Filtering and Change Detection, *John Wiley*,page 69 & Sons, 2000

[29] Jay A.Farrell. Aided Navigation: GPS with High Rate Sensors. *The McGraw-Hill*, 2008

[30] Narjes Davari and Pedro Aguiar and Joao Borges de Sousa. An AUV Navigation System Using an Adaptive Error State Kalman Filter Based on Variational Bayesian. Univeristy of Porto, DOI: 10.1109/AUV.2018.8729756, 2018

[31] Popescu, Gabriel. Pixel geolocation algorithm for satellite scanner data. 2014

[32] Reference Frame Definitions (GPS),“ [Online] `http://what-when-how.com/gps-with-high-rate-sensors/reference-frame-definitions-gps/`. Accessed: 2020-05-10

[33] D. Titterton and J. Weston. Strapdown Inertial Navigation Technology. *The American Institute of Aeronautics and Astronautics, second edition*,2014

[34] Understanding Euler Angles `http://www.chrobotics.com/library/understanding-euler-angles`, Accessed: 2020-05-10

[35] John Cloud. American Cartographic Transformations during the Cold War. *Engineering and Technology International Journal of Electrical and Computer Engineering Vol:9, No:5*, 2007

[36] Houetchak, Ludovic and Kamguia, Joseph and Loudi, Yap and Louis, Foyang. Reference Ellipsoid Parameters of Cameroon from GPS Data. *International Journal of Geosciences*, 2016

[37] ECEF - Wikipedia [Online] `http://what-when-how.com/gps-with-high-rate-sensors/reference-frame-definitions-gps/`. Accessed: 2020-05-10

[38] R. JUHANT and J. KNEZ and S. BLAŽIČ. ANALYSIS OF HIGH DYNAMIC CAR MANOEUVRES USING TWO TYPES OF LEVER-ARM CORRECTION. *International Journal of Automotive Technology vol. 17*, 2013

[39] C. W. Kang and N. I. Cho and C. G. Park. Approach to direct coning/sculling error compensation based on the sinusoidal modelling of IMU signal. *IET Radar, Sonar Navigation*, pages = 527-534, 2013

[40] YİĞİTER YÜKSEL . DESIGN AND ANALYSIS OF TRANSFER ALIGNMENT ALGORITHMS . Department of Electrical and Electronics Engineering - MIDDLE EAST TECHNICAL UNIVERSITY, 2005

[41] Assoc. Prof. Ing. Jan Roháč, Ph.D. and Ing. Martin Šipoš and Ing. Petr Bojda, Ph.D. Navigation unit 11-800 - technical documentation. *Navigation group of the Laboratory of Aircraft Intrumentation Systems, Department of Measurement, FEE CTU* 2018

Appendix A

List of Matlab files

A.1 Functions

- Provided functions
  - `comp_gravity.m`
  - `deg2rad.m`
  - `EA2DCM.m`
  - `ecef2NED.m`
  - `normDCM.m`
  - `rotateArray.m`
  - `skew.m`
  - `GNSS_lever_arm_comp.m`
  - `IMU_lever_arm_comp.m`
- Created functions
  - `dynamic_detection.m`
  - `process_data.m`
  - `monitoring_variables.m`
  - `GNSS_availability.m`
  - `time_update_KF.m`
  - `measurement_update_KF.m`
  - `ReadData.m`
  - `parameters_definition_err.m`
  - `KF_params.m`

A.2 Scripts

- `main_KF.m`

Appendix B
List of C files

B.1 Functions

- `kf.c`
- `kf_error_model.c`
- `kf_error_model.h`

Appendix C

List of data files

C.1 Data

- **`flight_data.mat`**
  Data structure
  01 - counter
  02 - time [10 ms] from start of the program
  03 - ADXRS453 rate X [deg/s]
  04 - ADXRS453 rate Y [deg/s]
  05 - ADXRS453 rate Z [deg/s]
  06 - Zeros
  07 - FXOS8700 acceleration X [mg]
  08 - FXOS8700 acceleration Y [mg]
  09 - FXOS8700 acceleration Z [mg]
  10 - Zeros
  11 - FXOS8700 magnetometer X [uT]
  12 - FXOS8700 magnetometer Y [uT]
  13 - FXOS8700 magnetometer Z [uT]
  14 - Zeros
  15 - Absolute air pressure [Pa]
  16 - Temperature [deg C]
  17 - Ublox GPS time (scale in raw CAN data = 1e4)
  18 - Ublox GPS fix
  19 - Ublox GPS number of satelites
  20 - Ublox GPS latitude (scale in raw CAN data = 1e7)
  21 - Ublox GPS longitude (scale in raw CAN data = 1e7)
  22 - Ublox GPS altitude (scale in raw CAN data = 1e5) [m]
  23 - Ublox GPS speed over ground [m/s]
  24 - Ublox GPS VDOP
  25 - Ublox GPS HDOP
  26 - Ublox GPS PDOP
  27 - Ublox GPS NED velocity down [m/s]
  28 - Ublox GPS NED velocity north [m/s]
  29 - Ublox GPS NED velocity east [m/s]
  30 - Ublox GPS heading [deg]

31 - Zeros

- **`experiment_car.mat`**
  Data structure
  01 - counter
  02 - time [10 ms] from start of the program
  03 - DMU11 acceleration X [g]
  04 - DMU11 acceleration Y [g]
  05 - DMU11 acceleration Z [m]
  06 - DMU11 rate X [deg/s]
  07 - DMU11 rate Y [deg/s]
  08 - DMU11 rate Z [deg/s]
  09 - GPS time (scale = 1e3)
  10 - GPS number of satelites
  11 - GPS latitude (scale in raw CAN data = 1e7)
  12 - GPS longitude (scale in raw CAN data = 1e7)
  13 - GPS altitude above the WGS84 ellipsoid [m]
  14 - GPS speed over ground [m/s]
  15 - GPS VDOP
  16 - GPS HDOP
  17 - GPS PDOP
  18 - GPS ECEF velocity x [m/s]
  19 - GPS ECEF velocity y [m/s]
  20 - GPS ECEF velocity [m/s]
  21 - GPS heading [deg]
  22 - INS NED position N [m]
  23 - INS NED position E [m]
  24 - INS NED position D [m]
  25 - INS NED velocity N [m]
  26 - INS NED velocity E [m]
  27 - INS NED velocity D [m]
  28 - INS Euler angles - roll [rad]
  29 - INS Euler angles - pitch [rad]
  30 - INS Euler angles - yaw [rad]