Politecnico di Torino

Master's Degree in
Mechatronic Engineering

Master's Degree Thesis

Robust Estimation Methods Using Factor Graphs in GNSS Applications

Relatori
Prof. Dr. Fabio Dovis
Co-relatori
Dr. Andrea Nardin
Dr. Simone Zocca

Candidato
Pelin Baysal

Anno Accademico 2023-2024

Abstract

Achieving high-accuracy and robust navigation solutions is crucial in various domains, such as autonomous vehicles, aviation, and mobile devices. In this context, Global Navigation Satellite Systems (GNSS) play a pivotal role in modern navigation and positioning applications due to their capability of providing absolute position fixes. However, many target applications have strict safety and precision requirements, which standalone GNSS is unable to achieve in harsh environments such as urban scenarios, thus requiring improvements in terms of accuracy and robustness.

Factor graphs have proven to be a powerful mathematical framework for modeling and solving complex estimation and optimization problems such as Simultaneous Localization and Mapping (SLAM). At its core, factor graphs represent relationships between variables using nodes and factors, where nodes correspond to variables of interest, while factors encode constraints or dependencies between these variables. By leveraging the graphical structure of the problem, factor graphs exploit sparsity and modularity to break down large-scale problems into smaller, more manageable components. Furthermore, these features allow for the seamless integration of additional constraint and measurement models to implement more advanced estimation techniques. Due to their increased flexibility, factor graphs have recently emerged as an alternative method for GNSS positioning with respect to traditional methods such as Extended Kalman Filter (EKF).

Several algorithms can be found in literature that are proven to further increase the accuracy and reliability in harsh environments, implemented on top of the GNSS solution since due to the flexible structure of the factor graphs, it is possible to integrate other methods. In this thesis, two robust estimation methods called Switch Constraints (SC), and Gaussian Max-Mixtures (GMM) were implemented in order to mitigate the effects of measurement outliers in urban environments. SC utilize switch functions that act like weighting constants that handle the erroneous data and adjust their weight to be closer to zero if the data deviates from the optimal. GMM introduces bi-modal or multi-modal Gaussian distribution to help defining the characteristics of the erroneous data so that they can fit into the model better and provide better accuracy. Those two methods were implemented on MATLAB on top the already existing receiver structure, and experimental datasets were collected in an urban scenario, in Torino, near Politecnico di Torino to test the algorithms. The results have shown that the errors have been reduced thanks to the robust estimation methods of SC and GMM in urban environment, compared to the standalone factor graph framework results.

Acknowledgements

Firstly, I would like to thank Prof. Fabio Dovis for letting me work in the field of satellite navigation systems for my thesis. It was a field that I felt passionate about when I took his class last year, and I am glad that I was able to pursue this topic for my thesis.

I am grateful for the support of my co-advisors, Dr. Andrea Nardin and Dr. Simone Zocca for taking time to evaluate the work that I have done and giving me their invaluable advice whenever I needed during the research. I am also thankful for Yihan Guo's support throughout the process, his expertise in this field guided me when I was challenged. I would also like to thank all the people in NavSAS research group who helped me during the experimental data collection.

The most special thanks go to my partner Kaan Aykol, who has been always by my side no matter what. Moving to another country for my Masters studies was a huge decision for me, and I am glad that we did it together. I would also like to thank my family; especially my mother, my sister and my father; for making me feel their support even from kilometers away. I am grateful for all of my friends, especially the ones I had from Bilkent; Irmak, Deniz, Evrim, Derin, Yagiz, Arda and Ezgi; whose unwavering belief in me always made me feel encouraged to keep going. I would also like to my friends that I made and have in Italy, I believe the same hardships we faced made us closer and stronger together.

I would like to share my heartfelt thanks to anyone who has been by my side during this journey. Without your support, I'd definitely have had a much harder time facing the challenges. I hope that I made you proud with this work.

Pelin Baysal

Contents

List of Tables

List of Figures

Acronyms

B | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **L** | **M** | **N** | **P** | **R** | **S** | **T** | **U** | **V** | **W**

B

Bayesian Net Bayesian Networks. 32, 34

BOC Binary Offset Carrier. 16

BPSK Binary Phase Shift Keying. 16

C

C/A Coarse-acquisition. 15

CAF Cross Ambiguity Function. 19, 20

CDF Cumulative Distribution Function. 61

D

DCS Dynamic Covariance Scaling. 80

DFT Discrete Fourier Transform. 19

DLL Delay Lock Loop. 21

DOP Dilution Of Precision. 22, 26, 27

E

EKF Extended Kalman Filter. 60, 62, 67, 71, 73

ESA European Space Agency. 15

EUSPA European Union Agency for the Space Programme. 15

F

FDMA frequency division-multiple access. 16

FFT Fast Fourier Transform. 19

FG Factor Graph. 3, 6, 31, 34–36, 38–44, 46–57, 60, 61

FGO Factor Graph Optimization. 4, 30, 33, 62, 67, 71, 73–77

FLL Frequency Lock Loop. 21, 22

FT Fourier Transform. 19

G

GDOP Geometric Dilution Of Precision. 6, 26, 27

GEO geostationary orbit. 16

GLONASS GLObal'naya NAvigatsionnaya Sputnikovaya Sistema. 14, 16, 58

GMM Gaussian Max-Mixtures. 44, 49, 51, 54–57, 60–62, 64–67, 71, 73–76, 79

GNSS Global Navigation Satellite System. 3, 6, 13, 15–17, 19, 22, 31, 42–50, 52–54, 56–58, 66

GPS Global Positioning System. 14, 16, 17, 29, 58

H

HDOP Horizontal Dilution Of Precision. 26

I

IDFT Inverse Discrete Fourier Transform. 19

IGSO inclined geosynchronous orbits. 16

L

LBS Location Based Services. 11, 79

LOS Line of Sight. 28

LS Least Squares. 24, 27, 28, 60–62, 67

M

MEO medium Earth orbit. 15, 16

ML Maximum-Likelihood. 20

N

NCO Numerical Control Oscillators. 21

P

PDF Probability Density Function. 31

PDOP Position Dilution Of Precision. 26

PLL Phase Lock Loop. 21

PRN Pseudorandom Noise. 13, 15, 16, 19–21

PVT Position, Velocity, Time. 22, 30, 31, 57, 58, 66, 79

R

RTK Real Time Kinematic. 57, 58, 66

S

SC Switch Constraints. 6, 44, 49–54, 56, 57, 60, 62, 64, 66, 67, 71, 73–76, 79, 80

SLAM Simultaneous Localization and Mapping. 32, 49

T

TDOP Time Dilution Of Precision. 26

ToA Time of Arrival. 17

U

UERE User Equivalent Range Error. 26, 28

V

VDOP Vertical Dilution Of Precision. 26

W

WLS Weighted Least Square. 27

Chapter 1

Introduction

Accurate and reliable positioning has become a fundamental requirement in many modern applications, ranging from autonomous driving, navigation, and logistics to disaster management and precision agriculture. Global Navigation Satellite Systems (GNSS) have become the backbone of Location Based Services (LBS), providing widespread access to positioning information, and allowing users to determine their state. However, their performance may often become compromised in challenging environments, such as urban areas with signal reflections or blockages. In these situations, errors such as multipath effects, signal interference can significantly degrade the reliability and accuracy of GNSS positioning.

Traditional estimation techniques, such as least squares and Kalman filtering, are efficient under ideal conditions but struggle to handle outliers or corrupted measurements effectively. Factor graphs (FG), a powerful probabilistic modeling framework, offer an alternative way of filtering by allowing the representation of complex relationships between variables while integrating measurement uncertainties [7]. Due to their flexible structure, FGs also allow adding robust estimation methods to help decrease the localization error which could be caused by multipath or signal blockage [8].

This thesis focuses on leveraging robust estimation methods within the FG framework, which address the challenges posed by unreliable GNSS measurements Specifically, the thesis aims to show that, by exploiting proper statistical modelling of the problem, robust estimation techniques can reduce errors in the positioning solution compared to the standalone factor graph optimization (FGO) solution.

The motivation for this research arises from the increasing need for GNSS solutions capable of maintaining high accuracy and reliability levels under diverse operational conditions. In pursuit of this objective, two key distinct robust estimation methods are explored and the following original contributions are obtained from the work of this thesis:

- **Switch Constraints (SC):** Switch constraints provide a mechanism for the system to evaluate the quality of each measurement and selectively reduce the influence of unreliable data by introducing a switch variable to be optimized [8], ensuring that erroneous measurements have minimal impact on the final position estimate. A SC algorithm is implemented on top of the existing factor graph scheme.

- **Gaussian Max-Mixtures (GMM):** Most applications only utilize uni-modal Gaussian distribution to characterize the input data. However, when there is multipath or other errors, a uni-modal distribution may become insufficient to filter the erroneous data, and GMM aims to solve this problem by introducing another distribution specifically model the outliers, and selects the most likely Gaussian distribution by maximizing it [11]. A mathematical formulation of GMM has been proposed in literature and integrated onto the factor graph structure.

- **Combination of SC and GMM (SC + GMM)**: So far, it has been observed that many applications have successfully implemented SC and GMM approaches separately and obtained successful results. Nevertheless, to the best of author's knowledge, it has not been seen that a combination of both methods applied at the same time to filter a dataset. After making sure that both SC and GMM implementations are successfully implemented and meaningful results are obtained, a combination of SC + GMM filters were tested, and promising improvements were obtained.

In order to test the robust estimation methods written above, two experimental datasets were collected using a GNSS receiver: one static and one dynamic, to test the performances of the SC, GMM and SC + GMM under different conditions (stationary and moving receiver). It has been shown that significant improvements were observed compared to the plain FG solution, thanks to the robust estimation methods.

By integrating these advanced techniques, the thesis aims to contribute to the development of navigation systems that can meet the demands of emerging technologies and critical applications, providing consistent performance in challenging scenarios.

Chapter 2

Satellite Navigation Systems

A satellite navigation system consists of a network of satellites to provide geo-positioning and timing information to an end-user for it to locate itself in an absolute reference frame. If that satellite navigation system covers the whole Earth, it is called Global Navigation Satellite System (GNSS). Nowadays, many modern applications, such as transportation, autonomous vehicles, precision agriculture, and logistics; rely on the usage of maps and user location data to function effectively. Therefore, the accuracy and precision of satellite navigation systems are of utmost importance.

In this first chapter, the history of the satellite navigation systems, a brief background and working principles of GNSS and its theoretical calculations are given in details so that the thesis can be fully grasped.

2.1 Global Navigation Satellite Systems (GNSS)

A GNSS network does not only consist of satellites orbiting the Earth; but it has three different segments; space, control and user segments. Three segments work together so that the applications that utilize GNSS can function properly [1].

Space Segment

The space segment of a given GNSS network is composed of a set of satellites (i.e. constellation), orbiting the Earth at an altitude between 20,000 km and 37,000 km, usually at the medium Earth orbit. The satellites transmit signals that contain information about their position, velocity, orbit, and identification code that is related to the satellite (i.e. Pseudorandom Noise (PRN) code). All the satellites in the constellation are expected to be synchronized via a stable atomic clock so that there would be no discrepancy between the transmission of the signal.

Control Segment

Control segment consists of stations on the ground that monitor the satellites in space. Specifically, the control segment is composed of three different stations: data uploading stations, master stations and monitor station. Data uploading stations are responsible for sending the updated data to the satellites, monitor stations are responsible for tracking the health, orbital status of the satellite data and finally the master station processes the data that is coming from the satellites, updates the satellite orbits and the time scale if necessary.

User Segment

The user segment consists of the devices that have GNSS receivers. The bodies could be transportation vehicles (such as ships, automobiles, aircrafts etc.), mobile devices, drones, or even standalone GNSS receiver boards. The users are able to receive signal from the satellites, which contain information regarding position, velocity and time info so that the users can determine their location, which is called positioning.

In Figure 2.1, one can see all three GNSS segments and their communication scheme. It should be noted that there is a bidirectional communication between the control segment and the space segment of GNSS, whereas there is only one way of communication between the users and the space segment; from satellites to users. The users cannot send a signal to satellites, also there is no communication between the control stations and the users.

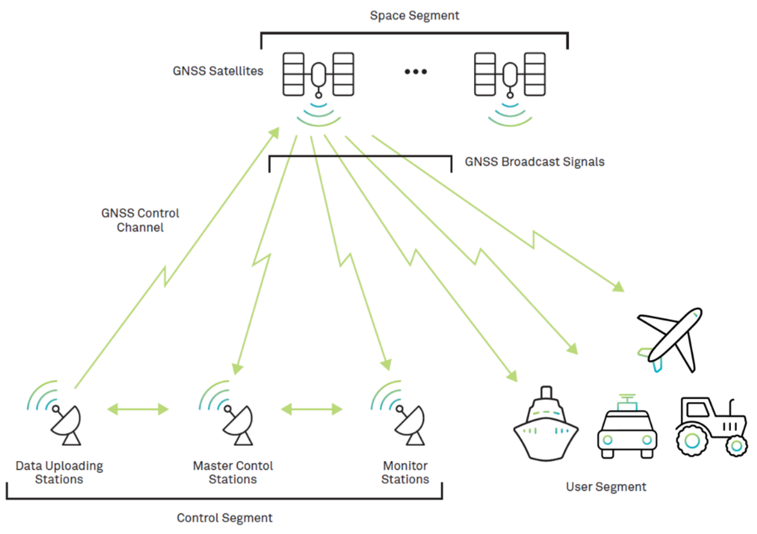


Figure 2.1: GNSS Architecture, that consists of space, control and user segments. [1]

2.2 GNSS Constellations

Currently, there are four satellite constellations that have global coverage; namely Global Positioning System (GPS), GALILEO, GLObal'naya NAvigatsionnaya Sputnikovaya Sistema (GLONASS) and BeiDou, developed by different regions around the world. Their brief histories are given below.

GPS

GPS is the GNSS system owned by United States of America. The GPS project started to be developed in 1973 by the U.S. Department of Defense and the constellation that consist of 24 satellites became fully operational in 1993. Even though it started as a system reserved for the military use, then it also became public and opened to civilian and commercial use. As of today, current number of usable GPS satellites are 31, which include currently used and spare ones [12]. Each GPS satellite's period is roughly 12 hours, thus rotates around the Earth twice a day and flies in the medium Earth orbit (MEO), around 20,200 km altitude [12]. All of the GPS satellites broadcast the same two frequencies: 1.57542 GHz (L1 band) and 1.2276 GHz (L2 band), and the newer GPS satellites also broadcast 1.176 GHz (L5 band) [13]. The Coarse-acquisition (C/A) PRN code, where each satellite has its own unique code to help them to be distinguished by the receiver, is for civilian use only and transmitted through L1 band. Each PRN code is orthogonal to each other, thus their cross-correlation is very low, so that the probability of signals coming from different satellites thought to be the same is very small. The C/A PRN codes have a frequency of 1.023 mega chips per second (Mchip/sec). The L2 band also transmits the P-code for military use only, whose encrypted version is called the *P(Y)* code.

One can see the GPS signal structure in Figure 2.2.

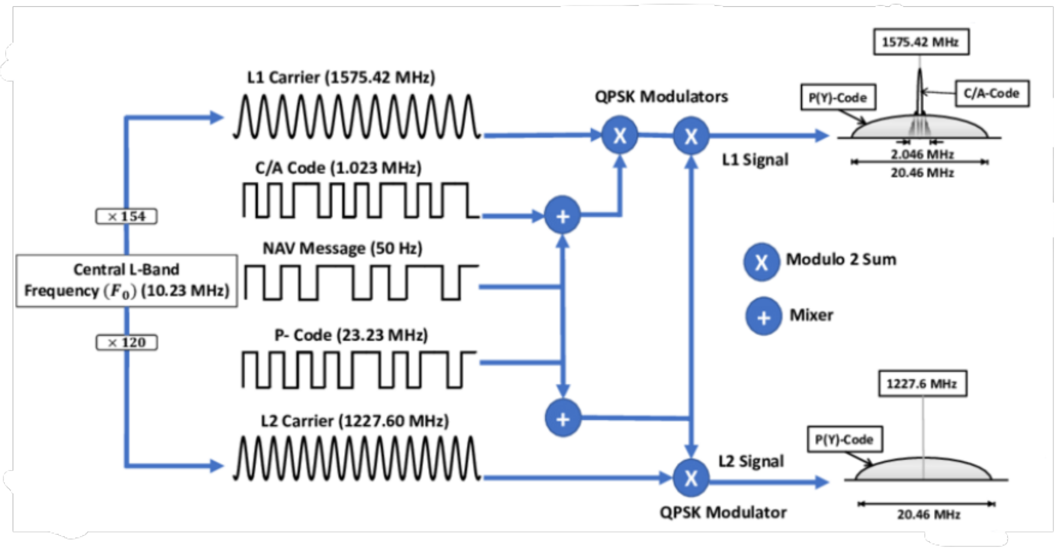


Figure 2.2: GPS Signal Structure [2]

GALILEO

GALILEO is the GNSS project developed and owned by the European Space Agency (ESA) and operated by the European Union Agency for the Space Programme (EUSPA). The GALILEO project started to be developed early 21st century and became operational in 2016 [14]. Similar to GPS, GALILEO has 30 usable satellites in space; 24 of them being currently in use and 6 of them are spare. Its orbital height is higher than the GPS', which is about 23,222 km and the orbital period is about 14 hours. GALILEO broadcasts signals in three main frequency bands: E1 (1575.42 MHz), E5 (1191.795 MHz) which consists of E5a (1176.45 MHz) and E5b (1207.14 MHz) bands, and finally E6 (1278.75 MHz) [15]. Different from the GPS' signal structure, GALILEO introduced a subcarrier for modulation, called Binary Offset Carrier (BOC), in order to reduce mutual interference among satellites.

GLONASS
GLONASS is the GNSS project of Russia, whose development began in 1976, for both civilian and military usage. It has 24 satellites in nominal use, which are located in the middle circular orbit around 19,100 km altitude, and an orbital period about 11 hours [16]. The GLONASS constellation is especially useful for high latitudes measurements, where other constellations' satellite measurements may be insufficient. Unlike GPS, all GLONASS satellites have the same PRN, but they use slightly different frequencies to be distinguished in L1 and L2 bands, called frequency division-multiple access (FDMA). The signals also are also modulated using Binary Phase Shift Keying (BPSK) similar to GPS.

BeiDou
BeiDou is the GNSS project developed and owned by China. At the beginning of the 21st century, the experimental BeiDou system with three satellites started to be developed as the first generation of BeiDou GNSS project. By 2012, second generation of BeiDou (called BeiDou-2), started to offer regional services to China and neighboring regions. The latest generation of BeiDou is the third one, called BeiDou-3, offers global coverage with 27 satellites MEO (global coverage), 3 satellites in inclined geosynchronous orbits (IGSO) (Asia-Pacific region coverage), and 5 satellites in geostationary orbit (GEO), covering China [17]. Its orbital period is approximately 13 hours.

2.3 Principles of GNSS

Since a brief history and the different constellations of GNSS throughout the globe have been discussed, one can move on to the technical aspects of GNSS, its working principles, main sources of errors.

2.3.1 Ranging and Triangulation/Trilateration

There are two most commonly used methods to determine a user's location via the satellites: which are called triangulation and trilateration [3]. In trilateration, the distances measured by satellites are used to determine the user location, whereas in triangulation the angles measured are user to determine the user location. GNSS constellations like GPS use the trilateration system for user location to be determined, whereas triangulation is usually used by surveyors in Earth applications to determine the location of an object.

In GNSS, satellites broadcast their signal that includes information related to navigation messages such as their location and time; and the GPS receiver tries to calculate its own location via the messages received from the satellites. One of the most basic but fundamental methods to calculate the range between the two objects is to use the method of Time of Arrival (ToA). The distance between the satellite (transmitter) and the receiver (GPS receiver) can be found as [18]:

$$R = c(T_{RX} - T_{TX}) = c \cdot \tau \tag{2.1}$$

If the transmission time T_{TX} is known perfectly and the received time T_{RX} is calculated by the receiver, the range can be easily calculated. Under normal circumstances, the clock of the receiver and the satellite is not perfectly synchronized, so a clock bias (b_u) occurs, but it can be fixed.

When the range between a satellite and the receiver is computed, the possible points that the receiver can be located are on the circle with the radius R whose center is the location of the satellite in 2D. So, any point that is R distance away from that satellite could be the location of the receiver. For example, if the receiver is on the blue circle around Satellite 1 in Figure 2.3 and the range is measured, no more information is gained beyond that with only one satellite. Thus, in order to find a more precise position of the receiver, signals from more satellites are required.

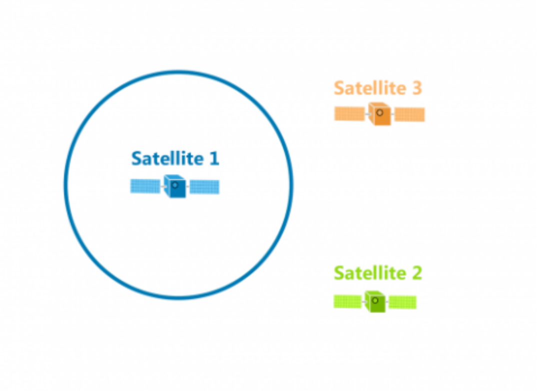


Figure 2.3: Satellite trilateration in 2D. Measurement from Satellite 1 is done and receiver is estimated to be on the blue circle. [3]

If it is assumed that the receiver's coordinates are in 2D, as in Figure 2.4 (right), and two circles intersect, so there could be two intersection points, which are the possible locations of the receiver. This is an improvement considering when there was only one satellite, as in Figure 2.3, where infinite number of points could be the receiver's location on the blue circumference.

Nevertheless, in order to be able to obtain a unique solution in 2D, at least 3 satellites are required. If the satellites are properly positioned around the receiver, the intersection of three measurements could be at only one point, which corresponds to the user position, as in Figure 2.4, right.

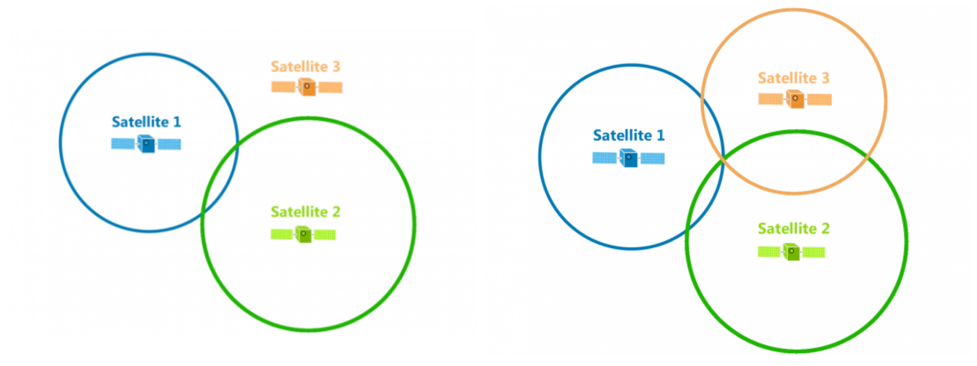


Figure 2.4: Satellite trilateration in 2D. The intersection of two satellite measurements (left) and the intersection of three satellite measurements (right). [3]

With three satellites in 2D as in Figure 2.4, the receiver can pinpoint its exact location via trilateration.

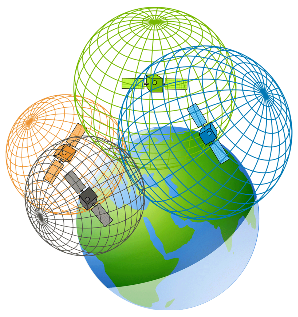


Figure 2.5: Satellite trilateration in 3D. [3]

Now, if one considers 3D coordinate system, the satellites broadcast signals as if they are in the center of a sphere. Thus, in order to obtain a a unique intersection point of the spheres so that the exact location of the receiver could be estimated, at least 4 satellites are needed, so that the trilateration algorithm could work. One could check Figure 2.5 for the intersection of four satellites.

Now that the trilateration procedure is given, one can move onto the acquisition and tracking processes that the receiver performs in order to obtain and track signals from the GNSS satellites.

2.3.2 Acquisition

In the acquisition process, the aim is to find from which satellites the receiver can obtain measurements (detection), and estimate roughly the delay and the Doppler frequency shift of the signal. Without determining which satellites are available, the signal tracking process becomes extremely difficult.

In order to acquire which GNSS satellites are in visibility, the correlation functions are used by the receiver. The GNSS receiver generated the local replicas of the satellite PRNs, and performs the correlation with the incoming signal to be determined whether it contains the specific PRN. Since the input signal is constantly received, circular correlation is generally performed which is also related to Discrete Fourier Transform (DFT), that is extensively used in signal processing [19]. The normalized circular correlation $R_{x,y}$ of signals $x[n]$ and $y[n]$ with length L can be expressed as:

$$R_{x.y}[m] = \frac{1}{L}\sum_{n=0}^{L-1} x[n]y[n+m] \tag{2.2}$$

If the satellite PRN that has been searched is observed in the received signal, an aurocorrelation value emerges from the rest of the correlated values on the plot. The relation between circular correlation function $R_{x,y}[m]$ and DFT is explained as:

$$IDFT\left\{X[k]Y^{*}[k]\right\} = R_{x.y}[m] \tag{2.3}$$

where Inverse Discrete Fourier Transform (IDFT) bridges the relation between circular correlation of the signals time domain and their frequency domain functions, $X[k]$ is the Fourier Transform (FT) of the signal $x[n]$ in the time-domain, $Y^{*}[k]$ is the conjugate of the FT of the signal $y[n]$ in the time-domain. Thus, the entire correlation function in the acquisition stage can be obtained by Fast Fourier Transform (FFT) operations.

The general form of the received signal from the j^{th} satellite can be expressed as [19]:

$$x_j[n] = AD(nT_s - \tau^D)C(nT_s - \tau)cos(2\pi(f_{IF} + f_d)nT_s + \varphi) \tag{2.4}$$

where A is the signal amplitude, T_s is the sampling frequency, $D(nT_s - \tau^D)$ is the navigation data with delay τ^D, $C(nT_s - \tau)$ is the PRN code with delay τ, φ is the phase of the received signal, f_{IF} is the intermediate frequency. The first step in the acquisition phase is to estimate the delay τ and the Doppler shift f_d. In order to achieve that, a search space is needed. There are usually 1023 code phases for τ and $\pm 5kHz$ to search in the frequency bin (sufficient for all ground applications). Considering the fact that for a stationary receiver, the Doppler shift can have a value for up to $\pm 5kHz$. A Cross Ambiguity Function (CAF) is formed to observe the peak at the estimated delay $\hat{\tau}$ and frequency $\hat{f}_d$. The formula for the CAF is as follows [20]:

$$R_{y,r}(\bar{\tau}, \bar{f}_d) = \sum_{n=0}^{L-1} y_{IF}(nT_s)c_i(nT_s - \bar{\tau})e^{i2\pi(f_{IF}+\bar{f}_d)nT_s} \tag{2.5}$$

where $c_i(t)$ is the PRN sequence of the i^{th} satellite that is being searched in the CAF function. The resulting search space is in 3D, where x-axis represents the delay bins $(\bar{\tau})$, y-axis represents the frequency shift $(\bar{f}_d)$, and the z-axis represents the squared modulus of the $R_{y,r}(\bar{\tau}, \bar{f}_d)$, where the relationship can be expressed as:

$$S^2_{y,r}(\bar{\tau}, \bar{f}_d) = |R_{y,r}(\bar{\tau}, \bar{f}_d)|^2 \tag{2.6}$$

An example CAF plot is given in Figure 2.6.

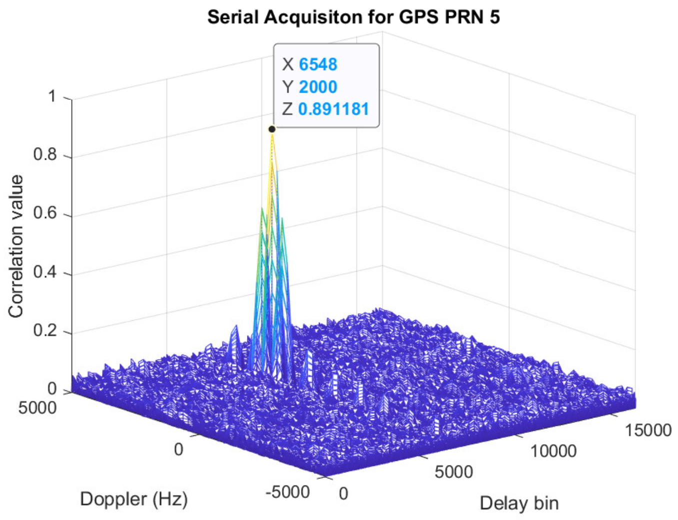


Figure 2.6: 3D CAF for GPS PRN 5. The peak is at delay bin = 6548, Doppler = 2 kHz. Max correlation value is 0.8911. [4]

In Figure 2.6, a peak was observed at $(\hat{\tau}, \hat{f}_d) = (6548, 2kHz)$, where the value of the peak is significantly higher than the noise floor of the CAF plot in 2.6, indicating the result that the signal was obtained from the GPS satellite of PRN 5. In order to properly determine the existence of a certain PRN in a CAF plot, the peak must be significantly higher than the noise floor, above a certain threshold. If that threshold is passed for a specific delay code and frequency value in a CAF plot, those values are taken as the estimated values for $(\bar{\tau}, \bar{f}_d)$. If a sufficiently high peak is observed in a given CAF plot, the estimated values from the $S_{y,r}(\bar{\tau}, \bar{f}_d)$ can be found via the Maximum-Likelihood (ML) approach:

$$(\hat{\tau}, \hat{f}_d) = \underset{\{\bar{\tau}, \bar{f}_d\}}{\operatorname{argmax}} |S_{y,r}(\bar{\tau}, \bar{f}_d)| \tag{2.7}$$

where the estimated values $\hat{\tau}, \hat{f}_d$ are the maximizing values of the CAF function of $S_i(\bar{\tau}, \bar{f}_d)$. After the rough values are estimated, the receiver can begin the tracking stage [21].

2.3.3 Tracking

The tracking stage aims to ensure that the local PRN codes in the receiver are perfectly synchronized with the incoming signals from the satellites. After the acquisition stage provides the initial estimate of $\hat{\tau}$ and $\hat{f}_d$, the tracking loops lock into the detected satellites to continuously track the signals for the finer estimate of the code and carrier parameters, so that the receiver can determine its location via trilateration.

To keep track of the incoming satellites, the receiver replicates the observed satellites' PRN and adjusts its parameters to guarantee the synchronization [5]. One can see the basic schematic of the tracking stage in the receiver in Figure 2.7.

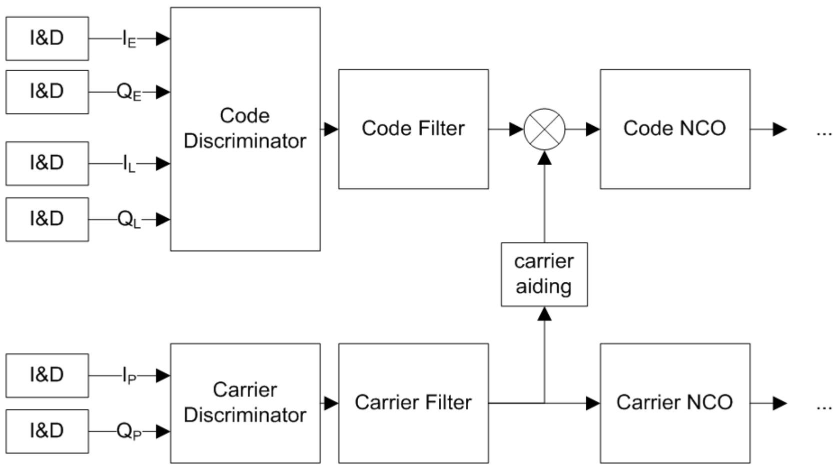


Figure 2.7: Basic structure of the tracking loop in the receiver [5]

The first block in the tracking stage can be listed as the integrate and dump ($I\&D$) units, which collect the correlators' outputs and separate them into the in-phase (I) and quadrature (Q) components. The E, P and L subscripts on the I and Q components refer to the Early, Prompt and Late correlators respectively [22]. Prompt (P) correlator is the aligned replica with the input signal whereas early (E) and late (L) correlators are either shifted earlier or later later replica with the incoming signal respectively. Depending on the correlator result, the receiver can detect whether the local replica is perfectly aligned or if it is not, how much correction is needed.

Then, the discriminators in the tracking stage process the I-Q components from the previous stage and provide measurable quantities such as code and carrier phase information. The filters try to remove the unwanted noise from the resulting signal coming from the discriminators. At last, Numerical Control Oscillators (NCO) take the filter outputs and then convert them into correction factors for Doppler shift and code delay, which are fed back to the Doppler wipe-off and local code generators blocks for fine estimation of the parameters [5].

For code tracking loop, the receivers commonly use the Delay Lock Loop (DLL). By observing the incoming signal's delay, the DLL provides a correction to the local replica code generators, to keep the local replica as much as aligned as possible with the transmitted signal, since an error of a millisecond may result in hundreds of kilometers [23], thus the precision is of utmost importance.

Carrier tracking loops may consist of only Phase Lock Loop (PLL), only Frequency Lock Loop (FLL) or a combination of both, depending on the receiver type. The PLL keeps track and tries to estimate the misalignment between the prompt correlator and the incoming signal phase, within the tracking loops in Figure 2.7 [24]. A common PLL is the Costas loop, which is a type of dicriminator indifferent to bit transitions [24]. On the other hand, FLL ignores the change or the difference in the phase and tries to catch up with the Doppler shift in the signal and provides frequency corrections [25].

2.4 GNSS Position, Velocity, and Time (PVT) Solutions

As the GNSS architecture, acquisition and tracking stages were laid out in the previous part, one can move onto the mathematical background of the user position and velocity calculation procedures. This is called the Position, Velocity, Time (PVT) solution, since along with the position and velocity, the variables regarding time, such as the clock bias and clock drift are also estimated. Timing information is also crucial in the GNSS systems, since most of the time the receiver time and the GNSS time is not synchronized, thus the difference is needed to be calculated. Below, the user position and velocity calculations are elaborated and the concept of Dilution Of Precision (DOP) is also explained.

2.4.1 User Position Calculation

The calculation of the user position can be estimated by knowing the range between the satellite and the user, which can be done via the pseudorange measurement model. The pseudorange from the j^{th} satellite to the user can be expressed as follows [26]:

$$\rho_j = r_j + c(t_u - t_j) + I_j + T_j + \epsilon_j \tag{2.8}$$

where ρ_j is the pseudorange between the satellite and the user, r_j is the true geometric range (Euclidean distance) between the satellite and the user, c is the speed of light in vacuum (299,792,458 m/s), t_u is the receiver clock bias, t_j is the clock bias of the satellite j, I_j and T_j are the ionospheric and tropospheric errors delays that result in measurement errors, ϵ_j represents the remaining errors which can be caused by not limited to multipath, noise and hardware errors. The reason why ρ_j is called pseudorange instead of range can be understood via the pseudorange measurement model in (2.8), since there is a mismatch between the measured distance and the actual range, due to the user and the satellite clock bias.

The actual range between the user and the satellite r_j can be formulated via Euclidean distance as follows:

$$r_j = \sqrt{(x_j - x_u)^2 + (y_j - y_u)^2 + (z_j - z_u)^2} \tag{2.9}$$

where $p_j = [x_j, y_j, z_j]$ is the satellite position and $p_u = [x_u, y_u, z_u]$ is the user position in Cartesian coordinate system. The term ct_u can be replaced by b_u and all the remaining errors can be collected inside ϵ_j for concision. One can assume the function the pseudorange only consist of the position and clock bias variables to solve it, since the errors in ϵ_j are not of interest for now, as below:

$$\begin{aligned} \rho_j &= f(x_u, y_u, z_u, b_u) \\ &= \sqrt{(x_j - x_u)^2 + (y_j - y_u)^2 + (z_j - z_u)^2} + b_u \end{aligned} \tag{2.10}$$

In order to have an approximation of the user position, one can assume that the function of the pseudorange for satellite j can be defined as follows [27]:

$$\begin{aligned}\hat{\rho}_j &= f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u) \\ &= \sqrt{(x_j - \hat{x}_u)^2 + (y_j - \hat{y}_u)^2 + (z_j - \hat{z}_u)^2} + \hat{b}_u\end{aligned} \tag{2.11}$$

Thus the approximated range is as follows:

$$\hat{r}_j = \sqrt{(x_j - \hat{x}_u)^2 + (y_j - \hat{y}_u)^2 + (z_j - \hat{z}_u)^2} \tag{2.12}$$

The relationship between the approximated variables $(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)$ and actual variables $(x_u.y_u, z_u, b_u)$ of the user position is as follows:

$$\begin{aligned}x_u &= \hat{x}_u + \Delta x \\ y_u &= \hat{y}_u + \Delta y \\ z_u &= \hat{z}_u + \Delta z \\ b_u &= \hat{b}_u + \Delta b\end{aligned} \tag{2.13}$$

where $\hat{p}_u = [\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u]$ denotes the approximated user location and $\Delta x, \Delta y, \Delta z, \Delta b$ denote the difference. Thus, one can also apply change of variables to (2.8) via (2.13) as below:

$$f(x_u, y_u, z_u, b_u) = f(\hat{x}_u + \Delta x, \hat{y}_u + \Delta y, \hat{z}_u + \Delta z, \hat{b}_u + \Delta b) \tag{2.14}$$

Nevertheless, (2.11) is non-linear in the user position $\hat{p}_u$, thus one has to apply the Taylor series expansion around a linearization point of $(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)$ to linearize the (2.11). The partial derivatives w.r.t to the approximated variables of the user position $[\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u]$ are:

$$\begin{aligned}\frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)}{\partial \hat{x}_u} &= -\frac{x_j - \hat{x}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)}{\partial \hat{y}_u} &= -\frac{y_j - \hat{y}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)}{\partial \hat{z}_u} &= -\frac{z_j - \hat{z}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{b}_u)}{\partial \hat{b}_u} &= 1\end{aligned} \tag{2.15}$$

Now, one can write the Taylor expansion to (2.14) as follows:

$$
\begin{aligned}
f(\hat{x}_u+\Delta x,\hat{y}_u+\Delta y,\hat{z}_u+\Delta z,\hat{b}_u+\Delta b) = & f(\hat{x}_u,\hat{y}_u,\hat{z}_u,\hat{b}_u)+\frac{\partial f(\hat{x}_u,\hat{y}_u,\hat{z}_u,\hat{b}_u)}{\partial \hat{x}_u}\Delta x \\
& +\frac{\partial f(\hat{x}_u,\hat{y}_u,\hat{z}_u,\hat{b}_u)}{\partial \hat{y}_u}\Delta y+\frac{\partial f(\hat{x}_u,\hat{y}_u,\hat{z}_u,\hat{z}_u)}{\partial \hat{z}_u}\Delta z \\
& +\frac{\partial f(\hat{x}_u,\hat{y}_u,\hat{z}_u,\hat{b}_u)}{\partial \hat{b}_u}\Delta b
\end{aligned}
\tag{2.16}
$$

Plugging the partial derivatives in (2.15), (2.8) and (2.11) to (2.16) one obtains:

$$
\rho_j=\hat{\rho}_j-\frac{x_j-\hat{x}_u}{\hat{r}_j}\Delta x_u-\frac{y_j-\hat{y}_u}{\hat{r}_j}\Delta y_u-\frac{z_j-\hat{z}_u}{\hat{r}_j}\Delta z_u+b_u \tag{2.17}
$$

Rearranging the expression in (2.17) to obtain the difference between the approximated pseudorange $\hat{\rho}_j$ and the actual pseudorange ρ_j, one can obtain:

$$
\hat{\rho}_j-\rho_j=\frac{x_j-\hat{x}_u}{\hat{r}_j}\Delta x_u+\frac{y_j-\hat{y}_u}{\hat{r}_j}\Delta y_u+\frac{z_j-\hat{z}_u}{\hat{r}_j}\Delta z_u-b_u \tag{2.18}
$$

To make the expression (2.18) simpler, one can apply the change of variables one more time, as follows:

$$
\begin{aligned}
\Delta\rho_j &= \hat{\rho}_j-\rho_j \\
a_{x,j} &= \frac{x_j-\hat{x}_u}{\hat{r}_j} \\
a_{y,j} &= \frac{y_j-\hat{y}_u}{\hat{r}_j} \\
a_{z,j} &= \frac{z_j-\hat{z}_u}{\hat{r}_j}
\end{aligned}
\tag{2.19}
$$

Thus, (2.18) becomes:

$$
\Delta\rho_j=a_{x,j}\Delta x_u+a_{y,j}\Delta z_u+a_{z,j}\Delta z_u-b_u \tag{2.20}
$$

Now, the pseurange equation for the j^{th} satellite is linear in Δx, Δy, Δz and b_u in (2.20) and it can be solved via Least Squares (LS). If one collects the pseudorange equations regarding n visible satellites, one can obtain:

$$
\begin{aligned}
\Delta\rho_1 &= a_{x,1}\Delta x_u+a_{y,1}\Delta z_u+a_{z,1}\Delta z_u-b_u \\
\Delta\rho_2 &= a_{x,2}\Delta x_u+a_{y,2}\Delta z_u+a_{z,2}\Delta z_u-b_u \\
&\vdots \\
\Delta\rho_n &= a_{x,n}\Delta x_u+a_{y,n}\Delta z_u+a_{z,n}\Delta z_u-b_u
\end{aligned}
\tag{2.21}
$$

The equations in (2.21) can be put in the matrix and vector form as follows:

$$\Delta\rho = \begin{bmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \vdots \\ \Delta\rho_n \end{bmatrix}, \quad H = \begin{bmatrix} a_{x,1} & a_{y,1} & a_{z,1} & 1 \\ a_{x,2} & a_{y,2} & a_{z,2} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ a_{x,n} & a_{y,n} & a_{z,n} & 1 \end{bmatrix}, \quad \Delta x = \begin{bmatrix} \Delta x_u \\ \Delta y_u \\ \Delta z_u \\ -\Delta b_u \end{bmatrix} \tag{2.22}$$

Thus, (2.21) can be expressed in matrix form via (2.22) as follows:

$$\Delta\rho = H\Delta x \tag{2.23}$$

In the (2.23), the unknowns are Δx and the $\Delta\rho$ term is computed using the pseudorange measurements, thus one would like to calculate Δx. If $n = 4$, thus only 4 satellites are used for the measurement, H is a square matrix. If the rows of H are linearly independent in that situation, H is invertible, thus one can obtain Δx directly as follows:

$$\Delta x = H^{-1}\Delta\rho \tag{2.24}$$

Nevertheless, if a reliable measurement is desired, it would be expected to have more than 4 satellites. In that case, H becomes a rectangular matrix, with number of rows (measurements) being higher than the number of columns (unknowns); and one can use the Moore–Penrose inverse (pseudoinverse) of matrix H.

$$\Delta x = (H^T H)^{-1} H^T \Delta\rho \tag{2.25}$$

It should also be reminded that there are 4 unknowns in the unknown vector Δx, being $(\Delta x_u, \Delta y_u, \Delta z_u, \Delta b_u)$; first three of them being the position coordinates of the user and the last one is the receiver clock bias. Thus, at least 4 equations are needed to find a unique solution, which means at least 4 visible satellites are needed too.

It should also be mentioned that the errors still exist in the calculation, denoted as ϵ_j in (2.8). If we also denote the errors in the pseudorange ρ as $d\rho$, the errors in the estimation vector x can be denoted as dx. The computation of dx can be found as follows, using the (2.25):

$$dx = [(H^T H)^{-1} H^T] d\rho \tag{2.26}$$

The matrix in the brackets ($(H^T H)^{-1} H^T$) in (2.26), defines the relationship between the pseudorange errors $d\rho$ and errors in the computation of the user position (x, y, z) and the clock bias (b). It should also be noted that the same matrix only depends on the geometry of the satellites and the user. Both of $d\rho$ and dx are considered to be vectors of Gaussian random variables with zero mean. The covariance of the dx can be computed as the expectation of $dxdx^T$. As a result, it can be found as:

$$cov(dx) = E[(H^T H)^{-1} H^T d\rho d\rho^T H (H^T H)^{-1}] \quad = (H^T H)^{-1} H^T cov(d\rho) H (H^T H)^{-1} \tag{2.27}$$

The covariance matrix of the pseudoranges $cov(d\rho)$ appears in the calculation of the covariance of the positions too (2.27), and the general assumption regarding $cov(d\rho)$ is that the errors between the measurements of the different satellites are not related, assumed to be independent, i.e. the errors are i.i.d (independent identically distributed). Thus, the covariance matrix of the error of the pseudorange measurements is diagonal, and the entries represent the variances which are the square of the satellite User Equivalent Range Error (UERE). Thus, the covariance matrix of the $d\rho$ can be written as:

$$cov(d\rho) = I_{n\times n}\sigma^2_{UERE} \tag{2.28}$$

where $I_{n\times n}$ is the identity matrix whose size is determined by the number of satellites (n), and σ^2_{UERE} is the variance of the satellite measurements. Since the covariance matrix in (2.28) is diagonal, the calculation of the covariance matrix dx in (2.27) reduces to:

$$\begin{aligned} cov(dx) &= (H^TH)^{-1}H^TH(H^TH)^{-1}cov(d\rho) \\ &= (H^TH)^{-1}cov(d\rho) \end{aligned} \tag{2.29}$$

The structure of $(H^TH)^{-1}$ determines the errors in the determined position of the user. As a result, $(H^TH)^{-1}$ is a 4×4 matrix, the square root of each diagonal entry represents the DOP of each variable of position (namely x_u, y_u, z_u and b_u). The matrix $(H^TH)^{-1}$ can be expressed in the following form:

$$(H^TH)^{-1} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} \\ d_{21} & d_{22} & d_{23} & d_{23} \\ d_{31} & d_{32} & d_{33} & d_{44} \\ d_{41} & d_{42} & d_{43} & d_{44} \end{bmatrix} \tag{2.30}$$

where d_{ii}, $i \in \{1,2,3,4\}$ correspond to the diagonal terms of $(H^TH)^{-1}$ in (2.30). One can find the DOPs using the diagonal elements of $(H^TH)^{-1}$. The most general form of DOP is Geometric Dilution Of Precision (GDOP), where it can be expressed as the square root of the trace of the $(H^TH)^{-1}$ matrix:

$$GDOP = \sqrt{d_{11} + d_{22} + d_{33} + d_{44}} \tag{2.31}$$

GDOP denotes the geometry factor that contributes to the error in the estimation of the position, and it is only a function of satellite and user geometry. Other DOP parameters are as follows: Position Dilution Of Precision (PDOP) (which considers x, y, z coordinates), Horizontal Dilution Of Precision (HDOP) (which considers only x, y coordinates in ENU reference frame)), Vertical Dilution Of Precision (VDOP) (which considers only z coordinate in ENU reference frame), Time Dilution Of Precision (TDOP) (which considers only the clock bias variable b). The formulation of the aforementioned DOPs are given as follows:

$$PDOP = \sqrt{d_{11} + d_{22} + d_{33}} \tag{2.32}$$

$$HDOP = \sqrt{d_{11} + d_{22}} \tag{2.33}$$

$$VDOP = \sqrt{d_{33}} \tag{2.34}$$

$$TDOP = \sqrt{d_{44}} \tag{2.35}$$

Thanks to the given DOP definitions from (2.32) to (2.35), one can also calculate the standard deviation in the position (σ_x), as the following:

$$\sigma_x = GDOP \cdot \sigma_{UERE} \tag{2.36}$$

Depending on the magnitude of the value of the GDOP, it is possible to comment on the reliability of the position estimation, based on the satellite-user geometry. The smaller the GDOP value is, the more reliable the estimation. Under ideal circumstances, the GDOP is smaller than 1, indicating the highest quality of estimation [6]. Between the GDOP values of 1-5, it can be said that the estimation results are highly reliable and meet most of the nonsensitive application requirements. If GDOP value is between 5-10, the estimation could be used nevertheless post-processing fix may be needed and an open sky condition is recommended to avoid more errors related to multipath, ionospheric and tropospheric errors. If GDOP values are larger than 10, the position estimations are poor and could be discarded unless a rough estimate is accepted.

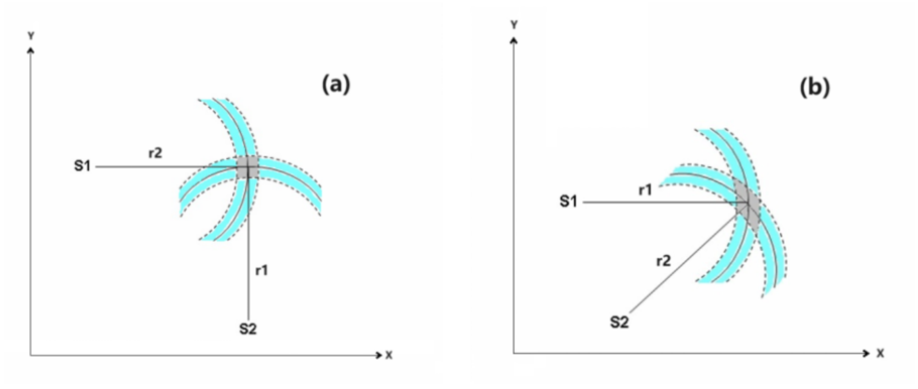


Figure 2.8: An example of the effect of GDOP on the precision of position estimation. a) The satellite user position vectors are orthogonal to each other resulting in a smaller area of intersection, and low DOP. b) The satellites are closer to each other, resulting in a larger area of intersection, and a higher DOP. [6].

One can see the geometrical example of high and low DOP geometries in Figure 2.8. It can be inferred that if the satellite positions are very close to each other, the intersection of the pseudorange uncertainties' area increases, which decreases the precision (i.e. dilutes) as a result the GDOP value increases.

Weighted Least Square (WLS) Estimation

During the LS estimation of the user position, it is assumed that the satellite measurements are i.i.d, but in real life that is hardly the case. Under those circumstances, the LS estimate becomes inadequate and there may be a need to make further improvements. It is shown that a weighting process can be implemented to the pseudorange measurements, on top of LS based on environmental conditions such as the noise level.

Again, if it is assumed that the pseudorange errors are Gaussian and the covariance of UEREs for the utilized satellites are given by the matrix $\boldsymbol{R}$, the optimized solution for the user position can be found as follows, instead of (2.25):

$$\Delta x = (H^T R^{-1} H)^{-1} H^T R^{-1} \Delta\rho \tag{2.37}$$

The matrix $\boldsymbol{R}$ in (2.37) is still diagonal, with different values in each diagonal entry, representing the reliability of each measurement. If the estimated variance of the noise is low, the weight would be high since one would like to prioritize precise measurements. Thus, in (2.37), the matrix $\boldsymbol{R}$ helps the estimation process by weighting each measurement and obtaining more precise results.

2.4.2 User Velocity Calculation

Similar to the user position calculation, the user velocity can be also calculated through the satellite measurements. Theoretically, the user velocity can be calculated as the derivative of the user position w.r.t time, as follows:

$$\dot{u} = \frac{du}{dt} \approx \frac{u(t_2) - u(t_1)}{t_2 - t_1} \tag{2.38}$$

where u is the user position, $\dot{u}$ is the user velocity, and the latter approximation holds if $(t_2 - t_1)$ is small enough. Due to the relative velocity between the satellite and the user, a shift in the frequency occurs on the received frequency, called the Doppler shift. The received frequency f_R approximation can be expressed in terms of the transmitted frequency f_T as follows:

$$f_R = f_T(1 - \frac{(v_r \cdot a)}{c}) \tag{2.39}$$

where v_r is the satellite-to-user relative velocity vector, a is the unit vector that points from the satellite to user along the Line of Sight (LOS) (which is calculated in the previous subsection when dealing with the LS estimation), and c is the speed of light (in vacuum, 299,792,458 m/s) or if not in the vacuum, the speed of propagation.

The dot product between the v_r and a represents the radial component of the v_r, where v_r can be expressed as the difference of the velocity vectors of the satellite (v) and the user; $v_r = v - \dot{u}$. The Doppler shift caused by the relative frequency can be expressed as:

$$\Delta f = f_R - f_T = -f_T \frac{(v - \dot{u}) \cdot a}{c} \tag{2.40}$$

Thus, one can write the received frequency from the j^{th} satellite considering the Doppler shift in (2.40), combining (2.39) and (2.40) as follows for the j^{th} satellite (related variables are subscripted as j):

$$f_{Rj} = \Delta f_j + f_{T_j} = f_{T_j} \left\{ 1 - \frac{[(v_j - \dot{u}) \cdot a_j]}{c} \right\} \tag{2.41}$$

However, the received signal frequency may be still subject to the error caused by the clock drift of the receiver ($\dot{t_u}$). If the measured estimate of the signal frequency is expressed as f_j, the received signal frequency f_{R_j} can be written as:

$$f_{R_j} = f_j(1 + \dot{t_u}) \tag{2.42}$$

Substituting (2.42) into (2.40) and after some algebraic manipulations, one can obtain the following:

$$\frac{c(f_j - f_{T_j})}{f_{T_j}} + v_j \cdot a_j = \dot{u} \cdot a_j - \frac{c f_j \dot{t}_u}{f_{T_j}} \tag{2.43}$$

Expanding the dot product into the vector components of $v_j = (v_{x_j}, v_{y_j}, v_{z_j})$ and $\dot{u} = (\dot{x}_u, \dot{y}_u, \dot{z}_u)$ results the following:

$$\frac{c(f_j - f_{T_j})}{f_{T_j}} + v_{x_j}a_{x_j} + v_{y_j}a_{y_j} + v_{z_j}a_{z_j} = \dot{x}_u a_{x_j} + \dot{y}_u a_{y_j} + \dot{z}_u a_{z_j} - \frac{c f_j \dot{t}_u}{f_{T_j}} \tag{2.44}$$

where the definition of $a_{t,j}$ ($t \in \{x, y, z\}$ for the j^{th} satellite, and $j \in \{1, ..., n\}$ for n observed and fixed satellites) was given in (2.19), and can be calculated during the position estimation in the receiver. The elements of the v_j vector can be obtained via the ephemeris data. f_{T_j} is known since it represents the nominal satellite transmission frequency, for example for GPS it is L1 (1575.42 MHz). f_j is expressed in terms of the receiver measurements. Thus, all the terms on the left hand side are known and can be replaced by a new term d_j, as a result, it can be written as follows:

$$d_j = \frac{c(f_j - f_{T_j})}{f_{T_j}} + v_{x_j}a_{x_j} + v_{y_j}a_{y_j} + v_{z_j}a_{z_j} \tag{2.45}$$

where d_j is also equal to the following expression, considering the fact that $\frac{f_j}{f_{T_j}} \approx 1$, one can remove them from the last term in (2.44) and obtain (2.46):

$$d_j = \dot{x}_u a_{x_j} + \dot{y}_u a_{y_j} + \dot{z}_u a_{z_j} - c\dot{t}_u \tag{2.46}$$

And one can apply the change of variables once again as $\dot{b}_u = c\dot{t}_u$ since it was already defined that $b_u = ct_u$ at the beginning of this chapter. So it could be thought that $\dot{b}_u = c\dot{t}_u$ is the derivative of $b_u = ct_u$.

It can be noticed that the (2.46) and (2.20) are very similar to the other, and the rest of the solution is very similar to user position calculation. Instead of linearized position ($\Delta x = [x_u, y_u, z_u]$) and pseudorange ($\Delta\rho$) variables in (2.20), now one can see the variables related to satellite and user velocity (d and $\dot{x}_u, \dot{y}_u, \dot{z}_u$; respectively). The vector/matrix form of (2.46) for observations coming from n satellites can be found below:

$$d = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix}, \quad H = \begin{bmatrix} a_{x_1} & a_{y_1} & a_{z_1} & 1 \\ a_{x_2} & a_{y_2} & a_{z_2} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ a_{x_n} & a_{y_n} & a_{z_n} & 1 \end{bmatrix}, \quad \dot{u} = \begin{bmatrix} \dot{x}_u \\ \dot{y}_u \\ \dot{z}_u \\ -\dot{b}_u \end{bmatrix} \tag{2.47}$$

It should be noted that the H matrix in (2.47) and (2.22) are the same. Thus, the user velocity equations become:

$$d = H\dot{u} \tag{2.48}$$

Using the pseudo-inverse of H, the user velocity vector $\dot{u}$ can be written as:

$$\dot{u} = (H^T H)^{-1} H^T d \tag{2.49}$$

It can also be noted that the (2.25) and (2.49) are very similar to each other and the matrix H plays a crucial role in finding both the user position and the user velocity.

Up till now, the background of the satellite navigation systems, the working principle, and how to find PVT solutions are thoroughly discussed in this chapter. Now, one can move onto the Factor Graph Optimization (FGO), its structure and how it can be utilized in finding a solution in GNSS systems, filtering and smoothing operations.

Chapter 3

Factor Graph Optimization

In this chapter, the general structure of the Factor Graph (FG) will be discussed so that the theoretical background of the thesis work can be fully understood. A FG is a powerful framework for representing and solving complex systems by breaking them down into simpler components. It allows the modeling of various probabilistic structures, such as Bayesian networks and Markov chains, by leveraging the known physical relationships between variables. One of the key advantages of a FG is its ability to express a large, complex objective function involving multiple variables as a combination of smaller, connected local functions. This simplification makes FG easier to solve graphical structures with a robust method, and makes it suitable to use in GNSS problems. FGs consist of two types of nodes: one being the variable nodes (i.e., the state vectors), which are the states to be estimated and initially unknown, and the other being the factor nodes (i.e. measurements, such as GNSS pseudorange), that encode the necessary equations and constraints on the variable node. The FGs started to be used in robotic applications [7], but thanks to their convenient structure, it is possible to extend the use case of FGs to GNSS PVT solution, signal processing, deep learning, computer vision and more. In the next sections, a more detailed approach to the background of the FGs and their use in GNSS scenarios are discussed.

3.1 Probabilistic modeling

In order to understand the FGs, one must have a firm grasp of probabilistic modelling, and especially the Bayesian networks in general. Since there is almost always an uncertainty in the sensor data in robotics and many other applications, it became a widespread practice to use the probability distributions to model the belief about the robot pose and estimation of its state. Thus, a continuous multivariate random variable $x \in \mathbb{R}^n$ is used to denote the related models, whose Probability Density Function (PDF) $p(x) \geq 0$ satisfies the following:

$$\int p(x)dx = 1 \tag{3.1}$$

In Simultaneous Localization and Mapping (SLAM) applications, which is a typical problem in robotics that is going to be used as an example in this chapter [7], one would like to model a set of unknowns X (such as the robot and landmark positions) that are based on the set of observations Z, which are the measurements provided by the sensors. In this case, that can be modelled as a conditional probability function:

$$p(X|Z) \tag{3.2}$$

Equation (3.2) implies the probability of X (of the robot position being at a certain point), given (conditioned) the measurements of Z (the sensor measurements). This procedure is called **probabilistic inference**, and that is where the probabilistic graphical models becomes beneficial [7]. Those models offer us a way to represent the complex probability distributions by leveraging their underlying structure. To elaborate more on the graphical models on the SLAM example which is in Figure 3.1, Bayesian networks will be the focus of the next subsection. In Figure 3.1, the x_1, x_2, x_3 represent the robot poses in time, whereas the l_1, l_2 represent the landmarks, whose positions are measured and will help determine the states.

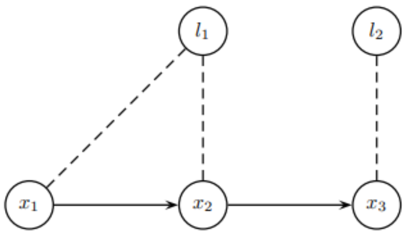


Figure 3.1: Toy SLAM example, where x_1, x_2, x_3 are robot poses and l_1, l_2 are landmarks

3.1.1 Bayesian Networks

Bayesian Networks (Bayesian Net), are a directed graphical model that exploits the conditional probability density to estimate the nodes using the set of random variables $\Theta = \{\theta_1, ..., \theta_n\}$. The joint probability density $p(\Theta)$ can be defined as:

$$p(\Theta) \triangleq \prod_j p(\theta_j|\pi_j) \tag{3.3}$$

where the variable π_j is related to the parent nodes of θ_j. The random variable set Θ is composed of states X as well as the measurements set Z, which can be shown that $\Theta = \{$X,Z$\}$. The Bayes net structure of the toy SLAM example in Figure 3.1 can be seen in Figure 3.2. The measurement variables z_1, z_2, z_3, z_4 shown in the boxes in Figure 3.2, represent the conditional relationship between the robot poses x_1, x_2, x_3 and landmarks l_1, l_2 when they become observed.

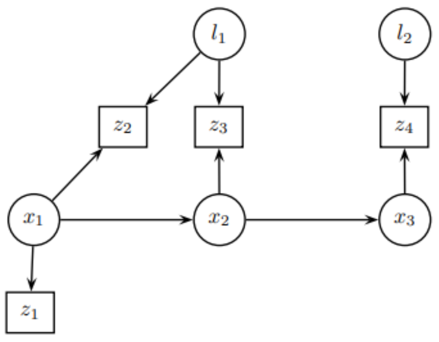


Figure 3.2: Toy SLAM example as Bayes Net

Thus, the joint density function $p(X, Z)$ can be obtained via the conditional probability density functions of the Bayes net in Figure 3.2 can be written as below:

$$\begin{aligned} p(X, Z) = {} & p(x_1)p(x_2|x_1)p(x_3|x_2) \\ & \times p(l_1)p(l_2) \\ & \times p(z_1|x_1) \\ & \times p(z_2|x_1, l_1)p(z_3|x_2, l_1)p(z_4|x_3, l_2) \end{aligned} \tag{3.4}$$

The four lines in the (3.4) can be described as below in order:

- Markov chain $p(x_1)p(x_2|x_1)p(x_3|x_2)$ on the poses x_1, x_2, x_3
- Probability density functions $p(l_1), p(l_2)$ on the landmarks
- Conditional probability $p(z_1|x_1)$ about the absolute measurement of x_1
- Conditional probabilities $p(z_2|x_1, l_1)p(z_3|x_2, l_1)p(z_4|x_3, l_2)$ with respect to the measurements of the landmarks (l_1, l_2) and robot poses (x_1, x_2, x_3, x_4)

While modelling the graphical models, usually a multivariate Gaussian distribution is assumed, whose probability function is [7]:

$$\mathcal{N}(\theta; \mu, \Sigma) = \frac{1}{|2\pi\Sigma|} \exp\left\{-\frac{1}{2}||\theta - \mu||^2_\Sigma\right\} \tag{3.5}$$

where Σ is the $n \times n$ covariance matrix, and $\mu \in \mathbb{R}^n$ is the mean of the multivariate Gaussian distribution in (3.5). The inside of the exponent in (3.5) indicates minus one half of the squared Mahalanobis distance, which will be used further in this study in FGO:

$$||\theta - \mu||^2_\Sigma \triangleq (\theta - \mu)^\intercal \Sigma^{-1}(\theta - \mu) \tag{3.6}$$

In order to estimate the states, one of the most used methods in statistics is Maximum A-Posteriori (MAP) inference to obtain information about the surroundings, which looks for the maximum of the posterior probability of $p(X|Z)$. Since in (3.4), the joint probability of $p(\Theta) = p(X, Z)$ is calculated, one can try to see which state or states in X allow $p(X, Z)$ to reach highest probability, given the set of measurements Z. The states in X may correspond to robot poses or the landmarks in the toy SLAM example in Figure 3.2. In order to compute the MAP estimate:

$$X^{MAP} = \underset{X}{\operatorname{argmax}} \; p(X|Z) \tag{3.7}$$

Using Bayes' Law:

$$X^{MAP} = \underset{X}{\operatorname{argmax}} \; \frac{p(Z|X)p(X)}{p(Z)} \tag{3.8}$$

Nevertheless, one can manipulate the terms of Bayes' Law to explain it in better terms and then adapt it to other graphical models such as the FGs. It is known that the measurements Z are given, thus the $p(Z)$ in (3.8) is just a scaling factor and it can be dropped since it does not affect the X^{MAP} value. Moreover, the conditional probability term $p(Z|X)$ is assumed to be a normalizing Gaussian density function in Z, not in X, which is in the scope of the interest. Thus, using the likelihood function $l(X; Z)$ in the MAP estimation is a more proper way to express what one would like to maximize (which are the states X) with respect to the constraints (which are the measurements Z). Since the likelihood function $l(X; Z)$ is related to the conditional probability of $p(Z|X)$, so that one could replace $p(Z|X)$ with $l(X; Z)$, as follows:

$$l(X; Z) \propto p(Z|X) \tag{3.9}$$

Replacing the (3.8) with the (3.10) enables us to represent MAP as the function of X, not Z, which is just a parameter. Thus, one could rewrite the Bayes' Law as below [7]:

$$X^{MAP} = \underset{X}{\operatorname{argmax}} \; l(X; Z)p(X) \tag{3.10}$$

Now, since required theoretical explanation for the graphical modelling has been set out, one can move onto the FGs to the next section.

3.2 Factor Graphs

Although Bayesian Net could be excellent for graphical modeling in many cases, FGs offer us a more effective method to carry out inference problem thanks to their properties.

Similar to Bayes net, in FGs one specifies the joint distribution functions as the product of factors. Nevertheless, the FGs can be used on more general cases since the functions can be any type of factorized functions $\phi(X)$, not only probability densities. In order to demonstrate the formulation of the FGs, one may take the toy SLAM example in Figure 3.1 and write the MAP estimation based on it. As a first step, the posterior function $p(X|Z)$ can be written as follows thanks to the Bayes' Law [7]:

$$\begin{aligned} p(X|Z) \propto\ & p(x_1)p(x_2|x_1)p(x_3|x_2) \\ & \times p(l_1)p(l_2) \\ & \times l(x_1; z_1) \\ & \times l(x_1, l_1; z_2)l(x_2, l_1; z_3)l(x_3, l_2; z_4) \end{aligned} \tag{3.11}$$

It is apparent that the conditional probability in (3.11), is a function of the unknowns only; namely the states and the landmarks.

Furthermore, one can also visualize the FGs easily, as in Figure 3.3, which is a FG representation of the toy SLAM example firstly introduced in Figure 3.1.

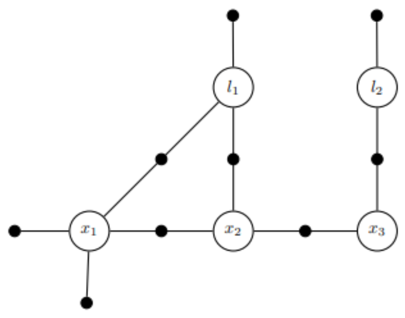


Figure 3.3: Toy SLAM example as a FG, where x_1, x_2, x_3 are robot poses and l_1, l_2 are landmarks, black dots represent the factors

Here, in Figure 3.3, instead of the explicit representation of the measurements Z as boxes and conditional densities in Figure 3.2, one can see the additional type of nodes called **factors** as black dots, between regarding states. It is worthy to note that the factors are only connected to the states that they are a function of. Considering this fact, it becomes very easy to associate 9 factors in (3.11) and 9 black dots in Figure 3.3 where there is a direct correspondent.

In mathematical terms, FG is a bipartite graph $\mathcal{F} = \{\mathcal{U}, \mathcal{V}, \mathcal{E}\}$ that consist of two types of nodes: variable nodes $x_j \in \mathcal{V}$ and the factor nodes $\phi_i \in \mathcal{U}$. An edge $e_{ij} \in \mathcal{E}$ can only exist between a factor node and a variable node, which makes the FG a bipartite graph. The neighboring variable nodes of a factor ϕ_i can be written as $\mathcal{N}(\phi_i)$, where the assignment of the variable nodes surrounding a factor is denoted as X_i, since they will be made of variable nodes -states and the landmarks- in the set X. Thus, the global function of a FG can be written as:

$$\phi(X) = \prod_i \phi_i(X_i) \tag{3.12}$$

In other words, the independency property in the FGs are encoded by the edges between a node and the factors, where each factor ϕ_i is a function of only the variables X_i in its adjacency set $\mathcal{N}(\phi_i)$ [7].

Considering the (3.12), one can write the global function in Figure 3.3 as below:

$$\begin{aligned}\phi(l_1, l_2, x_1, x_2, x_3) = &\ \phi_1(x_1)\phi_2(x_2, x_1)\phi_3(x_3, x_2) \\ &\times \phi_4(l_1)\phi_5(l_2) \\ &\times \phi_6(x_1) \\ &\times \phi_7(x_1, l_1)\phi_8(x_2, l_1)\phi_9(x_3, l_2)\end{aligned} \tag{3.13}$$

As it can be seen from the Figure 3.2 and 3.3, all Bayes net graphs can be easily converted into a FG, and vice versa, since there is a direct correspondence between the equations 3.4 and 3.13, line by line and factor by factor. It should also be noted that the functions ϕ_i do not have to adhere a specific type of function: they could be Gaussian, non-Gaussian, linear or non-linear etc., which is also another of the advantageous points of FGs compared to Bayesian networks. Thus, one can say that FGO is highly flexible since many kinds of factors can be added to the graph, based on the measurements.

3.2.1 MAP Inference for Nonlinear FGs

Since the global function $\phi(X)$ that defines the FG is constructed, now one can move onto the MAP estimation of it. Similar to what is done in Bayes net, one can also maximize the global function in FGs with respect to the states, which is the product of FG functions.

$$\begin{aligned}X^{MAP} &= \underset{X}{\operatorname{argmax}}\ \phi(X) \\ &= \underset{X}{\operatorname{argmax}} \prod_i \phi_i(X_i)\end{aligned} \tag{3.14}$$

If a Gaussian prior and likelihood is assumed, which is caused by the noise corrupting the measurements in the factors, the factor function can be written in the following form:

$$\phi_i(X_i) \propto \exp\Big\{-\frac{1}{2}||h_i(X_i) - z_i||^2_{\Sigma_i}\Big\}, \tag{3.15}$$

where $||.||_{\Sigma_i}$ is the Mahalanobis norm. The derivation of the MAP estimation steps using logarithm are indicated below:

$$\begin{aligned} X^{MAP} &= \underset{X}{\operatorname{argmax}} \phi(X) \\ &= \underset{X}{\operatorname{argmax}} \prod_i \phi_i(X_i) \\ &= \underset{X}{\operatorname{argmax}} \prod_i (\exp\left\{-\frac{1}{2}||h_i(X_i) - z_i||^2_{\Sigma_i}\right\}) \\ &= \underset{X}{\operatorname{argmin}} \prod_i (\exp\left\{\frac{1}{2}||h_i(X_i) - z_i||^2_{\Sigma_i}\right\}) \\ &= \underset{X}{\operatorname{argmin}} \ log \prod_i (\exp\left\{\frac{1}{2}||h_i(X_i) - z_i||^2_{\Sigma_i}\right\}) \\ &= \underset{X}{\operatorname{argmin}} \sum_i (\frac{1}{2}||h_i(X_i) - z_i||^2_{\Sigma_i}) \\ &= \underset{X}{\operatorname{argmin}} \sum_i (||h_i(X_i) - z_i||^2_{\Sigma_i}) \end{aligned} \tag{3.16}$$

To make the optimization easier, after plugging in the (3.15) into (3.13), one can take the negative logarithm of (3.13), drop the minus sign and 1/2 coefficient, and try to minimize the argument because the absence of the minus sign implies the negation of the objective function, and now the value that minimizes the function in (3.16) can be tried to found. Since taking the logarithm does not change the value that makes the X^{MAP} optimal, one can proceed to use that because it makes the optimization procedure simpler.

One should also note that the function $h(X_i)$ in (3.15) is usually a non-linear function, and in order to make the optimization procedure more straightforward, one may need to linearize the term $h(X_i)$ in (3.16). This procedure is elaborated in the next subsection.

3.2.2 Linearization

Thanks to the Taylor's expansion formula, one can linearize non-linear functions around a given linearization point. Using linearization methods, one can obtain the solution to the MAP estimation by solving a linear system rather than a nonlinear one, which in general is easier to solve. Considering the $h(.)$ nonlinear function, one can use the expansion below:

$$h_i(X_i) = h(X_i^0 + \Delta_i) \approx h(X_i^0) + H_i\Delta_i, \tag{3.17}$$

where the H_i is the multivariate partial derivative of the $h_i(X_i)$ at the linearization point X_i^0, and Δ_i is the state update vector, where $X_i = X_i^0 + \Delta_i$. The mathematical definition of H_i, which is the Jacobian of the measurement functions is as follows:

$$H_i \triangleq \left.\frac{\partial h_i(X_i)}{\partial X_i}\right|_{X_i^0} \tag{3.18}$$

If one plugs the Taylor expansion in (3.17) into the nonlinear MAP estimation 3.16, one can obtain the state update vector Δ^* as:

$$\begin{aligned}
\Delta^* &= \underset{X}{\operatorname{argmin}} \sum_i (||h_i(X_i) - z_i||^2_{\Sigma_i}) \\
&= \underset{X}{\operatorname{argmin}} \sum_i (||h(X_i^0) + H_i\Delta_i - z_i||^2_{\Sigma_i}) \\
&= \underset{X}{\operatorname{argmin}} \sum_i (||H_i\Delta_i - \left\{z_i - h(X_i^0))\right\}||^2_{\Sigma_i})
\end{aligned} \tag{3.19}$$

where the Δ^* is the solution to the locally linearized problem. As it was already described in (3.6), the norm used is (3.19) is a Mahalanobis norm, and there is an easy way to convert it to the l_2 norm to be used in the computations. Given a generic term e , one can convert it to l_2 using (3.6) and some algebra:

$$||e||^2_\Sigma \triangleq e^\intercal \Sigma^{-1} e = (\Sigma^{-1/2}e)^\intercal(\Sigma^{-1/2}e) = ||\Sigma^{-1/2}e||^2_2 \tag{3.20}$$

where $\Sigma^{-1/2}$ can be found via the Cholesky factorization of Σ. Now, again using the change of variables one can put the (3.19) in order.

$$\Delta^* = \underset{X}{\operatorname{argmin}} \sum_i ||\Sigma_i^{-1/2} H_i\Delta_i - \Sigma_i^{-1/2}\Big(z_i - h(X_i^0))\Big)||^2_2 \tag{3.21}$$

To make the form of the (3.21) more compact, one could apply the change of variables as follows:

$$\begin{aligned}
A_i &= \Sigma^{-1/2} H_i \\
b_i &= \Sigma^{-1/2}(z_i - h(X_i^0))
\end{aligned} \tag{3.22}$$

In that way, one can to turn the Mahalanobis form into l_2 norm in (3.19) and put the equation in a compact form. Thus, the (3.19) has become:

$$\begin{aligned}
\Delta^* &= \underset{\Delta}{\operatorname{argmin}} \sum_i (||A_i\Delta_i - b_i||^2_2) \\
&= \underset{\Delta}{\operatorname{argmin}}(||A\Delta - b||^2_2)
\end{aligned} \tag{3.23}$$

The structure of the information matrix A shows how the FG is constructed. Its rows reflect how the factors encode equations and constraints in the FG, and the number of rows are equal to the number of factors in general. The columns of A, represent the variables in the FG, which are usually states and landmarks to be determined, as in the toy SLAM example in Figure 3.3. The vector b corresponds to the prediction error at each stage. Due to the FGs' nature, the factor nodes are only connected to the variables nodes that they constrain. Thus, a factor is only connected to a few states, which makes the matrix A a sparse one (i.e. matrix A is not fully connected). Sparsity becomes an important factor in solving matrices, since it may result in more efficient computations and provide better accuracy for algorithms. That is one of the advantages of the FGs:

thanks to its sparsity property, it becomes easier to express and add constraints into the information matrix A.

An example of the matrix A and the prediction error b is provided below related to the toy SLAM example, which was provided in the Figure 3.3.

$$[A|b] = \begin{array}{c} \\ \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \\ \phi_7 \\ \phi_8 \\ \phi_9 \end{array} \begin{array}{c} \begin{array}{cccccc} \Delta l_1 & \Delta l_2 & \Delta x_1 & \Delta x_2 & \Delta x_3 & b \end{array} \\ \left[\begin{array}{ccccc|c} & & A_{13} & & & b_1 \\ & A_{23} & & A_{24} & & b_2 \\ & & A_{34} & & A_{35} & b_3 \\ A_{41} & & & & & b_4 \\ & A_{52} & & & & b_5 \\ & & A_{63} & & & b_6 \\ A_{71} & & A_{73} & & & b_7 \\ A_{81} & & & A_{84} & & b_8 \\ & A_{92} & & & A_{95} & b_9 \end{array}\right] \end{array} \tag{3.24}$$

The information matrix A and prediction error b related to the factor and variable nodes in the FG in 3.3 can be written in (3.24). Each ϕ, factor, which corresponds to a black dot in Figure 3.24, turns into a row of equation in matrix A. Thus, a total of 9 black dots, 9 factors, became 9 rows of matrix A. Moreover, matrix A has 5 columns, which correspond to 5 states to be estimated in Figure 3.3, namely x_1, x_2, x_3, l_1 and l_2. They were denoted adding a Δ on top of the matrix, because of the linearization process elaborated in the previous part.

How to construct and solve such a system in (3.24) will be explained in details, in the next subsection.

3.2.3 The Elimination Algorithm

There are many algorithms that can solve a FG, but a commonly used scheme exists under the name of the elimination algorithm. The elimination algorithm eliminates one variable at the time of a FG, thus it is computationally efficient and the MAP estimate of a sparse FG can be easily found. The algorithm basically converts a FG into a Bayes net, nevertheless only on the unknowns of the FG, which results in a straightforward MAP inference. This algorithm can be applied to any FG.

Thanks to the elimination algorithm, one can turn a FG in the following form:

$$\phi(X) = \phi(x_1, ..., x_n) \tag{3.25}$$

into a Bayes net in the following form:

$$p(X) = p(x_1|S_1)p(x_2|S_2)...p(x_n|S_n) = \prod_i p(x_i|S_i) \tag{3.26}$$

The S_i indicates the set **separator**, where the variable x_i is conditioned. It should be noted that in the elimination algorithm, the order of elimination is important because it also affects the remaining separators of other variables. If a variable is eliminated from a

FG, the variable will not be used in the rest of the factorized products in (3.26). When a variable x_i is eliminated, a single conditional probability $p(x_i|S_i)$ occurs as a factor.

When all variables become eliminated, the global function of the FG in (3.25) turns into a Bayes net an the probability function in (3.26). Thus, one can also say that the elimination algorithm turns $\phi(X)$ into $p(X)$ in n local factorization steps, n being the number of variable nodes.

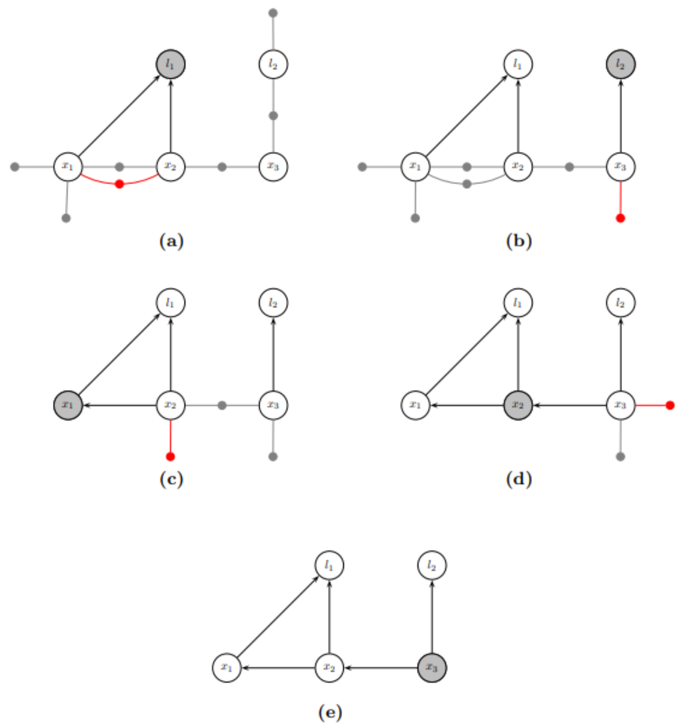


Figure 3.4: Elimination algorithm applied to the FG of the toy SLAM example in Figure 3.3. The elimination order is l_1, l_2, x_1, x_2, x_3 [7]. The eliminated variable at the indicated state is shaded gray. The red dots at each step represent the factors that are being added to the information matrix A regarding neighboring states, as a result of elimination algorithm.

In mathematical terms, when x_i is eliminated, the remaining partial FG is denoted as $\Phi_{j:n} \triangleq \phi(x_j, ..., x_n)$, which includes the variables that are not eliminated yet (from x_j to x_n). Given a partially eliminated FG in the form of $\Phi_{j:n}$, one must remove all the factors that are neighboring x_j, and multiply them with the joint probability of $\phi(x_j, S_j)$, which can be written using the conditional probability $p(x_i|S_i)$ as:

$$\phi(x_j, S_j) = p(x_i|S_i)\tau(S_j) \tag{3.27}$$

where $\tau(S_j)$ is the newly formed factor. When all the factors are eliminated, thus the separator $S(x_n)$ will be empty, thus the final variable yet to be eliminated will not be conditioned on any other node and the conditional probability function of x_j will just be the prior of it, solely $p(x_n)$.

A graphical example of the elimination algorithm in a FG is provided in Figure 3.4, in the order of $(l_1 \to l_2 \to x_1 \to x_2 \to x_3)$. Thus, the resulting Bayes net probability function factorization from $\phi(X)$ is as follows:

$$\begin{aligned} p(X) &= p(l_1, l_2, x_2, x_2, x_3) \\ &= p(l_1|x_1, x_2)p(l_2|x_3)p(x_1|x_2)p(x_2|x_3)p(x_3) \end{aligned} \tag{3.28}$$

As a reminder, in the problem outlined by the matrix in (3.24), the factors are in the form as in (3.23), which is:

$$\phi_i(X_i) = \exp\Big\{ -\frac{1}{2}||A_iX_i - b_i||_2^2\Big\} \tag{3.29}$$

where A_i is a matrix similar to the one in (3.24), which includes terms that connects states to its neighboring ones, via the encoding factors. Depending on at which step the elimination algorithm is, A_i may be composed of smaller set of state variables X_i, because of the already eliminated states.

In order to run the elimination algorithm on a variable x_j, one must remove all factors from $\phi_i(X_i)$ in (3.29), neighboring x_j and produce the intermediate factor of $\psi(x_j, S_j)$ as follows:

$$\begin{aligned} \psi(x_j, S_j) &\leftarrow \prod_i \phi_i(X_i) \\ &= \exp\Big\{ -\frac{1}{2}\sum_i ||A_iX_i - b_i||_2^2\Big\} \\ &= \exp\Big\{ -\frac{1}{2}||\bar{A}_j[x_j; S_j] - \bar{b_j}||_2^2\Big\} \end{aligned} \tag{3.30}$$

where $\bar{A_j}$ and $\bar{b_j}$ are the accumulated versions of A_i and b_i into one large matrix and one large vector respectively. In order to factorize $\psi(x_j, S_j)$ one can use many methods, but in this work, QR factorization will be used. The augmented matrix implied in (3.30) $[\bar{A_j}|\bar{b_j}]$ can be written using partial QR factorization as follows:

$$[\bar{A_j}|\bar{b_j}] = Q\begin{bmatrix} R_j & T_j & d_j \\ & \tilde{A}_\tau & \tilde{b}_\tau \end{bmatrix} \tag{3.31}$$

where R_j, T_j, and $\tilde{A_\tau}$ are matrices and d_j and $\tilde{b_\tau}$ are vectors in the system of equations. Specifically, R_j is an upper triangular matrix in (3.31) as a result of partial QR factorization. By plugging (3.31) into (3.30), $\psi(x_j, S_j)$ can be further factorized using QR factorization as follows:

$$\begin{aligned}\psi(x_j, S_j) &= \exp\Big\{-\frac{1}{2}||\bar{A}_j[x_j; S_j] - \bar{b_j}||_2^2\Big\} \\ &= \exp\Big\{-\frac{1}{2}||R_j x_j + T_j S_j - d_j||_2^2\Big\}\exp\Big\{-\frac{1}{2}||\tilde{A}_\tau S_j - \tilde{b_\tau}||_2^2\Big\} \\ &= p(x_i|S_i)\tau(S_j)\end{aligned} \tag{3.32}$$

As a result, the formula in (3.27) is obtained via the steps in (3.32). Here, the matrix Q is not incorporated since it does not change the norm of the matrix R since Q is orthonormal, thus discarded for this operation. The operation in (3.32) can be done until all variables are eliminated.

If Gaussian linearized factors are assumed as in (3.26), using the QR factorization, conditional densities can be written as:

$$p(x_j|S_j) \propto \exp\Big\{-\frac{1}{2}||R_j x_j + T_j S_j - d_j||_2^2\Big\} \tag{3.33}$$

Thus, the inner part of the exponential can be written as below considering the distribution is multivariate Gaussian function:

$$||R_j x_j + T_j S_j - d_j||_2^2 = (x_j - \mu_j)^\intercal R_j^\intercal R_j (x_j - \mu_j) \triangleq ||x_j - \mu_j||_{\Sigma_j}^2 \tag{3.34}$$

It can be further reduced to:

$$\begin{aligned}||R_j x_j + T_j S_j - d_j||_2^2 &= ||x_j - \mu_j||_{\Sigma_j}^2 \\ ||R_j x_j - (d_j - T_j S_j)||_2^2 &= ||\Sigma_j^{-1/2} x_j - \Sigma_j^{-1/2}\mu_j||_2^2\end{aligned} \tag{3.35}$$

As a result, the solutions to the covariance matrix Σ_j and the mean vector μ_j become:

$$\begin{aligned}\Sigma_j &= (R_j^\intercal R_j)^{-1} \\ \mu_j &= R_j^{-1}(d_j - T_j S_j)\end{aligned} \tag{3.36}$$

After the elimination algorithm is finished, the MAP estimation of each variable can be found via the back substitution since after the last variable is eliminated, it does not depend on any other variables and it can directly be computed. Going in the reverse direction of the elimination algorithm order, each state can be found MAP estimation, which is the mean (i.e. μ_j) in (3.36).

$$x_j^* = R_j^{-1}(d_j - T_j S_j^*) \tag{3.37}$$

Since the main theory of how to construct and solve FGs has been laid out theoretically in this chapter, now one can move onto the specific FGs employed in a GNSS scenario in the very next subsection.

3.2.4 Factor Graphs for GNSS

In the FGs that are specifically used for GNSS applications, usually the states that are estimated could be the receiver position, receiver velocity, tropospheric delay, carrier phase bias, the receiver clock bias and clock drift [8]. The factors that constrain the states in a typical GNSS application could be the pseudorange, prior state estimates, the dynamic equations. In this thesis; the receiver position and velocity, and the receiver clock bias and the clock drift (i.e. the derivative of the clock bias) are estimated; and comparisons were made based on the ground truth trajectory that is collected using another device. Thus, the state vector to be estimated becomes the following:

$$\Delta x = \begin{bmatrix} \Delta x_k & \Delta y_k & \Delta z_k & -\Delta b_k & \Delta \dot{x_k} & \Delta \dot{y_k} & \Delta \dot{z_k} & -\Delta \dot{b_k} \end{bmatrix}^T \tag{3.38}$$

where the first 4 variables are related to the receiver position (first 3 variables) and the last being the negative of the clock bias $(\Delta x_k, \Delta y_k, \Delta z_k, -\Delta b_k)$ and the latter 4 variables are related to the receiver velocity and the clock drift $(\Delta \dot{x_k}, \Delta \dot{y_k}, \Delta \dot{z_k}, -\Delta \dot{b_k})$ in (3.38). One can see a FG representation of a GNSS application in Figure 3.5.

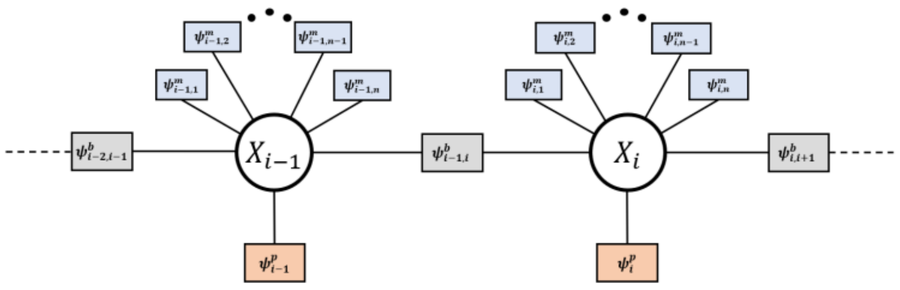


Figure 3.5: A typical FG for GNSS application [8]. The boxes represents factors and circles represent the states. Different colors of boxes indicate different types of factors. Red boxes indicate prior factors, grey boxes indicate motion factors, and blue boxes indicate GNSS factors.

In a typical GNSS FG as in Figure 3.5, the factors related to states to be estimated can be divided into three groups. Referencing to the Figure 3.5, the factors can be listed as follows:

Prior factor ψ^p_i: The prior belief about the state x_i, before other observations. The prior belief may depend on environmental characteristics or the specific dataset.

Motion factor $\psi^b_{i-1,i}$: The factor related to two successive states, a constraint that connects two state variables: x_{i-1} and x_i. May also be called the dynamic factor. This kind of factor can either be predicted or it can be observed via Inertial Measurement Unit

(IMU) sensors or wheel odometry [8].

GNSS factor $\psi^m_{i,1:n}$: The factor that relates each state to a GNSS observable, which could be pseudorange measurements or carrier phase measurements.

As it was detailed in the previous chapter, the aim is to integrate the related formulas of the factors into the information matrix A and prediction error b, so that one can conduct the MAP estimate of each state, in the form of $\phi = ||A\Delta - b||_2^2$, as in (3.23). Matrix A and column vector b will be constructed as in the (3.24), Then, the elimination algorithm will be run in order to find the optimum values of the states X. Thus, the objective function that is going to be minimized as a result of FG becomes the following equation:

$$X = \underset{x}{\operatorname{argmin}} \ \left(\sum \psi_{prior} + \sum \psi_{dynamics} + \sum \psi_{GNSS}\right) \tag{3.39}$$

It should be noted that, so far the robust estimation methods (which are in the scope of this thesis) have not been discussed or integrated yet. Firstly, a thorough analysis of typical factors will be completed in this chapter, then in addition to what is built, the robust estimation methods such as the Switch Constraints (SC) and Gaussian Max-Mixtures (GMM) methods will be analyzed and integrated in the next chapter.

Below, one can find the necessary formulations of the prior, dynamic and GNSS factors elaborated.

Prior Factor

The prior factor reflects what is best known of the current state based on the estimate of the previous state. Thus, one can deduce the prior factor estimation of the x_i^{th} state from previous state x_{i-1}'s covariance matrix Σ_{i-1} and the expected value $E(x_{i-1})$. In mathematical terms, the prior factor becomes the following:

$$\begin{aligned} \psi_i^p = \psi_{prior} &= (||x_{i-1} - \hat{x}_{i-1}||^2_{\Sigma_{i-1}}) \\ &= (||\hat{x}_{i-1} + \Delta x_{i-1} - \hat{x}_{i-1}||^2_{\Sigma_{i-1}}) \\ &= (||\Delta x_{i-1}||^2_{\Sigma_{i-1}}) \\ &= (||(\Sigma_{i-1})^{1/2}\Delta x_{i-1}||^2_2) \end{aligned} \tag{3.40}$$

Thus, it yields to:

$$\begin{aligned} A_{prior} &= \Sigma_{i-1} \\ b_{prior} &= 0 \end{aligned} \tag{3.41}$$

It should be noted that in this formulation of the prior factor, the coefficients in the matrix A and vector b are related to Δx_{i-1}, i.e. the previous state, not the currently estimated state of Δx_i.

Dynamic Factor

As it can be seen in the Figure 3.5, the dynamic factor $\psi^b_{i-1,i}$, is between two successive states of Δx_{i-1} and Δx_i, so it might be expected that the information matrix A and the vector b will include coefficients of both of the variables Δx_{i-1} and Δx_i. In this factor estimation, a general definition of the dynamic factor $\psi^b_{i-1,i}$ will be used, which is detailed below:

$$\psi^b_{i-1,i} = \psi_{dynamics} = (||f(x_{i-1}) - x_i)||^2_{Q_{i-1}}) \tag{3.42}$$

where the $f(.)$ is the dynamic model function and the Q_{i-1} is the related process noise covariance matrix. Since in some GNSS applications, there is not a direct approach to obtain the exact dynamic model of the vehicle, the model has to be chosen a-priori through research. One of the most used dynamic model to simulate the dynamics of a vehicle is the **constant-velocity model**, which assumes that the velocity is constant between two time instances [28], [29]. Through constant-velocity model, the relation between two successive time instances can be modelled as:

$$\hat{x}_i = F_{i-1}\hat{x}_{i-1} \tag{3.43}$$

where F_{i-1} is the state transition matrix [29], and it can be expressed as:

$$F_{i-1} = \begin{array}{r} \Delta x_{i-1} \\ \Delta y_{i-1} \\ \Delta z_{i-1} \\ -\Delta b_{i-1} \\ \Delta \dot{x}_{i-1} \\ \Delta \dot{y}_{i-1} \\ \Delta \dot{z}_{i-1} \\ -\Delta \dot{b}_{i-1} \end{array} \begin{bmatrix} 1 & 0 & 0 & 0 & T & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & T & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.44}$$

As it can be seen in (3.44), F_{i-1} is a square matrix, where columns and rows correspond to the same variable at the diagonal, and T is the sampling period, in this context time interval between two epochs. The constant velocity dynamics model adjusts the position variables (which are the first 3 variables of the Δx vector: Δx_{i-1}, Δy_{i-1}, Δz_{i-1}) according to the velocity of the receiver (which are the 5^{th}, 6^{th} and the 7^{th} variables of the Δx vector: $\Delta \dot{x}_{i-1}$, $\Delta \dot{y}_{i-1}$, $\Delta \dot{z}_{i-1}$) by the time between two successive epochs: T. Thus the factor becomes:

$$\begin{aligned} \psi_{dynamics} &= (||F_{i-1}x_{i-1} - x_i||^2_{Q_{i-1}}) \\ &= (||F_{i-1}\hat{x}_{i-1} + F_{i-1}\Delta x_{i-1} - \hat{x}_i - \Delta x_i||^2_{Q_{i-1}}) \\ &= (||[F_{i-1}, -I][\Delta x_{i-1}, \Delta x_i]^T - (\hat{x}_i + F_{i-1}\Delta x_{i-1})||^2_{Q_{i-1}}) \\ &= (||[F_{i-1}, -I][\Delta x_{i-1}, \Delta x_i]^T||^2_{Q_{i-1}}) \\ &= (||[Q^{-1/2}_{i-1}F_{i-1}, -Q^{-1/2}_{i-1}I][\Delta x_{i-1}, \Delta x_i]^T||^2_2) \end{aligned} \tag{3.45}$$

where I is the identity matrix. If the change of variables is applied to simplify the notation, one can obtain $F_Q = [Q^{-1/2}_{i-1}F_{i-1}]$, and $I_Q = [-Q^{-1/2}_{i-1}I]$, the information matrix A and vector b related to the dynamics factor become as follows:

$$\begin{aligned} A_{dynamics} &= [F_Q, I_Q]; \\ b_{dynamics} &= 0 \end{aligned} \tag{3.46}$$

It should also be noted that the the information matrix A includes the coeffiecients for both Δx_i and Δx_{i-1}, as it can be seen in (3.45).

GNSS Factor

Last but not the least, there will be factors related to the GNSS measurements, coming from each visible and utilized satellite at that moment, a total of n satellites. In Figure 3.5, the blue boxes refer to the GNSS factors, measurements coming from n satellites. The factors were shown as from $\psi_{i,1}^m$ to $\psi_{i,n}^m$ for x_i^{th} state. In Chapter 2, the details of how to obtained the PVT solution were thoroughly discussed, and the same approach for the FGs will be followed here.

The satellite measurements that come from j^{th} satellite in the i^{th} state were denoted as $\zeta_{i,j} = [\rho_{i,j}, \dot{\rho_{i,j}}]^T$, which are related to satellite's pseudorange measurement and its rate of change, and the observation function can be defined as $h_i(.)$, which is already detailed in Chapter 2. Thus, the GNSS factor for the j^{th} satellite can be written as below:

$$\begin{aligned}\psi_{i,j}^m = \psi_{GNSS}^j &= (||h_{i,j}(x_i) - \zeta_{i,j}||_{R_{i,j}}^2) \\ &= (||H_{i,j}\Delta x_i - h_{i,j}(\hat{x}_i) + \zeta_{i,j})||_{R_{i,j}}^2) \\ &= (||H_{i,j}\Delta x_i - (\hat{\zeta}_{i,j} - \zeta_{i,j})||_{R_{i,j}}^2) \\ &= (||R_{i,j}^{-1/2}H_{i,j}\Delta x_i - R_{i,j}^{-1/2}(\hat{\zeta}_{i,j} - \zeta_{i,j})||_2^2)\end{aligned} \tag{3.47}$$

where $R_{i,j}$ is the observation noise covariance associated to the j^{th} satellite, and can be estimated or found depending on the case. $H_{i,j}$ is the observation matrix, related to the PVT solution, already covered in Chapter 2. Applying the change of variables for the observation vector $z_{i,j} = \hat{\zeta}_{i,j} - \zeta_{i,j}$, one can obtain the following GNSS factor for the j^{th} satellite:

$$\psi_{GNSS}^j = \exp(||R_{i,j}^{-1/2}H_{i,j}\Delta x_i - R_{i,j}^{-1/2}z_{i,j}||_2^2) \tag{3.48}$$

Thus, the information matrix A and vector b becomes for the GNSS factor the following:

$$\begin{aligned}A_{GNSS,j} &= R_{i,j}^{-1/2}H_{i,j}, \\ b_{GNSS,j} &= R_{i,j}^{-1/2}z_{i,j}\end{aligned} \tag{3.49}$$

One can also combine all of the factors for n satellites into a single matrix A_{GNSS} and vector b_{GNSS}, obtaining the following for the i^{th} state:

$$\begin{aligned}A_{GNSS} &= R_i^{-1/2}H_i, \\ b_{GNSS} &= R_i^{-1/2}z_i\end{aligned} \tag{3.50}$$

For the sake of completeness, the full H_i and z_i that include observations coming from total of n satellites, are reported below:

$$z_i = \left[\begin{array}{c} \hat{\rho}_{i,1} - \rho_{i,1} \\ \vdots \\ \hat{\rho}_{i,n} - \rho_{i,n} \\ \hline \hat{\dot{\rho}}_{i,1} - \dot{\rho}_{i,1} \\ \vdots \\ \hat{\dot{\rho}}_{i,n} - \dot{\rho}_{i,n} \end{array}\right], \qquad H_i = \left[\begin{array}{cccc} a_{i,1}^T & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_{i,n}^T & 1 & 0 & 0 \\ \hline 0 & 0 & a_{i,1}^T & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & a_{i,n}^T & 1 \end{array}\right] \tag{3.51}$$

where $a_{i,j} = [(\frac{x_{i,j} - \hat{x}_{i,j}}{\hat{r}_{i,j}}), (\frac{y_{i,j} - \hat{y}_{i,j}}{\hat{r}_{i,j}}), (\frac{z_{i,j} - \hat{z}_{i,j}}{\hat{r}_{i,j}})]^T$ is the unit vector that points towards to the j^{th} satellite at the i^{th} state for Δx_i.

Running Elimination Algorithm

So far, a typical FG in a GNSS application and how to form related information matrix A and vector b is discussed in the previous subsections. Now, after forming the information matrix A and vector b, one can move onto solving them via the elimination algorithm discussed earlier.

In the scope of this thesis, a **multifrontal QR factorization** algorithm will be preferred, meaning that the factorization of Δx_{i-1} and Δx_i will be done using the QR factorization, instead of partial QR factorization method, in which one estimates Δx_{i-1} and Δx_i at two separate QR factorization steps. Using multi-frontal QR factorization decreases the number of steps by doing all the factorization at one step, instead of the number of sta, so it is computationally more efficient.

As it was elaborated in the previous subsections, there are three types of factors in a typical FG related to a GNSS example: namely prior factor, dynamic factor and GNSS factor. If one collects all three types of factors in the information matrix A and vector b, the following form is obtained:

$$[A|b] = \left[\begin{array}{cc|c} A_{prior} & & b_{prior} \\ F_Q & I_Q & b_{dynamic} \\ & A_{GNSS} & b_{GNSS} \end{array}\right] \tag{3.52}$$

where the individual terms were already introduced. It is worthy to note that the first column of A corresponds to the information regarding the previous state Δx_{i-1}, and the second column corresponds to the current state Δx_i.

Applying QR factorization to the matrix system $[A|b]$, one can obtain the following:

$$QR([A|b]) = QR\left(\left[\begin{array}{cc|c} A_{prior} & & b_{prior} \\ F_Q & I_Q & b_{dynamic} \\ & A_{GNSS} & b_{GNSS} \end{array}\right]\right) \tag{3.53}$$

$$QR\left(\left[\begin{array}{cc|c} A_{prior} & & b_{prior} \\ F_Q & I_Q & b_{dynamic} \\ & A_{GNSS} & b_{GNSS} \end{array}\right]\right) \rightarrow \left[\begin{array}{cc|c} R_{k-1} & T_{k-1} & d_{k-1} \\ & R_k & d_k \end{array}\right] \tag{3.54}$$

As a result of the QR factorization, an upper triangular matrix is formed, which contains the information regarding the states Δx_{i-1} and Δx_i. Using formulas (3.36) and

(3.37) and algebraic manipulations to the (3.54), one can determine the solution to the state Δx_i and its covariance matrix Σ_i as follows:

$$\begin{aligned} \Delta x_i &= R_i^{-1} d_i \\ \Sigma_i &= (R_i^T R_i)^{-1} \end{aligned} \tag{3.55}$$

Plugging Δx_i which was found in (3.55) to (3.54), which was also mentioned in (3.37), one can also determine the previous state of Δx_{i-1} as following:

$$\Delta x_{i-1} = R_{i-1}^{-1}(d_{i-1} - T_{i-1}\Delta x_i) \tag{3.56}$$

In fact, smoothing via back substitution is possible for many states using (3.56), if a fixed-lag number has been set and previous values have been stored. Since the values for $\boldsymbol{R},$ $\boldsymbol{T}$ and $\boldsymbol{d}$ are stored and after the current state value of Δx_i has been found, the corrections regarding previous states can be performed via the smoothing process. Thus, if a new variable representing the fixed-lag number $lag = 1{:}m$, m being the number of stored states, one can perform the smoothing via back substitution as follows:

$$\Delta x_{i-lag} = R_{i-lag}^{-1}(d_{i-lag} - T_{i-lag}\Delta x_{i-lag+1}) \tag{3.57}$$

So far, the construction and details regarding how to solve a generic FG regarding the GNSS application has been elaborated in this chapter with mathematical descriptions and related background. Now, one can move onto the methods of how to build more robust estimation for GNSS applications using FG in the next chapter.

Chapter 4

Robust Estimation Methods Using FGs in GNSS

So far, how to construct a FG for a typical GNSS application has been discussed in the previous chapter. As it was mentioned, thanks to the flexible structure of the FGs, it is easy to add new constraints, such as new terms in the objective function to minimize; to reduce the errors caused by multipath, atmospheric errors, or spoofing attacks [8]. This thesis investigates two robust methods that are shown to be effective against localization errors in degraded environments; namely Switch Constraints (SC) and Gaussian Max-Mixtures (GMM). Below, one can see the detailed theoretical background for both of the methods, and how to integrate them to an already existing FG, i.e. a FG formed in the last chapter.

4.1 Switch Constraints (SC)

Switch Constraints (SC) initially developed for the false loop closures problem in SLAM, nevertheless it has been shown that it can also be applied to the GNSS framework when dealing with multipath mitigation in urban environments [8], [30].

SC method introduces a new kind of variable node to be estimated, called switchable constraint to be optimized along with the position states in the FG. Every switchable constraint is connected to a factor in the graph. Switchable constraints can be thought of observation weights that can adjust the contribution level of each factor, depending on their reliability [11]. As a result, one needs to add switchable constraint terms to the objective function and introduce the switch constraint states, denoted as S.

The modified cost function will include the term related to switch constraints, the most important one shown below:

$$\psi^i_{switch} = ||\Psi(s_{i,j}).(h_{i,j}(x_i) - \zeta_{i,j})||^2_{\Sigma_i} \tag{4.1}$$

where the $(h_{i,j}(x_i) - \zeta_{i,j})$ term is related to the GNSS pseudorange factor in (3.47), $\Psi(s_{i,j})$ is the switch function that depends on the switch variable $s_{i,j}$. Switch functions

$\Psi(s_{i,j})$ are used in the cost function to adjust the weight of a measurement and the switch variables $s_{i,j}$ become the nodes in the factor graph to be estimated. The switch variables, $s_{i,j}$ are not directly used in the cost function because they may introduce discontinuities in the solution, which is unsuitable for optimization problems [30], thus a continuous switch function is defined. $\Psi(s_{i,j})$ is a linear function of $s_{i,j}$, $\Phi : \mathbb{R} \to [0,1]$ that maps it from continuous real number to the interval of [0,1] [30]. A commonly used and effective switch function can be found as follows:

$$\Psi_a(s_{i,j}) : \mathbb{R} \to [0,1] = \begin{cases} 0 & : s_{i,j} < 0 \\ \frac{1}{a} s_{i,j} & : 0 \leq s_{i,j} \leq a \\ 1 & : s_{i,j} > a \end{cases} \tag{4.2}$$

where a = 1 is shown to be a suitable choice [30]. The aim of the switch constraint is as follows: the lower the switch constraint $s_{i,j}$, lower the switch function $\Psi_a(s_{i,j})$, which adds a weight to its related factor and decreases its contribution to the calculation of the position state if necessary. Thus, FG with SC can be seen in Figure 4.1.

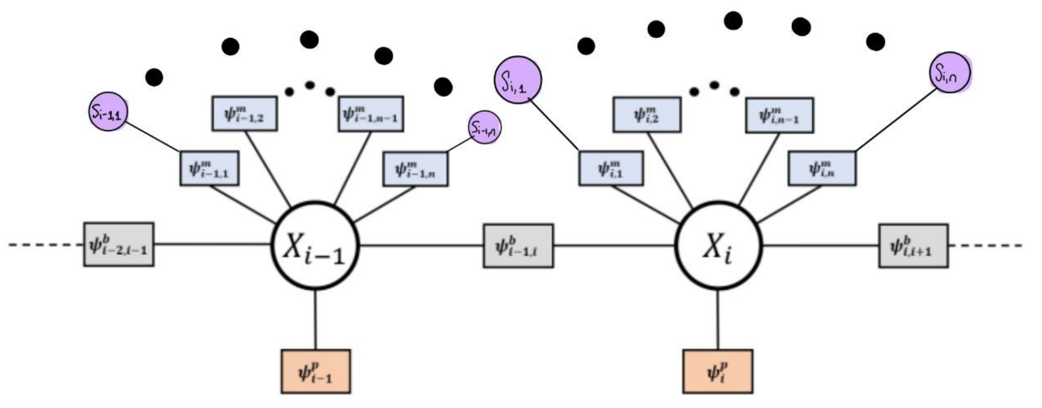


Figure 4.1: A FG with SC. The boxes represent the factors, whereas the circles represent the states to be estimated. Red boxes indicate prior factors, grey boxes indicate motion factors, and blue boxes indicate GNSS factors. Purple circles are switch nodes, connected to the GNSS factors. Each GNSS factor has its own switch state to be estimated.

In order not to make all switch variables $s_{i,j}$ zero, another term will be added to objective function, as follows:

$$\psi_{initialSwitch} = ||\gamma_i - s_i||^2_{\Xi} \tag{4.3}$$

where γ_i is the initial switch variable value (generally 1) and Ξ is the covariance matrix associated with the switch variables of the factors in the specified position state.

As a result, the objective function of the FG with SC can be written as follows [30], [11]:

$$X, S = \underset{x,s}{\operatorname{argmin}} \ (\sum \psi_{prior} + \sum \psi_{dynamics} + \sum \psi_{GNSS} + \sum_i \psi_{switch} + \sum \psi_{initialSwitch}) \tag{4.4}$$

Thus, along with switch function term ψ_{switch} and the term that forces the switch variables not to become all zero $\sum_i \psi_{initialSwitch}$, two terms are added on top of the initial FG structure.

4.2 Gaussian Max Mixtures (GMM)

The typical FG example and SC were subject to a uni-modal Gaussian distribution, meaning that only one Gaussian distribution function is used to characterize the behavior of the receiver position or velocity state. Nevertheless, in some cases, especially when there is multipath error, uni-modal Gaussian distribution becomes insufficient when trying to capture the true nature of the data. A proposed solution to this problem is to use the Gaussian mixture models, where multi-modal distributions are used (i.e. combination of more than one Gaussian distributions with different mean values μ, and variances σ^2 or covariance matrices Λ). One type of multi-modal Gaussian mixture models is the sum of multiple Gaussian components, as following [11]:

$$p(z_j|x) = \Sigma_j w_j \mathcal{N}(\mu_j, \Lambda_j) \tag{4.5}$$

where w_j is the weight for the j^{th} component to normalize the probability function $p(z_i|x)$. Nevertheless, the sum operator is not useful and usually complicates the operation since the logarithm of sums is not equal to the sum of logarithms (unlike the logarithm of products is equal to the sum of logarithms, as in (3.16)), even though the sum of Gaussian distributions may be sufficient to characterize the data. One solution to this problem is to use the *max* operator instead of *sum*, thus the distribution becomes [9]:

$$p(z_i|x) = \max_j w_j \mathcal{N}(\mu_j, \Lambda_j) \tag{4.6}$$

where $\mathcal{N}(\mu_j, \Lambda_j)$ for every j represents a different Gaussian distribution in (4.6). By using (4.6), one can put it inside the logarithm, and find the MAP solution in a straightforward fashion. One can see the difference between the bi-modal Gaussian, max-mixture and sum-mixture Guassian multi-model (bi-modal in this case) distributions in Figure 4.2.

4.3 Integration of Robust Estimation Techniques to FGs

In this chapter, two robust methods of SC and GMM were explained, nevertheless in order to integrate them to the already existing FG scheme, further steps are needed, which can be found below.

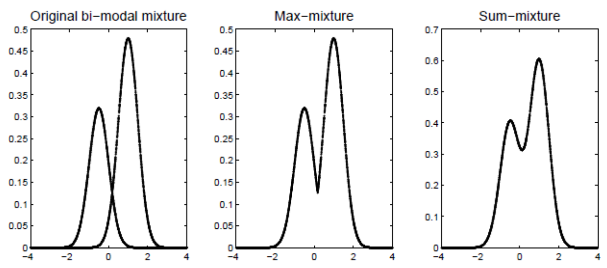


Figure 4.2: Comparison of Gaussian bi-modal, sum-mixture and max-mixture distributions [9]

4.3.1 Integration of Switch Constraints (SC)

As it was mentioned in (4.4), there are two additional terms regarding SC to be added to the objective function compared to the typical FG's, which is stated in (3.39). Those are:

1. ψ_{switch}: This term modifies the pseudorange residual, which is denoted as b_{GNSS} previously, by scaling it with $\psi(s)$, which depends on the switch variable. Formulated in (4.1).

2. $\psi_{initialSwitch}$: This terms acts as a regularization terms for the switch variables and typically pulls them towards the initial estimate of γ, which is usually 1, to prevent all switch variables from being 0. Formulated in (4.3).

In order to properly add those terms to the information matrix A, one may need to apply linearization process to both of the terms, since they are not linear in Δx, as it was shown in 3.2.2 to the factors in FG. The procedure for each term and adjusted information matrix A can be found below.

Adding ψ_{switch} Term

The switch function ψ_{switch} modifies the residuals and adjusts them if they cause a lot of errors in the estimation process. The formulation of ψ_{switch}, which was given in (4.1), includes the pseudorange residuals, which was elaborated in (3.47) and (3.48). Combining the terms, the ψ_{switch} term could also be written as:

$$\psi^i_{switch} = ||\Psi(s_{i,j}).(h_{i,j}(x_i) - \zeta_{i,j})||^2_{\Sigma_i} = ||\Sigma_i^{-1/2}\Psi(s_{i,j}).(h_{i,j}(x_i) - \zeta_{i,j})||^2_2 \tag{4.7}$$

Nevertheless, it must be noted that the information matrix does not only include Δx_{i-1} and Δx_i variables anymore, but they also include $\Delta s_{i,j}$, and $\Psi(s_{i,j})$ depends on the switch variable $s_{i,j}$. Thus, one also needs to find the coefficients for $\Delta s_{i.j}$. Using the Taylor expansion in (3.17) and finding the Jacobian in (3.18), one can also calculate the coefficients in the information matrix A for the switch variables $s_{i,j}$. In order to find the Jacobian for the switch variable $\Delta s_{i,j}$, let's denote it as $J_{S_{i.j}}$ one can take the partial derivative of the function inside the norm in (4.7).

$$J_{S_{i,j}} = \frac{\partial(\Psi(s_{i,j})(h_{i,j}(x_i) - \zeta_{i,j}))}{\partial s_{i,j}} = \frac{\partial \Psi(s_{i,j})}{\partial s_{i,j}}(h_{i,j}(x_i) - \zeta_{i,j})) \tag{4.8}$$

Since $(h_{i,j}(x_i) - \zeta_{i,j})$ does not depend on $s_{i,j}$, one can take it out of the partial derivative. Switch function $\Psi s_{i,j}$ was defined in (4.2), when $a = 1$ as it was shown in [30] this parameter provides a suitable switch function, its derivative becomes 1 when the switch variable is between 0 and 1, otherwise 0. Thus, $J_{S_{i,j}}$ becomes $(h_{i,j}(x_i) - \zeta_{i,j})$ in the linear region of $\Psi s_{i,j}$, and 0 if outside. Thus, the coefficients for $\Delta s_{i,j}$ in the information matrix A related to ψ^i_{switch} term, denoted as $A_{s,switch}$ can be found below:

$$A_{s,switch} = \Sigma_i^{-1/2} J_{S_{i,j}} = \begin{cases} \Sigma_i^{-1/2}(h_{i,j}(x_i) - \zeta_{i,j}) & : 0 \leq s_{i,j} < 1 \\ 0 & : o/w \end{cases} \tag{4.9}$$

For the coefficients of Δx_i in the ψ^i_{switch} term, one can again calculate the Jacobian using partial derivative, let's call it J_{X_i}.

$$\begin{aligned} J_{X_i} = \frac{\partial(\Psi(s_{i,j})(h_{i,j}(x_i) - \zeta_{i,j}))}{\partial x_i} &= \Psi(s_{i,j})\frac{\partial(h_{i,j}(x_i) - \zeta_{i,j})}{\partial x_i} \\ &= \Psi(s_{i,j})\frac{\partial(h_{i,j}(x_i))}{\partial x_i} \end{aligned} \tag{4.10}$$

where the term $\frac{\partial h_{i,j}(x_i)}{\partial x_i}$, Jacobian was defined as $H_{i,j}$, the observation matrix related to the GNSS measurements. It can easily be realized that the covariance matrix $\Sigma_i = R_{i,j}$ is the observation covariance matrix in (3.47), thus the coefficients for the Δx_i related to switch term ϕ_{switch}, denoted as $A_{x,switch}$ becomes:

$$\begin{aligned} A_{x,switch} &= \Sigma_i^{-1/2} J_{X_i} = R_{i,j}^{-1/2}\Psi(s_{i,j})H_{i,j} \\ &= \Psi(s_{i,j})R_{i,j}^{-1/2}H_{i,j} \end{aligned} \tag{4.11}$$

Applying the change of variables, written in (3.50), one can obtain:

$$A_{x,switch} = \Psi(s_{i,j})A_{GNSS} \tag{4.12}$$

Similarly, the vector b_{switch} can be found as below:

$$b_{switch} = \Psi(s_{i,j})b_{GNSS} \tag{4.13}$$

where SC scales the pseudorange measurements and related coefficients by the switch function, $\Psi(s_{i,j})$.

Adding $\psi_{initialSwitch}$ Term

For the $\psi_{initialSwitch}$ term, defined in (4.3), one needs the initial switch value of γ (generally 1) and the covariance matrix associated with the switch variables of the factors in the specified position state, Ξ.

$$\begin{aligned}\psi_{initialSwitch} &= ||\gamma_i - s_{i,j}||^2_{\Xi} \\ &= ||\Xi^{-1/2}\gamma_i - \Xi^{-1/2}s_{i,j}||^2_2\end{aligned} \tag{4.14}$$

The term s_i can be written as:

$$s_{i,j} = s^0_{i,j} + \Delta s_{i,j} \tag{4.15}$$

where $s^0_{i.j}$ is the current switch value and $\Delta s_{i,j}$ the difference, where the coefficients in the information matrix A depends on $\Delta s_{i,j}$. Thus, the $\psi_{initialSwitch}$ becomes:

$$\begin{aligned}\psi_{initialSwitch} &= ||\Xi^{-1/2}\gamma_i - \Xi^{-1/2}(s^0_{i,j} + \Delta s_{i,j}))||^2_2 \\ &= ||\Xi^{-1/2}\gamma_i - \Xi^{-1/2}s^0_{i,j} - \Xi^{-1/2}\Delta s_{i,j})||^2_2 \\ &= ||\Xi^{-1/2}\Delta s_{i,j} - \Xi^{-1/2}(\gamma_i - s^0_{i,j})||^2_2\end{aligned} \tag{4.16}$$

There is not a term related to Δx_i in (4.16), thus in the information matrix A, there will be zeros. The terms related to $\Delta s_{i,j}$ are the coefficients of it, as shown in (4.16). The rest of the terms, which are $\Xi^{-1/2}(\gamma_i - s^0_{i,j})$, form the vector $b_{initialSwitch}$. Thus:

$$\begin{aligned}A_{initialSwitch} &= \Xi^{-1/2} \\ b_{initialSwitch} &= \Xi^{-1/2}(\gamma_i - s^0_{i,j})\end{aligned} \tag{4.17}$$

In conclusion, the final form of matrix-vector system of $[A|b]$ including the SC terms is as follows:

$$[A|b] = \left[\begin{array}{ccc|c} A_{prior} & & & b_{prior} \\ F_Q & I_Q & & b_{dynamic} \\ & A_{GNSS} & & b_{GNSS} \\ & A_{x,switch} & A_{s,switch} & b_{switch} \\ & & A_{initialSwitch} & b_{initialSwitch}\end{array}\right] \tag{4.18}$$

The columns of A correspond to $[\Delta x_{i-1}, \Delta x_i, \Delta s_i]$, indicate the states to be determined in the FG structure. Again, this system can be solved via the multi-QR factorization, detailed in the last chapter.

4.3.2 Integration of Gaussian Max-Mixtures (GMM)

Unlike SC, GMM does not require an additional term to be added to the objective function of the typical FG. GMM is interested in finding the parameters for the multi-modal distributions that define the characteristics of the faulty and non-faulty data.

In GNSS data processing, when using bi-modal GMM, it is assumed that each observable can be modeled using two independent distributions: one distribution defines the data which is free of the outliers, while a second distribution represents the faulty data (i.e., the null hypothesis). The null hypothesis can be modeled as a Gaussian distribution centered at the mean of the error-free observable distribution, but with a larger variance caused by the errors such as multipath [11].

In the FG that is defined in the previous function, the estimated parameters are user position ($[\Delta x, \Delta y, \Delta z, -\Delta b]$) and velocity states ($[\Delta \dot{x}, \Delta \dot{y}, \Delta \dot{z}, -\Delta \dot{b}]$). In order to characterize the user position and the velocity, an observation covariance matrix $R_{i,j}$ is assigned in (3.47) for the i^{th} state node and j^{th} satellite. $R_{i,j}$ is a diagonal 2×2 matrix where the position variance is defined on the first diagonal and the velocity variance is defined on the second diagonal element. GMM aims to select the parameters that result in less error in each satellite observed measurement, which are position and velocity measurements of each satellite.

In the cases where uni-modal Gaussian distribution is used, the parameters are selected considering the non-faulty data, so they may not adequately represent the outliers. GMM defines another Gaussian distribution with another set of variances for position and velocity for them to define faulty data. It is recommended that the latter distribution can be modeled as a Gaussian distribution centered at the mean of the error-free observable initial distribution, but with a larger variance [11]. The variance parameters for position and velocity belong to the satellite measurements, which are the pseudorange and satellite velocity.

In the scope of this work, two sets variances were defined for position and velocity: $\sigma_{x,1}, \sigma_{x,2}$ and $\sigma_{v,1}, \sigma_{v,2}$. It is assumed that $\sigma_{x,1}, \sigma_{v,1}$ defines the characteristics of non-faulty measurement and $\sigma_{x,2}, \sigma_{v,2}$ defines the faulty measurement's characteristics. After pseudorange and velocity between the satellite and the user is measured and received by the receiver, the residual is calculated as follows:

$$e_i = z_i - h(x_i) \tag{4.19}$$

where e_i is the residual vector between z_i and $h(x_i)$, z_i is the vector of measurements of pseudorange or velocity of the i^{th} satellite, $h(x_i)$ is the vector of expected quantity of the measurements. In GMM, aim is to select the parameters that minimizes the error, where it can be formularized using the Mahalanobis distance, as follows:

$$(\hat{\sigma}_x, \hat{\sigma}_v) = \min_{\sigma_x, \sigma_v} \frac{1}{2}(e_i \Sigma_i^{-1} e_i^T) - \log(w_i) \tag{4.20}$$

where Σ_i is the $2{\times}2$ diagonal covariance matrix consisting the variances for pseudorange ($\sigma_{x,1}$ or $\sigma_{x,2}$) and velocity ($\sigma_{v,1}$ or $\sigma_{v,2}$). The minimizing variances for pseudorange and velocity values were selected and put in the covariance matrix Σ_i to proceed with the addition of the measurement factor to the FG. The aforementiones covariance matrix is actually equal to the $\Sigma_i = R_{i,j}$ noise covariance matrix, and it is already used in the elimination algorithm of the node of the FG, as from (3.47), to (3.50).

$w_i, i \in \{1{,}2\}$ is the weight assigned to the probability to the non-faulty and faulty data distribution, given in (4.6). Since it is a probability distribution, the sum of all the weights (total of 2 in this case, since it is bi-modal distribution) is equal to 1.

$$\sum_{i}^{2} w_i = w_1 + w_2 = 1 \tag{4.21}$$

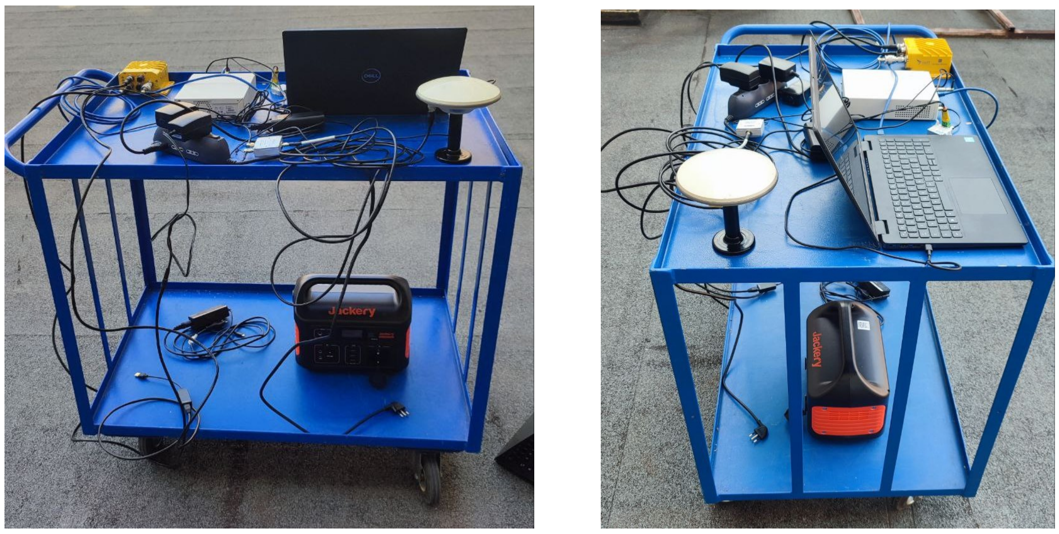
It is a common practice to choose the weight of non-faulty distribution higher (around 0.9) and faulty data to be lower (around 0.1) since the probability of obtaining too much faulty data is usually lower, nevertheless the weights can be also adjusted case by case.

So far, the integration of robust estimation methods of SC and GMM, and their integration to the FG structure are elaborated in this chapter. In the next chapter, the field testing that includes the experimental dataset collection and the results of the robust estimation methods are discussed. In addition to applying SC and GMM robust methods individually, a combination of them (denoted as SC + GMM) is also applied and the results were reported.

Chapter 5

Field Testing

In the previous chapters, from Chapter 2 to Chapter 4, a general overview of satellite navigation systems and how to find a PVT solution of the user given the satellite measurements using FGs and robust estimation methods using SC and GMM were given. In order to test the theory, an experimental set-up is constructed to collect GNSS data to be processed by the algorithms written. Thus, in order to collect GNSS data and complete the field testing, an experimental hardware setup is constructed and data was collected on October 23rd, 2024. One can see the experimental setup in Figure 5.1 (a) and (b).



(a) Experimental setup front view with power supply, antenna, RTK device, GNSS receiver

(b) Experimental setup side view (same equipment)

Figure 5.1: Experimental Setup to collect GNSS data, photos taken on Oct. 23, 2024.

The equipment was placed on a moving tray to make it easier to collect data in an open environment. The hardware setup consists of an antenna, a GNSS receiver, an Real Time Kinematic (RTK) device that provides precise positioning results, a power supply and computers to initialize and control the data collection. The GNSS receiver is the yellow device on the blue cart in Figure 5.1, Swift Duro by Carnegie Robotics, whose manual can be found in [10], [31]. It supports the following frequency bands and constellations: GPS L1/L2, GLONASS G1/G2, BeiDou B1/B2 and Galileo E1/E5b [31]. Even though only GPS constellation is utilized for this thesis work, it can be applied to other types of constellations as well.

Figure 5.2: Swift Duro GNSS receiver by Carnegie Robotics [10]

The RTK corrections were provided by the SPIN3 GNSS, which is the Interregional GNSS Positioning Service (SPIN3 GNSS) of the Piedmont Region, the Lombardy Region and the Autonomous Region of Valle d'Aosta [32]. This interregional network consists of 39 permanent GNSS stations distributed homogeneously across the Piedmont, Lombardy and Aosta Valley regions in Italy, and equipped with multi-constellation geodetic receivers open to the use of GPS, GLONASS and GALILEO constellations [32]. It is also able to provide RTK corrections, which is used as a true trajectory on the scope of this thesis work.

Several datasets were collected in order to test the performance of the algorithms written. The datasets were collected both in the static and dynamic conditions, so that the extent of the robust estimation becomes fully understood. The datasets were collected in Costo Castelfidardo, near Politecnico di Torino on October 23rd, 2024. Two datasets were selected, one static and one dynamic, to test the robust estimation methods. The datasets were analyzed, acquisition and tracking stages were completed, and PVT estimation methods are performed to obtain results, which one can find below in the next two sections.

5.1 Static Dataset

For the static dataset, the moving cart with the necessary equipment was moved to an outdoor space where the data collection started. One can see the actual location of the static data collection in Figure 5.3 obtained from the RTK device. The aim is to obtain a result as accurate as possible, close to the solution of the RTK device which is plotted in Figure 5.3.

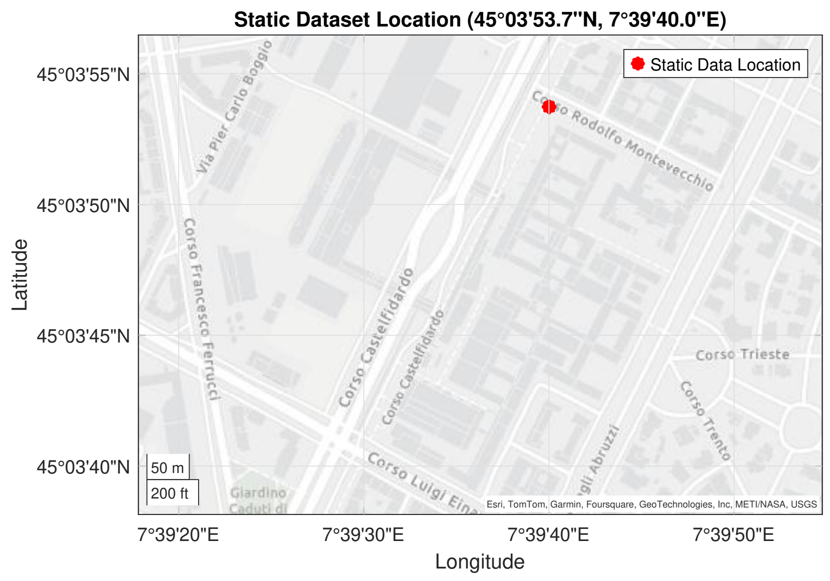


Figure 5.3: Static dataset location, near Politecnico di Torino, Corso Castelfidaro (45°03’53.7"N, 7°39’40.0"E)

Since the cart has not moved and was stable throughout the measurement duration, the single location point in Figure 5.3 will be taken as the true reference point when calculating errors. The true location spot of the static dataset obtained via Google Earth software can be also found in Figure 5.4 and Figure 5.5, where the environmental conditions can be better observed.

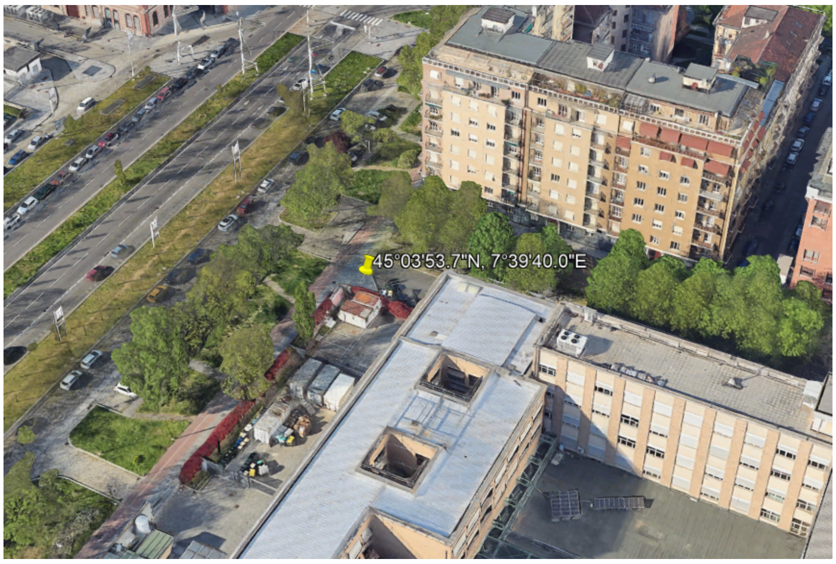


Figure 5.4: Static dataset location (yellow pin) on Google Earth, near Politecnico di Torino, Corso Castelfidaro (45°03’53.7"N, 7°39’40.0"E)

Figures 5.4 and 5.5 show the same geodetic location, from two different angles. It can be clearly seen that around the point of the true location, dense foliage and obstruction caused by the buildings can be observed; which are the characteristics of an urban area. These properties of urban areas may cause the signals to be reflected, or scattered through the reflective surface or the tree leaves; which ultimately causes multipath or even signal blockage; and results in erroneous data. The effects of these properties will be discussed later, along with the results of the filters.

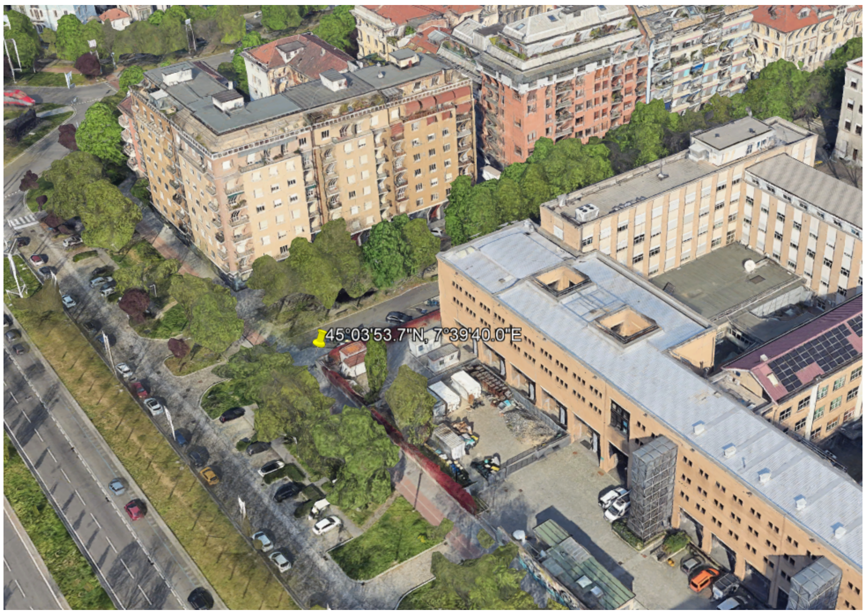


Figure 5.5: Static dataset location (yellow pin) on Google Earth, near Politecnico di Torino, Corso Castelfidaro (45°03’53.7"N, 7°39’40.0"E)

The static dataset collected consists of 2700 epochs to be processed, by 6 different methods:

1. LS
2. Extended Kalman Filter (EKF)
3. FG
4. FG with SC (denoted as only SC)
5. FG with GMM (denoted as only GMM)
6. the combination of SC and GMM filters together (denoted as SC + GMM)

It is worth noting that, to the best of the author’s knowledge, a study that combined both of the SC and GMM at the same time has not been found in the literature so far, thus it is an original contribution of this thesis. All of the 2700 epochs were processed by the aforementioned methods, and the time-series, Cumulative Distribution Function (CDF) plots and percentile bars are given, along with a discussion. One can also find the parameters that are used in the processing of the static dataset in Table 5.1.

Parameter Name	Method	Value
Epoch number	all	2700
Epoch Rate	all	0.1
Window size (ws)	FGO, SC, GMM, SC + GMM	10
Position variance ($\sigma^2_{x,1}, unit : m^2$)	all	20
Second positional variance ($\sigma^2_{x,2}, unit : m^2$)	GMM, SC + GMM	100
Velocity variance ($\sigma^2_{v,1}, unit : m^2/s^2$)	all	0.01
Second velocity variance ($\sigma^2_{v,2}, unit : m^2/s^2$)	GMM, SC + GMM	0.1
Initial Switch Value (γ)	SC, SC + GMM	1
Switch Variable Covariance Matrix (Ξ)	SC, SC + GMM	$diag(1)$

Table 5.1: Parameters used in the Static Dataset Processing

For the FG based algorithms, a window size (ws) of 10 is used throughout the processing; since it was observed that higher window size values did not provide a significant improvement, thus making 10 an optimal choice. The second velocity and position variances for GMM based methods were chosen at least 5 times or their original values to represent erroneous measurements for the static dataset.

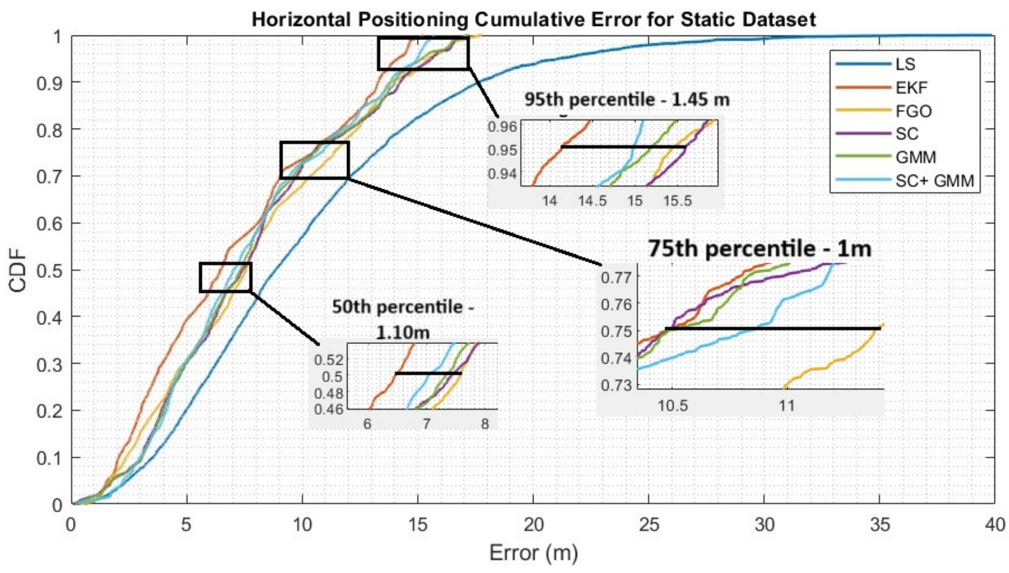


Figure 5.6: Horizontal (E+N) positioning error CDF of the static dataset in ENU reference frame

In Figure 5.6 and 5.7, one can observe the horizontal and vertical error CDFs of the static dataset in ENU reference frame respectively. For horizontal errors, the LS algorithm alone results in a maximum error of 39.85 m, as it can be inferred from the Figure 5.6. Nevertheless, EKF and FGO reduced the maximum error in horizontal positioning to 15.23 m and 17.78 m respectively. Robust estimation methods; SC, GMM and the combination of both (SC + GMM) further reduce the horizontal error of FGO as well, which can be seen on the zoomed portions of the percentiles on Figure 5.6. The maximum horizontal error of the SC algorithm was found to be 17.20 m, whereas for GMM the maximum horizontal error was 17.19 m. Nevertheless, the combination of SC and GMM algorithms gave us the most improved result in terms of the maximum error with 15.71 m (except EKF), a total of 1.49 m decrease and 9% improvement compared to the standalone FG implementation.

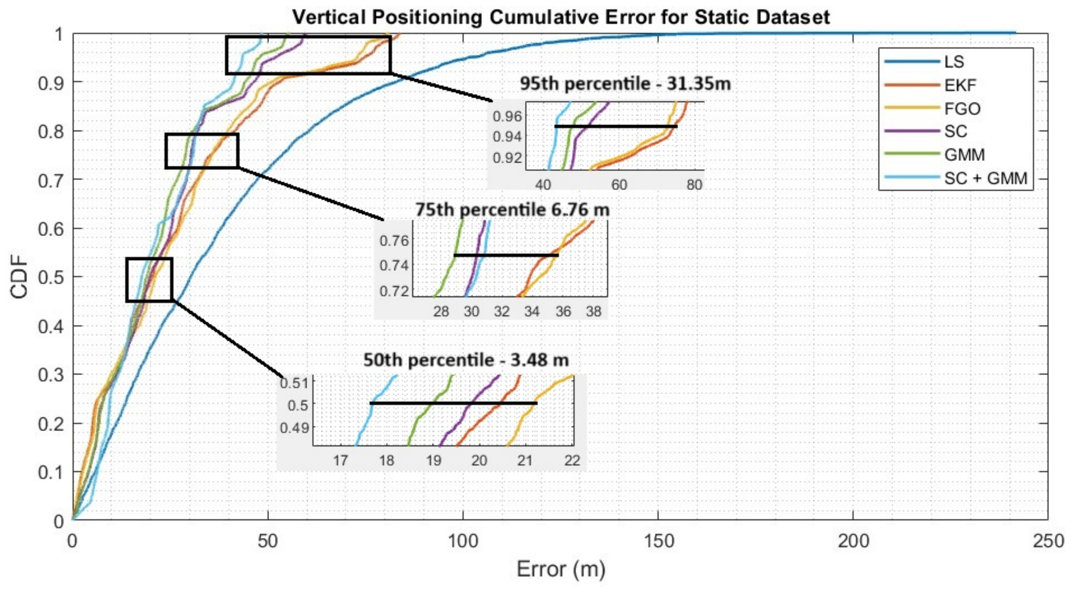


Figure 5.7: Vertical (U) positioning error CDF of the static dataset in ENU reference frame

If one checks the vertical error CDF in Figure 5.7, a similar plot can be seen. However, it can be noticed that the vertical errors were more than the horizontal errors for this static dataset. The maximum error for LS resulted in 241.7 meters. Nevertheless, the maximum error of EKF and FGO filters were found to be 83.89 m and 81.04 m respectively. It is also worth noting that even though EKF's performance was the best in horizontal error in Figure 5.6, in vertical errors except for the LS algorithm, EKF's total CDF error is the largest, with FGO following. Moreover, it is still possible to clearly observe the improvements provided by the robust estimation methods. The maximum vertical error of SC and GMM are 59.84 m and 55.6 m respectively, where the combination of SC and GMM performs even better with a maximum error of 48.55 m. The performances of robust estimation methods (SC, GMM and SC + GMM) were similar for other percentiles as well (such as 50 and 75) for vertical errors, reducing the error of the standalone FGO, which shows the effectiveness of the robust estimation methods. It can be calculated that the combination of SC and GMM (with a maximum vertical error of 48.55 m) provided a decrease of approximately 60% compared to FGO's solution (with a maximum error of 81.04 m).

A more detailed analysis can be done on time series of the horizontal and vertical errors of the static dataset, which are in Figure 5.8 and 5.9. Since the main objective of this thesis is to show the improvement of robust estimation methods compared to FGO, the Figures 5.8 and 5.8 only provides the plots for FGO, SC, GMM and SC + GMM and the comparisons will be made against FGO's performance.

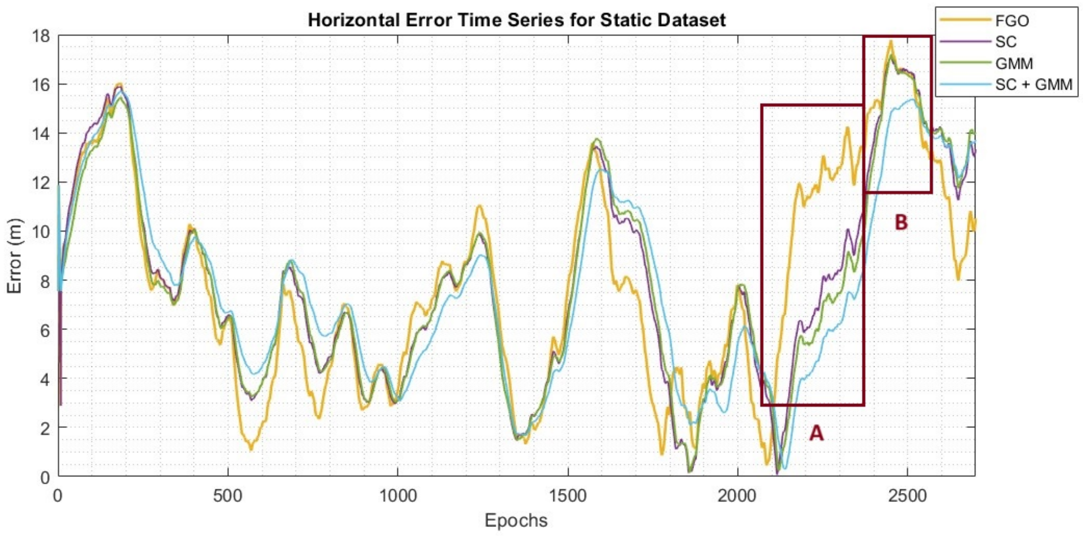


Figure 5.8: Horizontal (E+N) time series error for static dataset for 2700 Epochs, in ENU reference frame

It can be seen that the horizontal errors in time series for FGO and the robust methods follow a similar trend most of the time in Figure 5.8. Nevertheless, in the region represented with a rectangle A in Figure 5.8, the error of FGO is significantly higher than the rest, wherein the region B in Figure 5.8, FGO's error was 13.04 m and SC + GMM's error at in region B was found to be 5.94 m, a decrease of more than 54%. Moreover, in region B, the FGO's, SC's and GMM's error were approximately the same around 17 m, FGO's being 17.78 m, however SC + GMM's error was at 14.96 m in region B, indicating a difference of 2.82 m and a decrease of 15.9%, which is an improvement in the performance.

For the vertical errors in time series, the performance improvements thanks to the robust estimation methods even become more obvious in Figure 5.9. In the regions indicated as C, D and E in Figure 5.9, the FGO's errors peaked comparably higher than the robust estimation methods' peaks. In region C, FGO's error is around 38.47 m whereas SC + GMM's peak is around 21.38 m, which is a 39% improvement. Similarly in region D, FGO's error is around 52.68 m whereas the SC + GMM's peak is at 21.62 m, which indicates a 59% performance improvement. In region E, FGO's error is the highest being 81.05 m and SC + GMM's error is at 46.25 m, which also indicates a 43% performance improvement. It can be said that the robust estimation methods become more obvious and

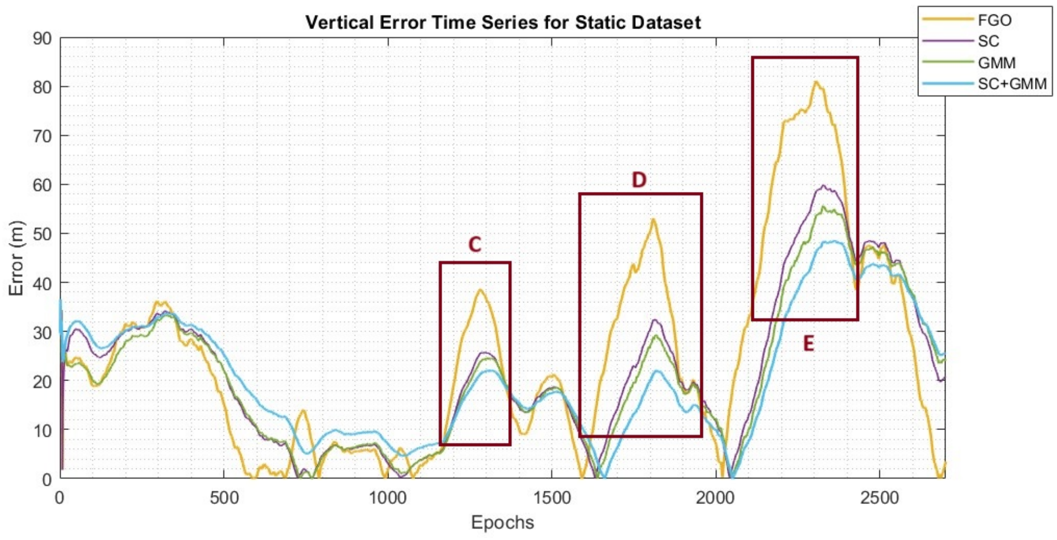


Figure 5.9: Vertical (U) time series errors for static dataset for 2700 Epochs, in ENU reference frame

effective in vertical positioning and decrease the errors more compared to the horizontal positioning results, as in Figure 5.8.

One can see the bar plots for horizontal and vertical errors of the static dataset in ENU reference frame in Figure 5.10 and Figure 5.11 respectively. The reported percentiles are 25, 50, 75, 95 and 100 (which is the maximum error). The LS method is excluded and only EKF, FGO and robust estimation methods (SC, GMM, SC + GMM) results were provided since the errors of LS are significantly higher than the others' as already shown in Figure 5.6 and Figure 5.7. Thus, there would be no need for further comparison against LS in the bar plots in Figure 5.10 and Figure 5.11.

As it can be seen in Figure 5.10, EKF performs the best in terms of horizontal errors. On the other hand, compared to FGO only, the least amount of errors belongs to the combination of SC + GMM method (except the 75th percentile, with a small margin) among the robust estimation methods. Nevertheless, SC and GMM alone also performs well, decreasing the overall error in almost all percentiles. In 25th percentile, the errors are more or less similar for FGO and robust estimation methods, with FGO's error being a little less than the others' nevertheless since it is a very low percentile it may not reflect the main characteristics of the filtering abilities of the methods. It can be seen that the robust estimation methods like SC, GMM and especially the combination of SC and GMM helped reduce the horizontal error.

The difference between the results of FGO and robust estimation methods becomes more obvious in vertical error percentile bar plots in Figure 5.11. As it was explained in the vertical error CDF in Figure 5.7 and vertical error time series in Figure 5.9, the EKF and FGO perform relatively more poorly in vertical axis compared to horizontal axis, where in horizontal axis the maximum error of FGO was 17.78 m whereas the maximum error in vertical axis of FGO was found to be 81.04 m. Thanks to the robust estimation

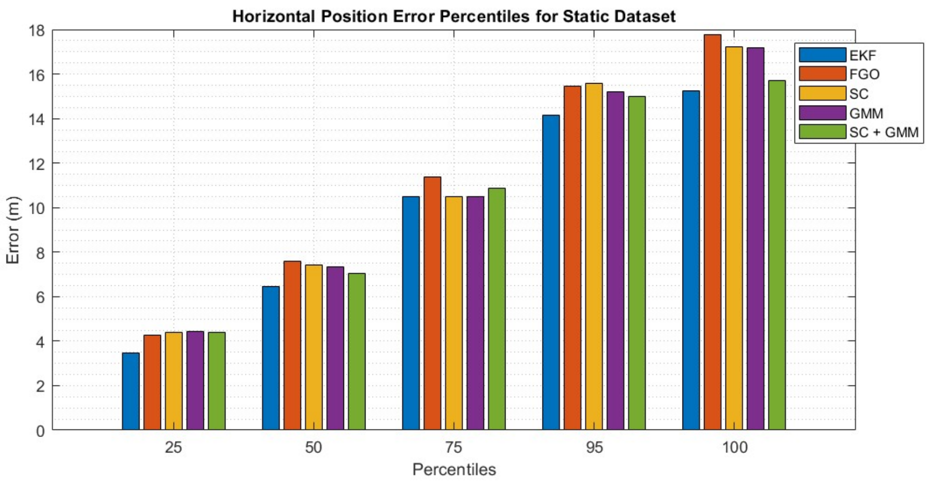


Figure 5.10: Percentiles for horizontal errors of the static dataset in ENU reference frame (for 25, 50, 75, 95 and 100th percentiles)

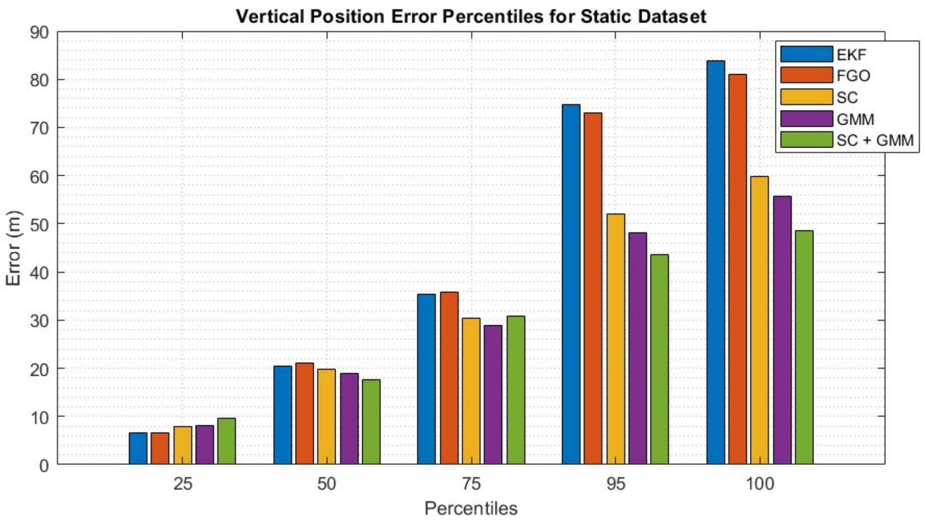


Figure 5.11: Percentiles for vertical errors of the static dataset in ENU reference frame (for 25, 50, 75, 95 and 100th percentiles)

methods, a substantial decrease can be observed in almost all percentiles in vertical axis in Figure 5.11. In 95th percentile, SC method decreased the error of FGO's from 72.94 m to 52.02 m, whereas in the same percentile GMM managed to decrease FGO's error to 48.09 m, and the combination of SC and GMM succeeded in reducing the error to 43.50 m; performing the best among all robust estimation methods. It can be said that, the EKF and FGO could not capture the true behavior of the static data in vertical axis and robust estimation methods managed to mitigate the errors especially in terms of the altitude.

It was also mentioned that the location where the static dataset was collected (45°03'53.7"N, 7°39'40.0"E), is prone to errors such as dense foliage and blocking caused by the nearby buildings, as they can be seen in Figure 5.4 and in Figure 5.5. Especially, the reflective surfaces on the buildings and scattering through the tree leaves may cause multipath and distort signals coming from the satellites and the PVT solution obtained by the GNSS receiver. It has been shown that the SC, GMM and the combination of both (SC + GMM) implemented on top of the FGO structure reduced the overall horizontal and vertical errors in the urban environment successfully.

In the next section, the results of the dynamic dataset will be analyzed.

5.2 Dynamic Dataset

For the dynamic dataset collection, the tray with the equipment was moved along the Corso Castelfidardo road, and the data was collected during the movement. One can see the trajectory of the dynamic dataset in Figure 5.12, collected by the RTK device. When the tracking results from the satellites were analyzed, it was seen that the dynamic dataset is composed of 1300 epochs, one can see the corresponding RTK trajectory in Figure 5.12. It should be noted that when the receiver gets closer to between the two bridges of Politecnico di Torino, signal distortions occurred in the RTK device, thus it should be kept in mind while evaluating the results obtained and filtered from the receiver.

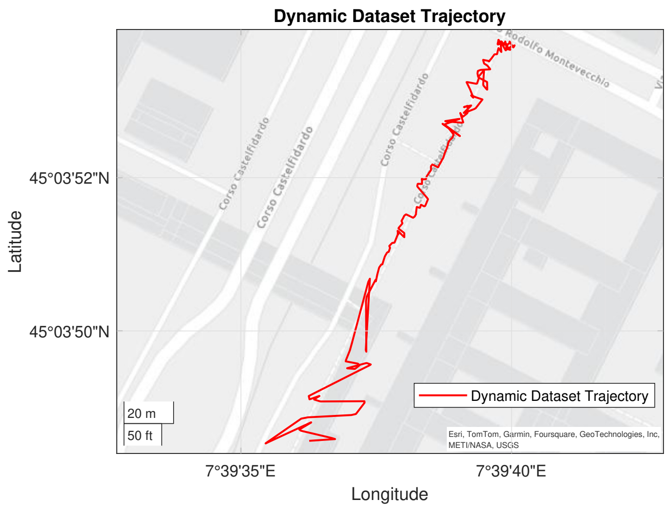


Figure 5.12: Dynamic Dataset RTK solution.

One can also see the dynamic trajectory on Google Earth application in Figure 5.13 and in Figure 5.14 in a slightly different angle.

The starting point of the dynamic dataset is the same as the static dataset location, which was (45°03'53.7"N, 7°39'40.0"E). The end point of the dynamic dataset corresponds to a location between two bridges of Politecnico di Torino in Corso Castelfidaro. As one

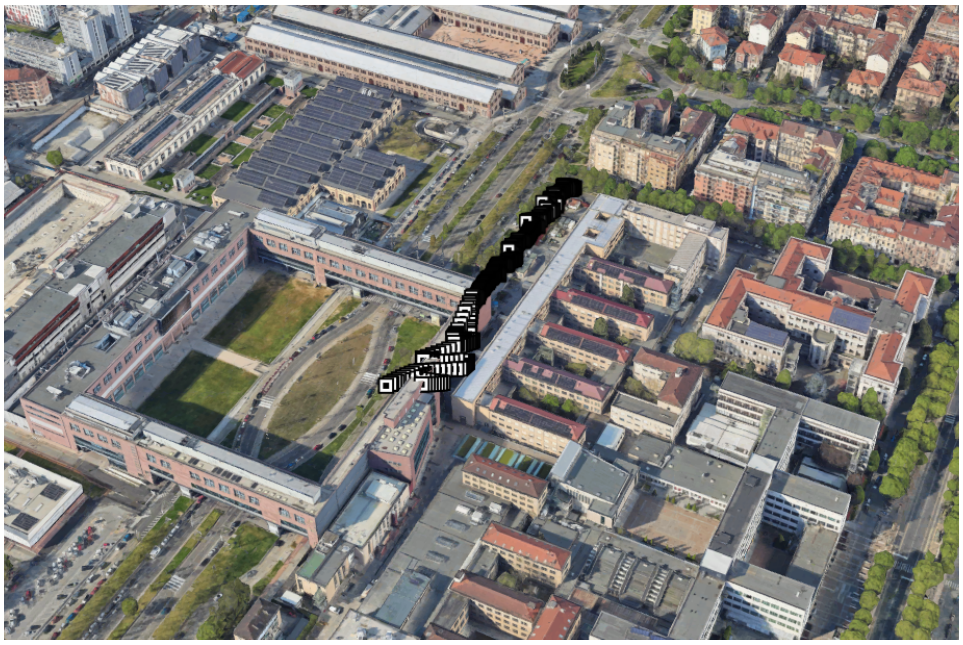


Figure 5.13: Dynamic dataset trajectory labeled with black boxes on Google Earth, near Politecnico di Torino, Corso Castelfidaro

can notice, the line of sight of the area, and especially the line of sight of the bicycle and walking road between two bridges is susceptible to be blocked by the nearby buildings, so it is not totally surprising that the dynamic trajectory became more erratic.

Similarly to the static dataset, six different methods are used to process the dynamic dataset, namely: LS, EKF, FGO, SC, GMM and the combination of SC and GMM. Unlike the static data set, in addition to the CDFs and error time series analyses, the trajectories found using the six filtering methods are separately shown in Figure 5.15 since the GNSS receiver was moving. One can also see the parameters used in the filters in the dynamic dataset processing in Table 5.2. Compared to the static dataset in Table 5.1, higher values of variances were selected for position and velocity. Since the receiver was moving this time, the uncertainty of erroneous measurement increases.

The results were successfully obtained, and the trajectories and the error plots are shown in the following figures separately. Firstly, one can see the six different trajectories obtained via different methods in Figure 5.15, along with the true trajectory obtained using the RTK device, plotted in black.

It can be easily realized from Figure 5.15 that, the shape of the trajectories of EKF, FGO, GMM and SC + GMM are similar to each other, and started to deteriorate when the receiver gets closer to the point between two bridges. However, it is possible to observe that in Figure 5.15, the trajectory obtained via SC follows a more straight trajectory compared to other estimation methods even though it slowly deviates through the end of the trajectory.

It is also possible to notice the gap between the true trajectory and the estimated

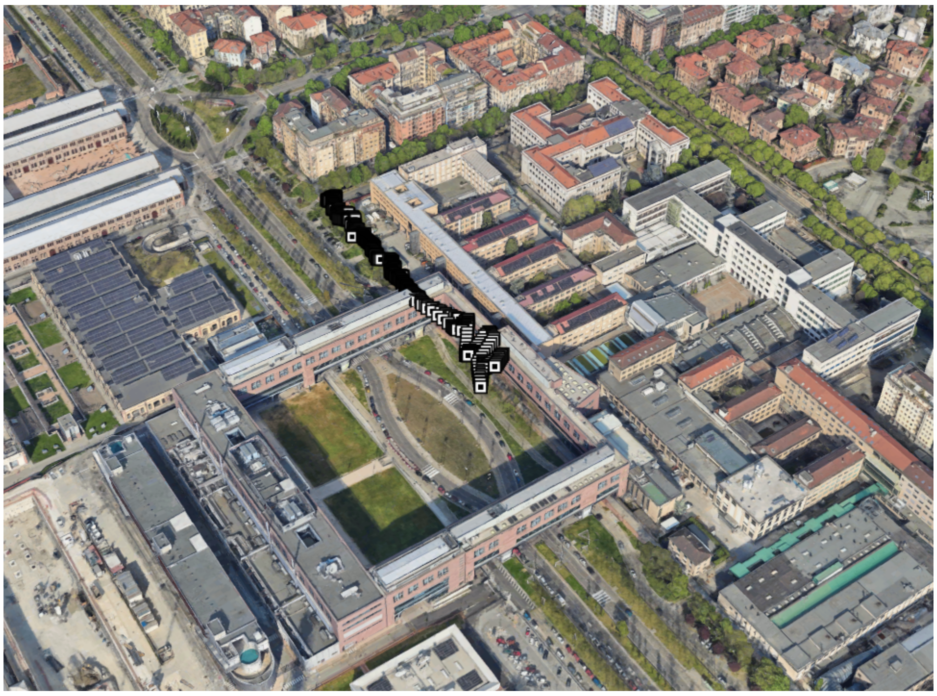


Figure 5.14: Dynamic dataset trajectory labeled with black boxes on Google Earth, near near Politecnico di Torino, Corso Castelfidaro

Parameter Name	Method	Value
Epoch number	all	1300
Epoch Rate	all	0.1
Window size (ws)	FGO, SC, GMM, SC + GMM	10
Position variance ($\sigma^2_{x,1}, unit : m^2$)	all	50
Second positional variance ($\sigma^2_{x,2}, unit : m^2$)	GMM, SC + GMM	200
Velocity variance ($\sigma_{x,1}v^2, unit : m^2/s^2$)	all	10
Second velocity variance ($\sigma_{v,2}v^2, unit : m^2/s^2$)	GMM, SC + GMM	50
Initial Switch Value (γ)	SC, SC + GMM	1
Switch Variable Covariance Matrix (Ξ)	SC, SC + GMM	$diag(1)$

Table 5.2: Parameters used in the Dynamic Dataset Processing

trajectories between the EKF and FGO methods, nevertheless this gap reduces thanks to the robust estimation methods such as GMM and SC + GMM; even though the shape of the trajectory remains similar.

To obtain more insights regarding the performance of the estimation methods, one should check the error plots regarding horizontal and vertical errors, CDF and time-series plots. One can check the horizontal CDF of the EKF, FGO, SC, GMM and the combination of SC and GMM methods in Figure 5.16. LS is not shown on the plot in Figure 5.16 due to the fact that its errors were already shown to be very large compared

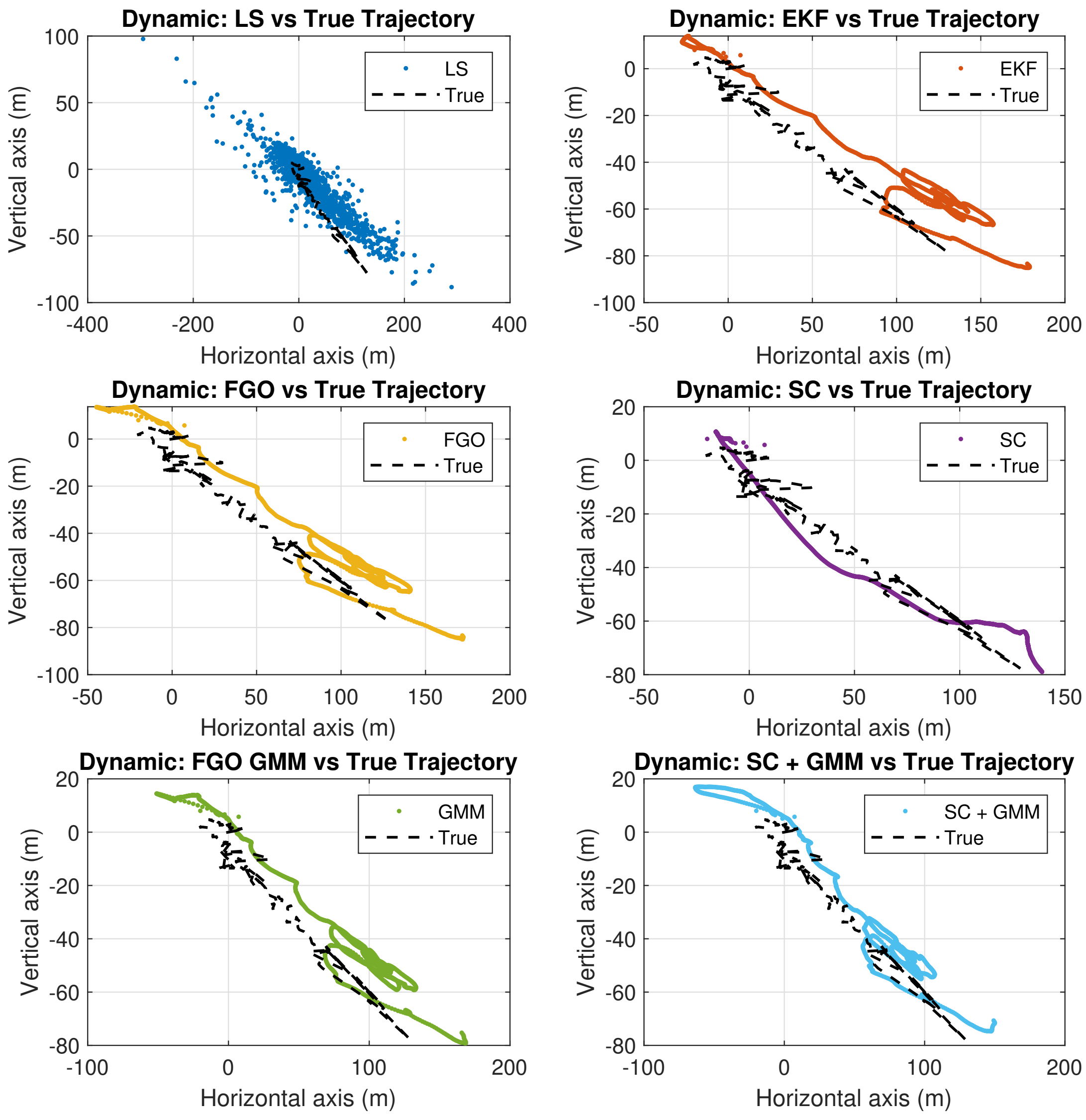


Figure 5.15: Trajectories obtained via the estimation methods for the dynamic dataset and the true trajectory. Used estimation methods are: LS, EKF, FGO, SC, GMM, SC + GMM.

to other methods in Figure 5.6.

As it can be seen in Figure 5.16, similarly to the static dataset, the SC + GMM performed the best among all 5 estimation methods of EKF, FGO, SC and GMM, in the $50^{th}, 75^{th}, 95^{th}$ percentiles. In the 95^{th} percentile, there is a 4 m difference between the SC

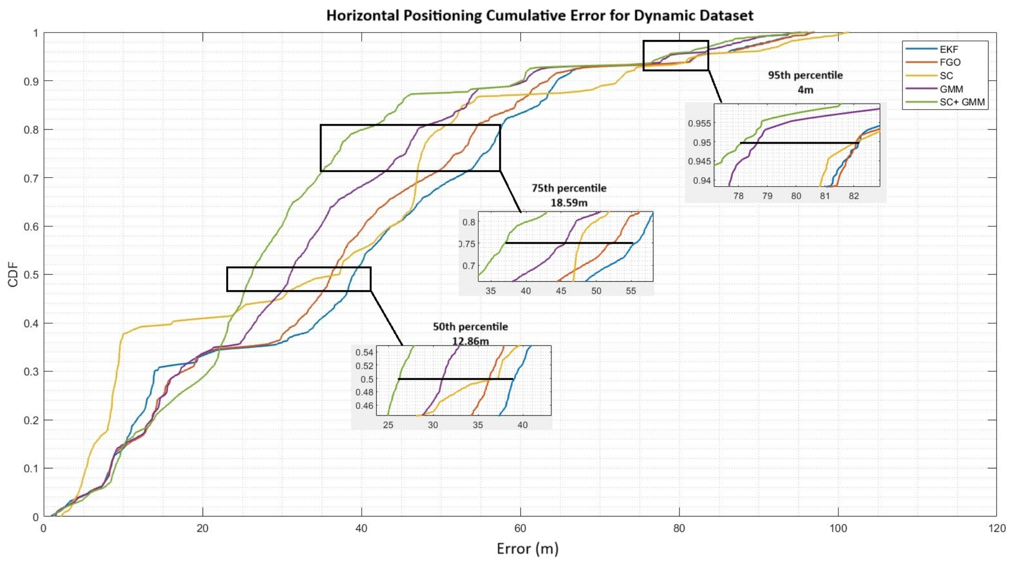


Figure 5.16: Horizontal CDF graph of EKF, FGO, SC, GMM and SC + GMM methods in ENU reference frame for dynamic dataset. The $50^{th}, 75^{th}, 95^{th}$ percentiles are zoomed and shown on the plot.

+ GMM method and the EKF (which is the worst performing one) in the horizontal error CDF's 95^{th} percentile, where SC + GMM's error is 78 m and EKF's error is around 82 m. GMM alone's performance in this percentile is also similar to SC + GMM's performance, with an error of 78.69 m. The maximum difference increases at the 75^{th} percentile, to 18.59 m, by a large margin. It can also be seen that the SC and GMM methods alone also perform better than the standalone FGO here, as almost in all percentiles, showing the effectiveness of the robust estimation methods. The error of FGO at the 75^{th} percentile is found to be 52.26, whereas SC's and GMM's errors are 47.57m and 45.58m respectively, indicating a performance improvement of 9% and 12% respectively at the 75^{th} percentile.

To specifically comment on the performance of SC: Even though its trajectory shape seems to be more straight than the others, because of the diversion towards the end of the trajectory, the maximum error increased and it may have caused extra errors. Nevertheless, it should also be noted that more than 40% of the horizontal errors of SC are within 20 m, higher than other estimation methods.

During static dataset horizontal estimation, EKF's performance was better than the other estimation methods, whereas during the dynamic set, EKF does not perform better than any other method here. It can be said that FGO based algorithms may work better than EKF under dynamic conditions based on horizontal errors. One can also see the vertical error CDF in 5.17.

Similarly to the static dataset, vertical errors are greater than the horizontal errors overall in the dynamic dataset, as it can be seen in Figures 5.6 and 5.7 for static, and Figures 5.16 and 5.17 for dynamic. Nevertheless, it is possible to observe more performance

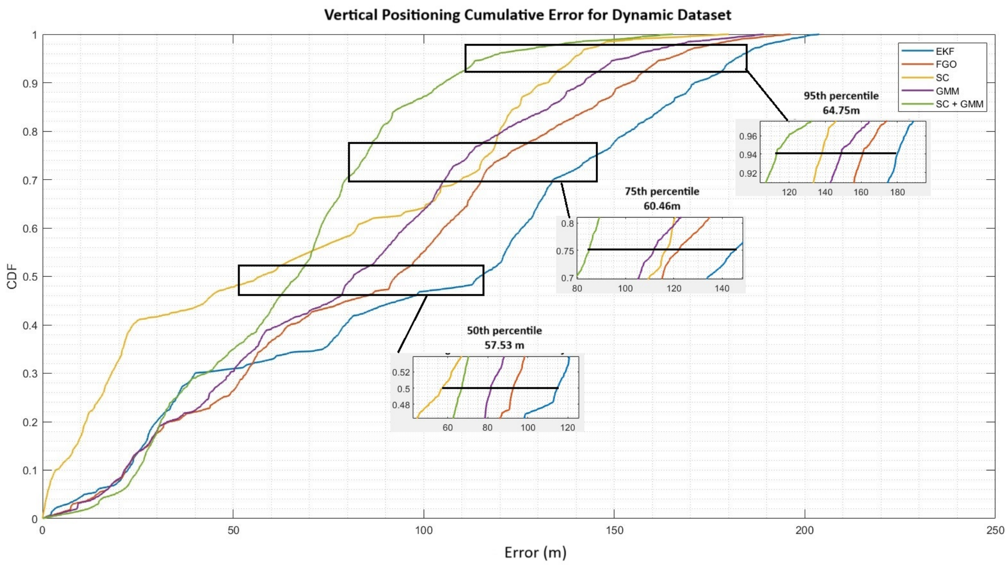


Figure 5.17: Vertical CDF graph of EKF, FGO, SC, GMM and SC + GMM methods in ENU reference frame for dynamic dataset.

improvements of robust estimation methods compared to FGO. For the vertical errors, SC and SC + GMM perform better in all percentiles shown in Figure 5.7. At the 95% percentile, the error of the SC + GMM is 116.17m, SC's error is 139.35m, GMM's error is 152.72m; whereas the FGO's error is 164.47 m. Thus, it is again possible to observe the improved performance thanks to the robust estimation methods. SC, GMM and SC + GMM improved standalone FGO's performance by 15%, 7% and 29% at 95th percentile respectively.

One can also check the horizontal and vertical error time series plots to gain more insight on the error distribution of different estimation methods at all the epochs, in Figures 5.18 and 5.19 respectively.

In Figure 5.18, one can see the horizontal errors in the dynamic dataset for the methods of EKF, FGO, SC, GMM and SC + GMM respectively. It can be seen in Figure 5.18 that EKF's horizontal errors are greater than FGO's, GMM's and SC + GMM's after the 200th epoch until the end. SC's performance seems to be the most improved until around the 650th epoch for both horizontal and vertical errors in the dynamic dataset in Figures 5.18 and 5.19, then most likely caused by the linear deviation which can be seen on the trajectory plot the error of SC increased rapidly. Nevertheless, all methods' errors increase in the last 100 epochs in both horizontal and vertical error time series plots, most likely to be caused by the satellite signal blockage.

In the first 200 epochs, SC + GMM's error in both horizontal and vertical error plot seems to be the most among the five estimation methods, nevertheless after 200 epochs, SC + GMM has one of the most consistent performances. Its errors are less than standalone FGO and GMM throughout all the epochs after the 200th. It can be said that after the

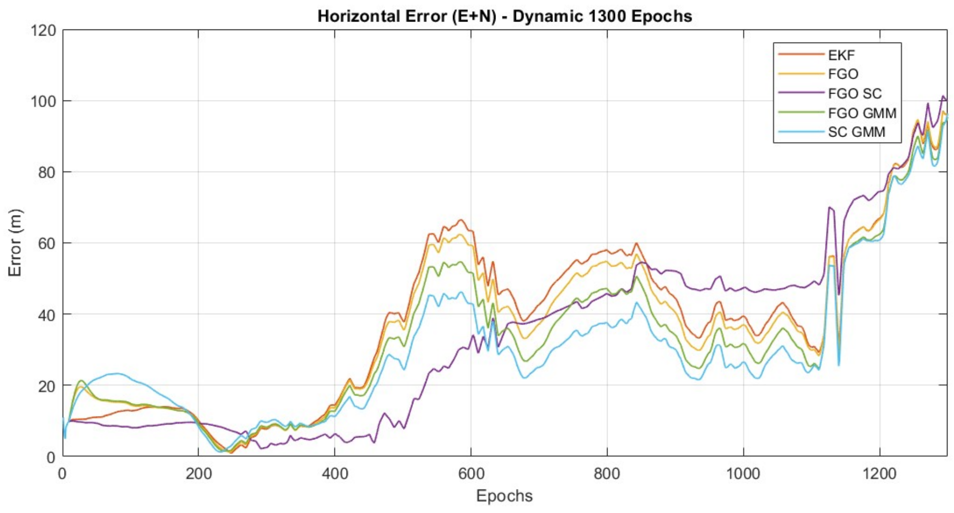


Figure 5.18: Horizontal (E+N) time series error for dynamic dataset for 1300 Epochs, in ENU reference frame

initial adjustment period, SC + GMM grasps the behavior of the trajectory. A similar explanation can be made for GMM as well, since it performed better than standalone FGO at all epochs as well, nevertheless still producing errors higher than SC + GMM's. It can be said that the integration of SC improved the performance of the estimation and made it more robust against potential errors, both in horizontal and vertical axes.

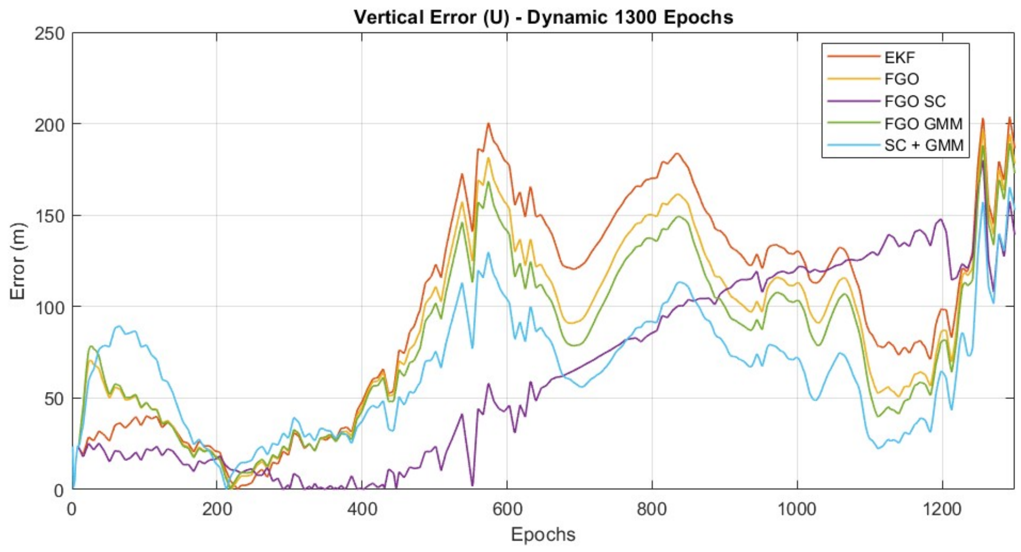


Figure 5.19: Vertical (U) time series error for dynamic dataset for 1300 Epochs, in ENU reference frame

In order to make a better comparison of all methods, bar charts representing the percentiles of horizontal and vertical errors of the dynamic dataset were provided in Figures 5.20 and 5.21 respectively. 25th, 50th, 75th, 95th and the 100th percentile results were shown as before for the methods of EKF, FGO, SC, GMM and SC + GMM. As it can be confirmed from the figure, the robust estimation methods, namely SC, GMM and the combination of them (SC + GMM), almost always decreased the error of FGO at all percentiles, except for SC at the 50th and the 100th percentile. But a similar situation was already observed with the static dataset's horizontal error results in Figure 5.10, where the SC's error was more than FGO's in 95th percentile and other's error values were similar to each other, so it was likely that a situation could occur for the dynamic dataset results as well. Nevertheless, it is possible to observe a significant decrease for the vertical error results, as in Figure 5.21.

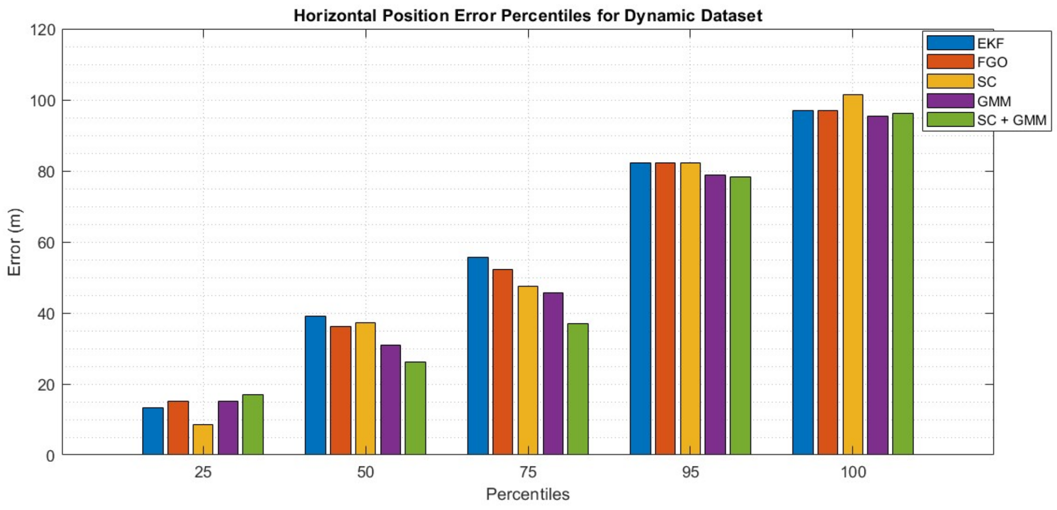


Figure 5.20: Percentiles for horizontal errors of the dynamic dataset in ENU reference frame (for 25, 50, 75, 95 and 100th percentiles)

In vertical axis errors, all robust estimation methods managed to decrease the error of FGO's, where the lowest errors belong to SC + GMM. At 95th percentile, FGO's error was 164.48m whereas SC + GMM's error is 116.18m, resulting in a 29.37% performance improvement. In the same percentile, SC's error was 139.35m and GMM's error is found to be 152.7m, improving the performance of standalone FGO once again. At 100th percentile, FGO's total vertical error is found to be 196.28m, whereas SC's, GMM's and SC + GMM's error were 180m, 189.2m and 165.37m respectively; improving the result by at least 7 meters. It is worth mentioning that SC has the smallest errors in 25th and 50th percentiles, meaning that 50% of SC's errors are within 57.24 meters, followed by SC + GMM, whose error value in 50th percentile is 67.02 meters, compared to FGO's 92.82 meters.

Considering all the results, it could be said that the robust estimation methods; namely SC, GMM and SC + GMM, improved the performance of FGO in GNSS applications,

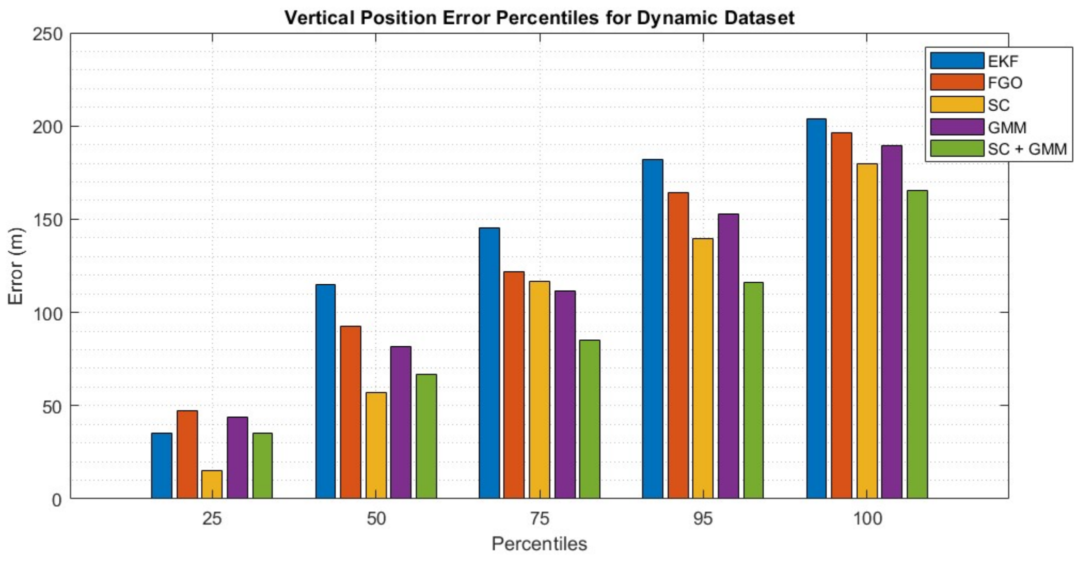


Figure 5.21: Percentiles for vertical errors of the dynamic dataset in ENU reference frame (for 25, 50, 75, 95 and 100th percentiles)

both in static and dynamic scenarios. The performance improvement is more significant and observable in vertical axis than in horizontal axis, considering the results obtained.

In the next section, the comparison of FGO and the robust estimation methods (SC, GMM and SC + GMM) could be found in details.

5.3 Final Comparison of Robust Estimation Methods against FGO

Since the aim of this thesis was to show the improvements of the robust estimation methods of SC, GMM and combination of two (SC + GMM) compared to the plain FGO solution, comparative tables reporting the errors and improvement percentages are given in this section. The improvement percentages of robust estimation methods compared to plain FGO solution are calculated as follows:

$$\delta = \frac{e_{FGO} - e_{Robust}}{e_{FGO}} * 100 \tag{5.1}$$

where δ is the percentage of improvement of the indicated robust estimation method in a given same percentile, e_{FGO} is the FGO's error (in meters), e_{Robust} is the error of the robust estimation method (SC, GMM or SC + GMM). A positive δ implies that the robust estimation method's error is less than the FGO's, whereas if δ is negative, it means that FGO's error is smaller than robust estimation method's error in that given percentage. In the following sections, you can find the analyses for the static and dynamic datasets.

Static Dataset

From Table 5.3 to 5.6, the static dataset's horizontal and vertical errors in meters (Table 5.3 and Table 5.5 respectively) and improvement percentiles (Table 5.4 and Table 5.6 respectively) are given.

%	25	50	75	95	100
FGO	4.27	7.57	11.38	15.44	17.78
SC	4.39	7.43	10.49	15.59	17.20
GMM	4.42	7.32	10.50	15.18	17.19
SC + GMM	4.41	7.03	10.85	14.98	15.71

Table 5.3: Horizontal Error (in m) of static dataset of FGO, SC, GMM and SC + GMM

%	25	50	75	95	100
SC	-2.71	1.94	7.85	-1	3.26
GMM	-3.7	3.24	7.75	1.66	3.31
SC + GMM	-3.23	7.16	4.68	3	11.65

Table 5.4: Horizontal Error Improvement (in %) of static dataset of SC, GMM and SC + GMM compared to FGO

The values in Table 5.3 and Table 5.5 can also be found in the bar plots of Figure 5.10 and 5.11, so they will not be elaborated further. Nevertheless, the Table 5.4 and Table 5.6 reflect the improvements in percentages. Even though there are negative improvements in lower percentiles like 25 in Table 5.3 and in Table 5.4, the improvements increase, and reach the maximum at 75^{th} percentile for SC and GMM. The SC + GMM algorithm managed to decrease the maximum error (i.e. 100^{th} percentile) by 11.65% for horizontal errors in static dataset, which is a great improvement.

The improvements become even more visible in vertical axis for all robust estimation methods in static dataset, where at the 95^{th} percentile SC, GMM and SC + GMM managed to decrease FGO's error by 28.68%, 34.08% and 40.36%; which is a significant number and higher than any of the horizontal percentage improvement values in Table 5.4. Once again, it may be an indication of the improvement capabilities of robust estimation methods in vertical axis. SC, GMM and SC + GMM also manage to decrease the maximum error of FGO by 26.17%, 31.39% and 40.09% respectively; similar to 95^{th} percentile vertical improvement results.

%	25	50	75	95	100
FGO	6.65	21.18	35.73	72.95	81.04
SC	7.95	19.83	30.39	52.02	59.84
GMM	8.06	19.00	28.97	48.09	55.60
SC + GMM	9.62	17.70	30.89	43.51	48.55

Table 5.5: Vertical Error (in m) of static dataset of FGO, SC, GMM and SC + GMM

%	25	50	75	95	100
SC	-19.50	6.38	14.94	28.68	26.17
GMM	-21.20	10.28	18.92	34.08	31.39
SC + GMM	-44.56	16.43	13.53	40.36	40.09

Table 5.6: Vertical Error Improvement (in %) of static dataset of SC, GMM and SC + GMM compared to FGO

Dynamic Dataset

One can see the tables for horizontal and vertical errors in Table 5.7 and Table 5.9, and percentage improvements in Table 5.8 and Table 5.10 respectively, compared to the FGO. It was presumed that the amount of errors could be greater in the dynamic dataset than the static dataset since the receiver was in motion, which happened as expected. In terms of percentages in Table 5.7, GMM provided the best performance improvement in 100^{th} percentile in horizontal axis with 3.31% improvement, and SC + GMM provided the best performance improvement in 95^{th} percentile in horizontal axis with 4.74% improvement.

SC's improvement is rather unstable throughout the percentages in Table 5.8, due to the deviations in its trajectory, thus the percentages going from negative to positive, nevertheless SC provided a more straightforward trajectory and its performance could be further improved in a further study.

The vertical errors of FGO, SC, GMM and SC + GMM for the dynamic dataset could be found in Table 5.9 and the percentages of improvement in vertical axis could be found in Table 5.10. The improvement percentages are smaller than in static dataset's as expected since the target was moving, but it is still possible to observe a significant improvement. Unlike the horizontal percentages, vertical percentages are all positive for all robust estimation methods. It may be another indication that the robust estimation methods are especially successful in the vertical axis on the dynamic dataset as well. The minimum improvement ratio for SC + GMM method is 15.74%, and it provides almost 30% improvement for the 95^{th} percentile. GMM's improvement ratios are smaller, but they still improve the results.

Considering all the information obtained from static and dynamic datasets, one can deduce that the combination of SC and GMM (SC + GMM) could be an promising choice since it mostly reduced the errors among all robust estimation methods tested.

%	25	50	75	95	100
FGO	15.03	36.27	52.27	82.11	96.97
SC	8.67	37.25	47.57	82.22	101.36
GMM	15.19	31.04	45.58	78.69	95.30
SC + GMM	16.92	26.24	37.01	78.23	96.25

Table 5.7: Horizontal Error (in m) of dynamic dataset of FGO, SC, GMM and SC + GMM

%	25	50	75	95	100
SC	42.28	-2.70	8.97	-0.1	-4.53
GMM	-1.08	14.41	12.79	4.17	3.31
SC + GMM	-12.63	27.66	29.20	4.74	0.74

Table 5.8: Horizontal Error Improvement (in %) of dynamic dataset of SC, GMM and SC + GMM compared to FGO

%	25	50	75	95	100
FGO	47.34	92.82	122.11	164.48	196.28
SC	15.28	57.24	116.87	139.35	180.01
GMM	43.59	81.56	111.75	152.72	189.22
SC + GMM	35.18	67.02	84.97	116.18	165.37

Table 5.9: Vertical Error (in m) of dynamic dataset of FGO, SC, GMM and SC + GMM

%	25	50	75	95	100
SC	67.72	38.33	4.28	15.28	8.28
GMM	7.91	12.12	8.47	7.14	3.59
SC + GMM	25.68	27.79	30.41	29.37	15.74

Table 5.10: Vertical Error Improvement (in %) of dynamic dataset of SC, GMM and SC + GMM compared to FGO

Chapter 6

Conclusion

GNSS is a powerful technological system that consist of satellites that allow the users determine their own location, which became a crucial element for many modern applications in recent years due to the increasing demand to LBS. Thus, obtaining accurate and reliable positioning services became essential for users. Nevertheless, in urban environments with tall buildings and dense foliage that cause dense multipath, traditional filters like least-squares and Kalman filtering may be insufficient to provide a stable and accurate PVT solution. A new estimation framework, which is able to provide more accurate and reliable solutions based on the graphical probabilistic model has emerged in recent years, called the factor graphs. Factor graphs consist of two types of nodes, one is the variable nodes to be estimated (such as the user position) and the other is the factor nodes (i.e. GNSS measurements) that encode the necessary equations and constraints on the variable node to be solved. Factor graphs are very flexible, and it is easy to integrate new factors to the states. These factors may come from various sensor measurements or prior information regarding the states. Since every factor in the graph represents a probabilistic constraint, the factors are expressed as terms in the cost function to be minimized, which makes the factors graphs easy to solve. This property of factor graphs also supports the integration of robust estimation methods for outlier mitigation.

The main aim of this thesis was to implement and integrate two robust estimation methods, namely Switch Constraints (SC) and Gaussian Max-mixtures (GMM) on top of the factor graph structure that was designed to solve positioning problem. The Switch Constraints method introduces switch variable states to be estimated, and dynamically adjusts the influence of potentially unreliable measurements in a factor graph that determines whether a measurement should influence the final state estimation, enhancing robustness against outliers. Whereas the Gaussian Max-Mixtures method introduces multimodal uncertainties by representing a measurement as a weighted combination of multiple Gaussian components, enabling the factor graph to account for complex error scenarios such as multipath effects. Both methods, and their combination (SC + GMM) were tested on experimentally collected static and dynamic datasets, and improvements were shown in this thesis.

To summarize, for both of the static and dynamic datasets, the improvements were observed both in horizontal and vertical axes compared to standalone factor graph optimization (FGO) solution. However, horizontal errors were less than the vertical errors and all of the robust estimation methods SC, GMM and SC + GMM were more successful in improving the vertical errors compared to standalone FGO solution. In vertical axis, SC showed improvements up to 28% percent, whereas GMM showed improvements up to 34%, and finally SC + GMM showed improvements up to 40% for the static dataset. For the dynamic dataset, in the 75^{th} percentile, SC showed 8.97% improvement, GMM showed 12.79% improvement, and SC + GMM showed 29.2% improvement horizontally. Also in the vertical axis, in the 95^{th} percentile of errors, SC showed 15.28% improvement, GMM showed 7.14% improvement, and SC + GMM showed 29.37% improvement in the dynamic dataset. The robust estimation methods also performed better than EKF, especially in the vertical axis in static and dynamic datasets. As a result, even though all of the robust estimation methods showed improvement in terms of percentages compared to standalone FGO, more significant improvements are observed when using both methods at the same time (SC + GMM), thus one can conclude that the most successful robust estimation method was the combination of SC and GMM (SC + GMM).

Future research can be done and implemented in this area. Another robust estimation method of Dynamic Covariance Scaling (DCS) is developed as an extension of SC, where the switch variables are not part of the optimization method, but variables calculated separately using the residual, current measurement uncertainty and a prior switch uncertainty [8]; which can be applied to GNSS applications as well. Different M-estimators could be used in the cost functions to enhance the robustness of the factor graphs. Also, the integration of different sensors, such as the inertial measurement units (IMU) could be added on top of the robust estimation methods to improve accuracy and reliability of positioning.

Bibliography

[1] Novatel Hexagon. An introduction to gnss: What are global navigation satellite systems? https://novatel.com/tech-talk/an-introduction-to-gnss/what-are-global-navigation-satellite-systems-gnss.

[2] Shah Zahida Khan, Mujahid Mohsin, and Waseem Iqbal. On gps spoofing of aerial platforms: A review of threats, challenges, methodologies, and future research directions. *PeerJ Computer Science*, 2021.

[3] GIS Geography. Trilateration vs triangulation in GPS. https://gisgeography.com/trilateration-triangulation-gps/. Accessed: 2024-11-09.

[4] Pelin Baysal. Satellite navigation systems lab report. Technical report, Politecnico di Torino, 2024. Course codes: 03LPXBG, 02LPXQW.

[5] European Space Agency. Tracking loops. https://gssc.esa.int/navipedia/index.php?title=Tracking_Loops.

[6] Tersus GNSS. What is DOP in GNSS? https://www.tersus-gnss.com/tech_blog/what-is-dop-in-gnss.

[7] Frank Dellaert and Michael Kaess. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, pages 6(1–2):1–139, 2017.

[8] Shounak Das, Ryan Watson, and Jason Gross. Review of factor graphs for robust gnss applications. https://doi.org/10.48550/arXiv.2112.07794, 2021.

[9] Edwin Olson and Pratik Agarwal. Inference on networks of mixtures for robust robot mapping. *The International Journal of Robotics Research*, 32, 2013.

[10] Carnegie Robotics. Duro specification and user manual. https://carnegierobotics.freshdesk.com/support/solutions/articles/154000199758-duro-specification-and-user-manual.

[11] Ryan M. Watson and Jason N. Gross. Robust navigation in gnss degraded environment using graph optimization. *Proceedings of the 30th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2017)*, 2018.

[12] Gps.gov: Space segment. https://www.gps.gov/systems/gps/space/.

[13] National Institute of Standards and Technology (NIST). Time and frequency: Z to G. https://www.nist.gov/pml/time-and-frequency-division/popular-links/time-frequency-z/time-and-frequency-z-g#:~:text=All%20GPS%20satellites%20broadcast%20on,on%20L5%20at%201176%20MHz. Accessed: 2024-11-09.

[14] ESA: European Space Agency. Galileo begins serving the globe. https://www.esa.int/Applications/Satellite_navigation/Galileo_begins_serving_the_globe.

[15] GMV. Galileo general introduction. https://gssc.esa.int/navipedia/index.php?title=Galileo_General_Introduction#cite_note-GALOSICD-8.

[16] Borealis Precision. Gnss knowledge: Glonass. https://www.gnss.ca/gnss/1361-glonass.

[17] China Satellite Navigation Office. Development of BeiDou navigation satellite system. In *Proceedings of the Munich Satellite Navigation Summit*, 2011.

[18] Penn State University. Why doesn't gps use triangulation? https://www.e-education.psu.edu/natureofgeoinfo/c5_p18.html. Accessed: 2024-11-09.

[19] R. Khan, S. U. Khan, R. Zaheer, and S. Khan. Acquisition strategies of GNSS receiver. In *Proceedings of 2011 International Conference on Computer Networks and Information Technology (ICCNIT)*, pages 119–124. Institute of Electrical and Electronics Engineers Inc., 2011.

[20] Inside GNSS. Gnss solutions: Understanding the basics of satellite navigation and tracking. https://insidegnss.com/wp-content/uploads/2018/01/june10-solutions.pdf, June 2010.

[21] Choi Seung Hyun, Kim Jae Hyun, Cheon Sig Shin, Sang Uk Lee, and Jae Hoon Kim. Acquisition and tracking schemes for a gps l5 receiver. In *2008 International Conference on Control, Automation and Systems*, pages 2214–2217, 2008.

[22] European Space Agency. Multicorrelator. https://gssc.esa.int/navipedia/index.php/Multicorrelator.

[23] European Space Agency. Delay lock loop (DLL). https://gssc.esa.int/navipedia/index.php?title=Delay_Lock_Loop_(DLL).

[24] European Space Agency. Phase lock loop (PLL). https://gssc.esa.int/navipedia/index.php?title=Phase_Lock_Loop_(PLL).

[25] European Space Agency. Frequency lock loop (FLL). https://gssc.esa.int/navipedia/index.php?title=Frequency_Lock_Loop_(FLL).

[26] Ronni Grapenthin. Pseudorange position estimation. phase lab, university of alaska fairbanks. http://www.grapenthin.org/notes/2019_03_11_pseudorange_position_estimation, 2019.

[27] Elliott D Kaplan and Christopher Hegarty. Understanding gps/gnss: principles and applications. https://d1.amobbs.com/bbs_upload782111/files_33/ourdev_584835O21W59.pdf, 2017.

[28] MathWorks. Mathworks matlab documentation center, motion model, state, and process noise. https://www.mathworks.com/help/fusion/ug/motion-model-state-and-process-noise.html.

[29] Nathanael L. Baisa. Derivation of a constant velocity motion model for visual tracking. https://doi.org/10.48550/arXiv.2005.00844, 2020.

[30] Niko Sunderhauf, Gerd Obst, Marcus Wanielik, and Peter Protzel. Multipath mitigation in gnss-based localization using robust optimization. *IEEE Intelligent Vehicles Symposium*, 2012.

[31] Swift Navigation. Duro product summary. https://www.swiftnav.com/sites/default/files/duro_product_summary.pdf.

[32] Spin GNSS. Spin GNSS official website. https://www.spingnss.it/.