BACHELOR'S FINAL THESIS

THESIS TITLE: Opportunistic Navigation with Iridium Next LEO Satellites

DEGREE: Double Bachelor's Degree in Aerospace Systems Engineering and Telecommunication Systems Engineering

AUTHOR: Carlos Acebes Cebrián

UCI ADVISOR: Prof. Zak M. Kassas

UPC ADVISOR: Prof. Francesc J. Robert

DATE: 16 July 2020

Title: Opportunistic Navigation with Iridium Next LEO satellites

Author: Carlos Acebes Cebrián

UCI Advisor: Prof. Zak M. Kassas

UPC Advisor: Prof. Francesc J. Robert

Date: 16 July 2020

Overview

The concept of opportunistic navigation arises from the future demands that autonomous vehicles will require in order to navigate in a reliable and accurate way in GNSS-challenged environments. Specifically, GNSS signals are not robust enough against intentional jamming attacks and they are unencrypted, making them accessible by hackers and completely spoofable.

Some alternatives that have been identified as timely sources of positioning are signals of opportunity, which cluster a broad spectrum including broadband LEO (Low Earth Orbits) satellite signals, AM/FM radio signals, Wi-Fi signals and even cellular LTE/4G signals, and which can be exploited for navigation although they were not transmitted for this purpose.

Particularly, LEO satellite signals have inherent attributes that make them even more desirable for opportunistic navigation. First, their received signal power is around 30dB higher than GNSS signals since they are located approximately twenty times closer to the Earth's surface. Second, they will be abundant in the following years since private companies are planning to aggregately launch thousands of broadband Internet satellites into LEO. Third, they will be diverse in frequency and direction since each broadband provider will deploy its satellites into unique constellations.

Unfortunately, there are several challenges with using LEO satellite signals for navigation as it is discussed throughout this document. For instance, there is a need of having specifically designed receivers that can extract navigation observables from LEO satellites and, furthermore, the internal clocks of LEO satellites are not as precisely synchronized as GNSS satellite clocks, requiring the receiver to account for extra timing shifts.

In this way, the present thesis addresses the problem of navigating opportunistically with Iridium Next LEO satellite signals by proposing a complete receiver architecture that allows to make Doppler measurements to satellite signals in order to obtain a PNT (Positioning, Navigation and Timing) solution.

Títol: Opportunistic Navigation with Iridium Next LEO satellites

Autor: Carlos Acebes Cebrián

Director UCI: Prof. Zak M. Kassas

Director UPC: Prof. Francesc J. Robert

Data: 16 de juliol del 2020

Resum

El concepte de navegació oportunista sorgeix de les exigències futures que requeriran els vehicles autònoms per a poder navegar de manera fiable i precisa en entorns on el GNSS no estigui disponible. Concretament, els senyals GNSS no són prou robustos contra els atacs intencionats i no estan xifrats, fent-los accessibles pels hackers i completament suplantables.

Algunes alternatives que s’han identificat com a fonts adients de posicionament són els senyals d’oportunitat, que agrupen un ampli espectre i que inclouen senyals de satèl·lits LEO (Òrbites Terrestres Baixes) de banda ampla, senyals de ràdio AM/FM, senyals Wi-Fi i, fins i tot, senyals cel·lulars de LTE/4G. Tots ells, es poden aprofitar per a la navegació tot i no haver estat transmesos amb aquest propòsit.

En particular, els senyals de satèl·lits LEO tenen atributs inherents que els fan encara més desitjables per a la navegació oportunista. En primer lloc, la seva potència de senyal rebuda és al voltant de 30 dB superior a la dels senyals GNSS, ja que es troben aproximadament vint vegades més a prop de la superfície de la Terra. En segon lloc, seran abundants en els propers anys, ja que algunes empreses privades tenen previst llançar de manera conjunta milers de satèl·lits de Internet de banda ampla en òrbites LEO. En tercer lloc, seran diversos en freqüència i direcció, ja que cada proveïdor de banda ampla desplegarà els seus satèl·lits en constel·lacions úniques.

Malauradament, hi ha diversos reptes relacionats amb l'ús de senyals de satèl·lits LEO per a la navegació, tal i com s’explica a través d'aquest document. Per exemple, és necessari tenir receptors específicament dissenyats que puguin extreure observables de navegació dels satèl·lits LEO i, a més a més, els rellotges interns dels satèl·lits LEO no estan sincronitzats tan precisament com els rellotges dels satèl·lits GNSS, exigint que el receptor tingui en compte els desfasaments en temps addicionals.

Així, la present tesi aborda el problema de navegar de forma oportunista amb els senyals dels satèl·lits LEO de Iridium Next, proposant una arquitectura de receptor completa que permet fer mesures Doppler als senyals dels satèl·lits per a obtenir una solució PNT (Posicionament, Navegació i Sincronització).

Acknowledgements

I would like to thank Professor Zak Kassas for having given me the opportunity of working in the ASPIN laboratory. His guidance and support during these months have really helped me to carry out this project.

I would like to thank Doctor Joe Khalife, who has been a good friend and who has been working side by side with me throughout this journey. From him, I have learned a number of concepts and practical knowledge that I am sure will be really useful in my future endeavors.

I would like to thank Professor Roger Rangel and the Balsells Foundation for having given me the chance to live in California and to conduct my final Bachelor’s thesis in an institution as prestigious as the University of California, Irvine (UCI). The financial support provided by the Balsells Fellowship has made possible the completion of this project.

I would like to thank Professor Francesc J. Robert for all his advice and for his way of understanding education. Attending his lectures and learning from his teaching methods have helped me develop important skills for my professional career.

I would like to thank my mother Pilar and my brother Pablo for being the best sources of support I can ask for. They are my strongest motivation and the reason why I keep going even in the toughest times. I really cannot express how grateful I am for the trust they put on me.

Finally, I would like to thank all my friends, both in Vilanova and Irvine, for being there always that I need them.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

1.1. Mission Statement and Motivation

The autonomous vehicles of the future will rely on several positioning sources to aid their inertial navigation systems in global navigation satellite system (GNSS)-challenged environments. Recently, it has been progressively revealed that GNSS signals are becoming to be obsolete since their accuracy, availability and robustness might be compromised in some scenarios that include intentional jammers and spoofers. Hence, alternative approaches that do not depend on GNSS and which can increase the reliability of autonomous flights are starting to be developed.

Potential alternatives to GNSS are signals of opportunity (SOPs), which have been considered as a reliable and desirable navigation source within the past decade. Some examples include broadband LEO (Low Earth Orbits) satellite signals, AM/FM radio signals, Wi-Fi signals and even cellular LTE/4G signals. In this way, opportunistic navigation consists on using the information that is already available in the surrounding environment of the autonomous vehicle and which can be exploited for navigation and positioning.

This approach, which consists on fusing several data obtained from signals of different topologies and characteristics, it is highly suitable since it allows the autonomous vehicles to build a spatio-temporal map of the environment within which they can localize themselves, and which might be more complete than what can be obtained with the possibilities offered by GNSS.

Particularly, LEO satellites have shown desirable attributes for opportunistic navigation since its received signal power is much higher compared to GNSS signals, the availability of their signals will be increased as several private companies such as SpaceX and Boeing are planning to aggregately launch thousands of broadband Internet satellites into LEO, and their signals will be more diverse in frequency and direction being that each provider will deploy the LEO satellites into unique constellations.

Previous publications have exhibited promising results with existing LEO constellations such as Orbcomm and Globalstar, obtaining PNT (Positioning Navigation and Timing) solutions in a totally opportunistic fashion. Nevertheless, there are still several LEO satellite constellations for which the possibilities for navigation that they can offer have not been studied so exhaustively.

In this way, the present thesis addresses the problem of navigating opportunistically with Iridium Next LEO satellite signals by proposing a complete receiver architecture that allows to make Doppler measurements to satellite signals in order to obtain a PNT solution.

1.2. Project Objectives and Scope

The present document covers the first stages that are necessary for obtaining a PNT solution in a totally opportunistic fashion with Iridium Next LEO satellite signals. Specifically, the following objectives have been identified:

- Understand the previous work that has been published using the Iridium Next constellation as a navigation source by conducting a literature review, and to identify uncovered subjects that may serve as research motivations
- Simulate Iridium Next signals with Matlab from the specifications provided by the owner of the constellation
- Design a complete receiver architecture based in Matlab that allows to acquire and to track the Doppler frequency from both simulated and real satellite signals
- Be able to predict the orbits of the satellites in order to determine optimum time windows for signal recording where the receiver has visible satellites over its location
- Design a navigation framework that allows to obtain a navigation solution from the Doppler measurements produced by the receiver to both simulated and real satellite signals

It might be recalled that this thesis aims to adopt an academic scope and thus it is not intended that the obtained results necessarily improve the possibilities offered by GNSS signals. While it is expected that both the signal tracking and the navigation solution obtained with simulations show promising results, the solution obtained in a real scenario aims to serve as discussion and motivation sources from which future work will be developed. This future stage will require from several experiments and research which are both out of the scope of the present project and for which will be required more time.

Finally, the completion of all the objectives has been carried out progressively by breaking down the complete problem into several parts that have been completed week by week. In this way, the entirety of this document is the result of the autonomous and regular work carried out for six months.

1.3. Project Structure

The structure of this thesis aims to exhibit a chronological order that allows the reader to follow the necessary steps to obtain the final results. On the other hand, the chapters that compose this document cluster all the different topics that are relevant to this research.

First, the need of finding reliable and desirable alternatives to GNSS (Global Navigation Satellite System) which will allow the vehicles of the future to navigate autonomously while ensuring signal robustness, availability and positioning accuracy is discussed. Besides, a summary of a literature review that was made with the goal of understanding other previous publications that used Iridium Next signals as a navigation source is given.

Second, a description of the characteristics and structure of the signals transmitted by Iridium Next satellites is presented. Moreover, the signal model that was used to simulate these signals as well as some implementation issues with Matlab are discussed.

Third, a complete receiver architecture which allows to acquire and to track Iridium Next satellite signals in order to produce Doppler measurements, which are used to obtain a navigation solution, is detailed. Furthermore, a code flowchart as well as some discussion about its Matlab implementation are provided.

Fourth, a method to predict the orbits of LEO satellites is discussed. Also, some results showing the ground track graphs of Iridium Next satellites are exhibited.

Fifth, the experimental setup including the required equipment for recording real satellite signals is detailed. Then, an exhaustive analysis from the collected data and the interpretation that leads to determine the nature of the recorded signals are made.

Sixth, the navigation framework that is implemented by means of an EKF (Extended Kalman Filter) and which is used to obtain the final navigation solutions both with simulated and real satellite signals is presented.

Seventh, the obtained results, which are the product of combining all the work and procedures detailed throughout this thesis are provided. Besides, an interpretation of those results as well as some improvement proposals are given.

Eight, the conclusions from the results that have been eventually reached as well as some future work discussion are provided. Finally, annexed to this document are included the appendices that contain the Matlab codes on which the signal simulator, the receiver, the orbit predictor and the navigation framework are implemented.

CHAPTER 2. OPPORTUNISTIC NAVIGATION

2.1. Historical Background

2.1.1. GNSS-Challenged Environments

The first satellite navigation system that provided a position, navigation and time (PNT) solution was developed by the US military in the late 1960s and was called Transit. As inferred, the system was designed to provide positioning services to the US Army and its allies during the contemporary wars that were taking place at that time, and although it was primitive and unable to provide a solution in real time with enough accuracy, it changed the world.

After the succeed of the aforementioned system, the US military continued on developing an improved version that would allow to increase the accuracy of the provided PNT solutions as well as the availability of its signals. In this way, by the end of the 1970s, the GPS (Global Positioning System) was born.

Although there are innumerable applications for which satellite navigation systems are highly useful, the beginnings of this technology were only focused on the military world. In fact, satellite positioning was seen as somehow valuable in a battle environment but was completely disregarded to be applied for civil applications. Then, the main enemy of the US by that time (Russia) started to develop its own system (GLONASS) in order maintain the competition. However, paradoxically, these two countries started what today is known as GNSS (Global Navigation Satellite Systems) and which is used globally in a vast variety of civil applications.

By the early 2000s, the US military accepted to make their GPS signals public and available to the rest of the world so they could be used for civil positioning and navigation. Nevertheless, they reserved the right to degrade their signals and make them useless by means of the selective availability in case that the national security of the US was threatened. Hence, China decided to develop its own system (BeiDou) after being worried about not being able to use GPS in case that the US denied them the access. Some years later, Europe developed Galileo by the 2011 in order to avoid the dependence as well on the US system.

The following table summarizes the main characteristics of the systems that compose the GNSS:

Table 2.1. GNSS Systems' Main Characteristics

	GPS	GLONASS	BeiDou	Galileo
Coding	CDMA	CDMA & FDMA	CDMA	CDMA
Altitude	20,180 km	19,130 km	21,150 km	23,222 km
Satellites	30	24	28	22
Frequency	L-band	G-band	B-band	E-band

From the previous table, over a hundred satellites are providing navigation and positioning services to the world and, furthermore, although the origins of satellite-based positioning were closely related with the military world, at present, the GNSS has become a backbone of the civil infrastructure. Moreover, since economically important sectors as banking & finance or the chemical industry are depending on GNSS to offer their services, it is important that the availability and reliability of navigation signals is as high as possible. Thereby, the concept of opportunistic navigation arises from the need of a reliable source of positioning in GNSS-challenged environments.

Apart from well-known interference sources as ionospheric delays or shadowing areas in urban environments that can result into a loss of coverage from GNSS signals, the main techniques to suppress navigation services are jamming and spoofing. Both are intentional attacks to the GNSS signals and the main difference between them is the result that they produce on the final user.

On the one hand, Jamming occurs willfully when a radiation of electromagnetic waves at the GNSS frequencies is produced by an external source that aims to intentionally degrade the service provided by the GNSS satellites. This technique consists on overpower the extremely weak signals that come from the satellites so they cannot be acquired and tracked properly by the receiver [1]. Recently, Personal Protection Devices (PDD) that can be used for jamming and which can be purchased easily online are being reported by an increased number of countries. Moreover, the possession of these light-weight devices is not equally regulated around the world and thus they suppose a real problem.

On the other hand, spoofing is a more sophisticated technique that consists on generating and transmitting fake GNSS signals with the intention to lead a GNSS receiver astray without being aware of the attack. In this way, the goal of spoofing is to make the receiver believe that it is in a different position than he actually is so it can be deviated from its original path and can be controlled externally by a third party. Besides, spoofing may consist on retransmit existing GNSS signals with the aim of altering the relative delays that are seen by the receiver and which lead to obtain an erroneous PNT solution. The following figure shows a schematic of a target being spoofed by a terrestrial source:

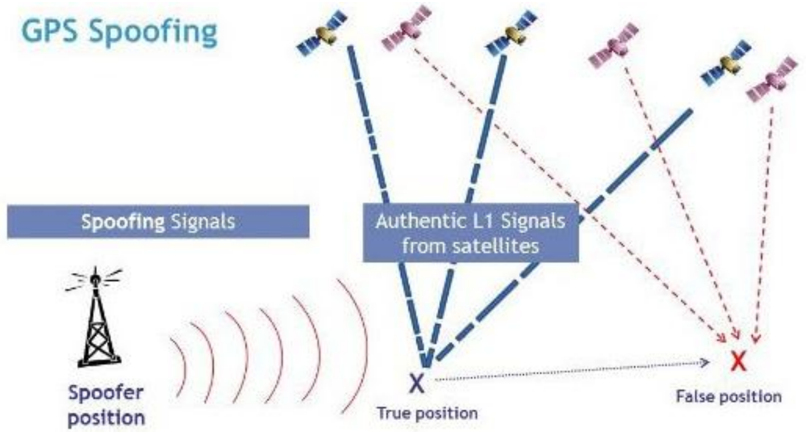


Fig. 2.1. GPS Spoofing Schematic [1]

In this way, it might be noticed that one of the dangers of spoofing is that it can eventually result into a receiver believing that he acquired the GNSS signals from inexistent satellites that are in a fake position and thus it will be deviated to a different position from the true one provided by the actual GNSS satellites.

Finally, there also exist other types of unintentional interference that may result into a loss of service for the users. Concretely, some GNSS bands are shared with TV harmonics, radars or airplane DME (Distance Measuring Equipment) systems that can interfere with GNSS signals. For all the above, a method to enhance the robustness of GNSS or even to become an alternative in case it becomes unavailable has started to be developed by means of opportunistic navigation.

2.1.2. Navigation with Signals of Opportunity

The concept of opportunistic navigation arises from the future demands that autonomous vehicles will require in order to navigate in a reliable and accurate way without the need of being supervised externally. As it was discussed in the previous subsection, GNSS will not meet the demands of these future system for the following reasons:

1. GNSS signals are received with an extremely weak power and might be unusable in deep urban and indoors environments
2. GNSS signals are not robust enough against intentional jamming attacks and unintentional interference
3. Civilian GNSS signals are unencrypted, they do not require authentication and, even more, they are specified in public documents that can be accessed by hackers, making them completely spoofable

In this way, opportunistic navigation consists on selecting wisely the signals that are already available in the surrounding environment of the autonomous vehicle and exploit them for positioning, navigation and timing. This information is processed by on-board receivers in order to allow the vehicle to localize itself in space and time and, furthermore, it is shared among the coexisting vehicles of the environment in order to achieve a global situational awareness.

The concept of signals of opportunity (SOPs) clusters a broad spectrum of signals that can be used as a primary source of navigation although they were not transmitted for this purpose [2]. Examples of SOPs include broadband LEO (Low Earth Orbits) satellite signals, AM/FM radio signals, Wi-Fi signals and even cellular LTE/4G signals. In fact, the great majority of the signals that are being transmitted at present can be used for positioning and navigation purposes if they are acquired in a pertinent way.

The figure on the following page illustrates a scenario where an autonomous car and two UAVs (Unmanned Aerial Vehicles) use signals from different communication services in order to increase the availability and accuracy of the PNT solutions that they could obtain if they only used GNSS.

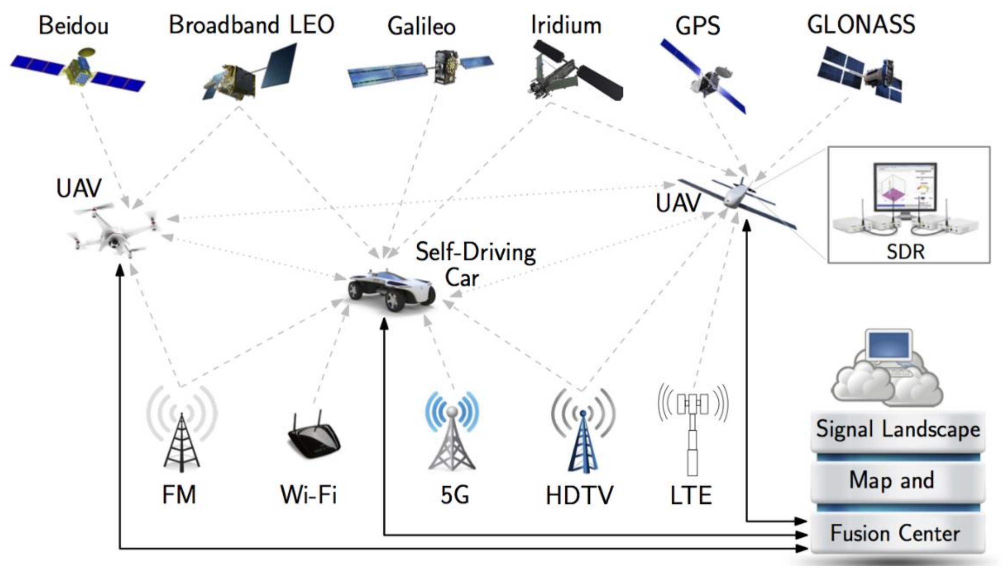


Fig. 2.2. Opportunistic Navigation Environment [2]

In the previous figure, the autonomous vehicles relay both on terrestrial and satellite signals in order to obtain a PNT solution. Recent research conducted by members of the Autonomous Systems and Perception Intelligence & Navigation (ASPIN) Laboratory has shown the promise of a submeter-accurate navigation solution for unmanned aerial vehicles when carrier phase measurements from cellular signals are used [3]. Nevertheless, satellite signals, particularly coming from LEO satellites, have inherent attributes that make them even more desirable for opportunistic navigation:

1. LEO satellites are located approximately twenty times closer to the Earth's surface compared to GNSS satellites, which are in Medium Earth Orbits (MEO). Then, the received signal power from LEO satellites is around 30dBs higher than GNSS signals
2. Private companies such as OneWeb, SpaceX or Boeing are planning to aggregately launch thousands of broadband Internet satellites into LEO, which will make them to become abundant as thousands in the following years
3. Each broadband provider will deploy their satellites into unique constellations that will use different frequency bands for transmitting their signals, making them diverse in frequency and direction

Besides, the Keplerian elements that are used to predict the orbits of LEO satellites and to know their locations, as it is discussed in chapter 5, are made publicly available by the NORAD (North American Aerospace Defense Command) and are updated daily in the two-line element (TLE) files. Moreover, some LEO constellations equip their satellites with GNSS receivers that broadcast their GNSS PNT solution to terrestrial receivers, allowing the autonomous vehicles to know where the LEO satellites are located.

Notwithstanding the foregoing, it might be recalled that LEO satellites are not intentionally designed for navigation and positioning, and so there are several challenges with using them for these purposes. More specifically, the following considerations must be considered when trying to obtain a PNT solution:

1. There is a need of having specifically designed receivers that can extract navigation observables from LEO satellites. Furthermore, the structure of the signals transmitted by satellites from different constellations may be different, requiring adapting the receivers as a function of the targeted constellation
2. The internal clocks of LEO satellites are not as precisely synchronized as GNSS satellite clocks [3], requiring the receiver to account for extra signal delays and timing shifts
3. Although the orbital parameters that are used to predict the orbits of LEO satellites are contained in the publicly available TLE files, the positions predicted from this information might be off by several kilometers

Fortunately, recent research carried out during the past decade [2] has proven that broadband LEO satellite signals can be used as a reliable navigation timing source if all the previous challenges are managed to be solved. Particularly, as it will be discussed throughout this thesis, it is possible to design receivers that base their operation on the Doppler effect that is produced on the satellites transmitted signals. The received frequency will differ from the transmitted one because of the movement of the satellites with respect to the receiver and, in this way, by tracking this frequency shift over a given time interval, the receiver can determine its location.

2.2. Iridium Next Related Literature Review

2.2.1. GNSS Backup Approach

Since the objective of this thesis is to design a receiver that can extract navigation observables for positioning in a totally opportunistic fashion using Iridium Next satellite signals, a literature review on this topic must be made. Thereby, we will be able to identify what has already been done and which may be useful for our work and, moreover, we will be able to identify what is missing to be produced in order to use it as a motivation for our designed receiver’s architecture. Finally, this section summarizes some previous works that used satellite signals from the same constellation to obtain a PNT solution.

In the recent years, some relevant publications demonstrated very promising results using Iridium Next signals but either in a partially opportunistic way or as a backup system for GNSS positioning.

Particularly, the following relevant publications have been identified to show navigation methods that serve as either backup or as augmentation systems for GNSS. Nevertheless, they lack a completely opportunistic approach due to the reasons that are described in the following page.

First, the work contained in [4] uses Iridium Next as an augmentation system for GPS. In this way, they assess the localization performance of a GPS system and an Iridium Next-Augmented GPS system. Hence, the navigation solution that they provide relies on GPS and could not be useful for an autonomous vehicle in a GNSS-challenged environment.

Other publications have shown navigation solutions using Iridium Next satellite signals that could be useful in GNSS-challenged environments. However, they lack an opportunistic approach and it appears to be uncertain if they could be truly applied. For instance, the work contained in [5] provides a navigation solution using Iridium Next but after having purchased a development kit directly to the company in order to design their receiver. Besides, the PNT solution given in [6] uses initial calibration signals that are transmitted encrypted and which can be only decoded by paying a fee to the constellation owner.

These previous approaches appeal for the rearrangement of the broadband protocol that LEO satellites are currently using in order to support navigation capabilities. Moreover, they support the idea of having simpler navigation algorithms at the cost of having to purchase navigation receivers to the constellation owners. As inferred, they do not conceive navigation using LEO satellites in a purely opportunistic fashion.

Finally, it might be recalled that they face the inconvenience of requiring significant changes to the existing infrastructure, which implies an elevated cost that either private companies who own the LEO constellations may not be willing to pay, or even they may charge the additional costs for these extra services to the final users. For all the above, using an alternative approach that exploits existing broadband LEO satellite signals opportunistically for navigation becomes more viable.

2.2.2. Opportunistic Approach

Intensive research has been carried out recently for the Iridium Next constellation, trying to chase a much more opportunistic approach and obtaining very promising results. As it was discussed previously, this way of focusing the problem of being able to obtain a PNT solution from LEO satellite signals appears to be the more convenient. In this way, this sub-section gathers some of the most relevant publications that have tried to follow this approach.

After an exhaustive literature review, it has been found that the work contained in [7] proposes a navigation framework that implements a positioning technique using Iridium Next signals and which can be used independently from GNSS in case it becomes unavailable. Particularly, it proposes a quadratic square accumulating instantaneous Doppler estimation algorithm that can improve the availability of the satellites signals even in weak signals environments. In this way, it produces an instantaneous Doppler positioning that allows the receiver to navigate.

The following figure shows the basic principle of the algorithm proposed in [7] and which is used for positioning with Iridium Next SOPs:

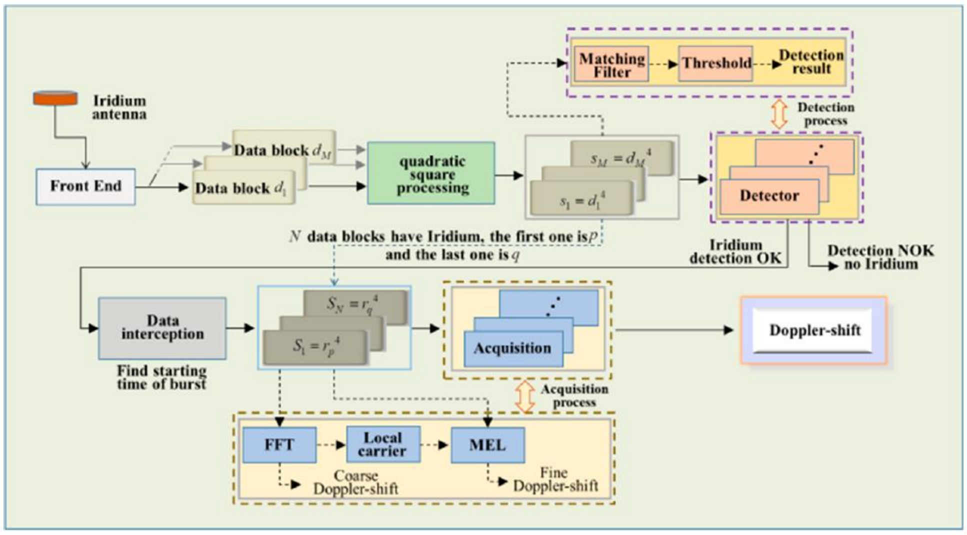


Fig. 2.3. Schematic of the estimation algorithm proposed in [7]

Although the aforementioned work shows a navigation framework that can produce Doppler measurements for positioning, it lacks to demonstrate consistent results that support the effectiveness of their proposed method in real environments. Moreover, it is mainly focused on the estimation theory techniques that must be considered to allow their algorithm work, leaving aside a detailed description of the different parts that compose the full receiver design.

Following in the same way, the work contained in [8] describes a method for joint synchronization and location using Iridium Next signals but does not neither describe the receiver's architecture on which this proposed technique is implemented. Hence, although it shows outstanding experimental results that demonstrate an absolute error from the obtained PNT in the order of 50 meters, it combines its algorithm with an already existing receiver for which the design is not detailed.

Thereby, it can be concluded that a complete and detailed receiver architecture that produces Doppler measurements from Iridium Next satellite signals in order to provide navigation and positioning services in an opportunistic fashion has not yet been shown.

2.3. Future Perspectives

It has been increasingly accepted by the scientific community that the future of navigation will tend to follow an opportunistic approach and will rely on several different types of signals that were not specifically designed for positioning.

Besides, although the future is still uncertain, it is becoming to be clear that autonomous vehicles will begin to be integrated solidly into our daily lives. For instance, global transportation companies such as amazon are starting to test some mechanisms and strategies to deliver their orders using autonomous vehicles. Furthermore, motoring companies such as Tesla have already commercialized self-driven cars that can be used autonomously in some states in the US.

On the other hand, there is a clear tendency that promises an increased availability of several signals suitable to be used for opportunistic navigation although they will not be designed for such purpose. Particularly, the 5G infrastructure, which is starting to be deployed at present, will suppose having available a huge quantity of terrestrial signals from which PNT solutions can be obtained.

In the same way, the arrival of IoT (Internet of Things) will suppose the launching of mega-constellation LEO satellites that will provide broadband Internet and whose signals are very desirable for opportunistic navigation as it was discussed in previous sub-sections. In fact, recent studies predict that over ten thousand satellites will be deployed into LEO in the following ten years, leading to have an enormous interconnected satellite network as shown on the following figure:

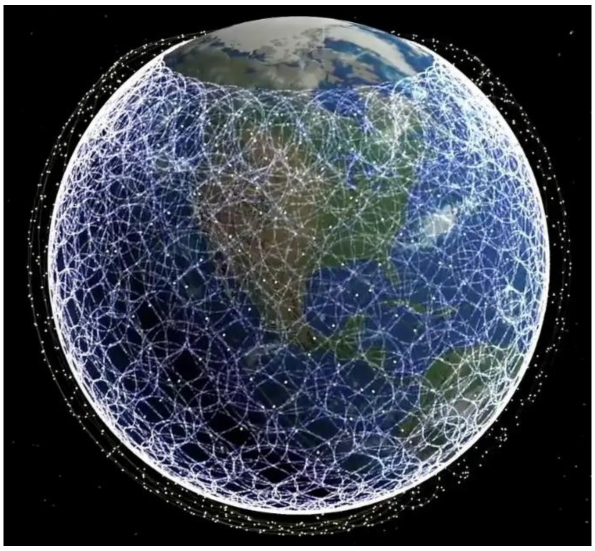


Fig. 2.4. Prediction of Mega-Constellation LEO Satellites Orbiting the Earth

Moreover, satellite signals which are specifically designed for navigation and which compose the GNSS infrastructure are becoming to be obsolete since they are spoofable, their availability is often compromised in urban and indoors environments, and the accuracy that they can offer for positioning is starting to become stagnant.

For all the above, the future of navigation appears to take an opportunistic tendency that will rely on signals of very different typologies and that will allow the vehicles of the future to obtain both more accurate and robust PNT solutions.

CHAPTER 3. IRIDIUM NEXT SATELLITE SIGNALS

3.1. Signal Characteristics

3.1.1. Frequency and Polarization

Iridium Next signals are transmitted over the L-band using the spectrum available from $1616 - 1626.5\ MHz$. In this way, there are 252 carriers in both the uplink and downlink channels, with carrier spacings of $41.6667\ kHz$ and with a required bandwidth of $36\ kHz$ [9]. These carrier frequencies are grouped into sub bands of 8 carriers, with the $32th$ group containing 4 carriers.

All the previous carriers are suitable to be assigned for both uplink and downlink channels, allowing to assign uplink carriers independently of the downlink ones. This will be useful when trying to obtain the navigation observables from the Iridium Next signals since the downlink channels that our designed receiver will use to navigate might be allocated into different parts of the total system spectrum as a function of time, which means that we will have a higher signal availability.

Iridium Next defines a small portion of $0.5\ MHz$ over the upper part of its assigned spectrum (from $1626 - 1626.5\ MHz$) which is used for paging and acquisition [9]. There are 5 simplex downlink carriers that employ the same frequency spacing as the standard channels and that require $35\ kHz$ of bandwidth. Those are interesting for our positioning purposes because they are used to transmit the Ring Alert messages and synchronization signals (unmodulated tones) that can be used to track the Doppler shift of the incoming signal, as it is discussed in chapter 4. The simplex channels are defined as it is shown in the following table:

Table 3.1. Iridium Next Simplex Downlink Channels

Use	Paging/ Acquisition	Paging/ Acquisition	Ring Alert	Paging/ Acquisition	Paging/ Acquisition
f_c [MHz]	1626.1042	1626.1458	1626.2708	1626.3958	1626.4375

On the other hand, Iridium Next signals are transmitted using RHCP (RightHand Circular Polarization) in both uplink and downlink channels. The reason of using this type of polarization is closely related to the fast changes in orientation that signals experience due to satellite motion. Since the angles of view from the receiver to the satellites are changing across the sky from horizon to horizon, the electric field of the signal rotates helically. This rotation can be either clockwise or counterclockwise with respect to the propagation direction of the signal. For the first case, it is said that the wave is RHCP, whereas it is said to be LHCP for the second case.

This previous explanation can be appreciated in the following figure:

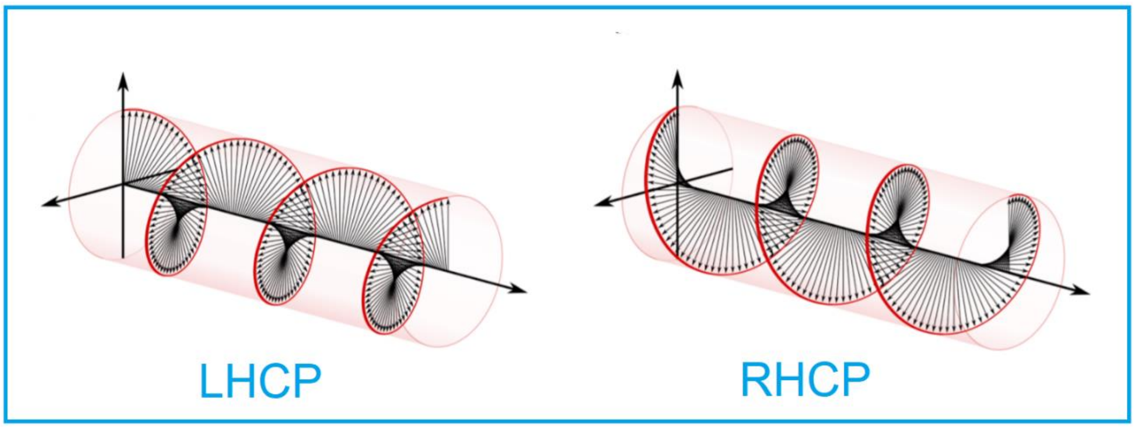


Fig. 3.1. Right-Hand and Left-Hand Circular Polarizations

In this way, it would be unpractical to transmit the signals using linear polarizations because it would require for the receiver antenna to constantly rotate in order to account for the inherent rotation of the received signal. Then, since it is only necessary that one end of the radio link uses a CP (Circular Polarization) antenna to transmit the signals in order to achieve all the rotation-independent benefits of CP, the receiver antenna can simply use a linear polarization to track the signal properly. Then, it should be noticed that CP inherently reduces the complexity of the receiver and allows us to collect satellite signals more easily.

3.1.2. Signal Structure and Access Technology

Iridium Next satellites transmit using TDD (Time Division Duplex) according to a TDMA (Time-Division Multiple Access) access technology. The TDMA frame is defined for each of the 32 sub bands and it is composed by 8 carriers that contain the uplink and downlink channels, plus a time slot used to allocate a simplex channel. The frame length is $90\ ms$ long and the simplex channel is $20.30\ ms$.

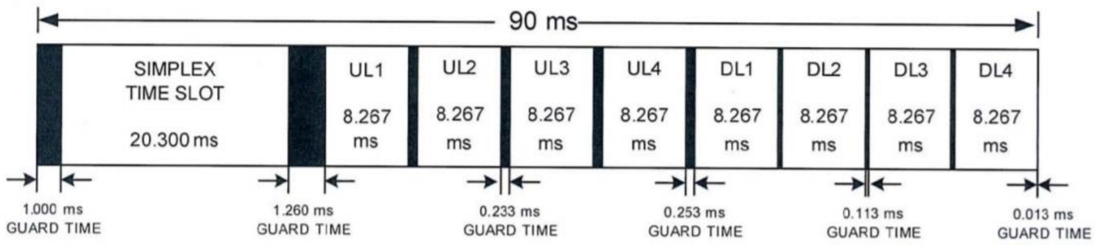


Fig. 3.2. TDMA Frame Structure [9]

The signal structure over the uplink and downlink channels consists on signal bursts that are sent periodically over the TDMA frame. Each burst is composed by an unmodulated tone, preceded by a unique word and the information data.

On the simplex channel, Iridium Next is transmitting the Ring Alert as well as paging/acquisition messages, which have the same burst structure as the standard carriers. In this way, the pure tone transmitted at the beginning of each burst will be used by our receiver to acquire and to track the Doppler, and to obtain a final navigation solution.

According to [5], the bandwidth of the bursts in the simplex channels is $26.6667\,kHz$ and the mean signal duration is $6.8\,ms$, instead of $20.30\,ms$ as defined by Iridium Next documentation [9]. Finally, the duration of the tones is $2.6\,ms$, creating an inherent signal duty cycle from the viewpoint of the receiver.

3.2. Signal Simulation

3.2.1. Signal Model

The designed receiver aims to track an unmodulated tone, which is acquired by periodically sampling any of the simplex downlink channels with a sampling period (T_s). In this way, the signal model is defined as:

$$r(i) = s(i) + n(i), \qquad i = 0,1,2,\dots \tag{3.1}$$

The channel noise (n) is assumed to be Complex Gaussian and can be expressed as:

$$n(i) = n_I(i) + j \cdot n_Q(i), \qquad i = 0,1,2,\dots \tag{3.2}$$

Where its In-phase (n_I) and Quadrature (n_Q) components are modeled as zero-mean complex white Gaussian noise with variance:

$$var(n) = {\sigma_n}^2 = \frac{N_0}{2T_s} \tag{3.3}$$

The data-meaningful part of the received signal is expressed as:

$$s(i) = \sqrt{A(i)} \cdot exp\{j2\pi f_D(i) \cdot iT_s + j\phi_s(i)\}, \qquad i = 0,1,2,\dots \tag{3.4}$$

Since the received signal corresponds to a pure tone, the absolute value of its amplitude should be unitary. However, due to the motion of the satellites, the distance between the receiver and the transmitting-end changes with time, and thus the received signal suffers a light attenuation. It is assumed that this attenuation increases with the square of the distance between the receiver and the satellites as a result of Free Space Path Loss. Then, the amplitude of the received signal is modeled as:

$$A(i) = 1 + \frac{1}{10}\left(\frac{d_0}{d(i)}\right)^2, \qquad i = 0,1,2,\ldots \tag{3.5}$$

The phase of the signal changes over time as a result of the Doppler shift, which is varying as a function of the distance between the receiver and the satellite. In this way, the signal phase is updated as:

$$\phi_s(i+1) = \phi_s(i) + 2\pi f_D(i) T_s, \qquad i = 0,1,2,\ldots \tag{3.6}$$

3.2.2. Doppler Shift and Distance Profiles Generation

It may be noticed that a realistic Doppler shift profile, as well as a distance profile are necessary to simulate the signal acquired by our receiver. To do so, we have used a Doppler profile simulator property of the ASPIN Laboratory that generates the Doppler shift time history produced by a LEO satellite, which is moving at a velocity compliant with the information given in the TLE files from the Iridium Next constellation.

Once the Doppler shift profile has been generated, it is possible to obtain the evolution of the distance between the receiver and the satellite by integrating the following expression:

$$f_D(t) = -\frac{f_c}{c} \cdot \frac{d(d(t))}{dt} \tag{3.7}$$

In this way, an expression for the distance profile during the time interval when the Doppler shift profile was defined can be found as:

$$d(t) = -c \cdot \int \frac{f_D(t)}{fc} dt + d_0 \tag{3.8}$$

The initial condition (d_0) is defined as the maximum possible distance between the receiver and the satellite, which is produced when the moving object is located at its apogee. For an Iridium Next satellite observed from Irvine, California this value leads to $d_0 = 2.25 \cdot 10^6\ m = 2250\ km$. Also, the carrier frequency is set to be $fc = 1626.2708\ MHz$ (the Ring Alert channel).

The generation of the profiles has been done by using that the height of the satellites orbits is $780\ km$, the typical Doppler shift produced by orbiting objects over this height is approximately between the interval $[-40\ kHz, 40\ kHz]$, and imposing a stationary receiver. Finally, the simulation time is set to be 10 minutes.

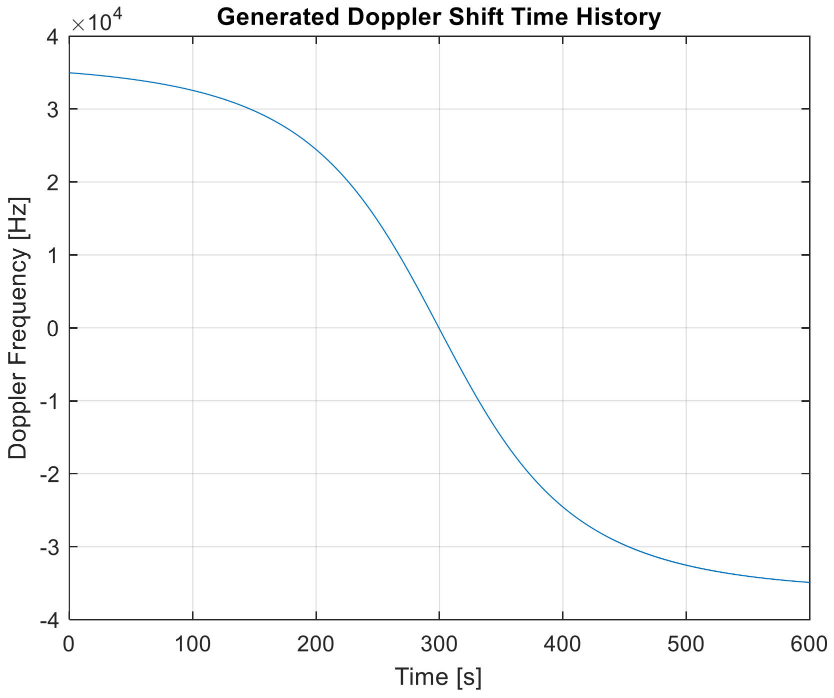


Fig. 3.3. Generated Doppler Shift Time History

It may be inferred from the previous figure that the satellite starts moving towards the receiver for the first part of the simulation, since the values of the Doppler shift produced on the received signal are positive. Then, the mid-point of the simulation corresponds to the moment where the satellite stays over the receiver location because the Doppler shift is null. Finally, the Doppler shift values (negative) of the last part of the simulation correspond to the satellite leaving backwards the receiver location.

In this way, the distance profile must show a satellite approximating to the receiver, passing over it and moving away, and this is what can be observed in the figure on the next page.

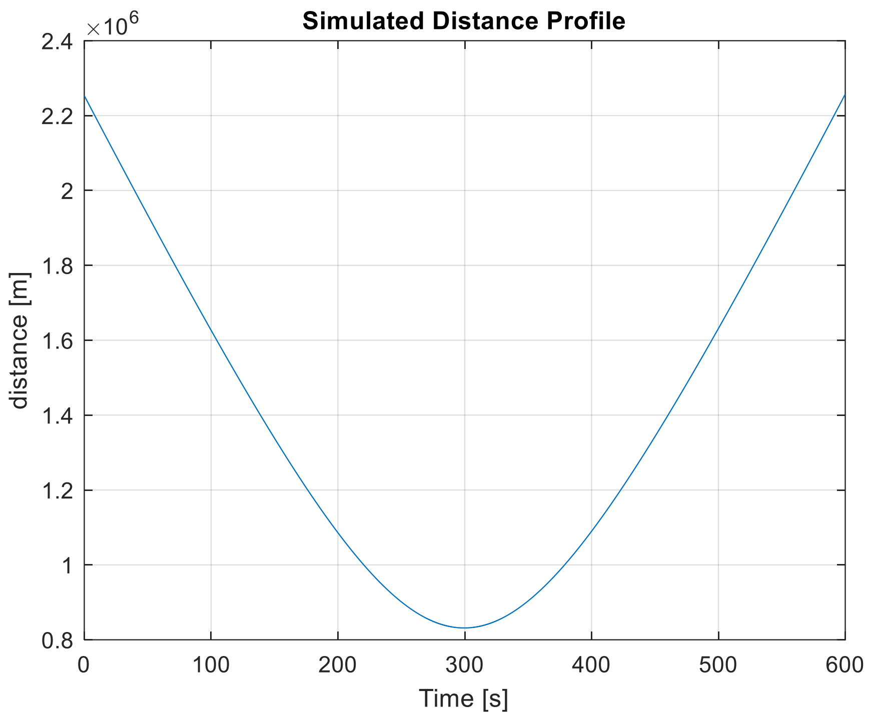


Fig. 3.4. Simulated Distance Profile

3.2.3. Timing Drift Effect

The acquired signal by our receiver is expected to be a sequence of pure tones that have a duty cycle (η) as it was previously discussed. In this way, Iridium Next satellites transmit signal bursts that contain an unmodulated tone at their beginning, and since the duration of a single tone ($2.6\,ms$) is lower than the duration of a burst ($6.8\,ms$), it can be said that the tone period ($6.8\,ms$) is higher than the tone duration ($2.6\,ms$).

The following expression is used to define the duty cycle of the tones, which is a measure of the percentage of time when the signal is received with an ON value:

$$\eta(\%) = \frac{T_{On}}{T_{On} + T_{Off}} \cdot 100 = \frac{2.6 \cdot 10^{-3}}{6.8 \cdot 10^{-3}} = 38.24\% \tag{3.9}$$

It may be noticed that the reception of a periodic signal in the time domain implies that the receiver must be synchronized with the transmitter so that the signal is acquired correctly. Unfortunately, timing effects such as propagation and clock biases play an important role by delaying in time the transmitted signals, so that the receiver acquires them with a timing shift. Then, this section describes how these effects are simulated in order to account for them during the receiver design.

The first important feature to include in our signal simulation is the effect of the timing delay due to wave propagation over the channel. In this way, the received tones will be shifted in time mainly due to the varying distance between the receiver and the satellite. Moreover, the effect of the clock bias of the satellites that results into synchronization issues between the transmitter and the receiver has been included. In the time domain, the received signal may be written as:

$$s_i(t) = \sqrt{A(t-\tau)} \cdot exp\{j2\pi f_D(t-\tau) + \phi_s(t)\} \tag{3.10}$$

Where the shift in time (τ) that affects both the amplitude and the phase of the received signal is modeled as:

$$\tau = \frac{d(t)}{c} + bias_{ct} \tag{3.11}$$

The clock bias of the satellites is assumed to be constant and it is modeled to include a pessimistic scenario, where this value tends to be high compared to the tone period (T_{tone}) of the satellite.

It is important to define the clock bias as a function of the tone period, since this parameter is not expected to be higher than one entire period. This can be explained after realizing that the received signal is periodic and thus any constant timing shift that might be applied to it ends to become circularly periodic.

The following figure shows an example of a periodic signal, with a period of N samples, being shifted by 2 samples:

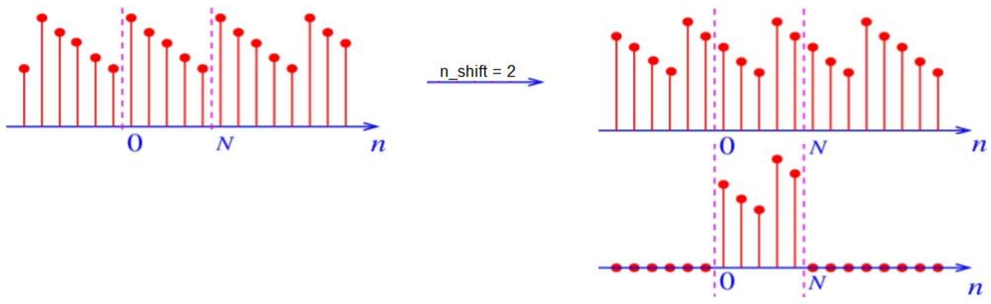


Fig. 3.5. Circular Shift of a Discrete-Time Signal Example

It may be noticed that the previous signal, when shifted, can only adopt some well-defined values since it is periodic in discrete time. Then, applying a time delay to the signal can be translated into circularly shifting its samples, which means to rearrange its samples by moving the final one to the first position, and shifting all the other samples to the next position. Then, any constant shift ends to become periodic as well as discretely defined.

Taking back to our sequence of pure tones, it can be concluded that, in fact, it is not possible to observe a timing shift higher than half of the tone period because of its time-periodic properties, which leads to define the actual timing shift between the following interval:

$$bias_{ct} \overset{\text{def}}{=} [-0.5T_{tone}, 0.5T_{tone}] \tag{3.12}$$

Knowing all the above, the clock bias is defined by the following expression, which implies that the timing shift of one tone out of five equals to its period if only a maladjustment due to synchronization issues is considered:

$$bias_{ct} \overset{\text{def}}{=} 0.2 \cdot T_{tone} \tag{3.13}$$

On the other hand, it is important to recall that the evolution of the timing shift (τ) is time-continuous, so it must be discretized to be implemented into the signal simulator. The time resolution of the simulator is $1/F_s$ since the minimum time interval that can be detected is the duration of a single sample. Then, it is possible to express the timing shift as a function of the number of samples that any given tone is shifted:

$$\tau_d(i) = \frac{1}{F_s}\tau = \frac{d(i)}{c} + 0.2 \cdot T_{tone} \tag{3.14}$$

The following figure is used to show the first 10 seconds of the timing shift evolution in order to show that it only adopts discrete values:

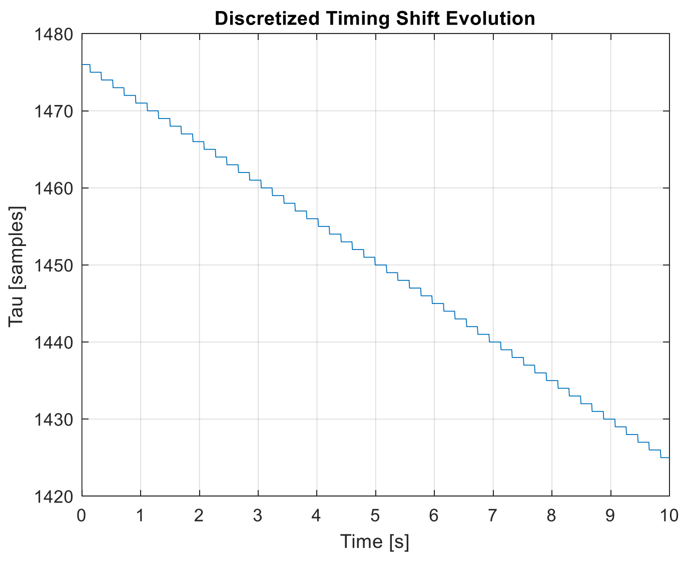


Fig. 3.6. First 10 seconds of the Discretized Timing Shift Evolution

Finally, a graphic representation of the discretized timing delay was obtained by using both the equation 3.14 and the distance profile that was shown in Fig. 3.4:

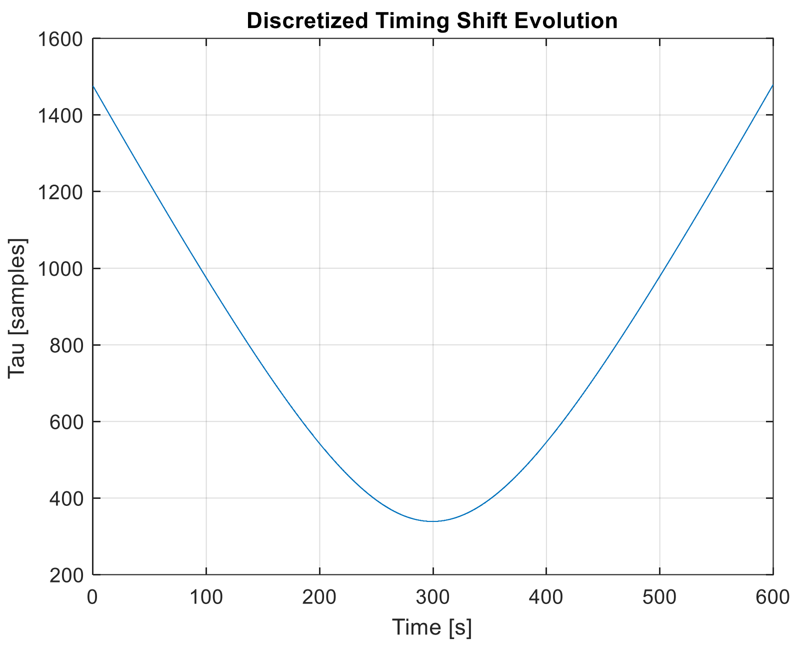


Fig. 3.7. Discretized Timing Shift Evolution

3.2.4. Matlab Implementation: Simulation Parameters

This section describes some of the main issues related to the implementation of the signal model described previously.

To start, it is necessary to define the sampling frequency (F_s) at which the received signal is being sampled. It was mentioned earlier that the burst bandwidth is $26.6667\ kHz$, so after applying the Nyquist Criteria, a minimum value for F_s that ensures anti-aliasing as well as the correct sampling of the tones is found:

$$F_s \geq 2 \cdot BW, \qquad Fs \geq 53.3334\ kHz \tag{3.15}$$

However, it is more useful to oversample the signal, since it allows to improve the resolution and the SNR (Signal-to-Noise Ratio), which appears to be highly interesting because satellite signals are received with low power. Also, it is important to set the sampling frequency to be an integer in order to avoid losing information due to decimal sampling.

In this way, the burst bandwidth is multiplied first by 3 to obtain an integer value of $F_s = 80\ kHz$, and then is again triply oversampled to obtain a final value for the sampling frequency as $F_s = 240\ kHz$.

Also, the noise variance is defined according to reference [10], which gives some typical values for this parameter as a function of the propagation conditions that apply for any given communications system (satellite communications for our purpose). Then, it is defined that ${\sigma_n}^2 = (400 \cdot 0.4)^2 = 25600$.

On the other hand, since the value of the noise variance is set to be high (according to ${\sigma_n}^2$) it is necessary to rescale the data-meaningful part of signal by a proportional factor of a power of two so that we can emulate the effect of amplifying the received signal with respect to the noise floor. In this way, the used scaling factor is 2^{10} and the expression of the received signal remains as:

$$r(i) = 2^{10} \cdot s(i) + n(i), \qquad i = 0,1,2,\ldots \tag{3.16}$$

Furthermore, it has been also necessary to discretize the equation 3.8 so that the distance profile can be generated in Matlab. Hence, the discretized expression for the distance profile is computed as:

$$d(i) = -c \cdot \sum_{i}^{N} \frac{f_D(i)}{fc} \cdot T_{int}\,, \qquad i = 0,1,2,\ldots \tag{3.17}$$

It should be also explained that integration period (T_{int}) in the previous expression equals the mean duration of a single burst ($6.8\,ms$) because, in this way, the minimum integrated entity will be a burst which only contains one tone. Then, only one tone is integrated at a time.

Once the simulator is run and the received signal is generated, it is saved into a binary file by writing its In-phase (I) and Quadrature (Q) components. Then, the expression for the received signal remains as:

$$r_{IQ}(i) = I_{r(i)} + jQ_{r(i)} = real\{r(i)\} + j \cdot imag\{r(i)\}, \qquad i = 0,1,2,\ldots \tag{3.18}$$

Finally, the following table summarizes the main simulation parameters:

Table 3.2. Main Simulation Parameters

F_s	σ_N	$scale$	T_{tone}	$tone\ duration$	$T_{simulation}$
$240 \cdot 10^3\ s^{-1}$	25600	2^{10}	$6.8 \cdot 10^{-3}\ s$	$2.6 \cdot 10^{-3}\ s$	$600\ s$

3.2.5. Signal Simulator Code Flowcharts

This section aims to show in a descriptive way the Matlab codes used for the signal simulation. For that purpose, we have used first a function that generates the received signal and then a main file that writes it into a binary file, as well as it allows to perform some operations that verify that the simulation was performed correctly.

To start, the aforementioned function allows to simulate our received signal as a function of some input parameters that are described in the following table:

Table 3.3. SimulateIridiumSignals(…) Function Input Parameters

Function[ ] = SimulateIridiumSignals(…)	
Input Parameter	Description
F_s	Sampling frequency
f_D	Doppler shift evolution generated externally
d	Distance profile generated externally
T_{int}	Integration period (equals the tone period)
T_{tone}	Tone Period
$tone\_duration$	Duration of a single tone
sim_time	Simulation time for which the signal is generated
$noise\_var$	Noise variance of the channel
$output\_file\_path$	Name of the file where the signal is saved

Moreover, the following list contains a description of some relevant points that need to be commented about the code. Mainly, it is justified the use of some predefined functions in Matlab:

- The circular shift that simulates the timing delay applied to the bursts of tones is implemented by using the Matlab function $circshift(A, k)$, which allows to circularly shift the positions in array A by k positions
- The channel noise is generated by using the Matlab function $rand(N)$, which returns a uniformly distributed random sequence of length N containting numbers in the interval $[0,1]$ (suitable to simulate a white complex Gaussian noisy channel)
- The signal reshaping is made by using the Matlab function $reshape(A, sz)$, which reshapes vector A using the size vector sz

Finally, the code flowchart for this function is shown in the figure on the next page.

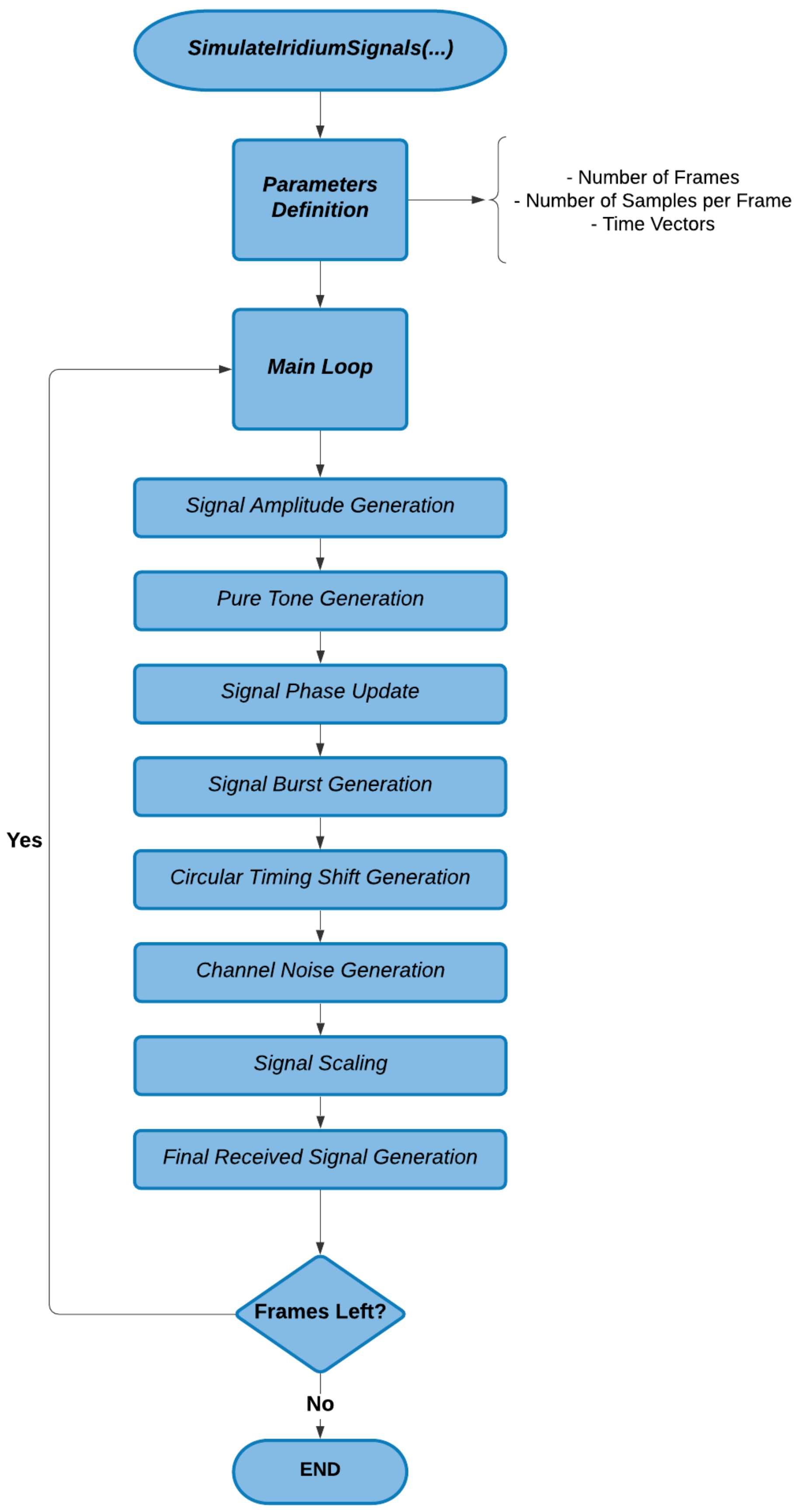


Fig. 3.8. SimulateIridiumSignals(…) Function Code Flowchart

On the other hand, a main file was used to generate the distance profile from a given Doppler shift time history as well as to read the simulated data, obtain some graphic representations of the received signal and perform some operation that verifies a proper signal simulation. The code flowchart is shown in the following figure:

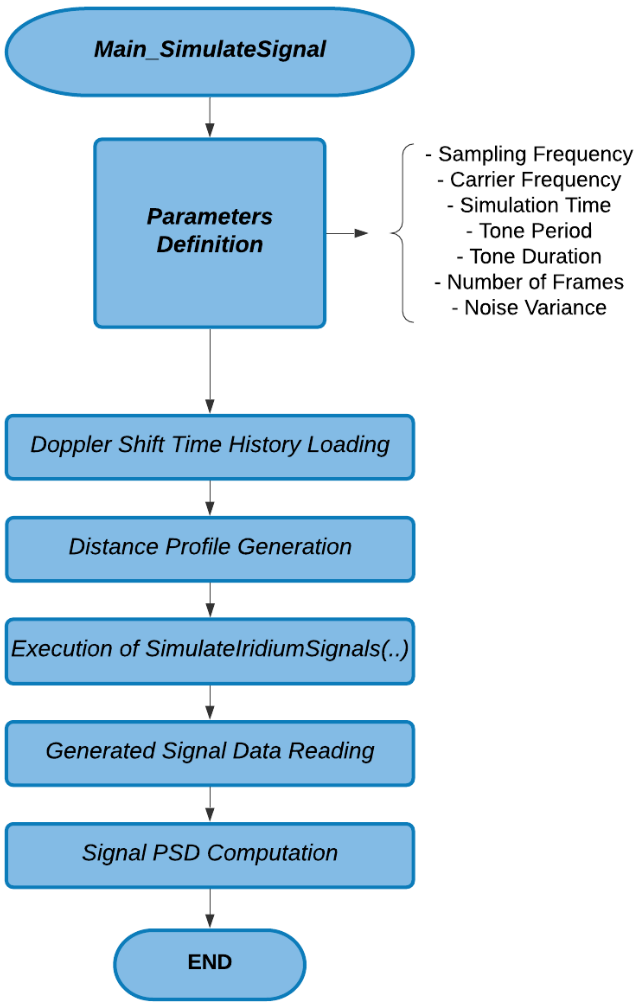


Fig. 3.9. Main_SimulateSignal Code Flowchart

3.2.6. Signal Graphic Representation

This section presents the simulated signal aiming to prove that it was generated correctly. Also, some PSD (Power Spectral Density) computations have been made in order to verify that the signal was generated according to both the Doppler shift and the distance profiles.

First, the signal time domain evolution without considering noise is shown. The figure on the next page is a representation of the firstly received 10 pure tones (each of them having a period of $6.8\ ms$ and a duration of $2.6\ ms$). In there, it may be noticed that tones are shifted in time so that the start of each of them does not necessarily match with a multiple of its period. For the same reason, the first tone isn't complete.

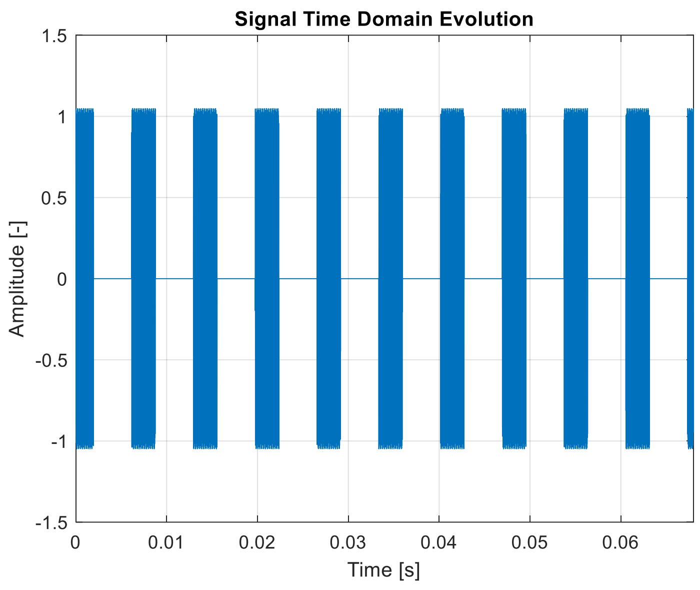


Fig. 3.10. Simulated Signal Time Domain Evolution with no Noise

Also, the following figure is the result of zooming over the first complete pulse during the interval $[6 \cdot 10^{-3}s, 9 \cdot 10^{-3}s]$. It can be appreciated that the shape of the envelope corresponds to an amplitude value which is modulated by a sinusoidal signal, which is in fact what it is expected to observe for our case where we are receiving a tone with variable amplitude.

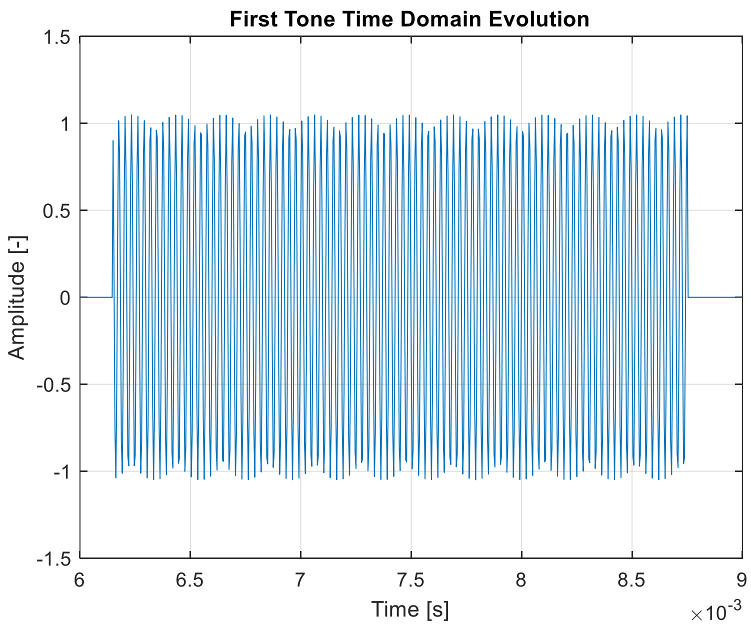


Fig. 3.11. First Simulated Tone Time Domain Evolution with no Noise

The following step has been to include the effect of the channel by applying white Gaussian noise to the signal shown in figures 3.10 and 3.11 in the same way that was described when we defined the signal model. The most important points that can be appreciated in the following figure are described:

- The amplitude of the signal has been rescaled by a factor 2^{10}. Then, when adding the effect of the noise, the maximum peaks are seen around an amplitude mean value of 1250
- The effect of the channel generates amplitude values in the order of magnitude of 500. However, there is still a high difference between the noise floor and the signal (roughly a factor 2.5)
- The envelope of the signal has been affected by noise, so its shape does not correspond to a pure sinusoidal anymore

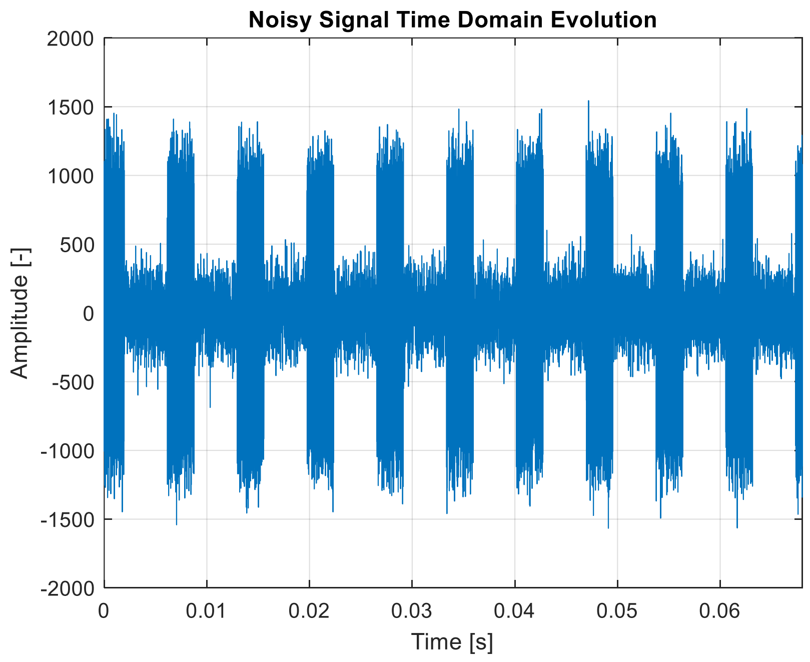


Fig. 3.12. Noisy Simulated Signal Time Domain Evolution

Once the signal has been demonstrated to be a sequence of tones, it is also interesting to inspect the IQ plot of the generated data. Since the signal is being simulated to go through a white Gaussian noisy channel, it is expected for the IQ plot to show higher concentration of samples over the origin. This can be explained because the ideal shape of the plot should be a Gaussian curve centered over the origin, which is in fact observed in Fig. 3.13.

Also, it may be useful to show the time evolution of the IQ samples in order to see if they are being transmitted by bursts as it is defined by Iridium Next documentation [9]. The figure on the following page shows a sequence containing IQ samples in a burst format.

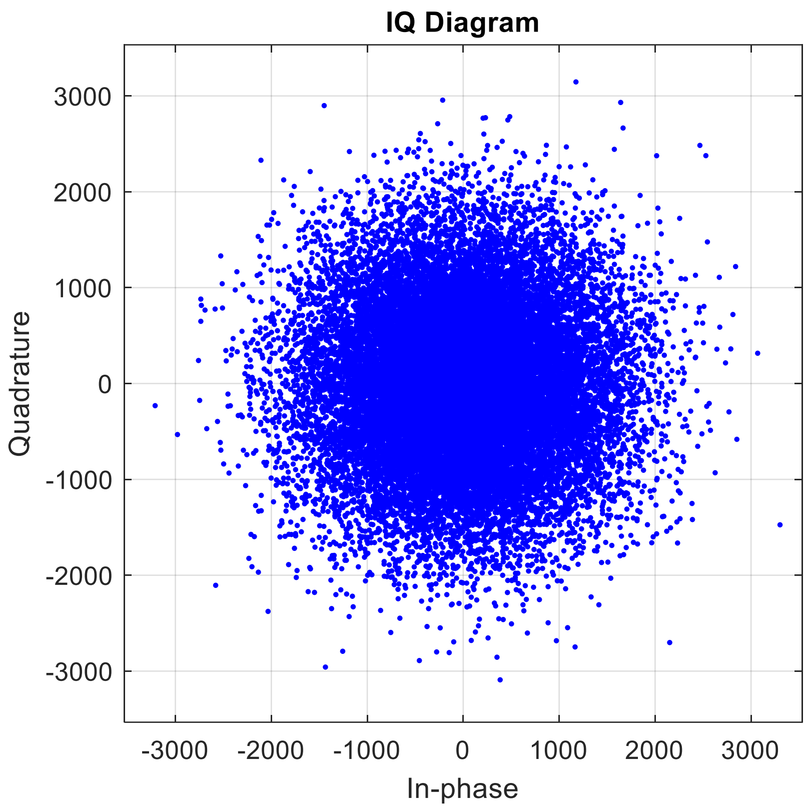


Fig. 3.13. IQ Diagram of the Simulated Signal

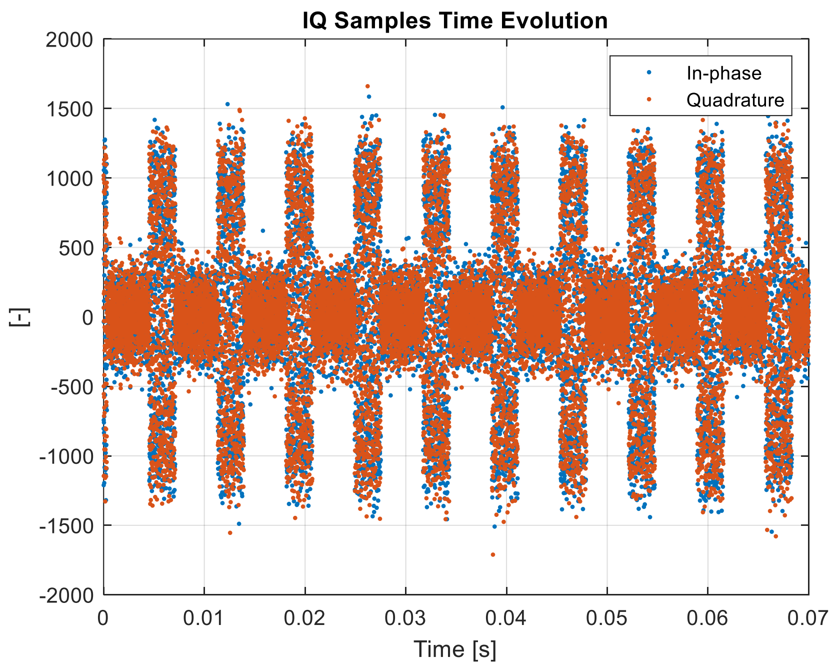


Fig. 3.14. IQ Samples Time Evolution of the Simulated Signal

Moreover, it is intended to demonstrate that the simulated signal was generated according to the Doppler shift profile that was defined in Fig. 3.3. This profile defines the Doppler frequency of the generated signal for all the simulation times, so it is expected that when computing the PSD (Power Spectral Density) estimate at any given time, a peak is observed exactly at the corresponding Doppler frequency.

The PSD was computed with Matlab using the function $pwelch()$, which allows to obtain an estimate of the power distribution over the frequency domain. The parameters used to perform such computations are defined in the following table:

Table 3.4. Pwelch(…) Function Parameters Definition

$pwelch(x[n], N_{window}, N_{overlap}, N_{DFT}, F_s)$	
Parameter	Value
$x[n]$: Input signal	Simulated IQ vector
N_{window}: Length of the Hamming window	500
$N_{overlap}$: Number of overlapped samples	50
N_{DFT}: Number of DFT points	2^{13}
F_s: Sampling frequency	$240\ kHz$

The function defined in the previous figure will compute the PSD estimate of the samples contained in the input vector. To do so, it will take segments of length N_{window}, overlapping $N_{overlap}$ samples from segment to segment. It will divide the frequency spectrum of F_s into the number of points defined by N_{DFT}. Then, it will generate a frequency vector of length N_{DFT} as well as a vector containing the PSD estimates associated to each frequency.

In this way, the following table contains a relationship between the maximum value observed on the PSD estimate and its frequency location for different simulation times:

Table 3.5. Frequency Location of the PSD estimate peaks

Simulation Time	Frequency	Maximum PSD estimate
$t = 100\ s$	$32.43\ kHz$	$27.72\ dB/Hz$
$t = 200\ s$	$24.45\ kHz$	$28.47\ dB/Hz$
$t = 400\ s$	$215.51\ kHz$	$28.46\ dB/Hz$
$t = 500\ s$	$207.53\ kHz$	$27.71\ dB/Hz$

Taking back to Fig. 3.3, it can be observed that the PSD estimate values shown in the previous table are symmetrical, which makes sense since the generated Doppler shift profile is an odd symmetry function. Moreover, for simulation times higher than $t = 300\ s$, the frequency values associated to the maximum values of the PSD estimate do not match with the Doppler frequencies of the simulated profile. Nevertheless, this is correct and can be explained by understanding that $pwelch()$, only computes PSD estimates for positive frequencies. Hence, the value of F_s must be subtracted to frequencies given by $pwelch()$, where the maximum PSD estimate is found. In this way, the following table shows the actual frequency locations of the peaks for simulation times $t = 400\ s$ and $t = 500\ s$:

Table 3.6. Actual Doppler Frequency Location for Maximum PSD estimates

Simulation Time	$pwelch()$ Frequency	Actual Frequency
$t = 400\ s$	$215.51\ kHz$	$215.51 - 240 = -24.49 kHz$
$t = 500\ s$	$207.53\ kHz$	$207.53 - 240 = -32.47\ kHz$

Finally, the following figures were obtained to show the frequency location associated to the maximum values of the PSD estimate at two different simulation times.

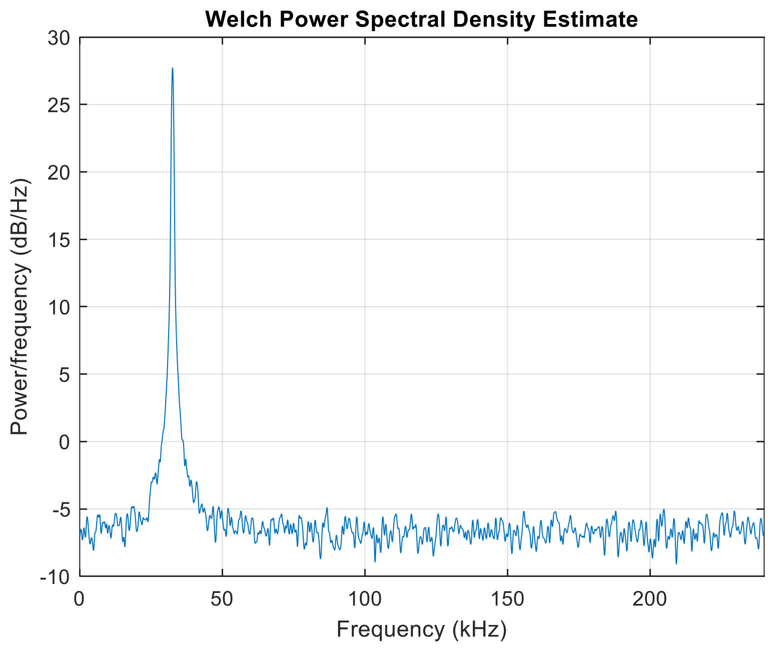


Fig. 3.15. Welch Power Spectral Density Estimate at Simulation Time = 100s

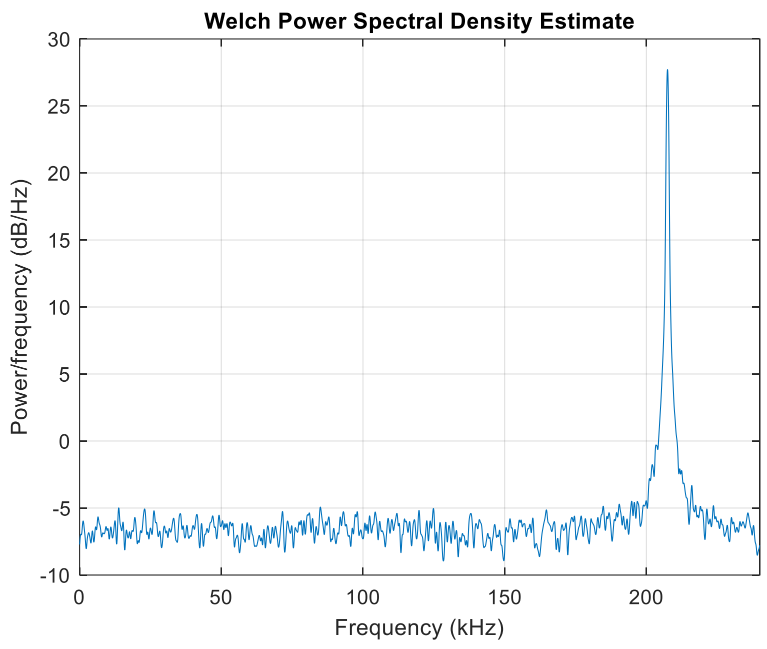


Fig. 3.16. Welch Power Spectral Density Estimate at Simulation Time = 500s

CHAPTER 4. RECEIVER DESIGN

4.1. Architecture Overview

The proposed receiver has been designed in order produce Doppler measurements from Iridium Next satellite signals which are useful to obtain a navigation solution. In this way, the receiver can acquire and track the Doppler frequency shift from the signals whose structure and main characteristics were presented in chapter 3.

The complete receiver architecture is shown in the following figure:

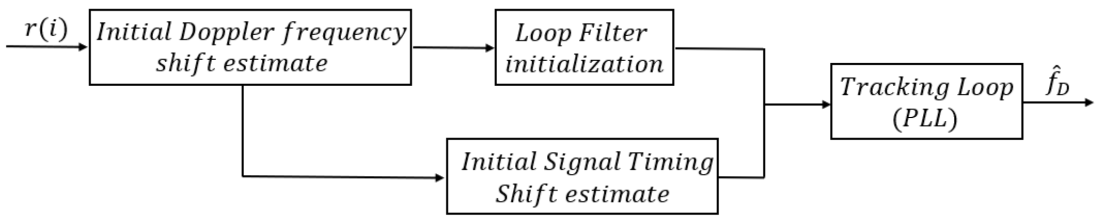

$r(i)$
Initial Doppler frequency shift estimate
Loop Filter initialization
Initial Signal Timing Shift estimate
Tracking Loop (PLL)
$\hat{f}_D$

Fig. 4.1. Proposed Receiver Architecture

The receiver implements a tracking loop based on a PLL (Phase-Locked Loop) that allows to track the phase of an incoming Iridium Next signal $r(i)$. This information, which is proportional to the Doppler frequency, allows to obtain estimates ($\hat{f}_D$) of the Doppler shift that satellite signals suffer when they are transmitted due to the motion of satellites with respect to the receiver. Finally, these estimates allow the receiver to position itself since they represent a measurement of the distance between the satellite and the receiver itself.

Since Iridium Next signals are transmitted over a TDMA (Time-Division Multiple Access) frame structure, the treatment of the incoming signal is made frame by frame. In this way, the first stage of the receiver consists on performing a PSD (Power Spectral Density) computation on the first received frame in order to obtain a first Doppler frequency shift estimate.

Then, this first Doppler estimate is used to initialized the loop filter contained in the tracking loop and, moreover, it is used to obtain an initial estimate of the distance between the receiver and the satellite, allowing to produce a first estimate of the timing shit that the signal suffered since it was transmitted.

Finally, these estimates are delivered to a PLL, which allows to produce a Doppler frequency shift estimate ($\hat{f}_D$) for each frame based on several control parameters from the previous frames that are detailed throughout this chapter. In this way, the receiver reconstructs the Doppler time history of the complete received signal.

4.2. Phase-Locked Loop Design

4.2.1. Structure and Block Diagram

The goal of our receiver is to acquire and to track the Doppler shift from the signals that Iridium Next satellites transmit, in order to estimate its position and to obtain a final navigation solution. For that purpose, it is necessary to implement a closed tracking loop that follows continuously up the phase from the incoming signal and uses it to produce an output signal whose phase is used to produce an estimate of the Doppler shift.

This control system is called a PLL (Phase-Locked Loop) and its working principle is based on generating an output signal whose phase is related to the phase of the input signal. The key of the operation is the ability to detect this phase difference between both signals, and to use this phase error to control the frequency of the loop. In this way, the Doppler shift estimate is obtained from the phase error between the input and the output signals.

The block diagram of the PLL that has been implemented on our receiver has been adapted from [11]. In there, the proposed structure is designed to be used while receiving simultaneously signals from multiple satellites of the same constellation. However, in our case, the receiver will only listen at one satellite at a time, so the structure must be redefined. In this way, our proposed block diagram for the PLL is shown in the following figure:

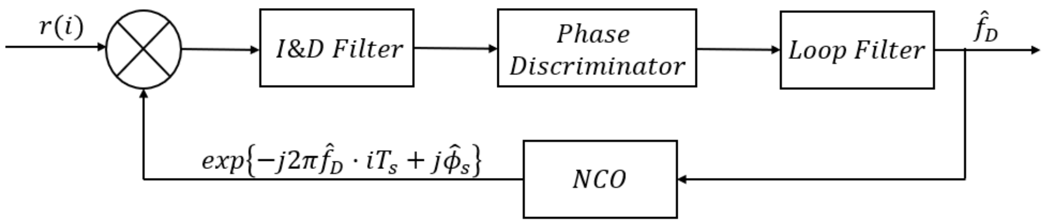


Fig. 4.2. Phase-Locked Loop Block Diagram

From the previous figure, it can be appreciated that the input signal from which the PLL is fed is the received signal itself, $r(i)$ which model was defined in chapter 3. Then, it is processed by multiple operations inside the closed loop and, finally, a Doppler shift estimate ($\hat{f}_D$) from the received signal is produced.

In this section, the different parts of the PLL will be described, at the same time that it is explained which operations are made to the input signal $r(i)$ in order to produce our desired output $\hat{f}_D$. Then, the stages of the PLL will be described in chronological order from when the signal is received until the Doppler shift estimate is produced, trying to give a mathematical description of the signal at each stage that aims to complete the overall understanding of the PLL.

Finally, the different parts that shape the PLL and whose details are given in the following sub-sections are listed as follows:

1. Numerically Controlled Oscillator (NCO)
2. Integrate and Dump (I&D) Filter
3. Phase Discriminator
4. Loop Filter

4.2.2. Numerically Controlled Oscillator

The first stage of the PLL consists on a Numerically Controlled Oscillator (NCO) that generates a sinusoidal wipe off signal $s_{NCO}[k]$ that is used to multiply the received signal $r(i)$, in order to obtain the phase error, which will be delivered to the phase discriminator of the PLL. The mathematical expression of $s_{NCO}[k]$ at time step $k+1$ is given by:

$$s_{NCO}[k+1] = exp\{-j2\pi\hat{f}_D[k] \cdot iT_s + j\hat{\phi}_s[k]\}, \qquad k = 0,1,2,\ldots \tag{4.1}$$

Where:

- $\hat{f}_D[k]$ is the current Doppler shift estimate maintained by the PLL at time step k
- $\hat{\phi}_s[k]$ is the current phase estimate maintained by the PLL at time step k

Then, given that our PLL processes the received signal frame-by-frame and denoting N the number of samples contained into one frame, an expression for the product between $r(i)$ and $s_{NCO}[k]$ at time step $k+1$ is given by:

$$p[k+1] = \frac{1}{N}\sum_{i=0}^{N-1} r(i)\, exp\{-j2\pi\hat{f}_D[k] \cdot iT_s - j\hat{\phi}_s[k]\} \tag{4.2}$$

Also, since the complete received signal is given by the following expression:

$$r(i) = \sqrt{A(i)} \cdot exp\{j2\pi f_D(i) \cdot iT_s + j\phi_s(i)\} + n(i), \qquad i = 0,1,2\ldots \tag{4.3}$$

The resulting product signal at time step $k+1$ that contains N samples corresponding to the current frame at this step is expressed as follows:

$$p[k+1] = \sqrt{A[k+1]} \cdot \exp\{j\Delta\phi_s[k+1]\} + n[k], \qquad k = 0,1,2,\ldots \tag{4.4}$$

Where:

- $A[k+1]$ is the amplitude of the current frame at time step $k+1$
- $n[k]$ is zero-mean white Gaussian noise with variance ${\sigma_n}^2$
- $\Delta\phi_s[k+1]$ is the phase error at time step $k+1$

As we discussed previously, the goal of this PLL stage is to obtain the phase error between the current frame of the received signal and the wipe off signal. At time step $k+1$, this variable contains information about the Doppler frequency (f_D) of the received signal at time step k, and the estimate of the Doppler frequency ($\hat{f}_D$) at the same time step. Moreover, the phase error $\Delta\phi_s[k+1]$ contains information of the phase estimate ($\hat{\phi}_s$) given by the PLL and the phase (ϕ_s) of the received frame. Finally, the expression of the phase error remains as:

$$\Delta\phi_s[k+1] = 2\pi T_{int}\left(f_D[k] - \hat{f}_D[k]\right) + \left(\phi_s[k] - \hat{\phi}_s[k]\right) \quad \textbf{(4.5)}$$

4.2.3. Integrate and Dump (I&D) Filter

Once the received signal has been wiped off by multiplying it with the sine wave generated in the NCO, the next stage on the PLL is in charge of integrating all the samples from the current frame in order to accumulate the total phase error that is scattered among all samples.

In this way, since the receiver acquires the signal frame-by-frame, at time step $k+1$, the phase error $\Delta\phi_s[k+1]$ of the current frame is distributed among all the N samples of this frame. Then, an intermediate stage between the NCO and the phase discriminator that integrates the phase from all the samples (the I&D filter) is needed.

Before moving on explaining the operation of the I&D filter, it might be noticed that the process of integrating the samples included in the current frame cannot be done by simply accumulating all of them. First, the received frames are made up from a combination of samples that contain information of the unmodulated tones and by samples that only contain noise. This is produced because Iridium Next signals are transmitted with a duty cycle and the tone duration is shorter than the tone period.

Then, the first step is to determine which is the optimum number of samples (N_{opt}) from the current frame that should be integrated at each step k in order to accumulate all the phase error while disregarding the samples that only contain noise. In this way, the sampler number (N_{opt}) is computed to ensure that the SNR on the current frame is maximum.

As we discussed in chapter 3, the duration of a tone is $2.6ms$ and the period of a tone is $6.8ms$. Then, defining the sampling frequency (F_s) at which the received signal is being acquired, the total number of samples contained into one tone can be computed as:

$$N_0 \stackrel{\text{def}}{=} N_{tone} = 2.6 \cdot 10^{-3} \cdot F_s \tag{4.6}$$

Also, considering the optimum number of samples (N_{opt}) to integrate on each frame, an expression for the integrated signal is given by:

$$s_{int} = \frac{1}{N_{opt}} \sum_{i=1}^{N_{opt}} (s_i + n_i), \qquad where\ s_i = \begin{Bmatrix} A, & if\ i \leq N_0 < N_{opt} \\ 0, & otherwise \end{Bmatrix} \tag{4.7}$$

Where s_i are the sample parts that may contain or not information about the tones, which is reflected into the signal amplitude (A) associated to the current frame, and n_i are the noisy sample parts.

On the other hand, the SNR is defined as the power ratio between the meaningful part of a signal and the background noise. Given that we are defining the channel noise to be a random variable, its power can be computed as the expected value of its mean squared $E(M^2)$. However, since the noisy samples (n_i) of the current frame are assumed to be IID (Independent and Identically Distributed) with zero mean, the noise has an expected value of zero. Then, its power is computed as its variance normalized by the total number of samples to be integrated and the expression for the SNR on the current frame is given by:

$$SNR = \left(\frac{1}{N_{opt}} \sum_{i=1}^{N_{opt}} s_i\right)^2 \cdot \left(\frac{{\sigma_n}^2}{N_{opt}}\right)^{-1}, \qquad {\sigma_n}^2 = var(n_i) \tag{4.8}$$

Hence, N_{opt} can be found by finding the optimum number of samples that maximize the previous SNR. Then, the mathematical problem consists on solving the following equation:

$$\frac{d(SNR)}{dN_{opt}} = \frac{d}{dN_{opt}} \left[\left(\frac{1}{N_{opt}} \sum_{i=1}^{N_{opt}} s_i\right)^2 \cdot \left(\frac{{\sigma_n}^2}{N_{opt}}\right)^{-1}\right] = 0 \tag{4.9}$$

Finally, it can be demonstrated that the expression of the SNR has an absolute maximum at N_0, and thus the previous equation is solved for $N_{opt} = N_0$. In this way, it has been proved that the optimum number of samples to integrate over one frame at each step of the PLL is the number of samples contained inside one tone duration (N_0), as it could be predicted since the samples contained outside from this interval only contain noise.

Nevertheless, it might be noticed that having obtained this sampler number is not enough to integrate correctly all the phase error ($\Delta\phi_s$) spread among all the samples from the current frame that contain the tone. In fact, it is necessary to know where the tone starts on each received frame in order to integrate N_0 samples from this tone start position.

This can be explained by remarking that the received tones might be delayed in time due to wave propagation, so the time instant where the tone starts is different for every received frame. Then, since time delay can be defined as the number of samples that each tone is shifted, a strategy to find the starting sample of the tone at each frame must be designed. If not, although the I&D filter integrates exactly N_0 samples at each frame, it could be the case that all the integrated samples only contain noise if the tone appears to be shifted more than N_0 and thus the complete phase error ($\Delta\phi_s$) of the frame would be lost. In this way, a method for tracking the timing shift of the signals is described in section 4.4.

Finally, assuming that the samples of the current frame have been integrated properly from the tone start, an expression for the integrated frame is given by:

$$s_{int} = \frac{1}{N_0}\sum_{i=1}^{N_0}\left(\sqrt{A}\cdot \exp\left\{2\pi T_{int}\left(\Delta f_{D_i}(i)\right) + \Delta\phi_{s_i}(i)\right\} + n(i)\right) \tag{4.10}$$

Furthermore, defining $\Delta\phi_{s_i}$ and Δf_{D_i} respectively as the phase difference and the Doppler shift associated to each sample of the current frame, the total phase of the current frame which is the aimed phase error ($\Delta\phi_s$) can be computed as:

$$\Delta\phi_s = \frac{1}{N_0}\sum_{i=1}^{N_0} 2\pi T_{int}\left(\Delta f_{D_i}(i)\right) + \Delta\phi_{s_i}(i) + n(i) \tag{4.11}$$

4.2.4. Phase Discriminator

The PLL stage that is used to detect the phase error ($\Delta\phi_s$) of the current frame is the phase discriminator. This element is essential to track properly the phase of the incoming signal and it is used to compute the phase difference between the input signal, $r(i)$ and the feedback auto-generated wipe off signal $s_{NCO}[k]$ that is produced by the NCO. Then, this computation that contains direct information about the Doppler shift of the current frame is delivered to the loop filter in order to obtain the final Doppler shift estimate that will be used to produce a navigation solution.

In order to explain the approach that has been followed to implement the phase discriminator into our receiver, let recall the type of signal that will be inputted to this PLL element. At time step k, the I&D filter produced an integrated signal (s_{int}) whose expression can be rewritten as:

$$s_{int}[k] = \frac{\sqrt{A}}{N_0}\exp\{\Delta\phi_s[k]\} + n[k] \tag{4.12}$$

Furthermore, using the Euler's formula given by the following expression:

$$e^{j\varphi} = \cos(\varphi) + j \cdot \sin(\varphi) \tag{4.13}$$

It is possible to define the output signal from the I&D filter as a complex IQ number in the form:

$$s_{int}[k] = \frac{\sqrt{A}}{N_0}\left(\cos(\Delta\phi_I[k]) + j \cdot \sin\left(\Delta\phi_Q[k]\right)\right) + n[k] \tag{4.14}$$

Then, defining $\Delta\phi_I$ as the phase error contained inside the real part of the signal and $\Delta\phi_Q$ as the complex phase error, an expression for the complete $\Delta\phi_s$ of the current frame at time step k is given by:

$$\Delta\phi_s[k] = \arctan\left(\frac{imag(s_{int}[k])}{real(s_{int}[k])}\right) = \arctan\left(\frac{\Delta\phi_Q[k]}{\Delta\phi_I[k]}\right) \tag{4.15}$$

In this way, it might be noticed that the implementation of the phase discriminator is straightforward since the operation that it has to perform is simply a phase computation. For that purpose, this stage must be able to detect the phase of a complex number in the interval $[-\pi, \pi]$ in order to avoid phase ambiguities, so the Matlab function $atan2$ (whose performance is discussed in section 4.5) will be used to include the discriminator into our receiver.

Moreover, it is important to recall that the expected value of the phase error is almost zero because this variable contains information about the differences between the signal phase (ϕ_s) and the Doppler shift (f_D) from the current frame and its estimates. Then, since it is expected that the estimates produced by the PLL are as close as possible to the actual signal values, the phase error must be small. Finally, this phase error will be filtered in the next PLL stage in order to disregard those frames that were received incorrectly and that produced a phase error overly high that it is not useful to obtain the final Doppler shift estimate.

4.3. Loop Filter Design

4.3.1. Block Diagram and Transfer Function

The most critical part in the design of the PLL is to choose an appropriate loop filter that bounds the Doppler shift estimate within an appropriate range according to what it is expectable to observe in signals transmitted by Iridium Next.

At each iteration of the PLL, the discriminator stage allows to calculate the signal phase error of the current frame ($\Delta\phi_s$), which is related to the Doppler shift and which is used as an input for the loop filter. Then, since it is expected that frames received from Iridium Next Satellites produce Doppler shifts within the interval $[-35\ kHz, 35\ kHz]$, the output phase given by the loop filter must be bounded according to this range. Moreover, the filter uses that the phase error of the current frame ($\Delta\phi_s$) must be ideally null. In this way, the loop filter will only select the phase values given by the discriminator stage that fulfill these conditions, leading to define it as an LPF (Low Pass Filter).

The design of the filter has been made by also considering the high dynamics of the environment, which implies that the filter order is chosen so that high rates of change on the Doppler shift can be tracked. Satellites are celestial bodies moving at high velocities, and the signals transmitted by them may be received with highly variable Doppler shifts depending on the distance between them and the receiver. Then, the filter response must be sharp and fast enough to account for these high rates, so the filter is chosen to be of second order.

The block diagram of the loop filter has been adapted from an existing architecture proposed in [12] which is shown in the following figure:

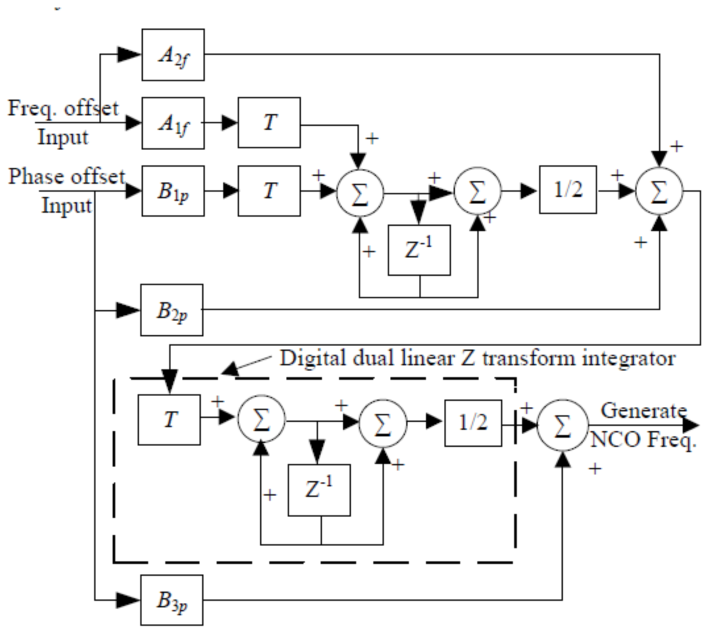


Fig. 4.3. Carrier Tracking Loop of 3rd order PLL defined by [12]

The previous figure shows a Carrier Tracking Loop that contains a 3rd order PLL as well as a 2nd order FLL (Frequency Locked Loop). However, since our receiver aims to track the Doppler shift only from the information given by the signal phase, the FLL stage can be removed.

Furthermore, the previous diagram can be simplified by translating it into the Laplace domain, where the Digital dual linear Z transform integrator is expressed as a generic Laplace integrator, which transfer function is well known and can be obtained as follows:

Let the Laplace transform $F(s)$ of a given function $f(t)$ be expressed as:

$$F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t)dt \tag{4.16}$$

The transfer function $H(s)$ of a given system in the Laplace domain as a function of an input signal $U(s)$ and an output signal $Y(s)$ can be computed as follows:

$$H(s) = \frac{Y(s)}{U(s)} \;\rightarrow\; Y(s) = H(s) \cdot U(s) \tag{4.17}$$

Then, it can be proved that to integrate $f(t)$ in the time domain is equivalent to define the following transfer function in the Laplace domain:

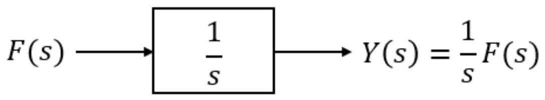


Fig. 4.4. Laplace Integrator

First, let define the Laplace transform of the integral of $f(\tau)$ (after applying the change of variable $t = \tau$) as:

$$\mathcal{L}\left\{\int_0^t f(\tau)d\tau\right\} = \int_0^\infty \left(\int_0^t f(\tau)d\tau\right) e^{-st}dt \tag{4.18}$$

Then, the previous expression can be integrated by parts applying:

$$\int_a^b u \cdot dv = u \cdot v \Big|_a^b - \int_a^b v \cdot du \tag{4.19}$$

Where the values of the integral are defined as shown in the following table:

Table 4.1. Integral Values for the Laplace Integrator Demonstration

$u = \int_0^t f(\tau) d\tau$	$v = -\frac{1}{s} e^{-st}$
$du = f(t) dt$	$dv = e^{-st} dt$

In this way, equation (4.19) can be rewritten as:

$$\mathcal{L}\left\{\int_0^t f(\tau) d\tau\right\} = \left[-\frac{1}{s} e^{-st}\right]_0^\infty \int_0^t f(\tau) d\tau - \int_0^\infty \left(-\frac{1}{s} e^{-st}\right) f(t) dt \tag{4.20}$$

Which leads to obtain the following expression by rearranging some terms and applying the Laplace Transform definition:

$$\mathcal{L}\left\{\int_0^t f(\tau) d\tau\right\} = -\frac{1}{s}\left[e^{-st} \int_0^t f(\tau) d\tau\right]_0^\infty + \frac{1}{s} F(s) \tag{4.21}$$

Then, the product in brackets can be expanded by evaluating its terms as:

$$\mathcal{L}\left\{\int_0^t f(\tau) d\tau\right\} = -\frac{1}{s}\left(e^{-s\infty} \int_0^\infty f(\tau) d\tau - e^{-s0} \int_0^0 f(\tau) d\tau\right) + \frac{1}{s} F(s) \tag{4.22}$$

Where the first integral is simplified because is being multiplied by an exponential that tends to zero, and the second integral is also simplified since both of its limits are zero. Finally, the following expression that proves the system shown in Fig. 4.4 is found:

$$\mathcal{L}\left\{\int_0^t f(\tau) d\tau\right\} = \frac{1}{s} F(s) \tag{4.23}$$

After having proved that the integrators shown in the diagram of Fig. 4.3 are equivalent to the Laplace integrator, and after having discussed that our receiver will only consider a PLL, a block diagram for the loop filter is presented in the figure on the next page.

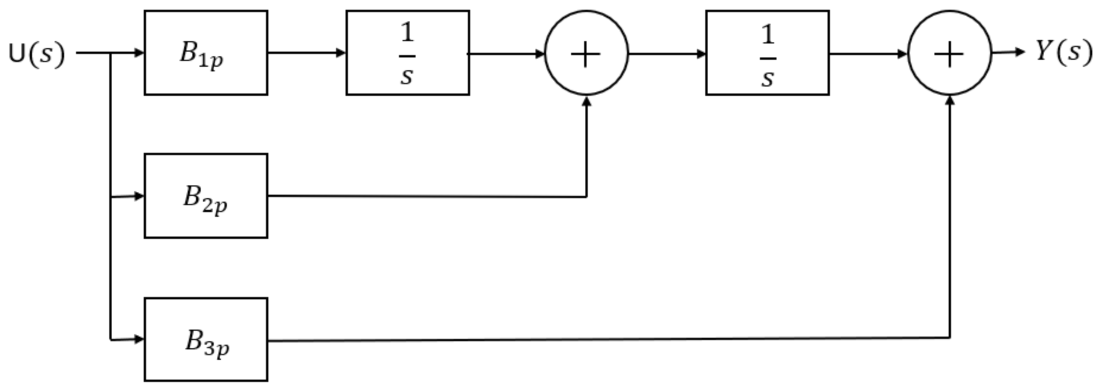


Fig. 4.5. Loop Filter Block Diagram

The previous model consists on a second-order loop filter since it uses two integration stages and so the order of the denominator of its transfer function is two. The expression of the output $Y(s)$ as a function of the input signal $U(s)$ remains as:

$$Y(s) = \left(\frac{1}{s}B_{1p}U(s) + B_{2p}U(s)\right)\frac{1}{s} + B_{3p}U(s) \tag{4.24}$$

Then, using the definition given in equation (4.17), the transfer function $F(s)$ of the loop filter is expressed as:

$$F(s) = \frac{Y(s)}{U(s)} = \frac{s^2 \cdot B_{3p} + s \cdot B_{2p} + B_{1p}}{s^2} \tag{4.25}$$

Particularizing the previous expression for the PLL implemented in the receiver, the input of the loop filter $X(s)$ is defined to be the output phase error given by the discriminator ($\Delta\phi_s$). In this way, the phase output $Y(s)$ is proportional to the Doppler shift of the current frame, which is used for our positioning purposes.

The coefficients (B_{ip}) of the transfer function in (4.25) can be expressed as a function of the noise bandwidth (B_L), which is an indicator of the stability of the loop and can be tuned in order to obtain a sharper response depending on the dynamics of the environment.

According to reference [12], the following expression defines the relationship between the natural frequency of the loop filter and its noise bandwidth:

$$B_L = 0.82\omega_L \tag{4.26}$$

Accordingly, the following table summarizes the parametrization of the loop filter coefficients as a function of the noise bandwidth:

Table 4.2. Loop Filter Coefficients Parametrization

$B_{1p} = 0.1{\omega_L}^3 = 0.1\left(\frac{B_L}{0.82}\right)^3$
$B_{2p} = 1.12{\omega_L}^2 = 1.12\left(\frac{B_L}{0.82}\right)^2$
$B_{3p} = 2.4\omega_L = 2.4\left(\frac{B_L}{0.82}\right)$

Finally, the expression for the transfer function $F(s)$ of the loop filter is given by:

$$F(s) = \frac{s^2 \cdot 2.4\left(\frac{B_L}{0.82}\right) + s \cdot 1.12\left(\frac{B_L}{0.82}\right)^2 + 0.1\left(\frac{B_L}{0.82}\right)^3}{s^2} \quad \textbf{(4.27)}$$

Setting the noise bandwidth (B_L) should be made according to the following considerations:

- High values of B_L ensure that the filter output, which is proportional to the Doppler shift estimate, converges faster to the actual Doppler value of the received signal. However, it increases the loop instability and thus the obtained output is noisier, which can be reflected into a larger phase error ($\Delta\phi_s$) of the signal
- Lower values of B_L produce a clearer and more accurate filter output but, as a side effect, the convergence of the Doppler shift estimate is produced in a slower rate and thus the number of iterations that the PLL needs to accomplish are increased

In this way, setting the noise bandwidth is closely related to the dynamics of the environment: when the Doppler shift rate is high, B_L should be set to a high value so the Doppler estimate converges quickly to the actual value. On the other hand, when the dynamics are lower, B_L should be decreased so that the accuracy of the Doppler estimate is increased. A further explanation about the selection of this parameter as well as a proposed approach to select it adaptively is given in section 4.3.4.

4.3.2. Continuous Time-Invariant State-Space Model

The state-space model of a physical system is a mathematical representation that describes its dynamics as a set of input, output and state variables related by first-order differential equations. In this way, the values of the output variables depend on the values of the state variables, whose values evolve in time in a way that depend on the externally imposed values of input variables.

Generically, the continuous time-invariant state-space model of a system is given by the following set of equations:

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t)\end{aligned} \qquad \textbf{(4.28)}$$

Where:

- $x(t)$ is defined to be the state vector
- $y(t)$ is defined to be the output vector
- $u(t)$ is defined to be the input vector
- A, B, C, D are defined to be the state-space matrices

For our particular case of a PLL design, it is useful to define the state-space model of the loop filter that describes the dynamics of the output phase (which is related to the Doppler shift) as a function of the input phase given by the discriminator ($\Delta\phi_S$). In this way, the output vector $y(t)$ may predict the actual evolution of the signal Doppler shift, yielding to the following definitions:

$$\begin{aligned}u(t) &\stackrel{\text{def}}{=} \Delta\phi_S(t) \\ y(t) &\stackrel{\text{def}}{=} 2\pi f_D(t)\end{aligned} \qquad \textbf{(4.29)}$$

Moreover, the dimensions of the state-space matrices are defined in the following table as a function of the number of inputs (p) and outputs (q), as well as the number of state variables (n):

Table 4.3. Definition of the State-Space Matrices Dimensions

Sate Matrix (A)	Input Matrix (B)	Output Matrix (C)	Feedthrough Matrix (D)
$\dim[A] = n\ x\ n$	$\dim[B] = n\ x\ p$	$\dim[C] = q\ x\ n$	$\dim[D] = q\ x\ p$

Finally, it may be noticed that the goal of deriving the state-space model of the loop filter is to find the state-space matrices that allow to obtain the Doppler shift estimate for each frame. Then, the derivation contained in the following pages justify the selection of matrices A, B, C, D.

Let first particularize the matrices dimensions for our case of the loop filter where we have one input $\Delta\phi_s(t)$, one output $2\pi f_D(t)$ and two state variables:

- $\dim[A] = 2\,x\,2$
- $\dim[B] = 2\,x\,1$
- $\dim[C] = 1\,x\,2$
- $\dim[D] = 1\,x\,1$

Then, the state-space model of the loop filter transfer function $F(s)$ is found in a CCF (Controllable Canonical Form). The reason of using this approach is because it gives a generalized representation of the system which is ensured to be always implementable. Besides, since the order of the denominator from $F(s)$ equals the order of the numerator, a CCF of the model can be found. Obtaining the state-space matrices is defined as follows:

Taking back to the loop filter transfer function $F(s)$ defined in (4.25), let define an intermediate variable $Z(s)$ that is included in $F(s)$ as follows:

$$F(s) = \frac{Y(s)}{U(s)} \cdot \frac{Z(s)}{Z(s)} = \frac{s^2 \cdot B_{3p} + s \cdot B_{2p} + B_{1p}}{s^2} \cdot \frac{Z(s)}{Z(s)} \qquad \textbf{(4.30)}$$

Then, solving for $Y(s)$ and $U(s)$ yields to:

$$\begin{aligned} Y(s) &= Z(s) \cdot \left(s^2 \cdot B_{3p} + s \cdot B_{2p} + B_{1p}\right) \\ U(s) &= Z(s) \cdot s^2 \end{aligned} \qquad \textbf{(4.31)}$$

At this point, let convert the previous expressions into their associated differential equations by applying the inverse Laplace transform of the n^{th} derivative of a continuous-time function by applying:

$$\mathcal{L}\left\{s^n F(s) - \sum_{i=1}^{n} s^{(n-i)} f^{(i-1)}(0)\right\}^{-1} = \frac{d^n f(t)}{dt^n} \qquad \textbf{(4.32)}$$

Where the summation term is cancelled since we are imposing the following initial condition on our studied system:

$$f^{(n)}(0) = \frac{d^n f(t=0)}{dt^n} = 0 \qquad \textbf{(4.33)}$$

Then, the coupled set of differential equations that define the state-space model of the loop filter are given by:

$$\begin{aligned} y &= \ddot{z} \cdot B_{3p} + \dot{z} \cdot B_{2p} + z \cdot B_{1p} \\ u &= \ddot{z} \end{aligned} \tag{4.34}$$

At this point, we can also define the state variables (x_n) of the model to be the intermediate variable (z) and its first time-derivative ($\dot{z}$). Moreover, the first time-derivative of both state variables ($\dot{x}_n$) are defined as follows:

$$\begin{aligned} x_1 &= z, \qquad \dot{x_1} = \dot{z} = x_2 \\ x_2 &= \dot{z}, \qquad \dot{x_2} = \ddot{z} = u \end{aligned} \tag{4.35}$$

After that, we can rewrite the expression of the output (y) as a function of the input (u) and the state variables (x_n):

$$y = u \cdot B_{3p} + \dot{z} \cdot B_{2p} + z \cdot B_{1p} \tag{4.36}$$

Furthermore, if the dynamical system is linear, time-invariant and finite-dimensional, the differential equations that describe its behavior can be written in matrix form. Then, the first coupled equation from (4.28) can be expressed as:

$$\dot{x} = Ax + Bu \rightarrow \begin{pmatrix} \dot{z} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \cdot \begin{pmatrix} z \\ \dot{z} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \cdot \ddot{z} \tag{4.37}$$

Developing the previous expression, the following equations are obtained:

$$\begin{aligned} \ddot{z} \cdot b_1 + \dot{z} \cdot a_2 + z \cdot a_1 &= \dot{z} \\ \ddot{z} \cdot b_2 + \dot{z} \cdot a_4 + z \cdot a_3 &= z \end{aligned} \tag{4.38}$$

Finally, solving for a_n and b_n, the state-space matrices A and B are found:

$$\begin{aligned} A &= \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ B &= \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{aligned} \tag{4.39}$$

Following in the same way, the second coupled equation from (4.28) can be expressed as:

$$y = Cx + Du = (c_1 \quad c_2) \cdot \begin{pmatrix} z \\ \dot{z} \end{pmatrix} + D \cdot u \tag{4.40}$$

Then, applying the definition of y given by equation (4.36) yields to:

$$u \cdot B_{3p} + \dot{z} \cdot B_{2p} + z \cdot B_{1p} = (c_1 \quad c_2) \cdot \begin{pmatrix} z \\ \dot{z} \end{pmatrix} + D \cdot u \tag{4.41}$$

Finally, solving for c_n the state-space matrices C and D are found:

$$\begin{gathered} C = (B_{1p} \quad B_{2p}) \\ D = B_{3p} \end{gathered} \tag{4.42}$$

4.3.3. Explicit Discrete Time-Invariant State-Space Model

The next step after having found the continuous time-invariant model of the loop filter is to discretize it and so to be implemented in Matlab. It may be noticed that the signal model given in chapter 3 is a mathematical representation in the discrete-time domain, so the signal phase and the Doppler shift associated to each frame are also given by discrete-time sequences. Then, the input and output vectors as well as the state vector must be discretized accordingly.

Let the state-space model of an explicit discrete-time system be written in terms of a recursive formula by using linear matrix difference equations as:

$$\begin{aligned} x[k+1] &= A_d x[k] + B_d u[k] \\ y[k] &= C_d x[k] + D_d u[k] \end{aligned} \tag{4.43}$$

Where:

- $x[k]$ is defined to be the discretized state vector
- $y[k]$ is defined to be the discretized output vector
- $u[k]$ is defined to be the discretized input vector
- $A_d, B_d, C_d,\ D_d$ are defined to be the discretized state-space matrices

The discretization of the continuous time-invariant state-space model defined in (4.34) can be obtained by using the Integral Method Approximation [13], which assumes that the system input is constant during the integration time (T_{int}) for which the model is being discretized. In this way, this method approximates the input signal by its staircase form as:

$$u(t) = u[k \cdot T_{int}], \qquad k \cdot T_{int} \leq t < (k+1)T_{int}, \qquad k = 0,1,2,\dots \tag{4.44}$$

Hence, this section describes the approach followed to obtain the discretized state-space matrices ($A_d, B_d, C_d,\ D_d$) from the continuous matrices (A, B, C, D) that were found in section 4.3.2.

First, let the impact of the approximation given by (4.43) into the solution of the state-space equations at instant $t = T_{int}$ to be found as:

$$x[T_{int}] = e^{A \cdot T_{int}} \cdot x[0] + \int_0^{T_{int}} e^{A \cdot (T_{int} - \tau)} \cdot Bu[0] d\tau \tag{4.45}$$

Then, by expanding the exponential term inside the integral from the previous expression, it is written as:

$$x[T_{int}] = e^{A \cdot T_{int}} \cdot x[0] + e^{A \cdot T_{int}} \int_0^{T_{int}} e^{-A \cdot \tau} d\tau \cdot Bu[0] \tag{4.46}$$

Afterwards, we can define an intermediate impact variable to be $\Phi(T_{int}) = e^{A \cdot T_{int}}$, yielding to define $x[T_{int}]$ as:

$$x[T_{int}] = \Phi(T_{int}) \cdot x[0] + \int_0^{T_{int}} \Phi(T_{int} - \tau) d\tau \cdot Bu[0] \tag{4.47}$$

Finally, comparing the previous expression with the general form of the discretized model given by equation (4.43), it can be concluded that the expressions for the discretized matrices A_d and B_d at instant $t = T_{int}$ are given by:

$$A_d(T_{int}) = \Phi(T_{int}) = e^{A \cdot T_{int}} \tag{4.48}$$

$$B_d(T_{int}) = e^{A \cdot T_{int}} \int_0^{T_{int}} e^{-A \cdot \tau} d\tau \cdot B = B \int_0^{T_{int}} e^{A \cdot (T_{int} - \tau)} d\tau \tag{4.49}$$

It should be noticed that the previous matrices were obtained for a specific time instant, but since our system is defined to be time-invariant, the same expressions apply for any given time. For instance, at time $t = (k+1)\, T_{int}$, the first coupled difference equation from (4.43) can be rewritten as:

$$x[(k+1)T_{int}] = \Phi[(k+1)T_{int} - kT_{int}]x[kT_{int}] + \int_{kT_{int}}^{(k+1)T_{int}} \Phi[(k+1)T_{int} - \tau]\, d\tau \cdot Bu[kT_{int}] \quad \textbf{(4.50)}$$

After some manipulation, the original expression in (4.43) that is evaluated for every multiple of T_{int}, which defines all the discrete-time domain, is found:

$$x[(k+1)T_{int}] = \mathrm{A}_d x[kT_{int}] + B_d u[kT_{int}] \quad \textbf{(4.51)}$$

In this way, equations (4.48) and (4.49) have been proved to be applicable at every time step k, so they can be evaluated at time instant $t = (k+1)\, T_{int}$ as follows:

$$A_d = \Phi[(k+1)T_{int} - kT_{int}] = \Phi(T_{int}) \quad \textbf{(4.52)}$$

$$B_d = \int_{kT_{int}}^{(k+1)T_{int}} \Phi[(k+1)T_{int} - \tau]\, d\tau B \quad \textbf{(4.53)}$$

Finally, the general expressions for the discretized matrices A_d and B_d are found:

$$A_d = e^{A \cdot T_{int}}, \qquad A \overset{\text{def}}{=} state\ matrix \quad \textbf{(4.54)}$$

$$B_d = B \int_0^{T_{int}} e^{A \cdot T_{int}} d\tau, \qquad B \overset{\text{def}}{=} input\ matrix \quad \textbf{(4.55)}$$

The obtention of the remaining discretized matrices C_d and D_d is made by simply evaluating the second coupled difference equation from (4.43) at instant $t = kT_{int}$:

$$y[kT_{int}] = C_d x[kT_{int}] + D_d u[kT_{int}] \quad \textbf{(4.56)}$$

Then, comparing the previous expression with equation (4.43) and recalling that our system is time-invariant, yields to prove:

$$C_d = C = (B_{1p} \quad B_{2p}), \qquad D_d = D = B_{3p} \tag{4.57}$$

Up to this point, we can move on to compute the particular A_d and B_d matrices using both equations (4.54) and (4.55), as well as the state-space matrices $C_d$and D_d that were given in (4.57). First, to find a solution for A_d requires considering the Taylor series expansion of the exponential function applied to the case where its argument is a matrix (M). This expansion is given by:

$$\mathrm{e}^M = I + M + \frac{M^2}{2!} + \frac{M^3}{3!} + \frac{M^4}{4!} + \cdots, \qquad where\ I \overset{\text{def}}{=} Identity \tag{4.58}$$

Recovering the expression for A_d given in (4.54), we can particularize the previous Taylor expansion yielding to the following:

Table 4.4. Taylor Series Expansion Particularization

$M = A \cdot T_{int}$	$M^n = A^n \cdot (T_{int})^n$	$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$

Then, it can be proved that the Taylor Expansion for our specific case is formed by only its two first terms, as demonstrates the following computation:

$$A^2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = 0 \quad \rightarrow A^n = 0, \qquad for\ n \geq 2 \tag{4.59}$$

In this way, we can calculate an expression for A_d as follows:

$$A_d = e^{A \cdot T_{int}} = I + A \cdot T_{int} = \begin{pmatrix} 1 & T_{int} \\ 0 & 1 \end{pmatrix} \tag{4.60}$$

Finally, the expression of matrix B_d is given by:

$$B_d = \int_0^{T_{int}} (I + At) B dt = \int_0^{T_{int}} \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} dt = \begin{pmatrix} 0.5 T_{int}^2 \\ T_{int} \end{pmatrix} \tag{4.61}$$

Once the discretized matrices for the state-space model of the loop filter have been found, it is necessary to define the initial conditions, so the filter is initialized properly, and its output is bounded. Then, the following assumptions are used:

- The initial signal phase error given by the discriminator ($\Delta\phi_s[k_0]$) is chosen to be the reference of our system, so it is set to zero. Then, $u[k_0] = 0$.
- The initial output from the filter is set to be the initial Doppler shift estimate that is computed when the receiver is initialized. Then, $y[k_0] = \hat{f}_{D_{initial}}$

Moreover, we defined our system to be in steady state because the variables that define the its behavior (the state-space matrices) do not change with time. Then, the coupled differential equations that define the state-space model can be written as:

$$\begin{gathered} x[k_0+1] = A_d x[k_0] = x[k_0] \\ y[k_0] = C_d x[k] \end{gathered} \tag{4.62}$$

The previous equations ensure that the system is initialized to be in steady state if and only if both equations are fulfilled simultaneously. Then, it is necessary to express the first equation in matricidal form so both conditions can be implemented on the receiver:

$$(A_d - I)x[k_0] = 0 \tag{4.63}$$

Then, the coupled equations from (4.62) can be written as:

$$\begin{pmatrix} 0 \\ y[k_0] \end{pmatrix} = \begin{pmatrix} 0 \\ \hat{f}_{D_{initial}} \end{pmatrix} = \begin{pmatrix} (A_d - I \\ C_d \end{pmatrix} \cdot x[k_0] \tag{4.64}$$

In this way, the previous expression can be solved for $x[k_0]$ so the state vector of the model is initialized correctly, and the performance of the filter is bounded:

$$x[k_0] = \begin{pmatrix} (A_d - I \\ C_d \end{pmatrix}^{-1} \cdot \begin{pmatrix} 0 \\ \hat{f}_{D_{initial}} \end{pmatrix} \tag{4.65}$$

4.3.4. Adaptive Filter Loop Bandwidth

An important feature to include in our receiver is the ability of changing the bandwidth of the loop filter as a function of the phase error detected by the discriminator of the PLL. In this way, the bandwidth can be narrower when the phase error is low and so the dynamics of the environment are affecting the receiver smoothly, and it can be wider when the dynamics increase and thus the phase error is higher.

First, let define the expression of the loop filter transfer function in terms of the noise bandwidth as:

$$F(s) = \frac{Y(s)}{U(s)} = \frac{B_L}{41} \cdot \frac{120s^2 + 56B_L \cdot s + 5B_L^2}{s^2} \tag{4.66}$$

Then, the goal of this section is to describe the technique that is used to adapt the parameter B_L as a function of the dynamics of the phase error ($\Delta\phi_s$) so that later on, the state-space matrices that define the model of the loop filter and which depend on B_L are recomputed. To do so, the first step is to find the frequency response of the loop filter by imposing that $s = j\omega$, so that an expression for $F(s)$ is given in the frequency domain:

$$F(j\omega) = \frac{B_L}{41} \cdot \frac{5B_L^2 + 56B_L j\omega - 120\omega^2}{-\omega^2} \tag{4.67}$$

Thereby, the following figure shows the frequency evolution of the absolute value of $F(j\omega)$:

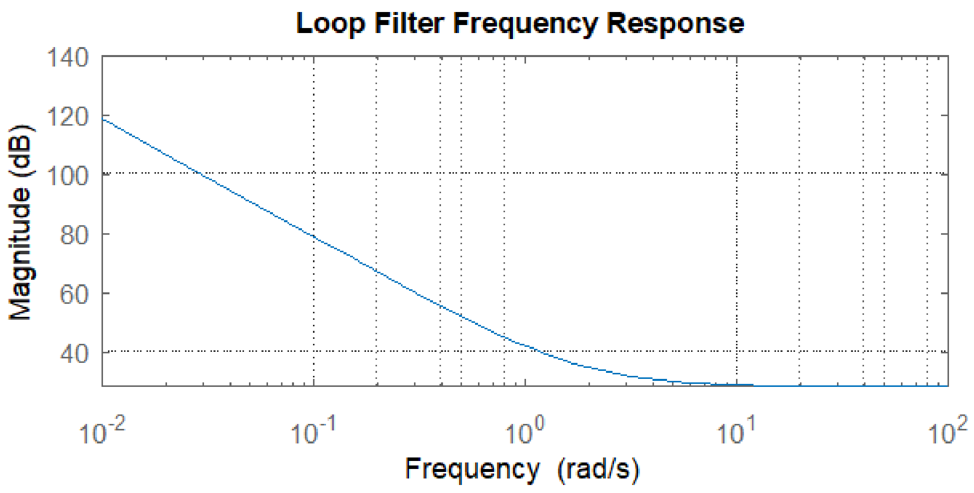


Fig. 4.6. Loop Filter Frequency Response

From the previous figure, it can be inferred that $|F(j\omega)|$ is mainly influenced by the effect of $-\omega^2$ for high frequencies, so the response of the filter appears to be very abrupt decreasing rapidly at a rate of $-40dB/dec$. This is interesting for our purposes since we are ensuring that no effect produced for high frequencies gets filtered, so the filter can reject those samples for which the phase error is not bounded on the interval $[-\pi,\ \pi]$. Moreover, it might be noticed that for low frequencies (close to the values that the phase error is expected to take) the frequency response of the filter can be approximated as:

$$F(j\omega) \cong \frac{5}{41} \cdot \frac{B_L^3}{-\omega^2} \tag{4.68}$$

The previous expression proves that the filter gain at the region close to the origin can be mainly controlled by the parameter B_L. Then, it might be interesting to analyze the gain of $F(j\omega)$ for different values of B_L, which is defined by:

$$G[dB] = 20\log_{10}(|F(j\omega)|) = 20\log_{10}\left(\left|\frac{5}{41} \cdot \frac{B_L^3}{-\omega^2}\right|\right) \tag{4.69}$$

After several iterations, it was observed that the dynamics of the simulated environment can be controlled by setting B_L on the range $[8, 16]$. This interval has been chosen trying to ensure a tradeoff between the obtained phase error and the convergence velocity to which it is achieved. On one side, for $B_L > 16$, the convergence velocity is higher, but the phase error is too large and so there is no point on increasing the noise bandwidth. On the other side, for $B_L < 8$, the phase error is achieved to be lower, but since the convergence speed is too slow, it is preferable to obtain a faster convergence although admitting larger phase errors.

In this way, we can obtain the gain of the loop filter at the origin region ($\omega = 10^{-3} rad/s$) for the extreme values of B_L that we determined. For the lower bound ($B_L = 8$), the following expression is used:

$$G_{BL=8} = 20\log_{10}\left(\left|\frac{5}{41} \cdot \frac{9^3}{-(10^{-3})^2}\right|\right) = 158.98dB \tag{4.70}$$

On the other hand, the gain at the lower bound ($B_L = 16$) is given by:

$$G_{BL=16} = 20\log_{10}\left(\left|\frac{5}{41} \cdot \frac{16^3}{-(10^{-3})^2}\right|\right) = 173.97dB \tag{4.71}$$

The following figures have been obtained using the expression (4.67) for $F(j\omega)$ and evaluating the noise bandwidth at $B_L = 8$ and $B_L = 16$. Then, it is demonstrated that the approximation for $F(j\omega)$ at the origin region which is given in (4.68) is correct because the gain values computed for both upper and lower bounds of B_L match with the values shown in the plots.

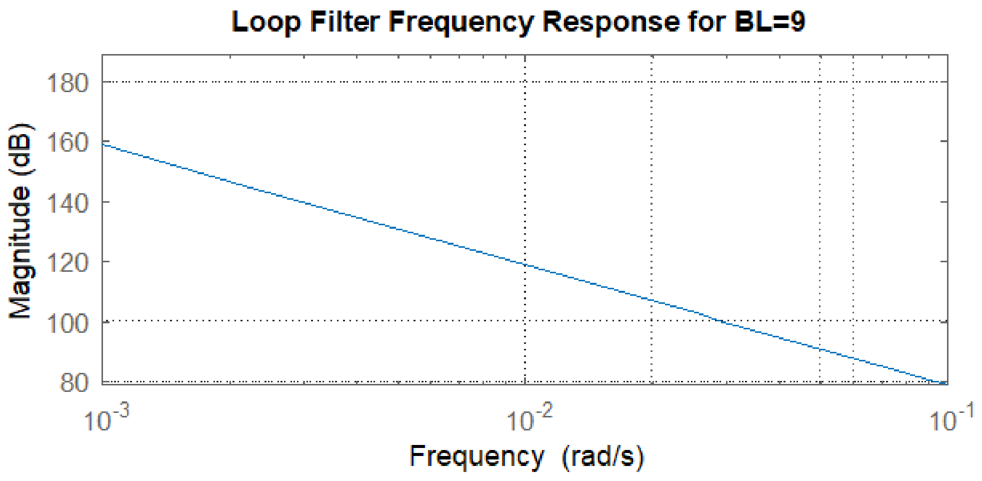


Fig. 4.7. Loop Filter Frequency Response for $B_L = 8$

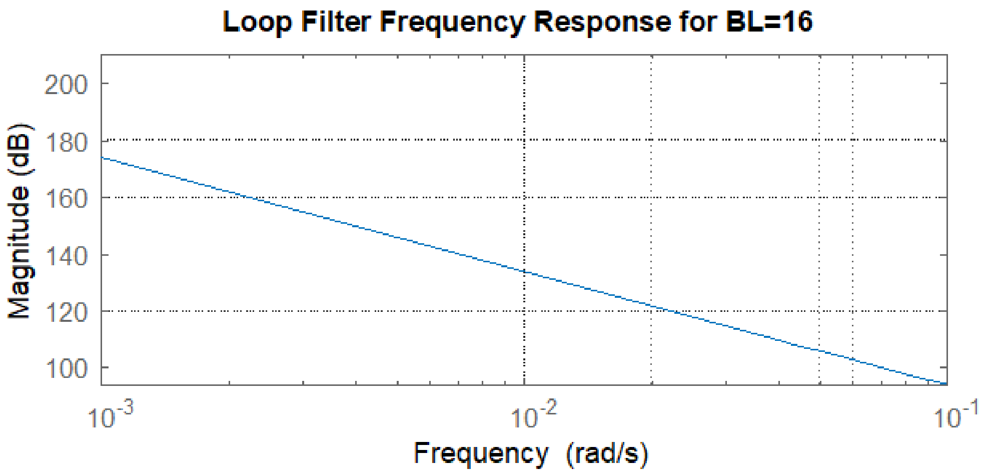


Fig. 4.8. Loop Filter Frequency Response for $B_L = 16$

Furthermore, it can be concluded that an effect of doubling B_L is translated on increasing the gain of the loop filter gain by roughly $15dB$. Then, since the gain increases so rapidly with the noise bandwidth, it is necessary to define a strategy to adapt this parameter in a slightly way so that the filter can react adaptively and for high and low dynamics environments without losing the sharpness of its frequency response.

The design strategy used to implement the adaptive bandwidth into the receiver has been made defining the following considerations:

- The adaptation of B_L is an iterative process that is produced as the received signal is acquired and it is made according to the variation of the phase error ($\Delta\phi_s$), which is the control parameter of the algorithm that is observed at each iteration to define the change amount of B_L
- An initial estimate of B_L is necessary to initialize the algorithm. This value has been set to be close to the upper bound so it can be decreasing as the phase error is reduced. Then, it is set that $B_L(k_0) = 14$
- The algorithm ensures that at each step, the value of B_L is set as low as possible to get the least noisy possible phase error but, at the same time, it is set to be high enough to ensure a fast convergence
- The phase error ($\Delta\phi_s$) is assumed to be uniformly distributed and its variance is bounded within the interval $[-\pi,\ \pi]$

Moreover, it is critical to determine the time instants when the phase error variance is computed and thus when B_L is changed adaptively. It has been observed that it is useful to define a time window as a function of the number of frames, so the algorithm decides how to change B_L after this time is elapsed. This method is also helpful since the phase error can be compared between the samples included into different time windows. In this way, at the step k, B_L is modified as a function of the phase error variance of the previous samples from the step $k-1$.

After an exhaustive assessment, it has been found that the optimum window length (N_{EP}) remains as:

$$N_{EP} = 0.01 N_{frames} \tag{4.72}$$

And the formula used to compute B_L at each step k is given by:

$$B_L[k] = a \cdot B_L[k-1] + b \cdot var(\Delta\phi_s[k-1]), \qquad k \geq 1 \tag{4.73}$$

$$where\ a = 0.9999, \qquad b = 0.0001$$

Also, in order to include the condition of $var(\Delta\phi_s) \stackrel{\text{def}}{=} [-\pi,\ \pi]$, the following conditions apply:

$$B_L[k] = a \cdot B_L[k-1] + b \cdot \pi, \qquad if\ var(\Delta\phi_s) > \pi \tag{4.74}$$

$$B_L[k] = a \cdot B_L[k-1] - b \cdot \pi, \qquad if\ var(\Delta\phi_s) < -\pi \tag{4.75}$$

From the previous expressions, a and b parameters are weighting the effect of increasing and decreasing B_L, and they are set to such values in order to ensure that at each step k, the change in B_L is very slight. In particular, from step $k-1$ to step k, the variation on $B_L[k]$ with respect to $B_L[k-1]$ can be a maximum of 0.004%. Then, it is achieved that the gain of the loop filter is controlled, and it does not tend to either 0 or ∞. Hence, the phase given by the discriminator is well filtered and the Doppler shift computed from this information is correct.

On the other hand, it might be recalled that the length of the sliding window (N_{EP}) is selected to ensure that the variance of the phase error obtained from the discriminator is well bounded, so we do not combine errors from different stages of the received signal. In other words, by keeping the length of this window to be short enough, we are ensuring that we are not comparing large phase errors from the first iterations of the PLL with the minimum errors that we eventually get in the last iterations when almost all the signal has been acquired.

In order to explain the operation of the adaptive bandwidth algorithm, the following code flowchart is shown:

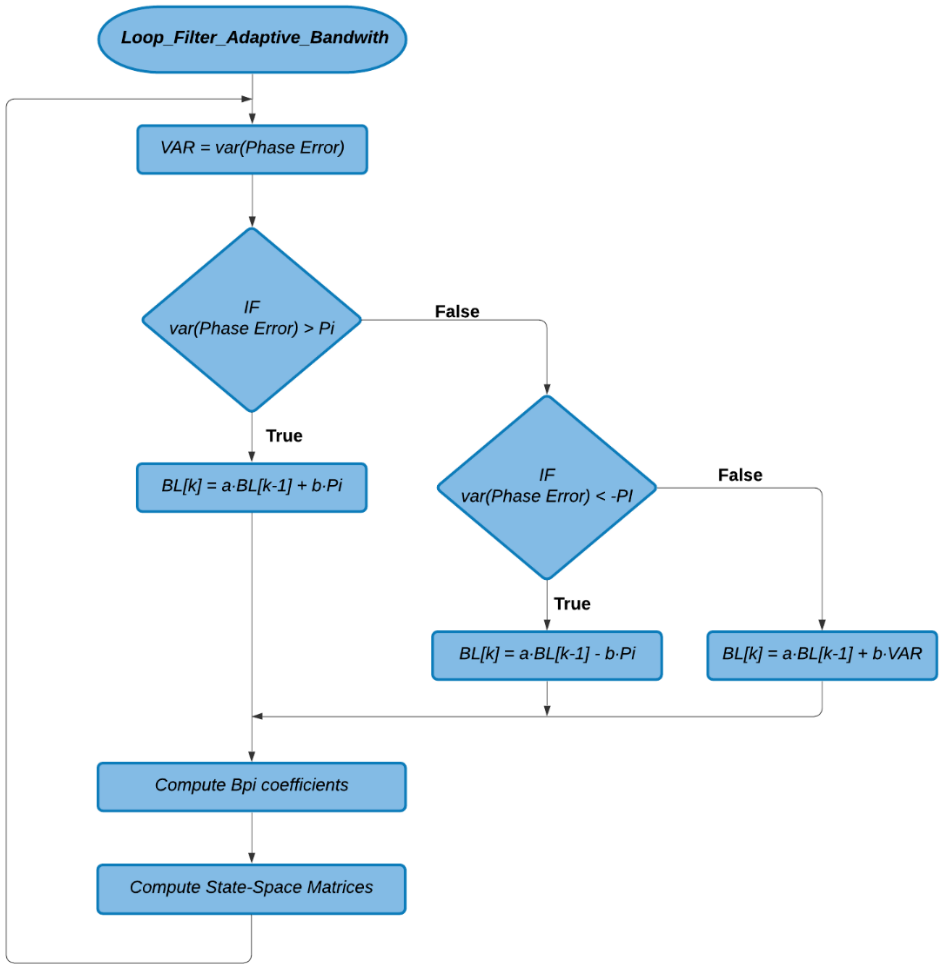


Fig. 4.9. Loop Filter Adaptive Bandwidth Code Flowchart

From the previous figure, the algorithm updates the noise bandwidth (B_L) of the loop filter after computing the error phase variance from the samples contained inside the sliding window (N_{EP}). Then, it takes the resulting value and decides whether is contained within the interval $[-\pi,\ \pi]$ or not. From this information, it computes B_L using a different formula. Finally, it computes the coefficients B_{ip} from which the state-space matrices are dependent and updates the value of these matrices accordingly.

It may be noticed that matrices A_d and B_d do only depend on the integration time (T_{int}) at which the received signal is being accumulated. Then, the algorithm uses this parameter as an input in order to compute these matrices as well.

Finally, the following figures are used to demonstrate the effectiveness of changing adaptively the noise bandwidth of the loop filter. All show the signal error phase evolution of the received signal throughout a simulation time of 100 seconds. This is, the phase error given by the discriminator stage ($\Delta\phi_s$) at each iteration of the PLL.

The first couple of figures (Fig. 4.10 and Fig. 4.11) were obtained by setting $B_L = 14$ for the entire simulation in the first one and, for the second one, by setting $B_L[k_0] = 14$ and then using the previously described algorithm to change it adaptively.

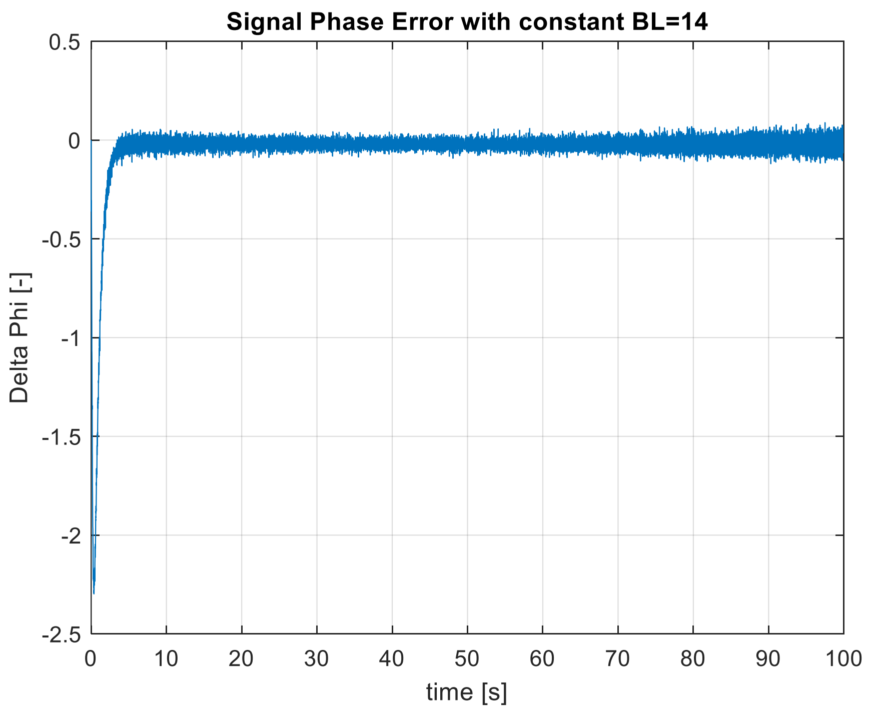


Fig. 4.10. Signal Phase Error Time Evolution with constant $B_L = 14$

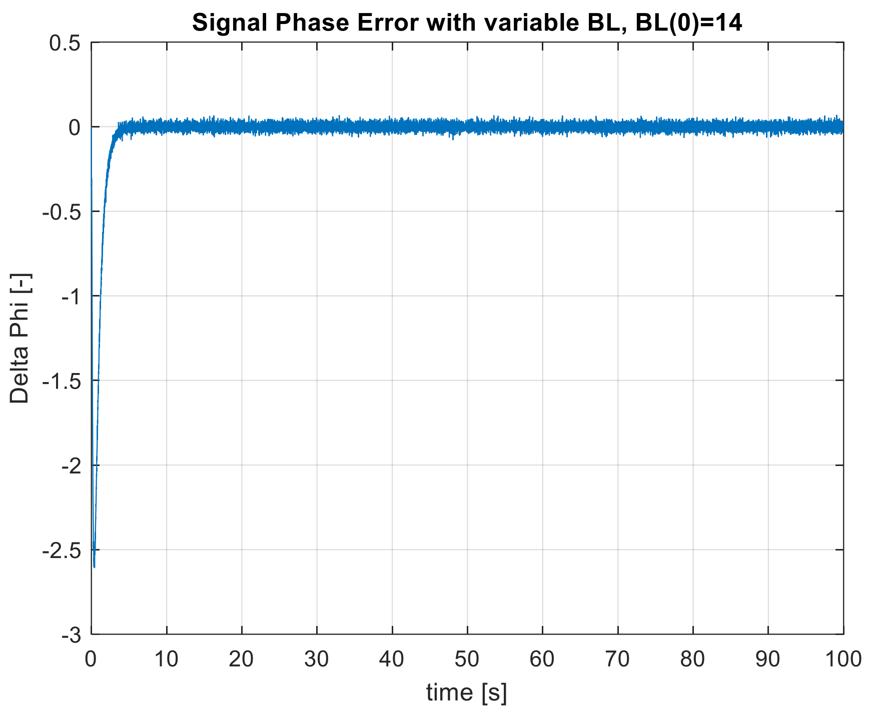


Fig. 4.11. Signal Phase Error Time Evolution with variable B_L $(B_L[k_0] = 14)$

From the previous figures, both phase error evolutions are close to 0, which indicates that the receiver is tracking the signal properly in both scenarios. This can be explained since the value of $B_L = 14$ was selected within the optimum interval of B_L which is $[8,\ 16]$. However, comparing them, it can be concluded that the signal phase error achieved when using an adaptive bandwidth is much less noisy.

Although Fig. 4.10 shows a decent phase error which could be much more than acceptable for standard communications systems, it is too much noisy for our purpose of obtaining a navigation solution from the tracked Doppler shift. Then, it is much more convenient to implement a technique as the one described over this section in order to achieve a phase error as closest as possible to 0.

On the other hand, it is also interesting to prove the effectiveness of the algorithm in a different scenario where the environment is much more demanding. In this case, we have selected a value for $B_L = 6$ which is under the interval $[8,\ 16]$ trying to emulate a scenario with high dynamics where the receiver is stressed.

On the next page, Fig. 4.12 was obtained by setting $B_L = 6$ for the entire simulation, while Fig. 4.13 was obtained by setting the initial B_L to this value and then changing it adaptively. It can be appreciated that our designed algorithm makes a huge difference since the second figure shows a signal phase error evolution that is able to converge to zero, whilst the first one shows a noisy error phase that, even though is bounded within $[-\pi,\ \pi]$, can't converge to the null value.

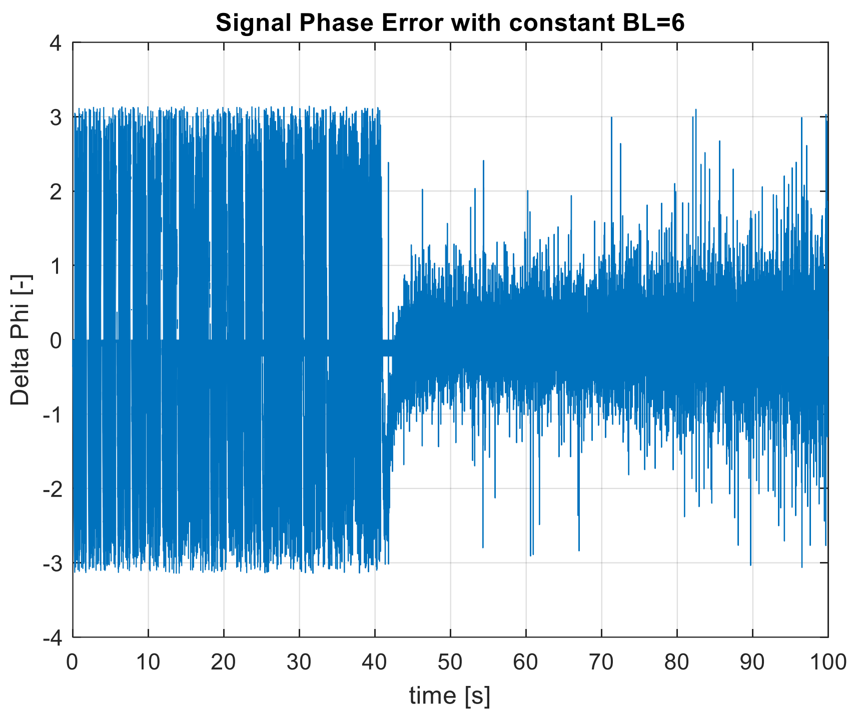


Fig. 4.12. Signal Phase Error Time Evolution with constant $B_L = 6$

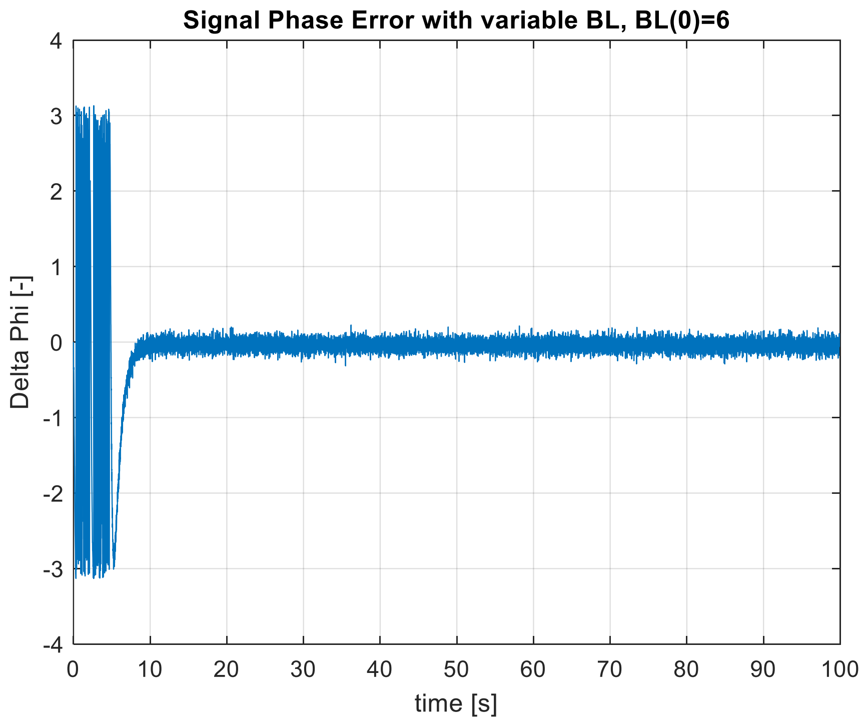


Fig. 4.13. Signal Phase Error Time Evolution with variable B_L $(B_L[k_0] = 6)$

4.4. Signal Timing Shift Tracking

4.4.1. Received Signal Power Threshold Approach

According to the signal definition given in chapter 3, our target signals are pure tones transmitted with a duty cycle, which means that the tone duration is shorter than the frame duration. In this way, the start of the frame might not necessarily match with the tone start since the received tones might be shifted in time and thus be received somewhere in between the current frame.

This could result into a problem when integrating the frames at each step of the PLL since the I&D filter must ensure that only the samples that contain information of the tone must be accumulated. Then, in order to guarantee that the total phase error ($\Delta\phi_s$) of the frame is integrated, a mechanism to detect the start of the tones that defines the start position of the integration window (whose size was demonstrated to equal the duration of a tone) needs to be implemented.

The first approach that has been used to detect the tone start consists on studying the transmitted signal and to define a threshold value for the expected received signal power (P_s) so it can be distinguished from noise. Hence, the receiver knows that the tone has started when the received signal power overcomes this threshold value.

This method can be implemented since the receiver processes the incoming signal frame by frame and only one tone is contained inside one frame. Moreover, it might be noticed that the effectiveness of this approach can be described in terms of the SNR of the current frame. Then, considering the expected noise variance of the channel (${\sigma_n}^2$) and the number of samples contained into one frame (N), an expression of the SNR for the current frame is given by:

$$SNR = \frac{P_s}{P_n} = \frac{{A_s}^2}{P_n} = \frac{{A_s}^2}{{\sigma_n}^2/N} \tag{4.76}$$

It may be inferred that for higher values of the SNR (which equals to say that the noise floor is low in comparison to the signal amplitude) the method will be effective because the receiver can distinguish clearly between noise and signal information. However, in a scenario where the tones are received with lower power and thus the noise floor is closer to the incoming signal, this method is not robust enough to track the start of the tones because there is not a clear frontier between noise and data.

Nevertheless, it is useful to study the viability of this approach since it is easy to implement and its computational cost is really low. Then, let first define the received signal amplitude using the model described in chapter 3 as a function of a parameter G that models the reception gain:

$$A_s(i) = \sqrt{G} \cdot \left(1 + \frac{1}{10}\left(\frac{d_0}{d(i)}\right)^2\right), \qquad i = 0,1,2,\ldots \tag{4.77}$$

Besides, considering that Iridium Next satellites orbits are located $780km$ above the Earth surface and defining $d_0 = 2250\ km$ as the maximum possible distance between the satellites and a receiver located in Irvine, California, a range of values for the received signal power (P_s) in terms of G can be obtained:

$$P_s[dBm] \in [30.83 + G(dB)\,, 35.26 + G(dB)] \tag{4.78}$$

Moreover, recalling that the channel noise for satellite communications can be modeled according to [10] to have a variance of $\sigma_n{}^2 = (400 \cdot 0.4)^2 = 25600$, that the sampling frequency of the receiver is set to $F_s = 240kHz$, and that the number of samples contained into one frame whose duration is $6.8ms$ is $N = 1632$, the expected noise power is computed as:

$$P_n[dBm] = 30dB + 10\log_{10}\left(\frac{\sigma_N{}^2}{N}\right) = 41.96dBm \tag{4.79}$$

Then, it can be concluded that the SNR of the received frame is mainly influenced by the reception gain (G), which means that the viability of this approach depends on the performance of the receiver and the characteristics of the channel.

In order to assess whether defining a threshold value for P_s that is suitable to disregard the noisy samples is effective or not, it is necessary to consider different scenarios. Then, the parameter G will be changed in order to simulate either a lower performance of the receiver or a noisier channel, and thus a threshold value for the received signal power will be chosen according to the SNR for each specific scenario. Finally, the time evolution of the phase error ($\Delta\phi_s$) given by the receiver will be studied to determine the effectiveness of this method. The following table summarizes the chosen parameters for each scenario:

Table 4.5. Set up Parameters of the Two Considered Scenarios

$G[dB]$	$P_s[dBm]$	$SNR[dB]$	$P_{s_{threshold}}[dBm]$
8	$[38.83, 43.26]$	$[-3.13, 1.30]$	42.5
14	$[44.83, 49.26]$	$[2.87, 7.13]$	44

From the previous table, the selected values of G have been chosen according to [14], which defines typical values for reception gains in satellite-ground communications. Also, they have been chosen to adopt this values so we can simulate a scenario where the SNR of the current frame is high and thus the threshold definition can be applied and, on the other hand, a scenario where the receiver performance is low and thus defining a threshold value implies losing some samples that contain useful data.

Furthermore, it might be noticed that the definition of the power threshold has been done trying to avoid the noise floor. Hence, it has been set to be slightly above the expected channel noise power. Finally, the following figures show the phase error ($\Delta\phi_s$) evolution (for a simulation time of $t = 100s$) given by the receiver for both considered scenarios:

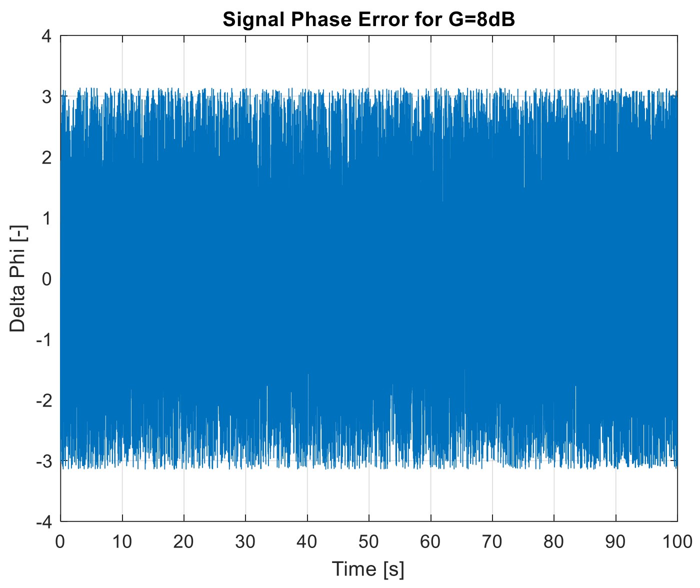


Fig. 4.14. Signal Phase Error time evolution for $G = 8dB$

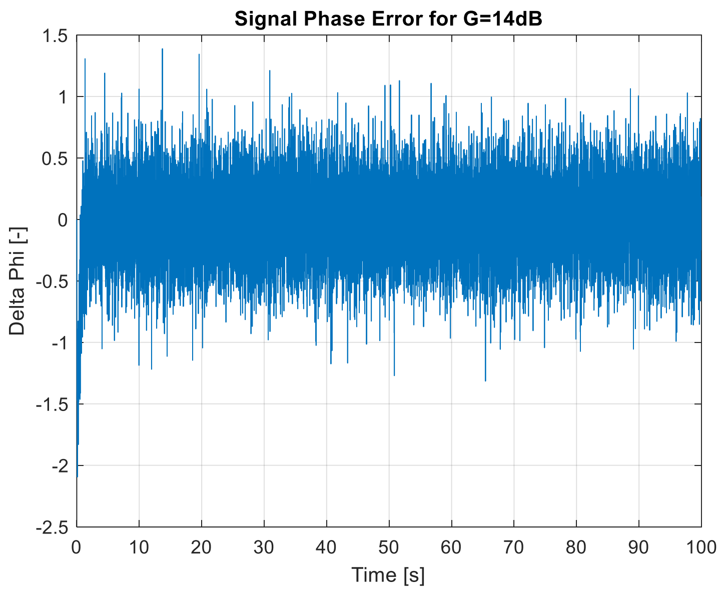


Fig. 4.15. Signal Phase Error time evolution for $G = 14dB$

From the figures on the previous page, we can conclude that the receiver is not able to track the incoming signal. For the first considered scenario, the SNR of the frames is too low and thus defining a threshold that determines the start of the tones makes no point, because noise cannot be clearly distinguished from data. Moreover, the phase error that it produces does not converge to any value and remains bounded inside the interval $[-\pi, \pi]$ just because the phase discriminator that was defined in sub-section 4.2.4 produces an output within this range.

On the other hand, for the second scenario where the SNR of the frames is increased, the receiver appears to be able to track the incoming signal since the phase error evolution shows a convergence tendency around 0. However, since the SNR appears to be still low, the receiver is still not able to decide clearly when the tone starts, which is reflected into an excessively noisy phase error.

In both cases, the Doppler shift estimate produced from the phase error given by the receiver would be too noisy and highly inaccurate. Then, an alternative approach to detect properly the start of the tones even in low SNR scenarios must be found.

4.4.2. Maximum Integrated Power Approach

The method described throughout this sub-section is a more conventional approach that allows the receiver to track time-delayed signals even when they are received with low SNR. More specifically, this approach consists of detecting the tone start on each received frame by finding the optimum position for a sliding window that gives the maximum integrated power over each frame. In this way, this position defines the start of the tone and the I&D filter will only integrate the samples contained inside this window ensuring that noisy samples are disregarded.

As it was discussed in sub-section 4.2.3, the optimum size (N_{opt}) of the sliding window that is used to maximize the SNR of the received frame corresponds to the duration of a tone ($2.6ms$). Then, considering the sampling frequency of the receiver to be $F_s = 240kHz$, it leads to define $N_{opt} = 624$.

Furthermore, the power contained inside the sliding window (P_w) can be computed by adding the power in all the window samples (s_i) as:

$$P_w = \frac{1}{N_{opt}} \cdot \sum_{k=1}^{N_{opt}} |s_i(k)|^2 \tag{4.80}$$

It might be noticed that the previous computation needs to be done iteratively for the different positions of the current frame, so the receiver can compare all the power values and decide which is the tone start position accordingly. Then, considering that each frame contains 1632 samples, the receiver will make 1009 power computations $(1632 - 624 + 1)$ for each frame. Hence, although this method has proved to be highly effective, its computational cost is higher, and it would be probably unfeasible to implement into the receiver if this power computations are made for each received frame.

For all the above, it will be necessary to define a strategy that allows to detect correctly the tone start for each frame without the need of the making power computations in all them.

First, let recall that the main source of the timing shift suffered by satellite signals is wave propagation, which depend mainly on the distance between the satellites and the receiver. Then, the expression of the timing shift (τ) for each received frame that was defined in chapter 3 can be approximated as:

$$\tau \sim \frac{d(t)}{c} \rightarrow \tau_d(i) = \frac{1}{F_s}\tau = \frac{d(i)}{c} \qquad \textbf{(4.81)}$$

Besides, the rate of change of τ between consecutive frames can be written in terms of the satellite velocity (v_{sat}) as:

$$\frac{d(\tau)}{dt} = \frac{1}{c}\cdot\frac{d}{dt}(d(t)) = \frac{1}{c}v_{sat} \qquad \textbf{(4.82)}$$

Then, knowing that the approximate orbital velocity of Iridium Next satellites is $27000km/h = 7500m/s$ and knowing that $c = 3\cdot 10^8 m/s$, it can be defined that the rate of change of τ is $2.5\cdot 10^{-5}$, which gives us information about how the start of the tone is shifted between consecutive frames. Moreover, since the sampling frequency is set to be $F_s = 240kHz$, and denoting k_0 the sample where the tone starts at the current frame and k_1 the starting sample of the tone acquired on the next frame, the difference between both tone starts in terms of samples ends to be:

$$\Delta k = k_1 - k_0 = F_s \cdot \Delta\tau_d = 6 \qquad \textbf{(4.83)}$$

It might be as well interesting to analyze which is the impact of assuming that the tone start is constant between frames in order to decide which is the maximum acceptable error that the receiver can commit. Defining the duration of a frame to be 1632 samples, this error can be expressed as:

$$\mathcal{E}(\%) = \frac{\Delta k}{1632}\cdot 100 = 0.37\% \qquad \textbf{(4.84)}$$

Then, defining a maximum acceptable error to be of the 10%, the receiver will assume the tone start to be constant for 27 consecutive frames and will reduce the necessary number of power computations by a factor 27, aiming that it remains close enough to the actual tone start in order to track the incoming signal accurately.

Finally, it might be recalled that this method is effective only for those scenarios where the timing shift (τ) of the received frames can be approximated as shown in (4.81). However, more realistic scenarios where the dynamics of the environment are high and thus the timing shift of the frames might be caused by multiple sources, may need an improvement of this approach that accounts for these considerations.

4.4.3. Tracking Improvements for High Dynamics Environments

In the previous sub-section, it was described that the receiver will find the actual tone start every 27 consecutive frames by finding the optimum position of a sliding window that gives the maximum integrated power. Nevertheless, it is important to recall that this period of update might be insufficient given the case that the dynamics of the environment are high.

Furthermore, considering that the signal is acquired during 10 minutes, the total number of received frames (recalling that the duration of a frame is $6.8 \cdot 10^{-3} s$) ends to be 88235. Then, the actual tone start will be found for only 3267 frames, which equals to say that the receiver will be using a shifted version of the actual tone start for almost the 96.3% of the total frames.

Hence, it becomes necessary to update the tone start of each frame as a function of the Doppler shift (f_D), with respect to the carrier frequency (f_c), that was acquired for the previous frame. In this way, the receiver is also accounting for the timing shift produced by the Doppler effect and will be able to correct the displacement of the tone start from frame to frame.

Since the PLL acquires the signal frame by frame, the following expression is used to update the tone start (k_{st}) at each iteration:

$$k_{start}(i+1) = k_{start}(i) - \frac{f_D(i)}{f_c} T_{int} \cdot F_s, \qquad i = 0,1,2,\ldots \tag{4.85}$$

Finally, in order to demonstrate the effectiveness of our proposed method and to justify its implementation, it is useful to compare it with the approach described in sub-section 4.4.2. Fig. 4.14 showed that the receiver was not able to track the incoming signal properly when the SNR was low and thus data cannot be distinguished from the noise floor. Then, the same simulated noisy signal has been delivered to our improved receiver in order to compare the phase error ($\Delta\phi_s$) that it produces over time.

The following figure shows the first 10 frames ($T_{frame} = 6.8 \cdot 10^{-3} s$) of the simulated noisy signal that was used to obtain both figures 4.14 and 4.15.

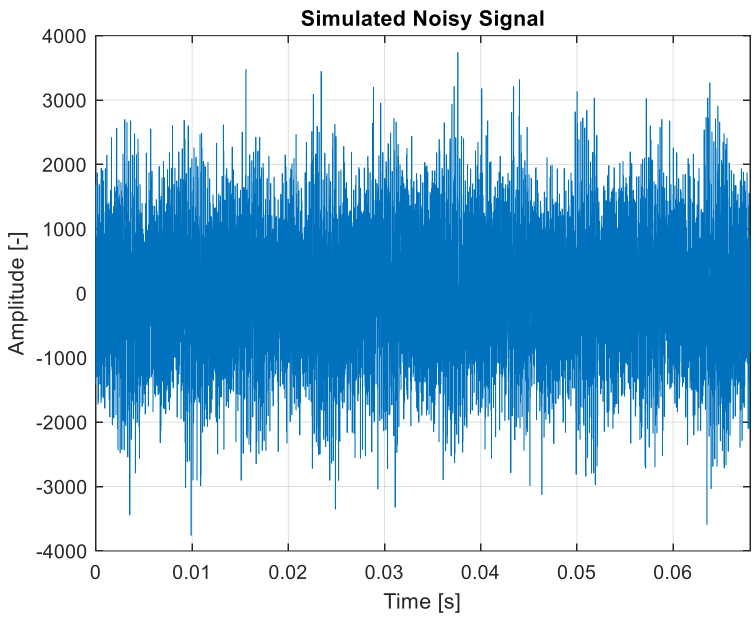


Fig. 4.16. First 10 Frames of the Simulated Noisy Signal

On the other hand, the following figure shows the phase error ($\Delta\phi_s$) produced by the receiver when it is using our improved method to track the tones start:

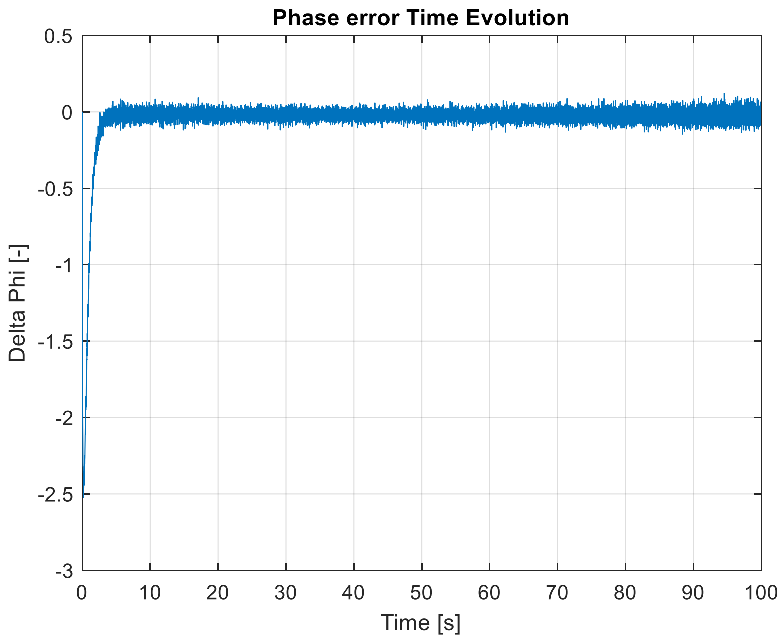


Fig. 4.17. Obtained Signal Phase Error time evolution with tracking improvements

Finally, it can be concluded that our proposed method can be used for high dynamics environments since it allows the receiver to detect correctly the tone start of each frame. This is reflected on Fig. 4.17, which shows that the phase error ($\Delta\phi_s$) converges to the null value with slight variations, which indicates that the receiver can track the Doppler shift from the incoming signal and use these measurements to obtain a navigation solution.

4.5. Matlab Implementation

4.5.1. Troubleshooting of Implementation Issues

This sub-section intends to describe how some of the issues concerning the receiver's implementation in Matlab were managed to be solved. Moreover, an overview of how the different stages of the receiver were implemented is given.

First, the complete received signal is stored by means of its IQ samples into a binary file. Also, the number of samples contained in one frame is defined as the product of the sampling frequency and the duration of a frame. In this way, the binary file is read frame by frame using the Matlab function $fread(\dots)$.

Second, the integration period (T_{int}) is defined to be equal to the duration of a single frame in order to ensure that all the phase information from the current frame is integrated at each step of the PLL at the I&D Filter stage.

Third, the initial Doppler frequency shift estimate ($\hat{f}_D$) is obtained by first performing a PSD computation on the first frame using the Matlab function $pwelch(\dots)$ and then finding the frequency position of the maximum peak of the PSD by means of the Matlab function $max(\dots)$.

Fifth, I&D Filter is implemented with the Matlab function $mean(x)$ which allows to accumulate all the values contained in the vector x. Besides, the phase discriminator is implemented using the Matlab function $atan2(y, x)$ which returns the fourth-quadrant inverse tangent of y, x ensuring that no phase ambiguities are produced.

Finally, the receiver decides how to update the loop filter bandwidth as well as the current frame timing shift comparing the current frame number with some predefined values using the Matlab function $mod(\dots)$.

4.5.2. Code Flowchart

The figure on the following page shows the code flowchart used to implement the receiver design in Matlab:

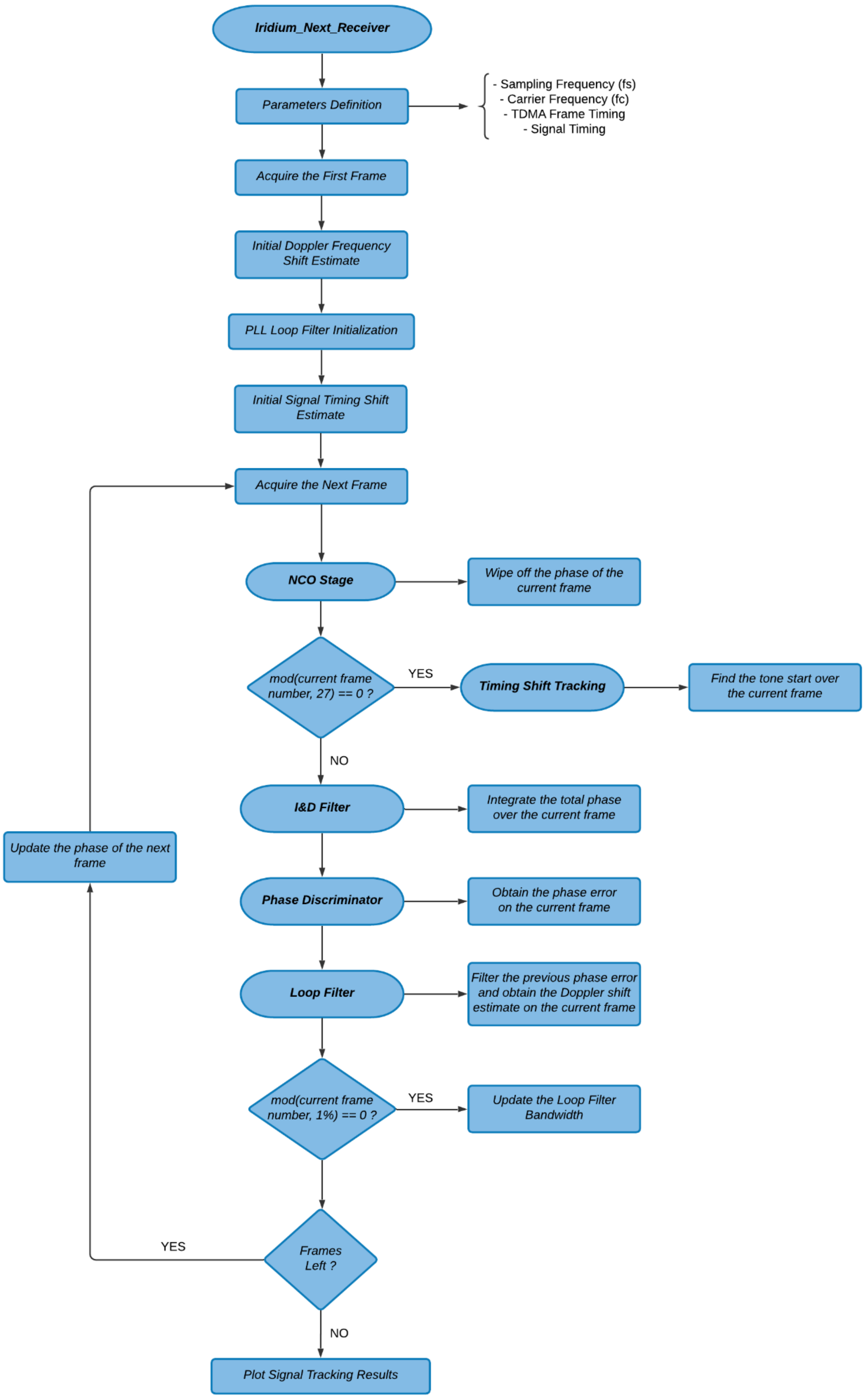

Iridium_Next_Receiver
Parameters Definition
- Sampling Frequency (fs)
- Carrier Frequency (fc)
- TDMA Frame Timing
- Signal Timing
Acquire the First Frame
Initial Doppler Frequency Shift Estimate
PLL Loop Filter Initialization
Initial Signal Timing Shift Estimate
Acquire the Next Frame
NCO Stage
Wipe off the phase of the current frame
mod(current frame number, 27) == 0 ?
YES
Timing Shift Tracking
Find the tone start over the current frame
NO
I&D Filter
Integrate the total phase over the current frame
Update the phase of the next frame
Phase Discriminator
Obtain the phase error on the current frame
Loop Filter
Filter the previous phase error and obtain the Doppler shift estimate on the current frame
mod(current frame number, 1%) == 0 ?
YES
Update the Loop Filter Bandwidth
YES
Frames Left ?
NO
Plot Signal Tracking Results

Fig. 4.18. Iridium Next Receiver Code Flowchart

CHAPTER 5. SATELLITE ORBITS PREDICTION

5.1. Iridium Next Constellation

Before moving on working with real satellite signals, it is necessary to study the movement of the satellites since it will allow us to determine which are the best windows of time for recording Iridium Next signals from a given location. In this way, this chapter goes into the detail of describing the necessary elements and calculations to predict the satellites orbits and to obtain a graphic representation of them.

The Iridium Next constellation consists of 75 active satellites that orbit the Earth in 6 different orbital planes spaced 30º apart [9]. Originally, the Iridium constellation was designed to incorporate 66 satellites (gathered in 6 groups of 11) in order to provide coverage for the entire Earth surface. Later, the company decided to enlarge the initial constellation (Next campaign) by launching 12 extra satellites in order to provide 24/7 real-time coverage, which would add two extra satellites on each of the original orbital planes. Unfortunately, 3 of them are not active since they experienced technical difficulties once they were launched and thus the current constellation remains with 75 satellites.

Iridium Next satellites use LEO (Low Earth Orbits) near-polar orbits that allow to cover the entire surface of the Earth since each satellite passes over each point of the Planet when it rotates on its axis. In this way, each satellite will pass over the equator at a different longitude on each of its orbits. Furthermore, satellites are chosen to adopt a Sun-synchronous orbit, which means that each successive orbital pass is produced at the same local time of the day. Hence, the coverage remains constant for each point over the Earth.

In order to retain the orbital synchronicity as the Earth revolves around the Sun throughout the year, the orbits of the satellites must be inclined so they can precess at the same rate. Then, the orbital period must be adjusted accordingly to produce the desired precession. Finally, the following table summarizes the main orbital parameters of Iridium Next satellites:

Table 5.1. Iridium Next Orbits Main Parameters

Altitude	Inclination	Period
$780\ km$	$86.4º$	$100.4\ mins$

It might be noticed that since the orbital period is approximately 100 minutes, Iridium Next satellites are non-geostationary. In fact, this orbital period is much lower than the Earth rotation period (approximately 24 hours) and, as a result, each satellite will complete 14 laps around the Earth every day. Hence, each satellite will cross the equator at 28 different longitudes every day although they are kept in the same orbital plane.

The following figure taken from [9] shows a polar representation of the satellite's orbital planes distribution. In there, satellites are represented by an empty circle and they are equally spaced into the 6 orbital planes. Furthermore, each satellite of the constellation is equally spaced to its direct neighbor (either from the same orbital plane or from the adjacent one) by forming an equilateral triangle:

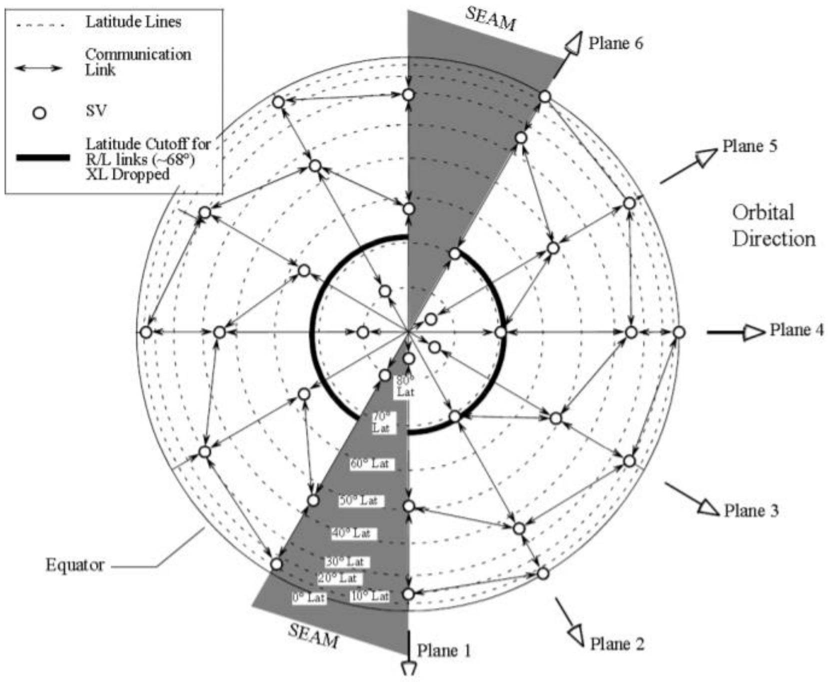


Fig. 5.1. Polar View of the Iridium Next Constellation [9]

Finally, this orbit topology was selected to provide full coverage, ranging from the equator to the poles, so that the service area of each satellite is overlapped with the areas generated by its neighbors. In this way, there are not shadow areas where service is not provided. Moreover, the satellites location is also designed to maximize the intersatellite miss distance at the polar crossing while maximizing the impact on the equatorial coverage.

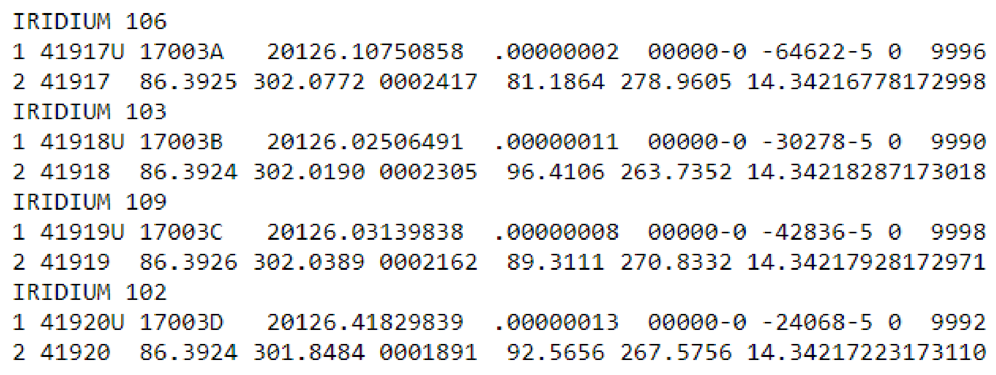
5.2. Orbital Elements

5.2.1. Two-Line Element Set Files Structure

The orbital information that is used to describe the trajectories of Earth-orbiting objects is contained in the TLE (Two-Line Element) files, which are made public and updated periodically according to the information transmitted by the satellites.

In this way, each LEO constellation has a TLE file that contains the orbital parameters of all its satellites for a given point in time (epoch) and which can be used to predict the ground track of the satellites.

The data contained in these files is encoded according to a two-line element set structure, which means that the files contain a list of lines that describe the orbital parameters of the satellites, and each satellite is defined by two lines. However, it is common to include an extra title line that indicates the satellite name, which is compliant to the standard definition given by the NORAD (North American Aerospace Defense Command). The following figure shows a fragment of an Iridium Next TLE file:



```
IRIDIUM 106
1 41917U 17003A   20126.10750858  .00000002  00000-0 -64622-5 0  9996
2 41917  86.3925 302.0772 0002417  81.1864 278.9605 14.34216778172998
IRIDIUM 103
1 41918U 17003B   20126.02506491  .00000011  00000-0 -30278-5 0  9990
2 41918  86.3924 302.0190 0002305  96.4106 263.7352 14.34218287173018
IRIDIUM 109
1 41919U 17003C   20126.03139838  .00000008  00000-0 -42836-5 0  9998
2 41919  86.3926 302.0389 0002162  89.3111 270.8332 14.34217928172971
IRIDIUM 102
1 41920U 17003D   20126.41829839  .00000013  00000-0 -24068-5 0  9992
2 41920  86.3924 301.8484 0001891  92.5656 267.5756 14.34217223173110
```

Fig. 5.2. TLE File Fragment of the Iridium Next Constellation

It can be seen from the previous figure that the data contained in each line fits into 70 columns. In this way, the following tables describe the fields of the TLE files according to the line and column number where they are located:

Table 5.2. Description of TLE File Line 1

Line 1	
Column	Description
01	Line Number of Element Data
03 - 07	Satellite Number
08	Classification (U = Unclassified)
10 - 11	International Designator (Last two digits of launch year)
12 - 14	International Designator (Launch number of the year)
15 - 17	International Designator (Piece of the launch)
19 - 20	Epoch year (Last two digits of year)
21 - 32	Epoch (Day of the year and fractional portion of the day)
34 - 43	First time derivative of the Mean Motion
45 - 52	Second time derivative of the Mean Motion
54 - 61	BSTAR drag term
63	Ephemeris type
65 - 68	Element number
69	Checksum (Modulo 10)

Table 5.3. Description of TLE File Line 2

Line 2	
Column	Description
01	Line number of element data
03 - 07	Satellite number
09 – 16	Inclination [Degrees]
18 - 25	Right Ascension of the Ascending Node [Degrees]
27 - 33	Eccentricity
35 - 42	Argument of Perigee [Degrees]
44 - 51	Mean Anomaly [Degrees]
53 - 63	Mean Motion [Rev/day]
64 - 68	Revolution number at epoch [Rev]
69	Checksum (Modulo 10)

5.2.2. Keplerian Elements

Satellites orbits are characterized by the Keplerian parameters, which are contained directly in the TLE files or can be easily derived from the information given in these files. These parameters are used to obtain the ground track of the satellites and are listed as follows:

- $\boldsymbol{t_0}$**:** Epoch (day of the year and fractional part of the day)
- $\boldsymbol{a}$**:** Major Semi Axis of the satellite orbit [m]
- $\boldsymbol{e}$**:** Eccentricity
- $\boldsymbol{i_0}$**:** Orbit inclination [rad]
- $\boldsymbol{\Omega_0}$**:** Right Ascension of the Ascending Node (RAAN) at t_0 [rad]
- $\boldsymbol{\dot{\Omega}_0}$**:** Rate of change of RAAN [rad/s]
- $\boldsymbol{\omega}$**:** Argument of the perigee at time t_0 [rad]
- $\mathbf{M_0}$**:** Mean Anomaly at time t_0 [rad]
- $\boldsymbol{n}$**:** Mean motion [rad/s]

First, the epoch (t_0) which indicates the time elapsed from the start of the current year, it is expressed in fractional time. This nomenclature consists on numerating the days starting from the January 1st and to identify the current time of the day as a fraction of a complete day (24 hours). As an example, 4:30am of 06/15/2020 equals to say 136.1875 (considering that 2020 is a lap year).

Second, the major semi axis (a) indicates the distance from the center of the elliptical orbit to the either the apogee or the perigee, and the eccentricity (e) shows the amount of deviation from a circular orbit by giving the ratio between the orbit foci divided by the major semi axis. Moreover, the orbit inclination (i_0) indicates the angle between the orbital plane of the satellites and the equator.

Third, the right ascension of the ascending node (Ω_0) is the geographic longitude with respect to the Greenwich meridian of the ascending node of the satellite orbit, which variation over time is given by $\dot{\Omega}_0$.

Fourth, the argument of the perigee (ω) shows the angle between the ascending node of the satellite orbit and the perigee in the direction of the satellite movement.

Fifth, the mean anomaly (M_0) is the angle travelled by the satellite after having passed the perigee. When the satellite is moving away from the perigee towards the apogee, M_0 is positive. On the contrary, when it is moving towards the perigee after having reached the apogee, M_0 is negative. Finally, the mean motion (n) indicates the angular speed required for a satellite to complete one orbit.

It might be noticed that there is a mismatch between the units of the previous parameters and the TLE files. In there, Keplerian elements are not necessarily given in SI units due to historical reasons, which means that they must be converted before being able to be used in order to produce the ground track of the satellites.

Moreover, the TLE files are missing two of the previously defined parameters, which are the Major Semi Axis (a) and the Rate of change of the RAAN ($\dot{\Omega}_0$) but, however, both can be derived from the information contained in the TLE files using to the following equations:

$$a = \frac{{\mu_E}^{1/3}}{n^{2/3}} \tag{5.1}$$

$$\dot{\Omega}_0 = \frac{d\Omega_0}{dt} = \frac{-3}{2} \cdot J_2 \cdot \left(\frac{R_E}{a(1-e^2)}\right)^2 \cdot n \cdot \cos(i_0) \tag{5.2}$$

Where:

- $\mu_E = 3.986004418 \cdot 10^{14}\ m^3 s^{-2}$ (Earth Standard Gravitational Parameter)
- $J_2 = 1.08262668 \cdot 10^{-3}$ (Earth Space Flight Constant)
- a: Major Semi Axis [m]
- e: Eccentricity
- $R_E = 6378 \cdot 10^3\ m$ (Earth Radius)
- n: Mean motion [rad/s]
- i_0: Orbit inclination [rad]

Finally, it is recalled that orbit corrections to account for slow changes in the Keplerian elements are not considered since the accuracy obtained without accounting for these corrections is enough for our scope.

5.3. Satellites Ground Track

5.3.1. ECEF Coordinates Calculation

The ECEF (Earth-Centered, Earth-Fixed) coordinates of a celestial object orbiting the Earth represent its $[x, y, z]$ position with respect to the center of mass of Earth, according to a given geodetic System. Particularly, we are considering the WGS-84, which is the most widely used system in order to define the coordinate's system fundamental and derived constants. The parameter description for this model is listed in the following table:

Table 5.4. WGS-84 Parameter Definition

WGS-84 Parameter	Value
a_E: Earth's Major Semi Axis	6,378,137 m
$e^2{}_E$: Earth's First Eccentricity Squared	0.00669437999014
$\dot{\Omega}_E$: Earth's Angular Velocity	$7.2921151467 \cdot 10^{-5}\ rad/s$
GM_E: Gravitational Constant	$3{,}986{,}004.418 \cdot 10^{8}\ m^3 \cdot s^{-2}$

The first step to obtain the ground track of Iridium Next satellites is to compute their ECEF coordinates at the current time using the Keplerian elements described in the previous section. In this way, in order to visualize these elements on a $[x, y, z]$ coordinate system, the following figure is shown:

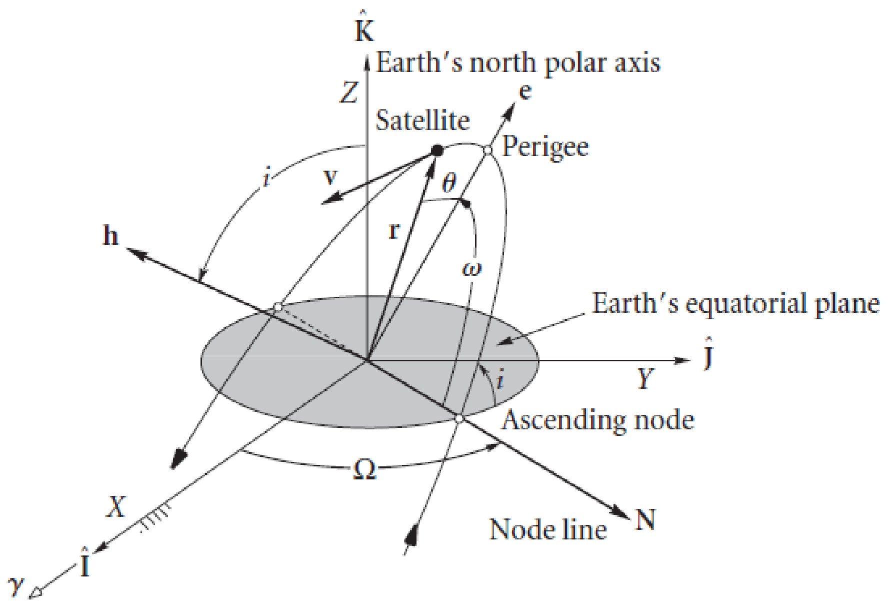


Fig. 5.3. Keplerian Elements $[x, y, z]$ Representation

At this point, we can obtain the ECEF coordinates for each satellite at current time as described in this sub-section. For such purpose, an adaptation of the procedure described in [15] has been used.

First, it is necessary to obtain the time (t_k) as the difference between the current time (t) and the epoch of the TLE file (t_0) as shown as follows:

$$t_k = t - t_0 \tag{5.3}$$

Second, the mean anomaly (M_k) at time t_k is computed as:

$$M_k = M_0 + n \cdot t_k \tag{5.4}$$

Third, the eccentric anomaly (E_k) at time t_k is computed by solving iteratively for E_k the following equation:

$$M_k = E_k - e \cdot \sin(E_k) \tag{5.5}$$

Moreover, the previous considerations should be considered:

- $E_k(0) = M_k$
- $E_k(n) = M_k + e \cdot \sin(E_k(n-1))$
- $|E_k(n) - E_k(n-1)| < 10^{-8}$

Fourth, the true anomaly (ν_k) at time t_k can be computed from its sine and cosine expressions, which ensure that no ambiguity is produced:

$$\sin(\nu_k) = \frac{\sin(E_k) \cdot \sqrt{1-e^2}}{\cos(E_k) \cdot (1-e)} \tag{5.6}$$

$$\cos(\nu_k) = \frac{\cos(E_k) - e}{1 - e \cdot \cos(E_k)} \tag{5.7}$$

In order to find ν_k, it is possible to define an auxiliary complex number (C) whose real and imaginary parts are the previous expressions:

$$C = \cos(\nu_k) + j \cdot \sin(\nu_k) \tag{5.8}$$

Then, ν_k is derived from the argument (α) of this complex number (C):

$$\tan(\alpha) = \frac{\sin(\nu_k)}{\cos(\nu_k)} = \tan(\nu_k) \tag{5.9}$$

$$\alpha = \arg(C) = \tan^{-1}(\nu_k) = \nu_k \tag{5.10}$$

Fifth, the argument of the satellite latitude (u_k) at time t_k can be computed as:

$$u_k = \nu_k + \omega \tag{5.11}$$

Sixth, the satellite orbit radius (r_k) at time t_k with respect to the Earth's center of mass defined by WGS-84 is computed as:

$$r_k = a \cdot (1 - e \cdot \cos(E_k)) \tag{5.12}$$

Seventh, the longitude of the ascending node (Ω_k) at time t_k considering the angular speed of Earth rotation ($\dot{\Omega}_E = 7.2921151467 \cdot 10^{-5}\ rad/s$) is given by:

$$\Omega_k = \Omega_0 + \dot{\Omega}_0 \cdot \mathrm{t}_k - \dot{\Omega}_E \cdot t \tag{5.13}$$

Eighth, the $[x, y]$ coordinates of the satellite at time t_k within its orbital plane are given by the following expressions:

$$x_p = r_k \cdot \cos(u_k) \tag{5.14}$$

$$y_p = r_k \cdot \sin(u_k) \tag{5.15}$$

Taking back to Fig. 5.3, the ECEF coordinates of the satellite can be obtained by projecting the previous $[x_p, y_p]$ coordinates on the cartesian axes by applying the following rotation matrix (R):

$$R = \begin{pmatrix} \cos(\Omega_k) & -\cos(i_0) \cdot \sin(\Omega_k) & 0 \\ \sin(\Omega_k) & \cos(i_0) \cdot \cos(\Omega_k) & 0 \\ 0 & \sin(i_0) & 0 \end{pmatrix} \tag{5.16}$$

In this way, the satellite ECEF coordinates at time are computed by applying the following matrix relationship:

$$\begin{pmatrix} x_{ECEF} \\ y_{ECEF} \\ z_{ECEF} \end{pmatrix} = R \cdot \begin{pmatrix} x_p \\ y_p \\ 0 \end{pmatrix} \tag{5.17}$$

Finally, the satellite ECEF coordinates at time t_k are given by the following expressions:

$$x_{ECEF} = x_p \cdot \cos(\Omega_k) - y_p \cdot \cos(i_0) \cdot \sin(\Omega_k) \tag{5.18}$$

$$y_{ECEF} = x_p \cdot \sin(\Omega_k) + y_p \cdot \cos(i_0) \cdot \cos(\Omega_k) \tag{5.19}$$

$$z_{ECEF} = y_p \cdot \sin(i_0) \tag{5.20}$$

5.3.2. LLA Coordinates Calculation

The next step after the ECEF coordinates of any given satellite from the constellation have been computed, is to convert them into LLA (Latitude, Longitude and Altitude) coordinates. This step is necessary since the ground track is a representation of the projection of the satellites' trajectories on the Earth map, where each point is defined as a pair of latitude & longitude coordinates.

The conversion from ECEF to LLA coordinates is derived from [16] and it can be understood by considering the figure on the following page, where a representation of both systems of coordinates are shown. There, it can be inferred that Longitude (λ) is the angle defined from the Prime Meridian Plane by rotating about the z-axis until the satellite position. Moreover, Latitude (ϕ) is defined as the angle defined from the Equatorial Plane by rotating about the x-axis.

In this way, we can conclude that it is necessary to rotate the ECEF coordinates position to the new LLA reference axis by means of the algorithm described throughout this sub-section.

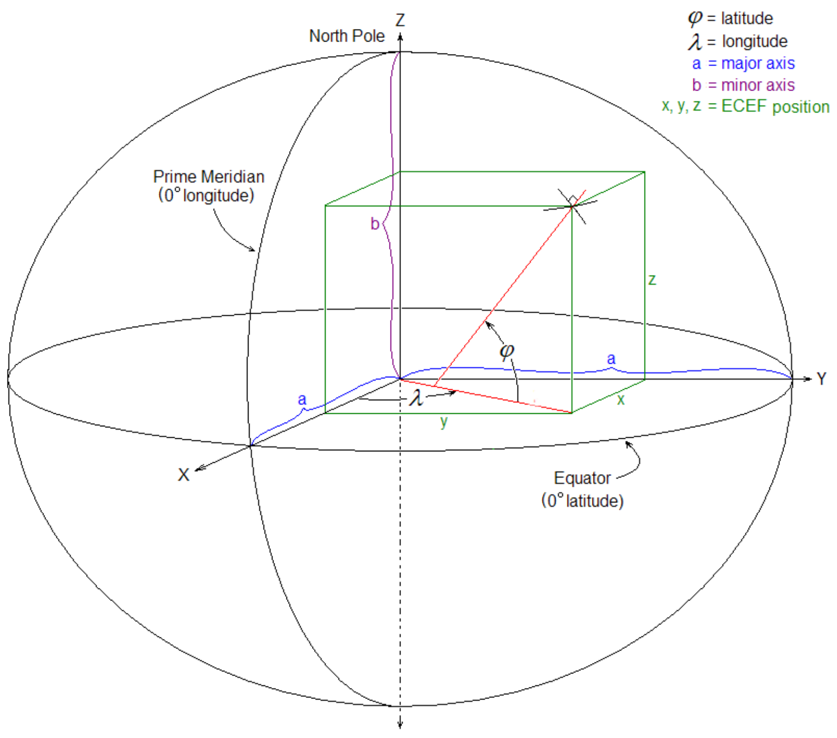


Fig. 5.4. ECEF and LLA Coordinates Systems Representation [16]

The first step to rotate the satellite's ECEF coordinates to LLA at current time is to compute the satellite orbit radius (r) as follows:

$$r = \sqrt{{x_{ECEF}}^2 + {y_{ECEF}}^2} \tag{5.21}$$

Then, a series of auxiliary parameters (E, F, G, C) needed to compute the satellite circumference arch are calculated by using the WGS-84 parameters that were defined in table 5.4 as well as the previously computed ECEF coordinates:

$$E^2 = {a_E}^2 - {b_E}^2, \qquad {a_E}^2 = {b_E}^2 + {c_E}^2, \qquad e_E = \frac{c_E}{a_E} \tag{5.22}$$

$$F = 54 \cdot {b_E}^2 \cdot {z_{ECEF}}^2 \tag{5.23}$$

$$G = r^2 + (1 - {e_E}^2) \cdot {z_{ECEF}}^2 - {e_E}^2 \cdot E^2 \tag{5.24}$$

$$C = \frac{{e_E}^4 \cdot F \cdot r^2}{G^3} \tag{5.25}$$

From the previous auxiliary parameters, the satellite circumference arch (s) is computed using the following expression:

$$s = \sqrt[3]{1 + C + \sqrt{C^2 + 2 \cdot C}} \tag{5.26}$$

Moving on, two extra auxiliary parameters (P, Q) are computed in order to obtain the distance between the center of the new coordinates system and the satellite position. Their expressions are given by:

$$P = \frac{F}{3 \cdot \left(s + \frac{1}{s} + 1\right)^2 \cdot G^2} \tag{5.27}$$

$$Q = \sqrt{1 + 2 \cdot {e_E}^4 \cdot P} \tag{5.28}$$

In this way, the distance between the center of the LLA coordinates system and the current satellite position is computed using the following expression:

$$r_0 = -\frac{P \cdot {e_E}^2 \cdot r}{1 + Q} + \sqrt{\frac{1}{2} \cdot {a_E}^2 \left(1 + \frac{1}{Q}\right) - \frac{P \cdot (1 - {e_E}^2) \cdot {z_{ECEF}}^2}{Q \cdot (1 + Q)} - \frac{1}{2} P r^2} \tag{5.29}$$

Furthermore, the final two auxiliary parameters (U, V) that are necessary to compute the new LLA coordinates can be computed as follows:

$$U = \sqrt{{z_{ECEF}}^2 + (r - r_0 \cdot {e_E}^2)^2} \tag{5.30}$$

$$V = \sqrt{{z_{ECEF}}^2 \cdot (1 - {e_E}^2) + (r - r_0 \cdot {e_E}^2)^2} \tag{5.31}$$

Then, the new z coordinate of the satellite (z_0) is computed as:

$$z_0 = \frac{{b_E}^2 \cdot z_{ECEF}}{a_E \cdot V} \tag{5.32}$$

Finally, the LLA coordinates of the satellite at current time are computed by combining all the previous parameters as it is shown in the following page:

$$h = U \cdot \left(1 - \frac{{b_E}^2}{a_E \cdot V}\right) \tag{5.33}$$

$$\phi = arctg\left(\frac{z_{ECEF} + e'^2 \cdot z_0}{r}\right), \qquad where\ e' = \frac{c_E}{b_E} \tag{5.34}$$

$$\lambda = arctg\left(\frac{y_{ECEF}}{x_{ECEF}}\right) \tag{5.35}$$

5.3.3. Map of Satellite Ground Tracks

The ground track of Iridium Next satellites can be obtained with *Matlab* by computing their LLA coordinates at current time, and then updating them periodically within a given simulation time. This can be made by first downloading the TLE file at current date from *CelesTrack* and obtaining the satellites' orbital parameters from there. After that, the LLA coordinates can be derived following the steps described in the previous sub-sections.

In this way, the following figure shows an Earth map containing the expected ground track of all Iridium Next satellites for the next 8 minutes, where it can be seen how the satellites are equally spaced into 6 orbital planes:

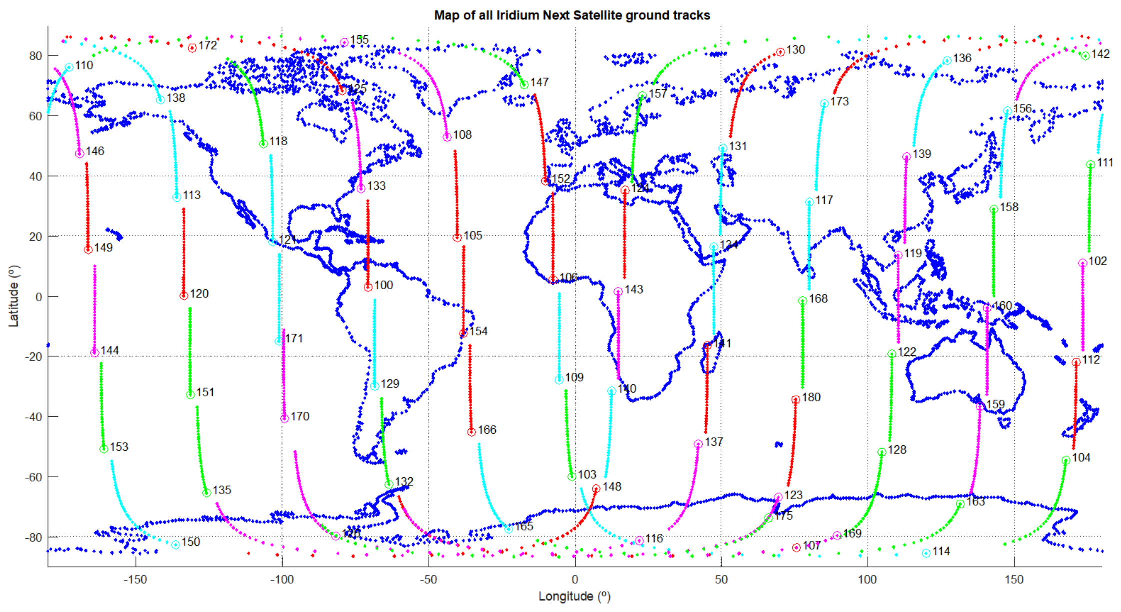


Fig. 5.5. Map of all Iridium Next Satellite Ground Tracks for the Next 8 Minutes

Also, the following code flowchart describes the operation of the Matlab code that was implemented to obtain the ground track of the satellites:

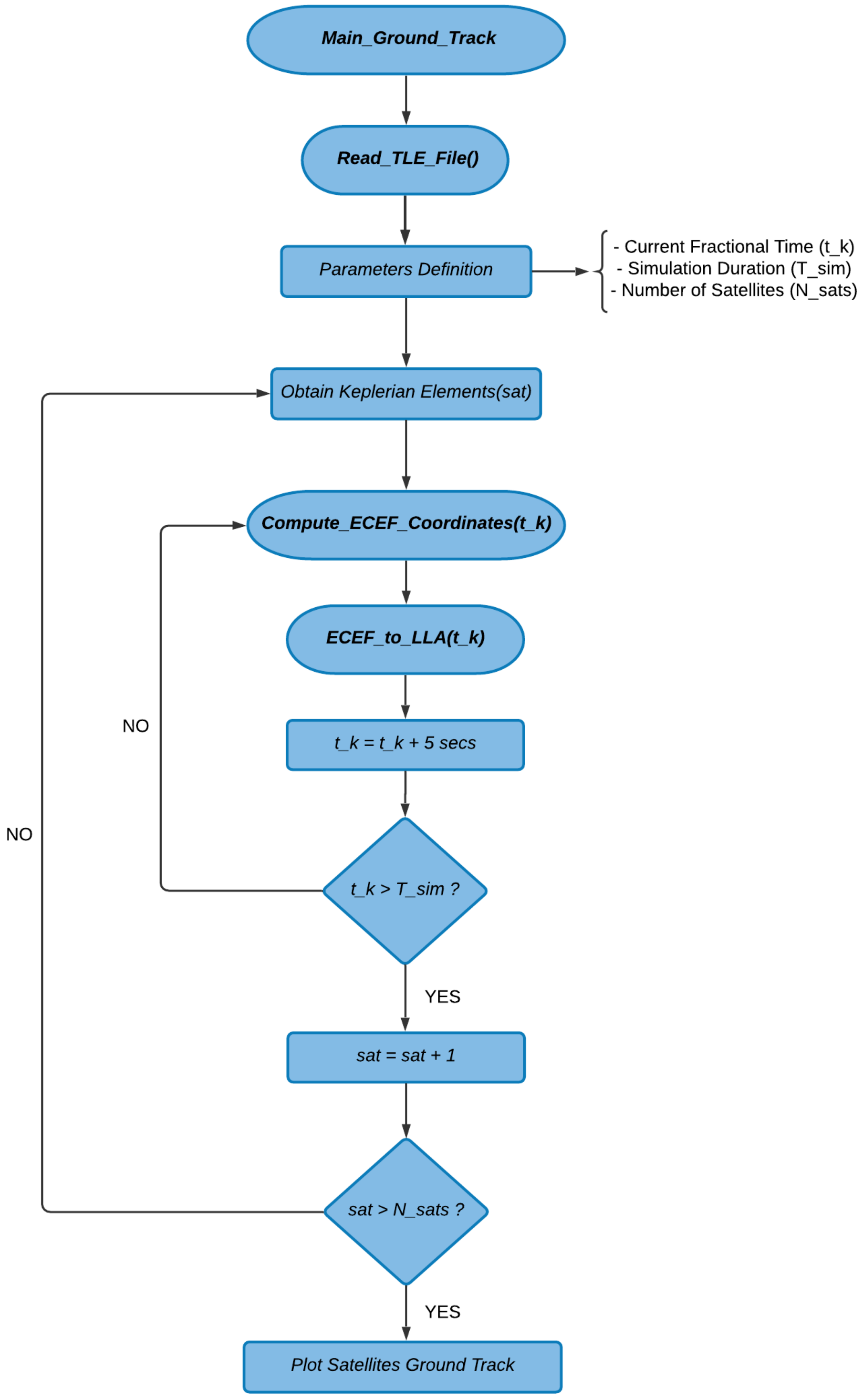

Main_Ground_Track
Read_TLE_File()
Parameters Definition
- Current Fractional Time (t_k)
- Simulation Duration (T_sim)
- Number of Satellites (N_sats)
Obtain Keplerian Elements(sat)
Compute_ECEF_Coordinates(t_k)
ECEF_to_LLA(t_k)
t_k = t_k + 5 secs
t_k > T_sim ?
NO
YES
sat = sat + 1
sat > N_sats ?
NO
YES
Plot Satellites Ground Track

Fig. 5.6. Main_Ground_Track Code Flowchart

Moreover, in order to verify that the ground track of the satellites was obtained correctly, we have plotted the expected trajectory of a given satellite from the constellation (Iridium 104) for one orbital period (100.4 minutes). In the following figure, it is observed that the selected satellite is expected to complete one orbit during this period since it will cover an entire lap around the planet. Then, we can conclude that its expected ground track was computed properly:

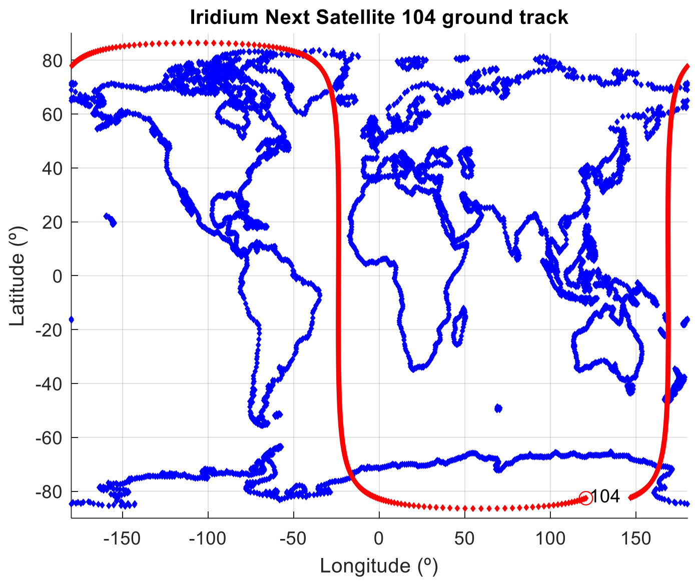


Fig. 5.7. Iridium Next Satellite 104 Ground Track for One Orbital Period

Finally, the following figure shows the expected ground track of the same satellite for 24 hours. It can be seen that it will cross the equator by 28 different locations, which means that it will complete 14 laps around the Earth as we discussed:

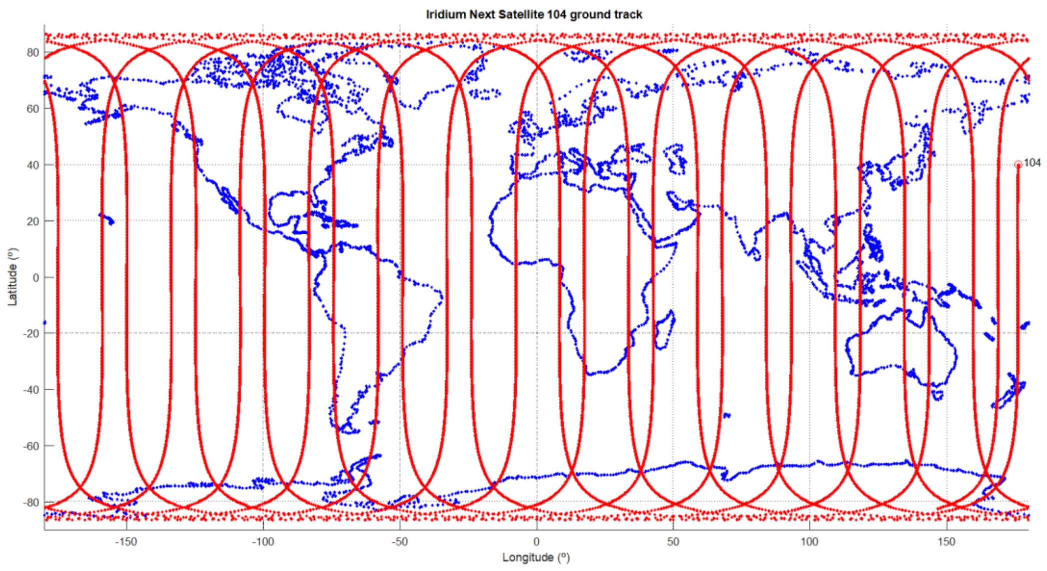


Fig. 5.8. Iridium Next Satellite 104 Ground Track for 24 hours

CHAPTER 6. SIGNAL RECORDING

6.1. Experimental Setup

6.1.1. AT1621-12 Iridium Next Antenna

In order to guarantee that the signal recording is performed properly and to ensure that the selected antenna does not increase the complexity of the experimental setup, it has been necessary to choose an antenna directly manufactured by the owner of the Iridium Next constellation.

As we discussed in previous chapters of the present document, ideally, it would have been more convenient to choose an antenna from a different manufacturer in order to approach the problem of navigating with Iridium Next satellite signals in a totally opportunistic fashion. However, since the scope of this thesis does not pretend to cover an antenna design/adaptation and since the receiver is independent from the selected antenna, it has been considered that the opportunistic navigation approach is maintained regardless the chosen antenna.

In this way, the selected antenna is the model AT1621-12 [17] from Iridium Next and its main electrical characteristics are listed in the following table:

Table 6.1. Electrical Specifications of the antenna AT1621-12

Parameter	Value
Frequency	1616 – 1626.5 MHz
Polarization	Right Hand Circular
Maximum Gain	4dBic
VSWR	$\leq$ 2.0:1
Impedance	50Ω
Power Handling	10W, 15% duty cycle
Operating temperature	-40ºC to 85ºC

From the previous table, the gain of the antenna is given in *dBic* units, which is a measurement in decibels of the antenna directivity referenced to circularly polarized theoretical isotropic radiator.

Moreover, the radiation coverage is given in the datasheet [16] of the antenna as a function of a measurement coordinate system which defines the main plane patters as azimuth (plane x-y) and elevation (plane y-z). In this way, both planes are orthogonal and are sufficient to define the directions on which the antenna is more directive. Besides, the orientation of the antenna over both planes is given as the sweep polar angles with respect to the origin of the planes. Hence, the azimuthal plane corresponds to $\theta = 90^{\circ}$, and the horizontal plane matches with $\varphi = 90^{\circ}$.

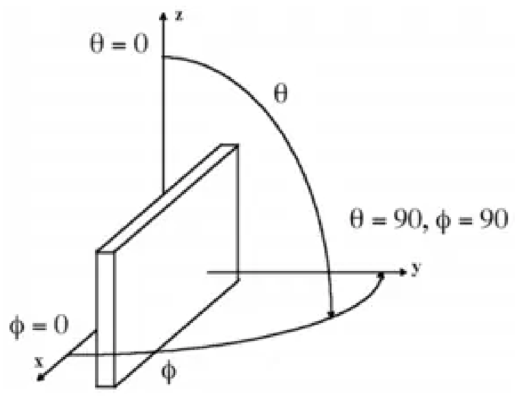


Fig. 6.1. Antenna Measurement Coordinate System

In this way, the following table summarizes the radiation coverage of the antenna AT1621-12, extracted from [16]:

Table 6.2. Radiation Coverage of the Antenna AT1621-12

Gain [dBic]	Sweep angle, θ
4.0	$\theta = 0^{\circ}$
-1.0	$0^{\circ} < \theta < 75^{\circ}$
-2.5	$75^{\circ} \leq \theta < 80^{\circ}$
-4.5	$80^{\circ} \leq \theta < 85^{\circ}$
-7.5	$\theta = 90^{\circ}$

Finally, the AT1621-12 antenna has been selected since it delivers an improved reception for satellite communications, it includes a magnetic mount that allows to allocate the antenna on the roof of a car and makes it portable, and it is specifically designed for acquiring Iridium Next satellite signals. Finally, the following figure shows a picture of the antenna:

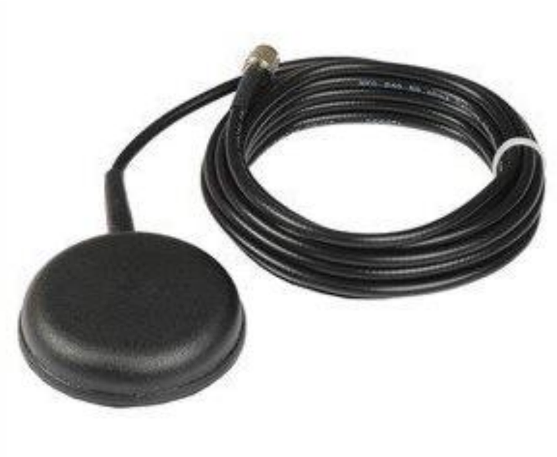


Fig. 6.2. AT1621-12 antenna picture

6.1.2. Universal Software Radio Peripheral

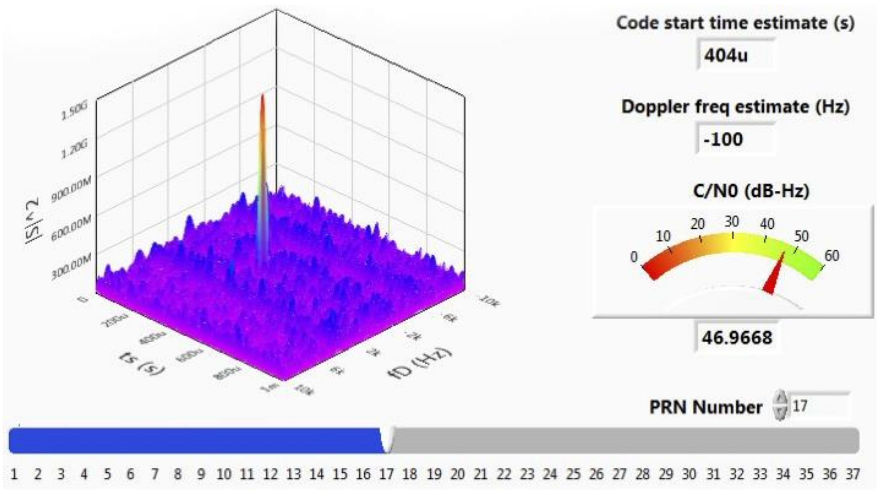
The device that allows for the sampling and acquisition of the signals that are captured by the AT1621-12 antenna is a USRP (Universal Software Radio Peripheral), which is highly interesting since it provides a software-defined RF architecture that helps the user to rapidly design systems with custom signal processing.

Particularly, we are using the USRP RIO (Reconfigurable I/O architecture) model USRP-2954 from the manufacturer of National Instruments [18] whose main characteristics are listed in the following table:

Table 6.3. Overview of features of the USRP-2954

Parameter	Value
RF Frequency Range	10MHz to 6GHz
Maximum Bandwidth	160MHz
Sampling Frequency Range	200MS/s
Gain Range	0dB to 31.5dB
Number of RX Channels	2

The previous radio supports different graphic interfaces that allow to control its RF parameters as well as to analyze the captured spectrum. For such purpose, a proprietary software of the ASPIN laboratory which is the Multichannel Adaptive Transceiver Information Extractor (Matrix) has been used. This interface treats all the ambient signals in the environment which are not necessarily transmitted for positioning or navigation sources as potential signals of opportunity. In this way, the software allows the user to rapidly detect opportune signals from which navigation and timing information can be obtained, which is of outmost interest for the purpose of the present thesis.



Code start time estimate (s): 404u

Doppler freq estimate (Hz): -100

C/N0 (dB-Hz): 46.9668

PRN Number: 17

Fig. 6.3. Interface of the Matrix Software

It might be recalled that we are using the USRP-2954 from a user point of view, which means that it is not pretended to cover the internal operation and performance of the device. Hence, we have used the radio as a black box that allows to visualize the signals available over the sampled spectrum and to record the RF signals of opportunity from Iridium Next satellites.

Finally, the following picture shows a detailed schematic of the radio peripheral obtained from [17]:

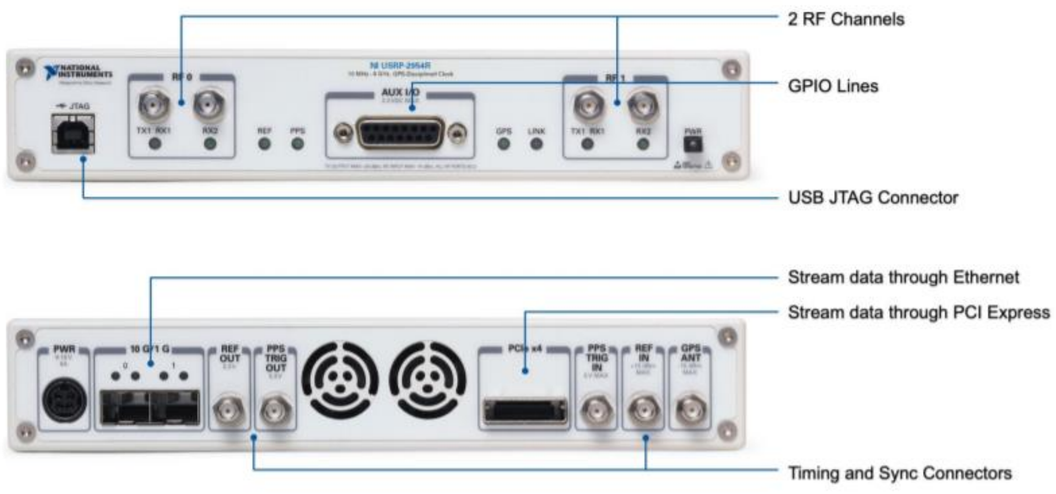

2 RF Channels
GPIO Lines
USB JTAG Connector
Stream data through Ethernet
Stream data through PCI Express
Timing and Sync Connectors

Fig. 6.4. Detailed View of the USRP-2954 [17]

6.1.3. Selection of Location and Recording time

The final step regarding the experimental setup concerns the selection of the recording location as well as the selection of the best time that will allow for having visible satellites over the receiver location.

On the one hand, the recording location has been selected so that the experimental setup can be deployed outdoors in a place that has a certain elevation with respect to the ground and it is not shielded by any surrounding building. In this way, we have selected the top roof of the parking building from the Engineering Gateway of the University of California, Irvine (UCI) whose LLA coordinates are given in the following table:

Table 6.4. LLA coordinates of the recording place

Latitude	Longitude	Altitude
33.643301	-117.837981	40

On the other hand, the selection of the recording time has been made by means of the satellite orbit predictor that was described in chapter 5. Particularly, several simulations were performed in order to know which is the expected time at which some Iridium next satellites will pass over the receiver location that was defined in table 6.4.

Then, the following table describes the window time at which the data recording was performed:

Table 6.5. Definition of the Recording Time Window

	Start Time	End Time	Duration
PST Time	21:20:11h	21:24:29h	4mins 18 secs
UTC Time	04:20:11h	04:24:29h	

Besides, in order to demonstrate that a satellite was visible from the selected location at the specific recording time, a plot of the expected ground track has been obtained. Particularly, the following figure shows the expected trajectory of the Iridium Next satellite 128 (IRRNTX42811) for 10 minutes from the start time that was defined in table 6.5:

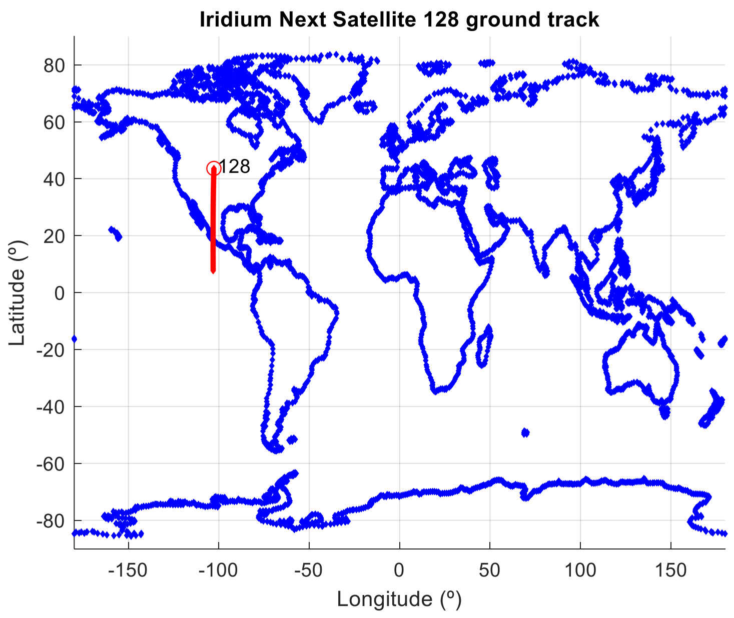


Fig. 6.5. Expected Ground Track of Iridium Next Satellite 128 for 10 minutes

Finally, a sky plot of the previous satellite was obtained from the information given in the TLE files that are published by the NORAD and whose access is public via online. The figure on the following page proves that the window time for recording is optimum since the observed satellite will cross the receiver's location from horizon to horizon.

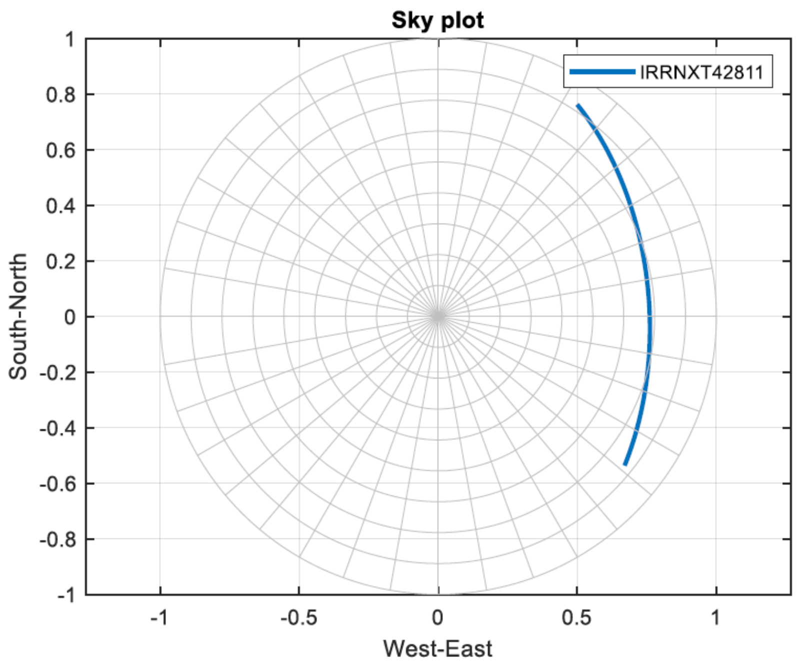


Fig. 6.6. Expected Sky Plot of Iridium Next Satellite 128 for 10 minutes

6.2. Data Analysis

6.2.1. Interpretation of the Recorded Signals

Once the signal recording was performed, it is necessary to assess whether the recorded signals correspond to the Iridium Next satellite that was crossing over the receiver’s location or they are the result of having acquired interferent sources that are not useful for our navigation purposes.

As it was previously discussed, the recorded signal is stored into a binary file as a set of IQ samples. In this way, the first step is to ensure that the size of the file and thus the number of samples that were recorded are compliant with our expectations according to the following considerations:

- The recording duration is 4 minutes and 18 seconds
- The sampling frequency was set to f_s = 2.4 MHz
- 1 Sample = I (in-phase) + Q (Quadrature)
- The size of each sample is I (int16) + Q (int16) = 32 bits = 8 Bytes

Then, since the file size is 2,582,500,000 Bytes, the following computation proves that the recorded signal was stored correctly:

$$2.5825 \cdot 10^9 \; Bytes \cdot \frac{1 \; IQ}{4 \; Bytes} \cdot \frac{1 \; s}{2.5 \cdot 10^6 \; IQ} = 258.25 \; s = 4'18" \tag{6.1}$$

After that, the next step is to analyze the specific frequency band (1626MHz-1626.5MHz) on which the downlink simplex channels of Iridium Next are defined according to [9]. The following table gathers the center frequencies of these channels:

Table 6.6. Iridium Next Downlink Simplex Channels

	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
f_c [MHz]	1626.1042	1626.1458	1626.2708	1626.3958	1626.4375

Furthermore, it is necessary to design a strategy that allows to accurately guarantee the presence of satellite signals on this band. In this way, the interpretation of the recorded signals has been realized by computing the PSD (Power Spectral Density) over the simplex downlink frequency band for different time instants.

Since the motion of satellites produces a Doppler effect on the frequency of the received signals, it is expected that, if the signal was transmitted by a satellite, the peak of the PSD will be moving over time towards the left on the frequency axis. This can be explained since the receiver is stationary and the satellite will be approaching it and then moving away. Hence, the expected Doppler time history that the receiver observes on its acquired signals must be decreasing progressively.

The following figure shows the PSD computations that were performed over the targeted frequency band for different time instants with increments of 30 seconds:

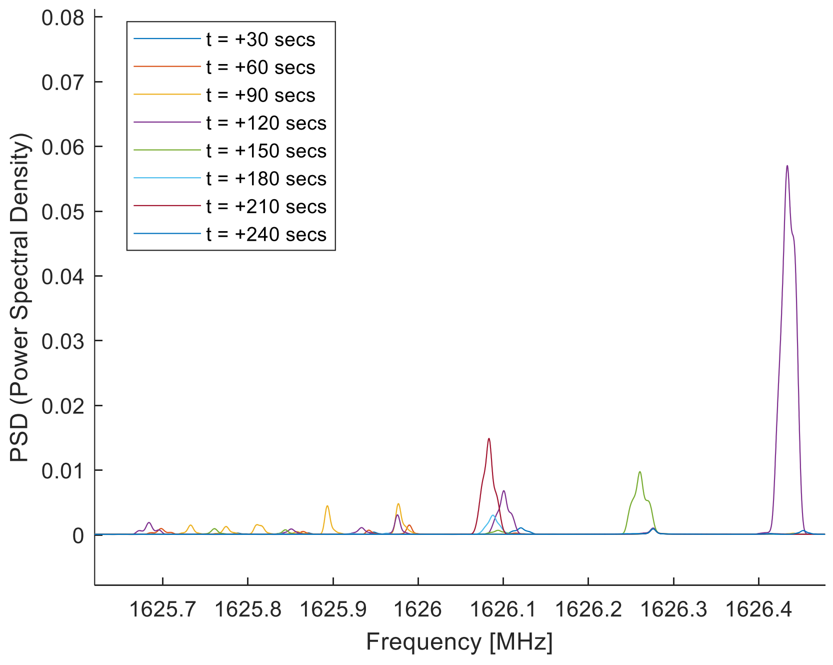


Fig. 6.7. PSD computations at different time instants for the entire band

From the previous figure, we can clearly conclude that the recorded signals correspond to moving emitters since the peaks of the PSD are shifting in frequency over time. Moreover, given the information of table 6.6, it can be as well inferred that a satellite signal contained on the first Iridium Next downlink simplex channel ($f_c = 1626.1042$MHz) was recorded.

Finally, the following figure shows a zoom over the center frequency of this channel, which contains the PSD computation for different time instants. In there, it can be appreciated that the peaks of the PSD are moving towards the left, which means that the Doppler frequency is decreasing throughout the recording.

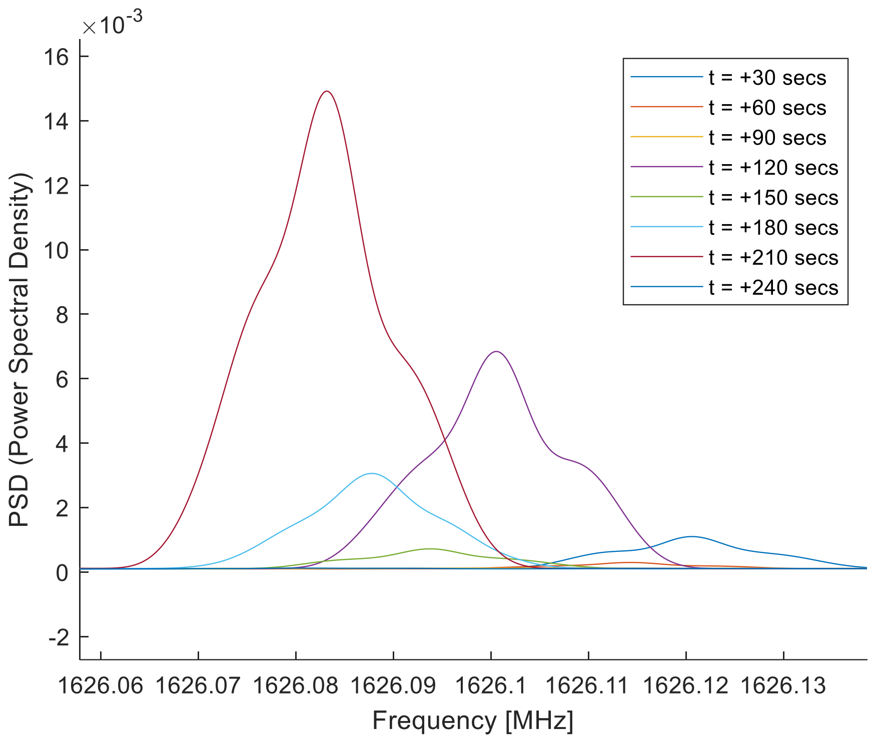


Fig. 6.8. PSD computations at different time instants for $f_c = 1626.1042$MHz

6.2.2. Doppler Frequency Time History Reconstruction

After having guaranteed that the recorded signal corresponds to a moving emitter which was most likely transmitting on the first Iridium Next downlink channel, it is necessary to completely verify that it was in fact transmitted by the Iridium Next satellite (IRRNTX42811) which was visible from the receiver's location at the specific recording time.

For such purpose, it is possible to use the information given in the TLE files in order to obtain the expected Doppler frequency time history that a signal transmitted by the satellite Iridium Next 128 should have experienced. The figure on the following page shows this expected Doppler frequency profile.

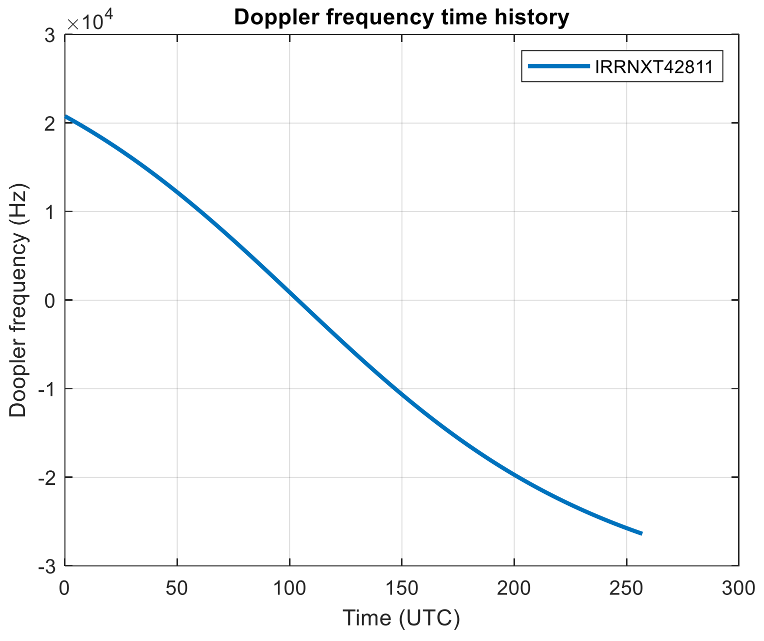


Fig. 6.9. Expected Signal Doppler Frequency from the Satellite IRRNTX42811

The final step is to reconstruct the observed Doppler frequency throughout the recording time by using the information from Fig. 6.8, and to compared it with the expectancy predicted on the TLE files, which was shown in Fig. 6.9.

In this way, the center frequency of the PSD peaks from Fig. 6.8, which are associated to different time instants and which represent the Doppler frequency shift observed by the receiver, will be saved in order to generate the observed Doppler time history. Besides, the bandwidth of the peaks has been also analyzed in order to compare it with the specification values given in the Iridium Next documentation [9] that states the downlink simplex channels to have a bandwidth of 35kHz. Finally, the following table summarizes this information:

Table 6.7. Measured center frequency and bandwidth of the PSD peaks

Time offset	Center Frequency (f_c)	Bandwidth ($\Delta f = f_2 - f_1$)
$t = +30\ s$	$f_c = 1626.12057722168$ MHz	$f_2 = 1626.14010847168$ MHz
		$f_1 = 1626.10173261719$ MHz
		$\Delta f = 38.376$ kHz
$t = +60\ s$	$f_c = 1626.11424482422$ MHz	$f_2 = 1626.13110578613$ MHz
		$f_1 = 1626.09753645020$ MHz
		$\Delta f = 33.569$ kHz
$t = +90\ s$	$f_c = 1626.10890424805$ MHz	$f_2 = 1626.11668623047$ MHz
		$f_1 = 1626.10356367188$ MHz
		$\Delta f = 31.123$ kHz
$t = +120\ s$	$f_c = 1626.10051191406$ MHz	$f_2 = 1626.12034833984$ MHz
		$f_1 = 1626.08037031250$ MHz
		$\Delta f = 39.978$ kHz

$t = +150\ s$	$f_c = 1626.09379804688\ \text{MHz}$	$f_2 = 1626.10928571777\ \text{MHz}$
		$f_1 = 1626.07724226074\ \text{MHz}$
		$\Delta f = 32.043\ \text{kHz}$
$t = +180\ s$	$f_c = 1626.08777082520\ \text{MHz}$	$f_2 = 1626.10623395996\ \text{MHz}$
		$f_1 = 1626.06862104492\ \text{MHz}$
		$\Delta f = 37.613\ \text{kHz}$
$t = +210\ s$	$f_c = 1626.08319318848\ \text{MHz}$	$f_2 = 1626.10386884766\ \text{MHz}$
		$f_1 = 1626.06190717773\ \text{MHz}$
		$\Delta f = 41.962\ \text{kHz}$

From the previous table, it is observed that all the values of the measured PSD peak bandwidth are around the expected value of 35kHz with an approximate variance of ±5kHz, which leads to conclude that is highly probable that the recorded signal was produced by the satellite IRRNTX42811.

Furthermore, considering that the satellite transmitted the signal at a center frequency of 1626.1042MHz, the difference between this expected center channel and the measured center frequencies from Table 6.7 correspond to the Doppler shift that the signal suffered until it was acquired by the receiver. In this way, the following figure shows the measured Doppler frequency time history, which is the results of these subtractions:

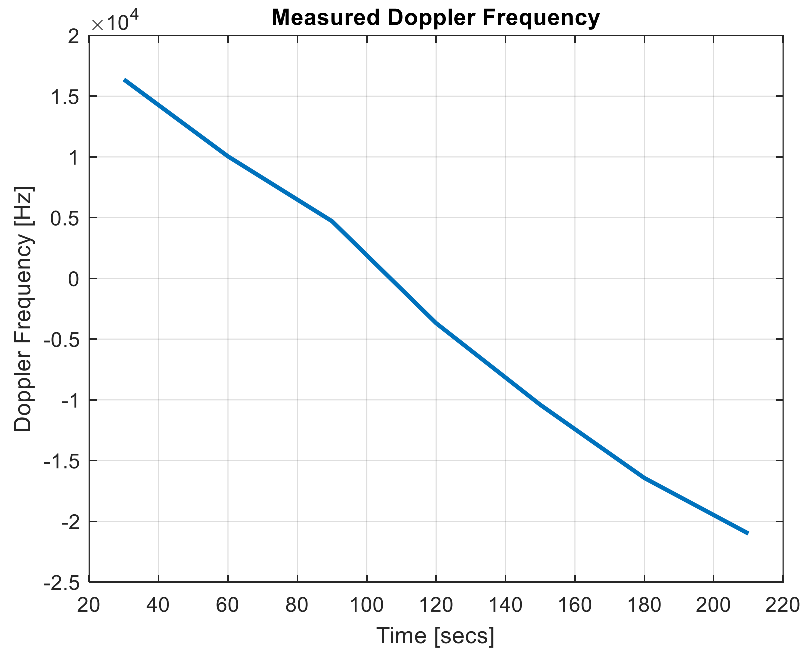


Fig. 6.10. Measured Signal Doppler frequency from satellite IRRNTX42811

Finally, it has been proved that the recorded signal was transmitted by an Iridium Next satellite and thus can be used for obtaining a navigation solution given that the comparison between the expected and the measured signal Doppler frequencies on the figure of the following page match for almost all time instants.

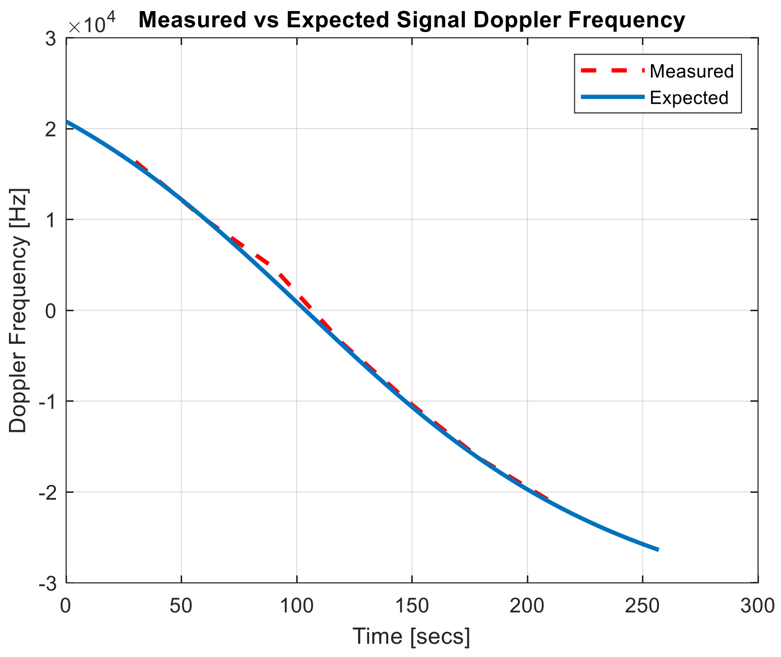


Fig. 6.11. Measured vs Expected Signal Doppler Frequency

As a final remark, it might be noticed that the measured Doppler when the satellite was above us (the expected value is zero) is not as accurate as the rest of the measures due to the high dynamics that are produced at this moment. The Doppler frequency observed by the receiver is related to the projection of the acceleration vector of the satellite on the pointing vector joining the receiver. In this way, since the direction of this vector is highly variable when the satellite is above the receiver's location, the measured Doppler frequency losses accuracy.

CHAPTER 7. NAVIGATION FRAMEWORK

7.1. Model Description

7.1.1. Problem Formulation

The obtention of a PNT (Positioning, Navigation and Timing) solution requires from the definition of a framework on which our designed receiver and the Iridium Next satellites will be placed. The formulation of the navigation problem considers a stationary receiver which makes Doppler frequency measurements to the available satellites and uses this information to estimate its position by means of an EKF (Extended Kalman Filter).

Besides, the navigation solution will not be produced in real time since it is necessary for the receiver to first track the Doppler frequency from the satellite signal and then to iterate this estimates with an EKF in order to obtain the final navigation solution, which will be an estimate of its stationary position.

First, the receiver state consists of its constant clock drift ($\dot{\delta t}_{rec}$) and its position three-dimensional vector defined as:

$$r_{rec} \triangleq [x_{rec}, y_{rec}, z_{rec}]^T \tag{7.1}$$

Besides, the three-dimensional position and velocity vectors of the each l-th LEO satellite at time step k are obtained from the TLE files and are defined as shown:

$$r_{LEO,l}(k) \triangleq \left[x_{LEO,l}, y_{LEO,l}, z_{LEO}\right]^T, \qquad k = 0,1,2\ldots \tag{7.2}$$

$$\dot{r}_{LEO,l}(k) \triangleq \left[\dot{x}_{LEO,l}, \dot{y}_{LEO,l}, \dot{z}_{LEO,l}\right]^T, \qquad k = 0,1,2\ldots \tag{7.3}$$

Finally, it is considered that each l-th LEO satellite has a different clock drift ($\dot{\delta t}_{LEO,l}$) which it is not known by the receiver and it is assumed to be constant.

7.1.2. Pseudorange Rate Measurement Model

This section pretends to model how the Doppler measurements produced by the receiver can be translated to pseudorange rate measurements. In this way, at time step k, the pseudorange rate measurements to the l-th LEO satellite are given by:

$$z_{LEO,l}(k) \triangleq c\frac{\hat{f}_{D,l}(k)}{f_{c,l}}, \qquad k = 0,1,2\,... \tag{7.4}$$

Where:

- $\hat{f}_{D,l}(k)$ is the Doppler estimate from the $l\text{-}th$ LEO satellite at time k
- $f_{c,l}$ is the carrier frequency at which the $l\text{-}th$ LEO satellite is transmitting

Moreover, the pseudorange rate measurements to each $l\text{-}th$ LEO satellite can be expressed as:

$$z_{LEO,l}(k) = \frac{\dot{r}_{LEO,l}^{T}(k)\left[r_{rec} - r_{LEO,l}(k)\right]}{\left\|r_{rec} - r_{LEO,l}(k)\right\|} + c\Delta\dot{\delta}t_l + v_{LEO,l}(k) \tag{7.5}$$

From the previous expression, the clock bias ($\Delta\dot{\delta}t_l$) between the receiver and each satellite is defined as:

$$\Delta\dot{\delta}t_l = \dot{\delta}t_{rec} - \dot{\delta}t_{LEO,l} \tag{7.6}$$

Finally, the measurement noise ($v_{LEO,l}$) present at each measurement of the receiver to each satellite is modeled as zero-mean white noise with variance $\sigma_{LEO,l}^2$.

7.2. Extended Kalman Filter

7.2.1. Operating Principle

The EKF (Extended Kalman Filter) is used for estimating the receiver's position by using measured empirical data that is physically related to this position and has inherent a certain random component.

Then, given a series of physical parameter values such as $g = f(x)$ for which direct measurements cannot be made, it is possible to estimate g from the empirical measurements made on x, which is an observable function from which direct information can be obtained.

It might be noticed that while our receiver is able to make Doppler measurements to satellite signals by implementing a tracking loop, it cannot perform direct measurements on the environment to estimate its position. However, since it was previously shown in (7.4), the Doppler frequency is proportional to the pseudorange rate of the receiver and, hence, by obtaining accurate measurements from the Doppler shift and using them as input to an EKF, the receiver will be able to obtain an estimate of its current position.

The motivation of using an EKF instead of a standard Kalman filter arises from the need of having an estimator that can be applied to nonlinear systems, which are the most common in the engineering field. In this way, the EKF linearizes the model of a nonlinear system in order to estimate its dynamics by means of state-space techniques and recursive algorithms [19].

The EKF consists of two differentiated steps (prediction and correction) which are iterated progressively over time. First, the system state is predicted using its known dynamical model. After that, the second step consists on correcting the first estimate with the observation model by using the criteria of minimizing the error covariance of the estimator. Finally, this procedure is repeated for each time step recursively using the state of the previous step as initial value.

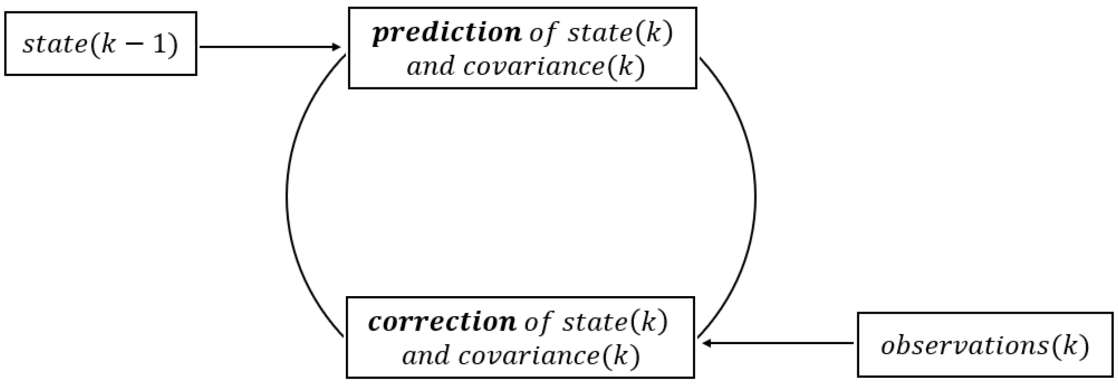

$state(k-1)$
$\textbf{prediction}$ $of\ state(k)$ $and\ covariance(k)$
$\textbf{correction}$ $of\ state(k)$ $and\ covariance(k)$
$observations(k)$

Fig. 7.1. Circuit of the Extended Kalman Filter

7.2.2. EKF Model

The model of the EKF that is used on our navigation framework considers two satellites and the receiver. In this way, the state to be estimated is composed by the variables that are aimed to be estimated, which are the position of the receiver and the clock biases from the satellites:

$$x \triangleq \left[r_{rec}^{T}, c\Delta\dot{\delta}t_{LEO,1}, c\Delta\dot{\delta}t_{LEO,2}\right] \tag{7.7}$$

Besides, the EKF is designed to produce an estimate $\hat{x}(k|m)$ of $x(k)$ using all the pseudorange rate measurements from time step 1 to $m \leq k$. By way of clarification, the notation $\hat{x}(k|m)$ represents the estimate of x at time k given observations up to and including at time $m \leq k$.

The EKF computes as well the estimation error covariance $\mathrm{P}(k|m)$ which is defined as follows:

$$\mathrm{P}(k|m) \triangleq \mathrm{E}[\tilde{x}(k|m)\tilde{x}^T(k|m)] \quad \textbf{(7.8)}$$

Where the estimation error ($\tilde{x}$) is defined as the difference between the current estimate at time k and the actual value as:

$$\tilde{x}(k|m) \triangleq [x(k) - \hat{x}(k|m)] \quad \textbf{(7.9)}$$

The equations that describe the dynamics of the EKF consists of two sets of coupled equations that are known as time predict and time update equations. In particular, given prior estimates such as $\hat{x}(k|k)$ and $\mathrm{P}(k|k)$, the EKF state and estimation error covariance time-update equations are expressed as:

$$\hat{x}(k+1|k) = F \cdot \hat{x}(k|k) \quad \textbf{(7.10)}$$

$$\mathrm{P}(k+1|k) = F \cdot \mathrm{P}(k|k) \cdot F^T + Q \quad \textbf{(7.11)}$$

For the case of our proposed navigation framework, the state matrix ($\hat{x}$) consists of a column vector containing the three-dimensional coordinates of the receiver's position as well as the clock drifts of the two LEO satellites:

$$\hat{x} \triangleq \left[x_{rec},\ y_{rec},\ z_{rec},\ c\Delta\dot{\delta}t_{LEO,1},\ c\Delta\dot{\delta}t_{LEO,2}\right]^T \quad \textbf{(7.12)}$$

Furthermore, F is defined to be the identity matrix with dimensions 5x5, and Q is defined to be the process noise covariance. Theoretically, since x is a constant vector, Q should be a zero matrix. Nevertheless, in order to prevent the estimation error covariance from converging to zero, Q is chosen to be an identity matrix with dimensions 5x5 but multiplied by a factor $\epsilon = 10^{-12}$.

On the other hand, EKF state and covariance measurement predict equations are given by:

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \mathrm{K}(k+1)\mathcal{V}(k+1) \quad \textbf{(7.13)}$$

$$\mathrm{P}(k+1|k+1) = [I - \mathrm{K}(k+1)H(k+1)]\mathrm{P}(k+1|k) \quad \textbf{(7.14)}$$

Where $\mathrm{H}(k+1)$ is the measurement Jacobian defined in the following way:

$$\mathrm{H}(k+1) = \left[h_{LEO,1}^T(k+1),\ h_{LEO,1}^T(k+1)\right]^T \tag{7.15}$$

Each element $h_{LEO,i}^T(k+1)$ corresponds to the partial derivative with respect to the position and a unitary gain placed on the $i\text{-}th$ position in order to include the effect of the clock drift of each LEO satellite:

$$h_{LEO,1}^T(k+1) = \left[\frac{\partial h_1(x,k)}{\partial r}, 1{,}0\right] \tag{7.16}$$

$$h_{LEO,2}^T(k+1) = \left[\frac{\partial h_2(x,k)}{\partial r}, 0{,}1\right] \tag{7.17}$$

And each derivative of $h_i(x,k)$ can be obtained as shown:

$$\frac{\partial h_i(x,k)}{\partial r} = \frac{\dot{r}_{LEO,i}(k+1)}{\left\|\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right\|} - \left[\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right] \cdot \frac{\dot{r}_{LEO,i}^T(k+1) \cdot \left[\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right]}{\left\|\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right\|^3} \tag{7.18}$$

Finally, the measurement Jacobian remains as:

$$\mathrm{H}(k+1) = \begin{bmatrix} \frac{\partial h_1(x,k)}{\partial r} & 1 & 0 \\ \frac{\partial h_2(x,k)}{\partial r} & 0 & 1 \end{bmatrix} \tag{7.19}$$

On the other hand, $\mathrm{K}(k+1)$ is the standard Kalman gain, which can be computed using the following expression:

$$K(k+1) = \mathrm{P}(k+1|k) \cdot \mathrm{H}(k+1)^T \cdot S^{-1}(k+1) \tag{7.20}$$

Where:

$$S(k+1) = \mathrm{H}(k+1) \cdot \mathrm{P}(k+1|k) \cdot \mathrm{H}^{\mathrm{T}}(k+1) + R \tag{7.21}$$

In the previous expression, R is defined to be the measurement noise covariance matrix, whose diagonal elements are the noise variance associated to the measurements produced to each LEO satellite:

$$R = \begin{bmatrix} \sigma_{LEO,1}^2 & 0 \\ 0 & \sigma_{LEO,2}^2 \end{bmatrix} \tag{7.22}$$

Finally, the last element of the prediction equations is the innovation vector, which is formed according to:

$$\mathcal{V}(k+1) = \begin{bmatrix} z_{LEO,1}(k+1) \\ z_{LEO,2}(k+1) \end{bmatrix} - \begin{bmatrix} \hat{z}_{LEO,1}(k+1) \\ \hat{z}_{LEO,2}(k+1) \end{bmatrix} \tag{7.23}$$

Where $z_{LEO,i}(k+1)$ is computed according to (7.5) and $\hat{z}_{LEO,i}(k+1)$ is obtained by means of the following equation:

$$\hat{z}_{LEO,i}(k+1) = \frac{\dot{r}_{LEO,i}^T(k+1)\left[\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right]}{\left\|\hat{r}_{rec}(k+1|k) - r_{LEO,i}(k+1)\right\|} + c\Delta\dot{\delta}t_i(k+1|k) \tag{7.24}$$

In summary, the following table contains a description of the dimensions of all EKF model variables:

Table 7.1. Description of the EKF model variables

Variable	Definition	Dimensions
$\hat{x}(k\|k)$	State Vector	5x1
$\mathrm{P}(k\|k)$	Estimation Error Covariance	5x5
Q	Process Noise Covariance	5x5
$\mathrm{H}(k+1)$	Measurement Jacobian	2x5
R	Measurement Noise Covariance	2x2
$K(k+1)$	Standard Kalman Gain	5x2
$\mathcal{V}(k+1)$	Innovation Vector	2x1

7.2.3. Filter Initialization

Once the equations of the model for the EKF were described, it is necessary to select appropriately some initialization parameters in order to ensure that the performance of the filter is adequate, and its produced position estimate for the receiver is as accurate as possible.

First, a selection of a prior estimate $\hat{x}(0|0)$ for the state vector is required in order to initialize the EKF. Theoretically, if the performance of the filter was optimal and it was iterated tending to infinity, a random estimate for this variable could be selected and the filter would be able to converge to the true position. Unfortunately, we must bound the initial estimate in order to improve the convergence velocity of the filter, so we are using a multivariate normal distribution of the true position of the receiver (which is known) with covariance $\mathrm{P}(0|0)$.

Second, the initial estimation error covariance $\mathrm{P}(0|0)$ is set aiming to approach a pessimistic scenario where the measurements made by the receiver are not accurate enough and thus the EKF produces huge estimation errors, such as position errors in the order of 1km and clock drifts in the order of the seconds:

$$\mathrm{diag}[\mathrm{P}(0|0)] = [10^6, 10^6, 10^6, 10,10] \tag{7.25}$$

Third, the velocities and positions of the satellites are obtained from the information of the TLE files by using SPG4 (Simplified General Perturbations) orbit determination software.

Fourth, the measurement noise covariance matrix (R) is an indicator of the veracity given to the measurements produced by the receiver. Specifically, lower values for the individual covariances of each LEO satellite imply that the measurements are accurate and thus the noise introduces lower errors. On the contrary, higher values assume imprecise measurements for which the causes might be either known or not. Then, the selection of these values is a trade off such as lower values will increase the performance of the filter but will simulate a less realistic scenario, and higher values might simulate a more truly environment but at the risk of lowering in excess the performance of the EKF.

$$R = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.3^2 \end{bmatrix} \tag{7.26}$$

Finally, the product factor (ϵ) of the process noise covariance matrix (Q) is set to be in the order of 10^{-12}.

7.2.4. Matlab Implementation

The figure on the following page contains the code flowchart of the implementation of the EKF into Matlab.

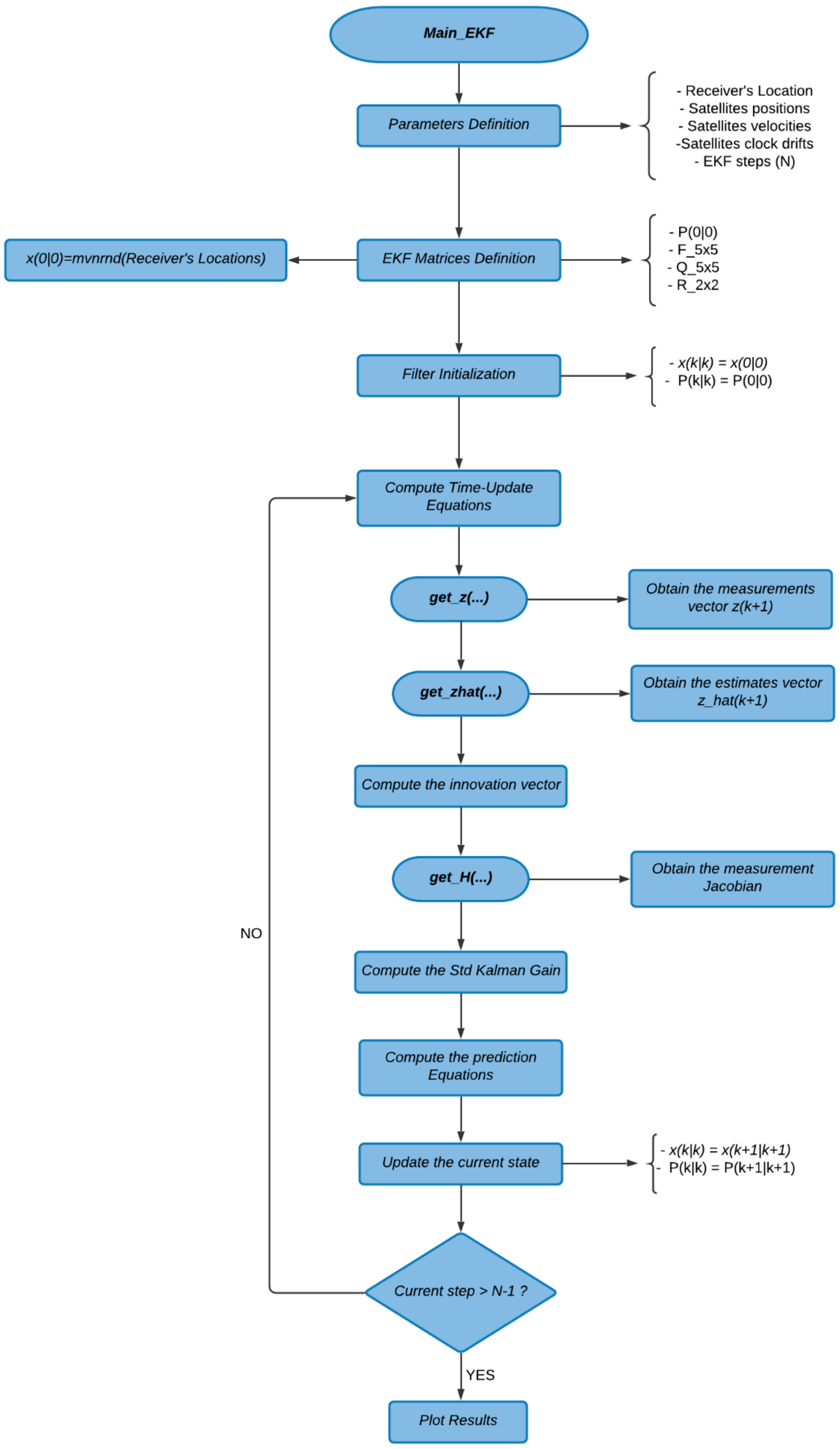


Fig. 7.2. Main_EKF Code Flowchart

CHAPTER 8. RESULTS

8.1. Simulation Environment

8.1.1. Simulated Signal Tracking

This section exhibits the results obtained by the receiver after having been able of acquire and to track an Iridium Next simulated satellite signal. Besides, the initialization parameters that were set up in order to track the Doppler frequency from the simulated signal are shown in the following table:

Table 8.1. Receiver's Initialization Parameters in Simulation

Parameter	Value
Sampling Frequency: f_s	$240\ kHz$
Carrier Frequency: f_c	$1626\ MHz$
Skipping Time	$20s$
PSD Estimate Time	$0.1s$
Integration Time: T_{int}	$6.8 \cdot 10^{-3}s$
Frame Duration: T_{frame}	$6.8 \cdot 10^{-3}s$
Tone Period: T_{tone}	$6.8 \cdot 10^{-3}s$
Tone Duration	$2.6 \cdot 10^{-3}s$
Loop Filter Initial Noise Bandwidth: $B_{L,0}$	13

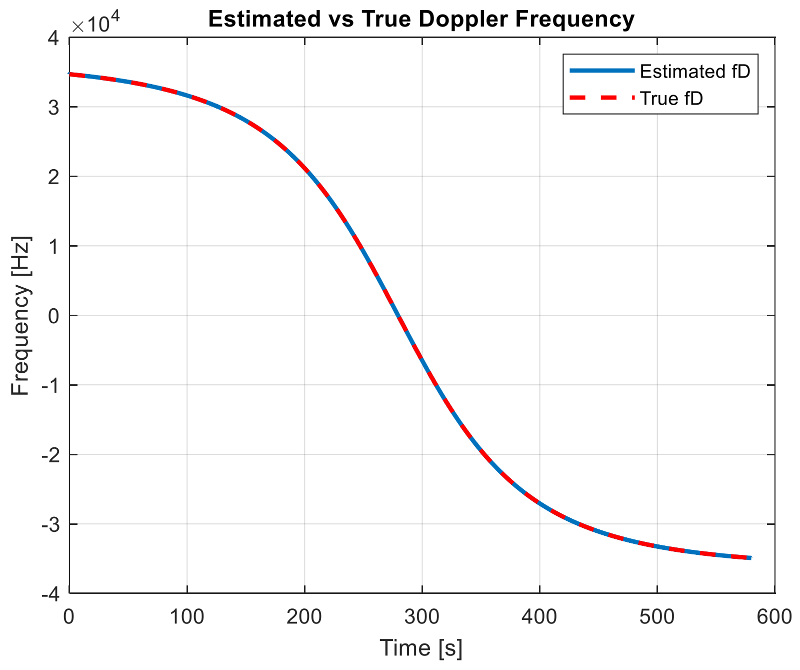


Fig. 8.1. Estimated vs True Doppler Frequency in Simulation

The figure on the previous page shows a comparison between the true Doppler of the simulated signal, which was obtained from the TLE files using SPG4 software as it was described in chapter 3, and the Doppler frequency estimate produced by the receiver.

It might be noticed that the tracked Doppler frequency matches almost exactly for the entire simulation, which was set to 10 minutes. Moreover, the tracking is not produced for the entire simulation time since it is necessary for the receiver to skip the first 20 seconds in order to improve the accuracy of the first Doppler estimate computed using the PSD (Power Spectral Density) for 0.1 seconds.

On the other hand, the absolute error of the Doppler estimates compared to the true values is shown in the following figure:

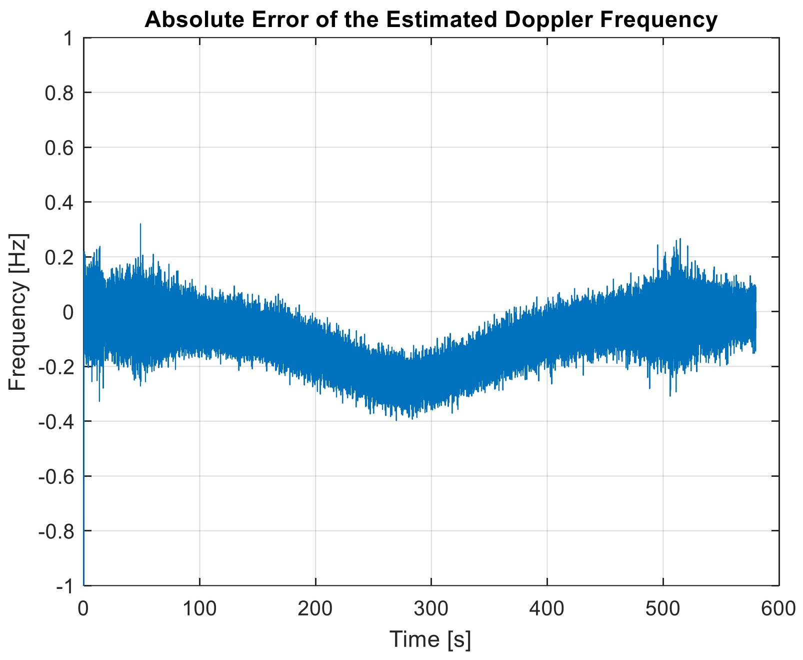


Fig. 8.2. Absolute Error of the Estimated Doppler Frequency in Simulation

From the previous figure, it is seen that the absolute error variance at the beginning of the acquisition is higher compared to the rest of the simulation. This can be explained since the receiver is initialized with a first Doppler frequency estimate that might not be enough accurate. Moreover, the initial signal phase is assumed to be zero, which is not a realistic value, and which produces an initialization lag. In the same way, since the initial signal timing shift is computed from the initial Doppler estimate, it also can produce a higher error for the initial frames.

On the other hand, it is observed that the maximum error is produced in the mid part of the simulation. This was also expected since, on this stage, the true Doppler frequency of the signal is close to zero, the dynamics of the system are higher and thus the receiver is more demanded and performs a worse estimate.

Finally, the following figure shows the signal phase error throughout the simulation time. As it was previously discussed, it is expected that the evolution of this variable converges to zero since, by definition, it is a measure of the difference between the true and the estimated Doppler, as well as between the true and the estimated signal phase.

Furthermore, it is observed that the worse values of the phase error are produced at the same stages of the simulation where the absolute error of the estimated Doppler frequency is maximum since both magnitudes are by definition related.

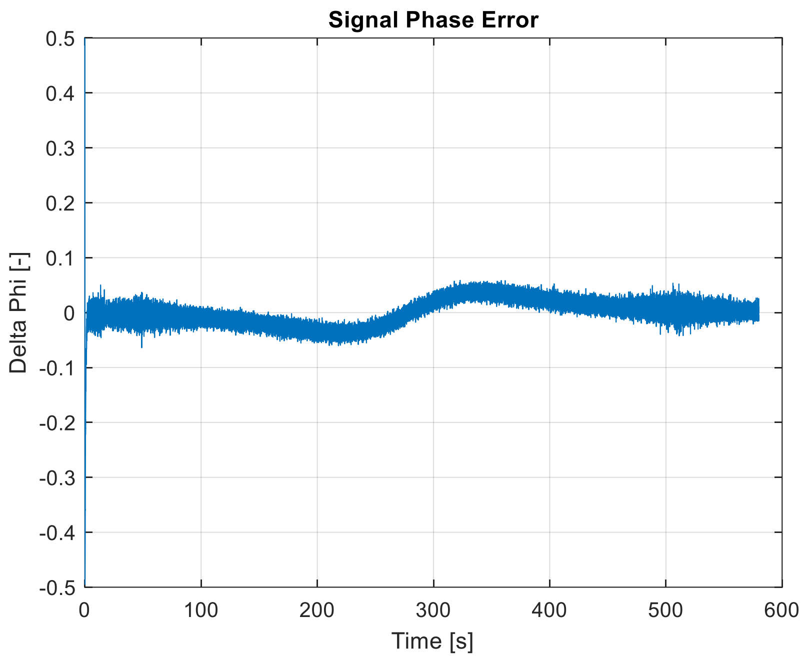


Fig. 8.3. Signal Phase Error in Simulation

8.1.2. Simulated Navigation Solution

Once the receiver demonstrated to be able to obtain accurate results for the Doppler frequency tracking in the simulation environment, the following figures show the results obtained after feeding these estimates to an EKF.

The true location of the receiver, which is assumed to be known in order to compare it with the estimate produced by the filter, is shown in the following table:

Table 8.2. True position of the receiver in simulation

LLA	Latitude	Longitude	Altitude
	33.643301º	-117.837981º	40 m
ECEF	x_{rec}	y_{rec}	z_{rec}
[m]	-2482107.57446003	-4700181.05692380	3513599.39891489

Moreover, the initialization parameters of the EKF are listed as follows:

$$\mathrm{diag}[\mathrm{P}(0|0)] = [10^6, 10^6, 10^6, 10,10] \tag{8.1}$$

$$\mathrm{diag}[\mathrm{Q}] = [10^{-12}, 10^{-12}, 10^{-12}, 10^{-12}, 10^{-12}] \tag{8.2}$$

$$R = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.3^2 \end{bmatrix} \tag{8.3}$$

$$\hat{x}(0|0) \ = mvnrnd\left(r_{rec}, c\Delta\dot{\delta}t_{LEO,1}, c\Delta\dot{\delta}t_{LEO,2}\right) \tag{8.4}$$

On the other hand, the error produced by the EKF at each step $k+1$ between the true position and the estimated position of the receiver for each three-dimensional coordinate (x, y, z) is given by:

$$\varepsilon(k+1) = [\hat{x}(k+1), \hat{y}(k+1), \hat{z}(k+1)]^T - r_{rec} \tag{8.5}$$

Besides, the upper and lower bounds of the estimation error standard deviation at each step $k+1$ for each three-dimensional coordinate (x, y, z) can be computed as:

$$\sigma_{+} = 3\sqrt{\left[\mathrm{P}_{xx}(k+1|k+1), \mathrm{P}_{yy}(k+1|k+1), \mathrm{P}_{zz}(k+1|k+1)\right]} \tag{8.6}$$

$$\sigma_{-} = -3\sqrt{\left[\mathrm{P}_{xx}(k+1|k+1), \mathrm{P}_{yy}(k+1|k+1), \mathrm{P}_{zz}(k+1|k+1)\right]} \tag{8.7}$$

In this way, the figures on the following pages show the time history of the error produced on the estimated receiver’s position by the EKF for the entire simulation duration.

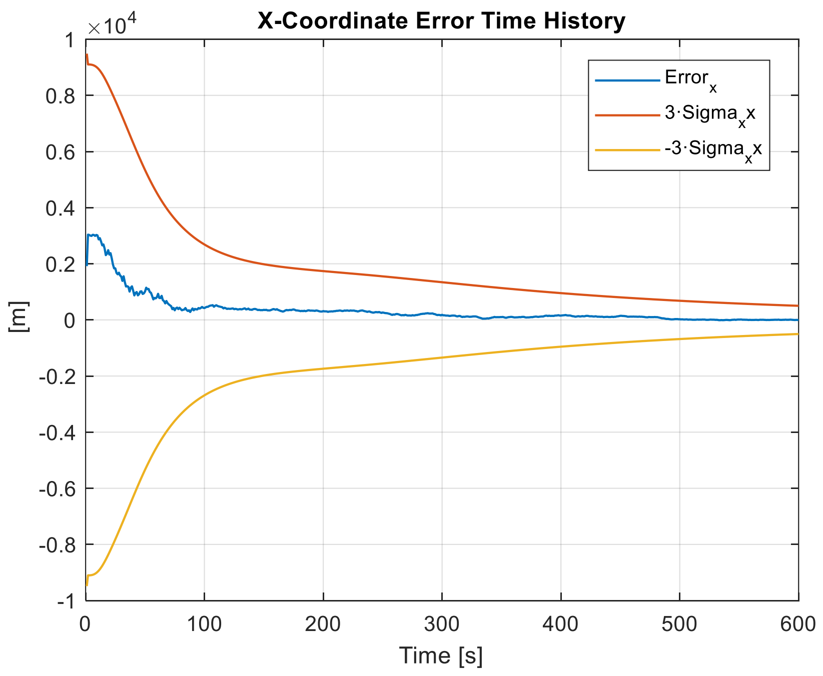


Fig. 8.4. X-Coordinate Error Time History in Simulation

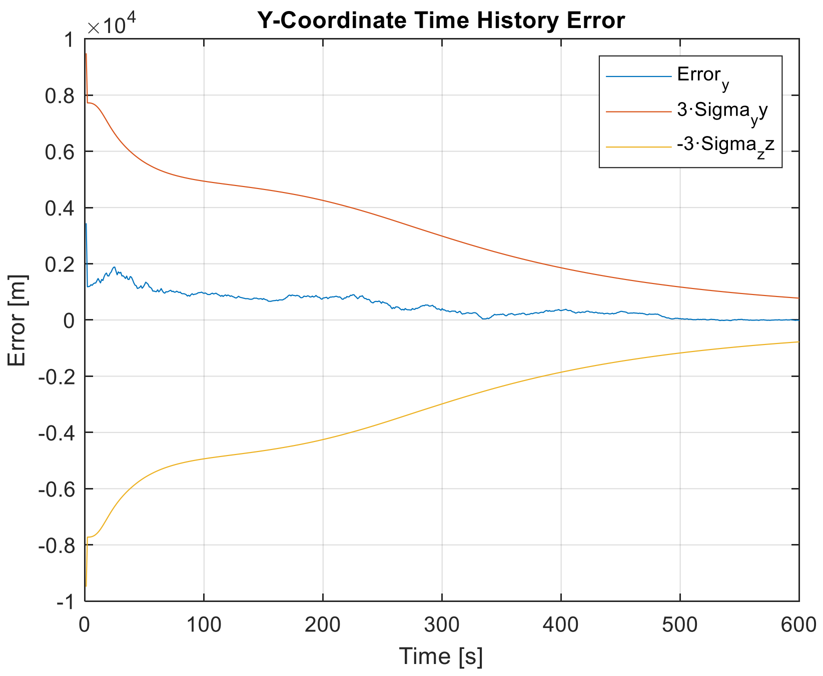


Fig. 8.5. Y-Coordinate Error Time History in Simulation

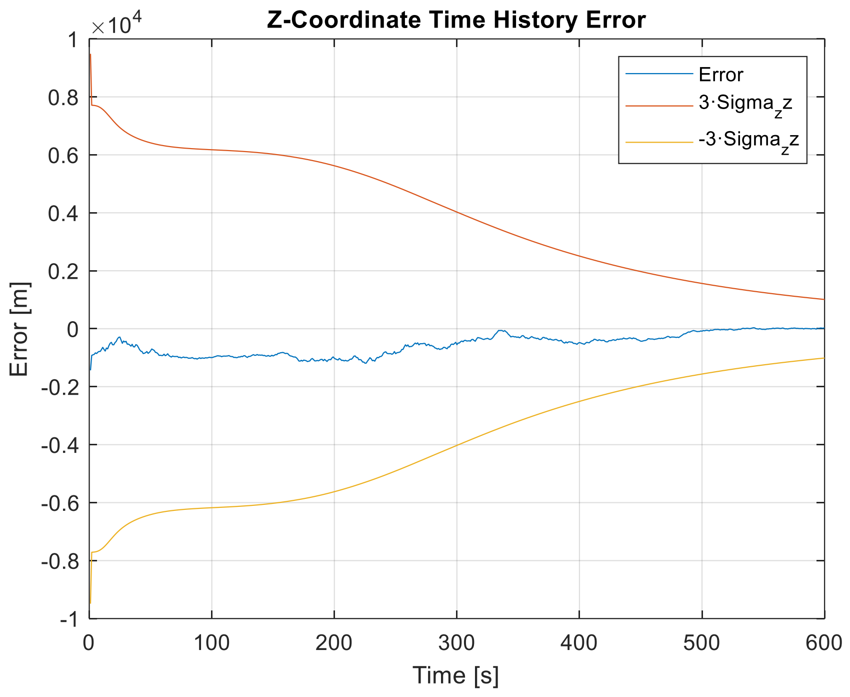


Fig. 8.6. Z-Coordinate Error Time History in Simulation

From the previous figures, it is observed that the error produced by the EKF on each coordinate of the receiver's estimated position converged to zero. This might be explained since the Doppler estimates that the receiver was able to produce from the simulated signal were highly accurate. Then, the more accurate the Doppler measurements, the more accurate is the EKF predicted position.

Furthermore, the estimation errors produced on each coordinate as well as the final estimated location of the receiver are given in the following table:

Table 8.3. Estimation Errors and Receiver's Position Estimate in Simulation

ε_x	ε_y	ε_z
6.3348 m	17.0054 m	32.3984 m
$\hat{x}_{rec}$	$\hat{y}_{rec}$	$\hat{z}_{rec}$
-2482113.90923264 m	-4700198.06235169 m	3513631.99733095 m

And the total error on the estimated receiver's position vector with respect to the true receiver's location is computed as:

$$\varepsilon_{Total} = \|\hat{r}_{rec} - r_{rec})\| = 37.3091\ m \tag{8.8}$$

As it could be expected, the position estimation error is lower in the x, y coordinates since it is easier for the proposed model of the EKF to relate the Doppler measurements information to a two-dimensional map location. Besides, it is also intuitive that since the satellites are always above the receiver, it is easier to obtain a good geometry in the x, y plane instead on the vertical coordinate, and thus the estimates on the vertical plane are worse.

Finally, the following figure shows the true and the final estimate positions of the receiver's in the simulation environment. It is observed that the true position is located on the roof top of the Engineering Gateway Building of UCI and the estimated receiver's position is off by only a few meters.

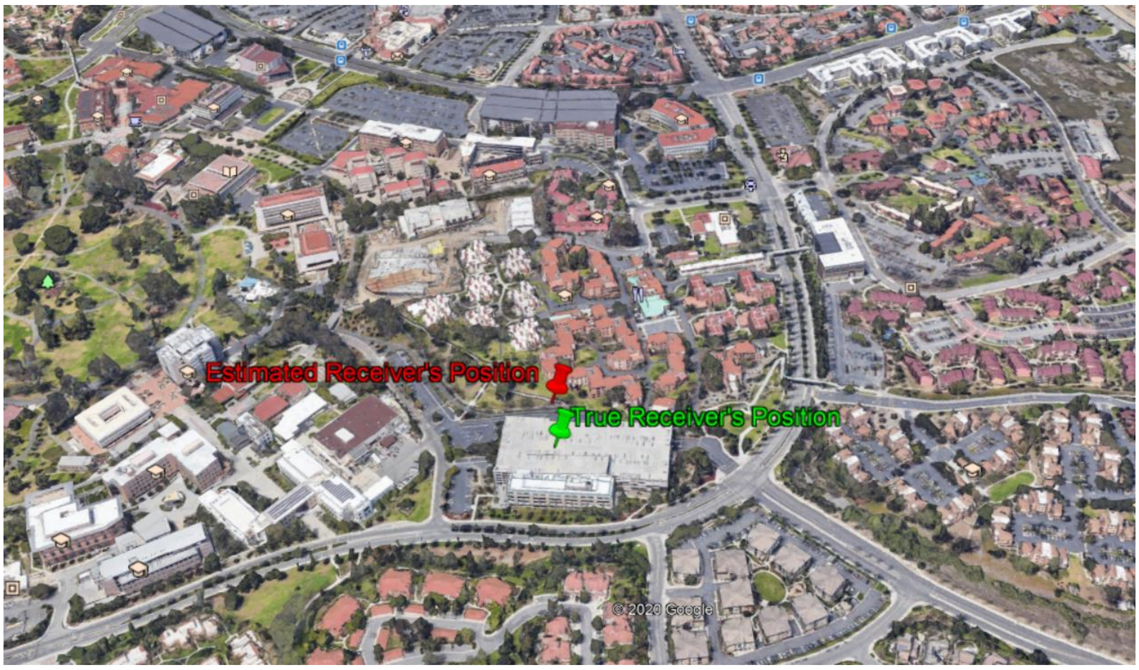


Fig. 8.7. True vs Estimated Receiver's Location in Simulation

8.2. Scenario with Iridium Next Satellite Signals

8.2.1. Iridium Next Satellite Signal Tracking

This section presents the experimental results obtained from a real Iridium Next satellite signal. Particularly, the initialization parameters that were set up on the receiver in order to track the Doppler frequency from the recorded signal that was described in chapter 6 are shown in the table on the following page:

Table 8.4. Receiver's Initialization Parameters in the Real Scenario

Parameter	Value
Sampling Frequency: f_s	$2.4\ MHz$
Carrier Frequency: f_c	$1626.1042\ MHz$
Skipping Time	$5s$
PSD Estimate Time	$0.1s$
Integration Time: T_{int}	$5.6 \cdot 10^{-3}s$
Frame Duration: T_{frame}	$5.6 \cdot 10^{-3}s$
Tone Period: T_{tone}	$5.6 \cdot 10^{-3}s$
Tone Duration	$1.8 \cdot 10^{-3}s$
Loop Filter Initial Noise Bandwidth: $B_{L,0}$	13

Some parameters were redefined comparing to the simulations. Particularly, the sampling frequency has been selected according to the experimental set up defined when the signal was recorded. Moreover, the carrier equals the first Iridium Next simplex downlink channel on which the satellite that was available during the recording was transmitting.

Besides, the frame and signal timing parameters have been modified according to the values that were observed when the recorded signal was analyzed in chapter 6. Similarly, it has been necessary to skip the initial 5 seconds of the recorded file in order to improve the accuracy of the Doppler frequency tracking.

Finally, the following figure shows a comparison between the measured Doppler frequency from the recorded signal, the true Doppler given by the TLE files and the estimation produced by the receiver:

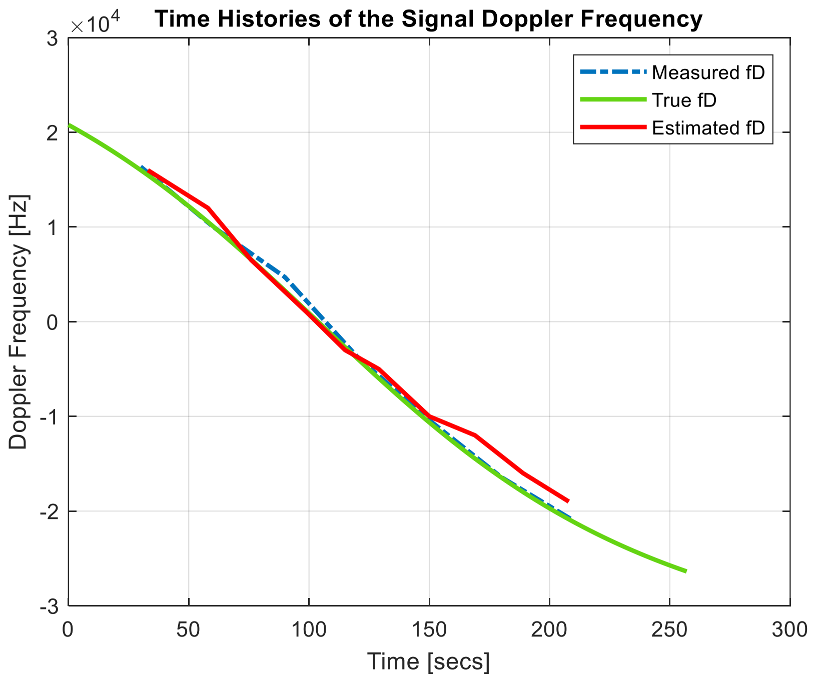


Fig. 8.8. Time Histories of the Signal Doppler Frequency

It is observed that the Doppler estimates produced by the receiver to a real satellite signal are much worse compared to the estimates obtained in the simulation environment. In fact, the previous figure provides valuable information that might explain this behavior. To start, the Doppler measurements that were made to the recorded signal do not coincide with the Doppler profile predicted from the TLE files, which means that experimental setup introduced sources of error that are reflected in the Doppler estimates.

Second, the timing information of the recorded signal does not match exactly with the information provided in the Iridium Next documentation [9], which means that the duration of the recorded tones is not constant anymore and the duration of the frames has a variance of approximately 5ms. Hence, although the initialization of the receiver has been modified, it is expectable that its tracking performance is worse. Finally, given the scope of the present thesis, it is considered that the obtained Doppler tracking with a real satellite signal shows promising results.

On the other hand, the following figure shows the signal phase error produced by the receiver from the recorded signal. It is observed that the error is not able to converge to zero, as is it was observed in the simulation environment. However, it is bounded between the interval $[-0.2\,rad, 0.2\,rad]$, which equals to say that the Doppler estimate might be off by $[-500\,Hz, 500\,Hz]$. In the same way, it will be proved in the following sub-section that committing this error on the signal phase estimate will suppose an error propagation in the receiver's estimated position, which could be off by hundreds of meters

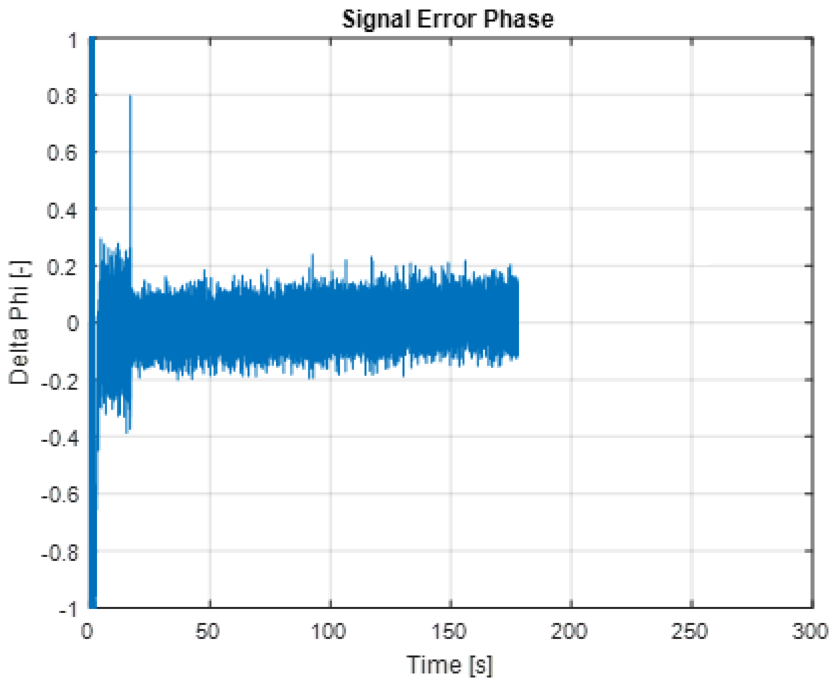


Fig 8.9. Signal Phase Error from the Real Satellite Signal

8.2.2. Iridium Next Navigation Solution

Although the Doppler frequency tracking from the recorded Iridium Next satellite signal is not accurate enough for obtaining a precise navigation solution, it is possible to use this information with the EKF proposed in chapter 7 in order to obtain an estimate of the receiver's location.

It might be noticed that the signal recording was performed in the roof top of the engineering gateway building, whose coordinates are given in Table 8.2. Then, the receiver's position estimate will be compared to this true location. Moreover, the model of our proposed EKF must be redefined since only one satellite was available and visible for the receiver when the recording was made. In this way, the initialization parameters of the EKF considering only one satellite signal are listed as follows:

$$\mathrm{diag}[\mathrm{P}(0|0)] = [10^9, 10^9, 10^9, 10] \tag{8.9}$$

$$\mathrm{diag}[\mathrm{Q}] = [10^{-12}, 10^{-12}, 10^{-12}, 10^{-12}] \tag{8.10}$$

$$R = 0.2^2 \tag{8.11}$$

$$\hat{x}(0|0) \ = mvnrnd\left(r_{rec}, c\Delta\dot{\delta}t_{LEO}\right) \tag{8.12}$$

From the previous expressions, the new model for the EKF in case of having visible only one satellite consists of estimating the receiver's three-dimensional position as well as the satellite clock drift. Then, the dimensions of the model matrices are reduced accordingly.

Besides, the error $\varepsilon(k+1)$ produced by the EKF at each step $k+1$ between the true position and the estimated position of the receiver holds with the definition given in (8.5). In the same way, the expressions for the upper and lower bounds of the estimation error standard deviation (σ_+ and σ_-) at each step $k+1$ maintain the definitions given in (8.6) and (8.7).

Finally, the figures on the following pages show the time history of the error produced on the estimated receiver's position by the EKF using the Doppler measurements that the receiver obtained from the real satellite signal.

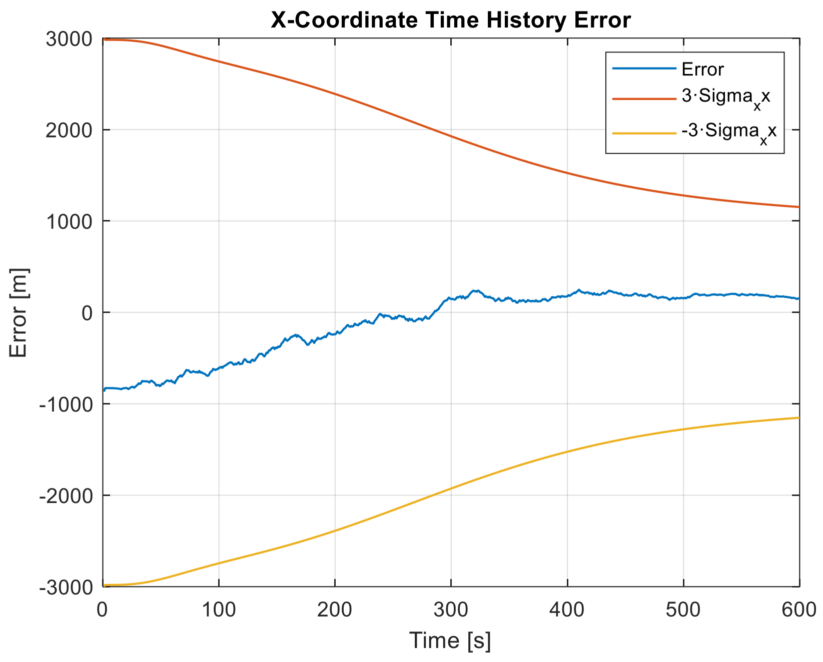


Fig. 8.10. X-Coordinate Error Time History from a Real Satellite Signal

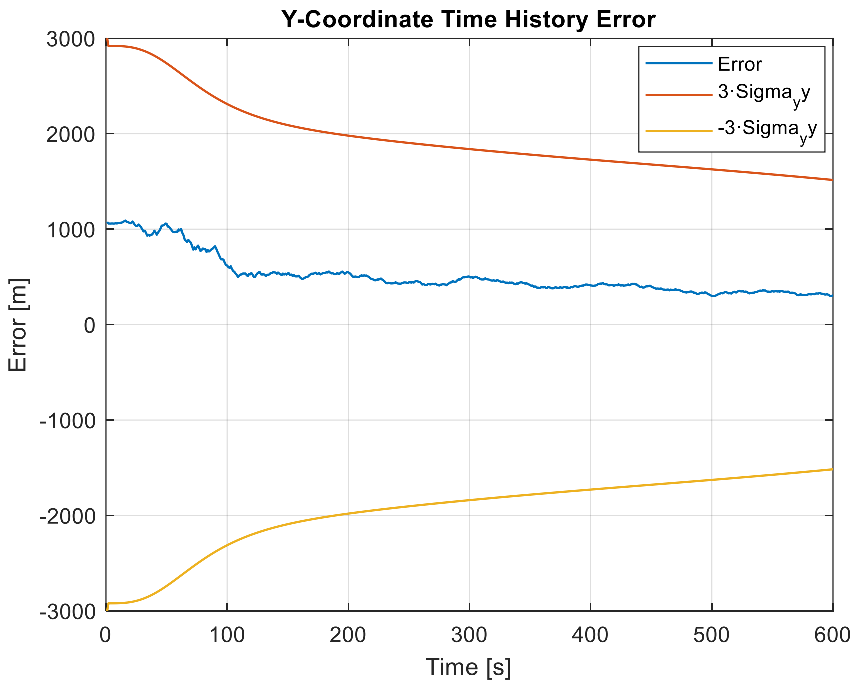


Fig. 8.11. Y-Coordinate Error Time History from a Real Satellite Signal

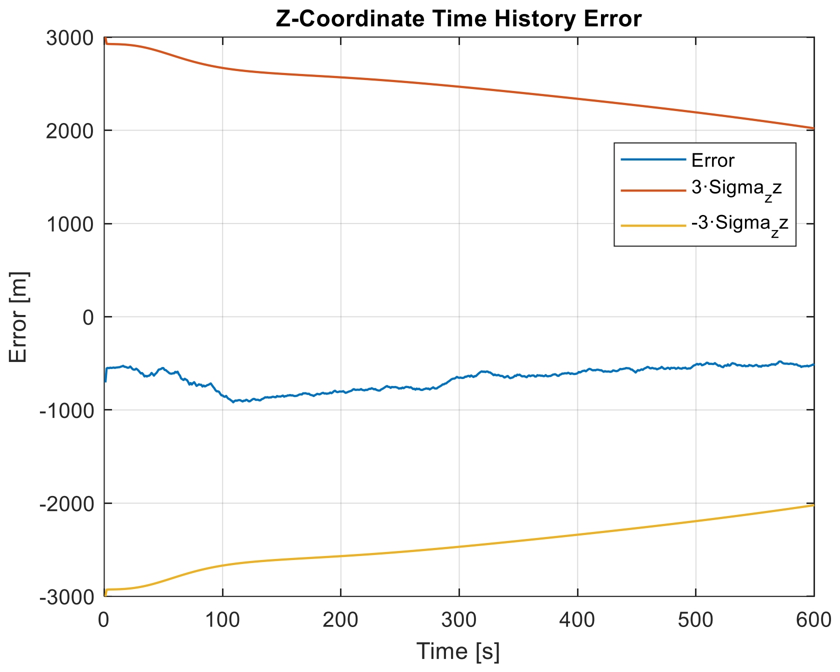


Fig. 8.12. Z-Coordinate Error Time History from a Real Satellite Signal

From the previous results, it is observed that the error produced by the EKF on each coordinate of the receiver's estimated position is not able to able to converge to zero. As it was expected, the Doppler estimates that the receiver produced from the real satellite signal were not enough accurate and thus the estimated position carries out larger errors. Furthermore, the geometry obtained with only one satellite is not enough for obtaining an accurate navigation solution.

In this way, the following table gathers the estimation errors produced on each coordinate as well as the final estimated location of the receiver:

Table 8.5. Estimation Errors and Receiver's Position Estimate

ε_x	ε_y	ε_z
155.1889 m	309.2010 m	521.4482 m
$\hat{x}_{rec}$	$\hat{y}_{rec}$	$\hat{z}_{rec}$
-2481952.38560240 m	-4699871.85590503 m	3513077.95074485 m

And the total error on the estimated receiver's position vector with respect to the true receiver's location is computed as:

$$\varepsilon_{Total} = ||\hat{r}_{rec} - r_{rec})|| = 625.7772\, m \tag{8.13}$$

Moreover, as it occurred in the simulation environment, the position estimation error is lower in the x, y plane since it is easier for the receiver to obtain a good geometry for estimating the $[x, y]$ coordinates. Finally, the following figure shows the true and the final estimate positions of the receiver after having implemented the EKF on the Doppler estimates made to the real satellite signal. While the true position is located on the roof top of the Engineering Gateway Building of UCI, the estimated receiver's position is off by several meters as it was previously discussed.

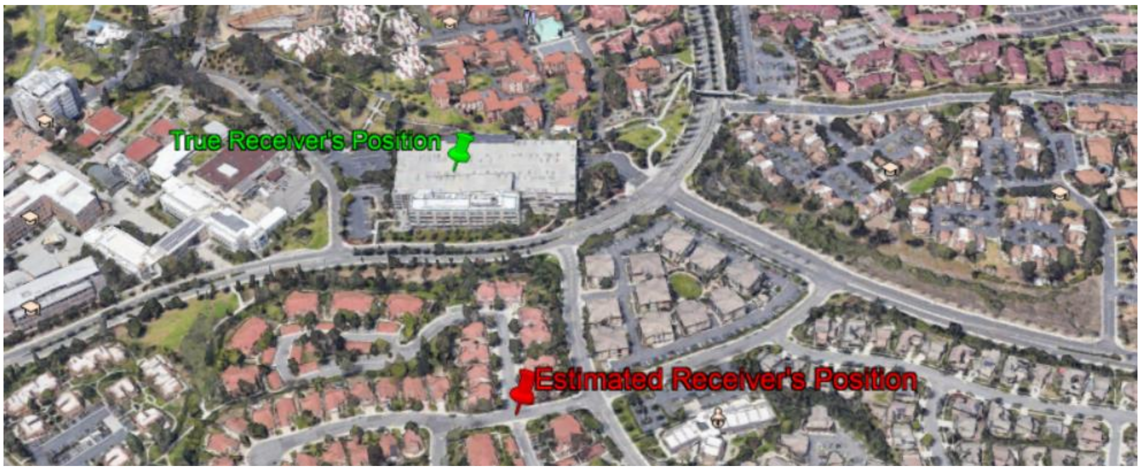


Fig. 8.13. True vs Estimated Receiver's Location obtained from a satellite signal

8.3. Improvements Proposal

The results obtained from using a real Iridium Next satellite signal have been highly useful since several research work might arise from its interpretation. In this way, the following aspects have been identified to be susceptible of improvement in order to obtain a more accurate navigation solution:

- The receiver should be improved in order to account for variable frame and tone durations, as well as for considering different frame topologies
- The experimental set up should be adapted in order to ensure that at least two satellites are visible from the receiver's location, allowing for a better geometry that would improve the final navigation solution
- The EKF model initialization parameters should be assessed in order to improve the error convergence velocity and thus the estimated receiver's position

CHAPTER 9. CONCLUSIONS

In view of the results, it can be concluded that the proposed receiver has been able to track the Doppler frequency from the incoming signal in the simulation environment. It was observed that the absolute error between the true Doppler and the estimate produced by the receiver is less than 0.4Hz for the entire simulation. Moreover, the phase error committed by the receiver is less than 0.1rads.

In the same way, the navigation solution obtained from the Doppler tracking in simulation has shown promising results. Not only the total error committed in the receiver's position estimate is roughly 37m, but also the proposed model for the EKF introduces an error of less than 18m for the x, y coordinates.

On the other hand, the Doppler frequency tracking from a real satellite signal proved a good performance of the receiver. Although the phase error ended up being higher (0.2rads) comparing to the simulation environment, the receiver demonstrated to be able to track the Doppler frequency without straying to far from the expected profile predicted from the TLE files.

Unfortunately, the navigation solution obtained from these Doppler measurements in a real scenario committed a total error of approximately 625m in the receiver's position. Nevertheless, this can be explained since the Doppler measurements produced by the receiver were not enough accurate and, moreover, the navigation solution was obtained by only using one satellite, which is not enough for the receiver to have good satellite geometry.

Finally, given the scope of the present thesis and the initial goals that were proposed in chapter 1, it can be concluded that all the planned project stages were successfully completed, and the proposed objectives were finally achieved.

9.1. Future Work

The work gathered in this document aims to serve as the beginning of a major research project that will be carried out for several more months. Particularly, the receiver will be improved so it can account for variable frame lengths, as well as for different frame topologies. In this way, it is expected that the achieved Doppler tracking performance remains closer to the simulation environment.

Besides, receiver will include an additional altimeter that will allow to have extra measurements on the z-coordinate so that the error committed in the receiver's position estimate over this coordinate is reduced. In the same way, the EKF will improve its performance if more empirical information is included in the model.

Finally, it is expected to include Doppler and altitude measurements from three satellites in order to reduce the total error of the receiver's position estimate.

REFERENCES

[1] P. Misra, P. Enge, "Global Positioning System: Signals, Measurements and Performance", 2001.

[2] Z. Kassas, "Collaborative Opportunistic Navigation, "*IEEE Aerospace and Electronic Systems Magazine ,* vol. 28, no. 6, pp. 38-43, 2013.

[3] Z. Kassas, J. Khalife, "Opportunistic UAV navigation with carrier phase measurements from asynchronous cellular signals," in *IEEE Transactions on Aerospace and Electronic Systems*, 2019.

[4] L. Xiao, Z. Lei, "Analysis of iridium-augmented GPS positioning performance," in *IET International Radar Conference* , 2018.

[5] H. Benzerrouk, Q. Nguyen, F. Xiaoxing, A. Amrhar, A. V. Nebylov and R. Landry, "Alternative PNT based on Iridium Next LEO Satellites Doppler/INS Integrated Navigation System," in *IEEE ICINS*, 2019.

[6] Z. Tan, H. Qin, L. Cong and C. Zhao, "New Method for Positioning Using IRIDIUM Satellite Signals of Opportunity," in *IEEE Access*, 2019.

[7] Z. Tan, H. Qin, L. Cong and C. Zhao, "Positioning Using IRIDIUM Satellite Signals of Opportunity in Weak Signal Environment," in *Electronics*, 2019.

[8] M. Leng, L. Lei and S.Razul, "Joint Synchronization and Localization using Iridium Ring Alert Signal," in *IEEE ICICS*, 2015.

[9] Iridium Communications Inc, "Iridium Next Engineering Statement," 2008.

[10] J. Coon, M. Sandell, M. Beach and J. McGeehan, "Channel and Noise Variance Estimation and Tracking Algorithms for Unique-Word Based Single-Carrier Systems," in *IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS*, 2006.

[11] J. Khalife, Z. Kassas, "Receiver Design For Doppler Positioning With LEO Satellites," in *IEEE ICASSP*, 2019.

[12] X. Gao, Y. Li and J. Bao, "Efficient Carrier Acquisition and Tracking for High Dynamic and Weak Satellite Signals," in *Journal of Communications*, 2016.

[13] Z. Kowalczuk, "On Discretization of Continuous-Time State-Sapce Models," in *IEEE Transactions on Circuits and Systems* , 1991.

[14] G. M. Kowalski, "Satellite Data Communications Link Requirements for a Proposed Flight Simulation System", 1994.

[15] J. Olmos, "Ground Track of GPS Satellites", EETAC, 2018.

[16] K. Osen, "Accurate Conversion of Earth-Fixed Earth-Centered Coordinates to Geodetic Coordinates", HAL, 2017.

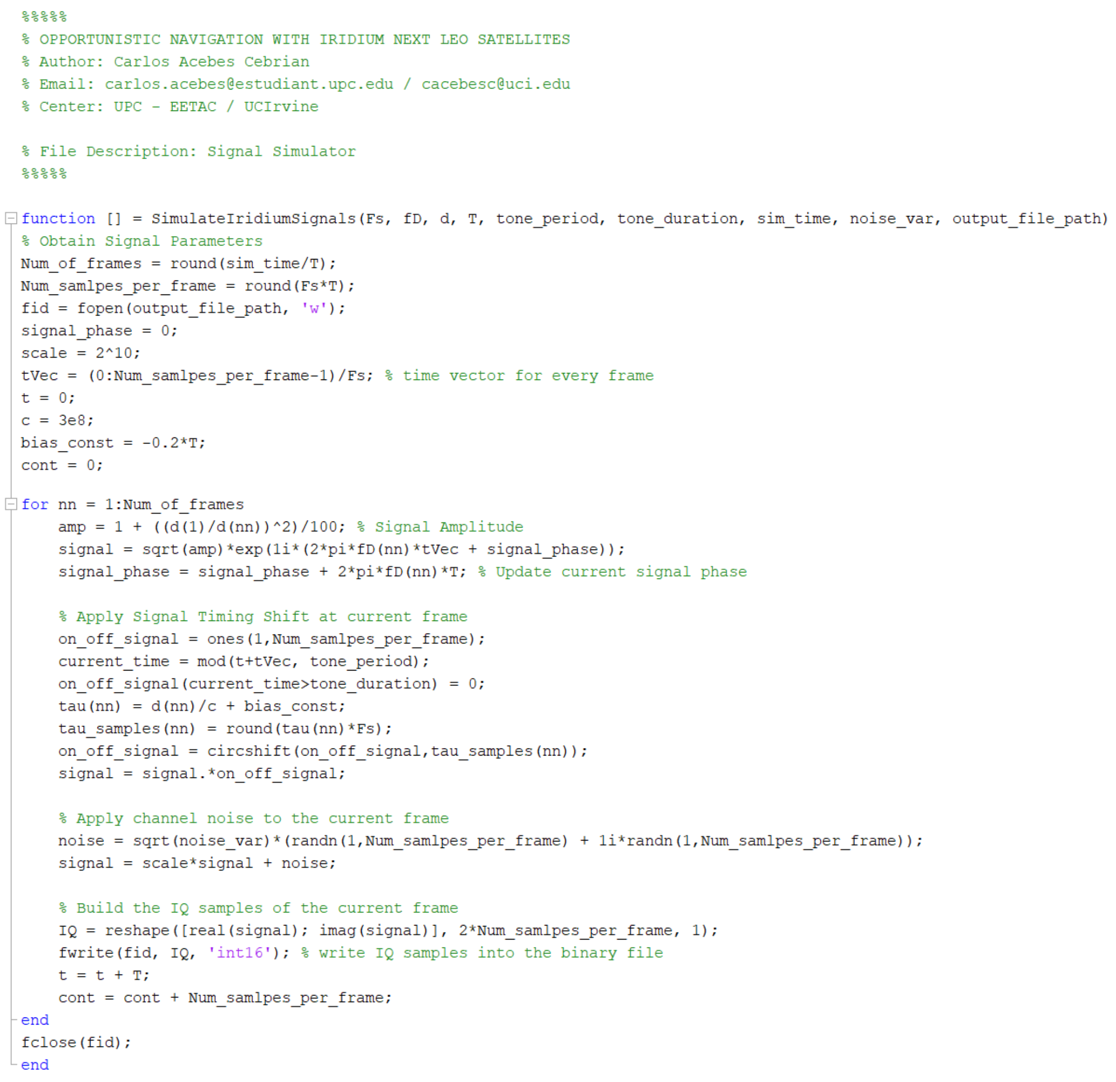
[17] AT1621-12 Datasheet, "https://satellitephonestore.com/sites/default/files/at1621-12_g.pdf," 2010.

[18] USRP-2954 Datasheet, "https://www.ni.com/pdf/manuals/375725c.pdf," 2017.

[19] Y. Kim, H. Bang, "Introduction to Kalman Filter and its Applications," 2018.

APPENDIX A. SIGNAL SIMULATOR MATLAB CODES

A.1. Signal Simulator



```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Signal Simulator
%%%%%

function [] = SimulateIridiumSignals(Fs, fD, d, T, tone_period, tone_duration, sim_time, noise_var, output_file_path)
% Obtain Signal Parameters
Num_of_frames = round(sim_time/T);
Num_samlpes_per_frame = round(Fs*T);
fid = fopen(output_file_path, 'w');
signal_phase = 0;
scale = 2^10;
tVec = (0:Num_samlpes_per_frame-1)/Fs; % time vector for every frame
t = 0;
c = 3e8;
bias_const = -0.2*T;
cont = 0;

for nn = 1:Num_of_frames
    amp = 1 + ((d(1)/d(nn))^2)/100; % Signal Amplitude
    signal = sqrt(amp)*exp(1i*(2*pi*fD(nn)*tVec + signal_phase));
    signal_phase = signal_phase + 2*pi*fD(nn)*T; % Update current signal phase

    % Apply Signal Timing Shift at current frame
    on_off_signal = ones(1,Num_samlpes_per_frame);
    current_time = mod(t+tVec, tone_period);
    on_off_signal(current_time>tone_duration) = 0;
    tau(nn) = d(nn)/c + bias_const;
    tau_samples(nn) = round(tau(nn)*Fs);
    on_off_signal = circshift(on_off_signal,tau_samples(nn));
    signal = signal.*on_off_signal;

    % Apply channel noise to the current frame
    noise = sqrt(noise_var)*(randn(1,Num_samlpes_per_frame) + 1i*randn(1,Num_samlpes_per_frame));
    signal = scale*signal + noise;

    % Build the IQ samples of the current frame
    IQ = reshape([real(signal); imag(signal)], 2*Num_samlpes_per_frame, 1);
    fwrite(fid, IQ, 'int16'); % write IQ samples into the binary file
    t = t + T;
    cont = cont + Num_samlpes_per_frame;
end
fclose(fid);
end
```

A.2. Main of Signal Simulator

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Main Signal Simulator
%%%%%

clear all
clc
close all
fclose('all');

Fs = 80e3*3; % Hz (Sampling Frequency)
fc = 1626e6; % Hz (carrier frequency)
c = 3e8; % m/s
output_file_path = 'IQ_Samples.bin';
sim_time = 10*60; %s
T = 6.8e-3; % s/frame
Num_frames = round(sim_time/T);
tone_period = 6.8e-3; %s
tone_duration = 2.6e-3; % s (Actual value 2.6e-3 s)
noise_var = (400*0.4)^2;

load('d_fD_tVec.mat');
SimulateIridiumSignals(Fs, fD, d, T, tone_period, tone_duration, sim_time, noise_var, output_file_path);
figure; plot(tVec,d)
figure; plot(tVec,fD)
```

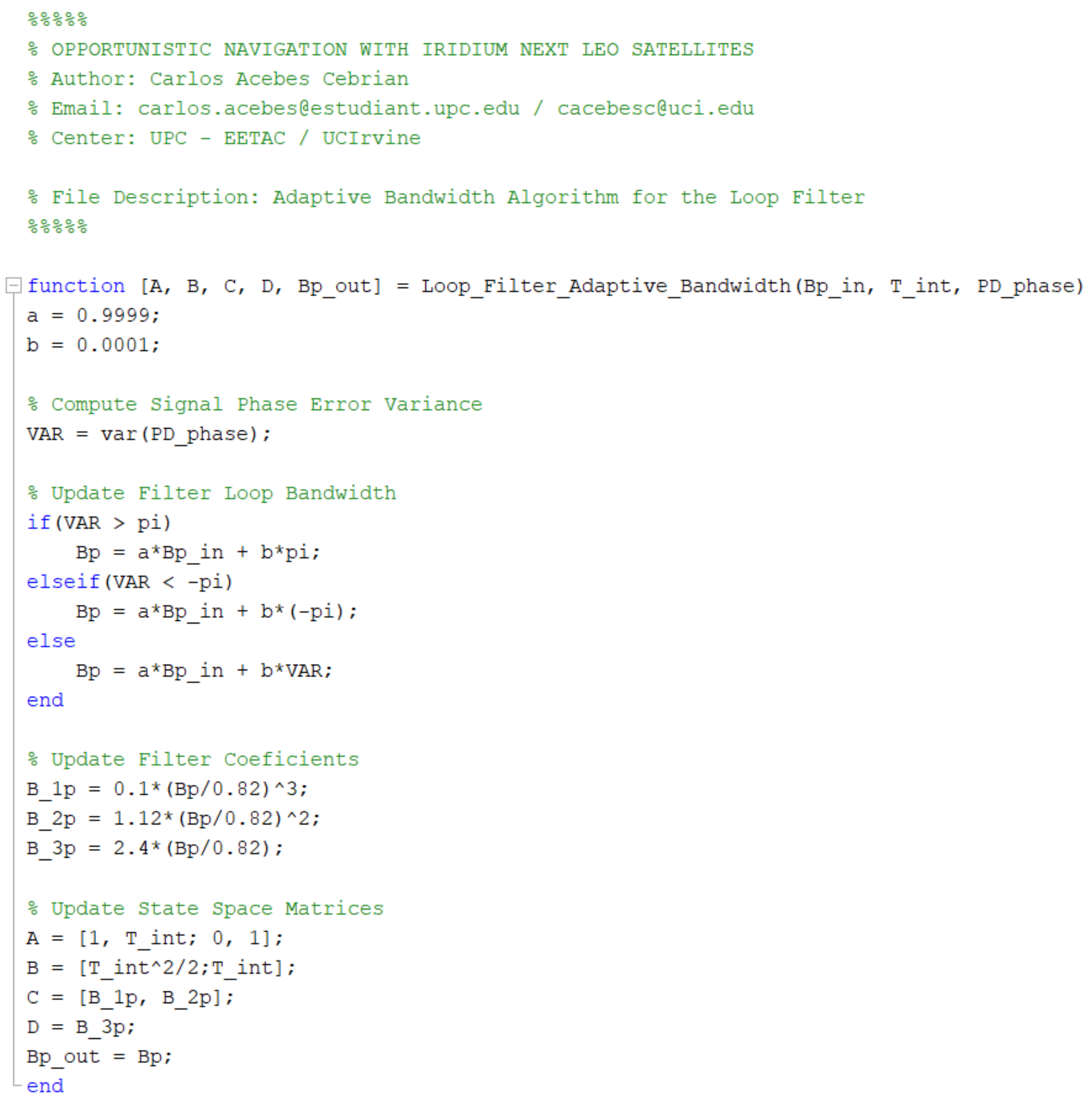
APPENDIX B. RECEIVER'S MATLAB CODES

B.1. Iridium Next Signals Receiver

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Iridium Next Signals Receiver
%%%%%

clear all
clc
close all
fclose('all');

output_file_path = 'IQ_Samples.bin';
Fs = 80e3*3; %Hz
fc = 1626e6; % Hz (carrier frequency)
test_duration = 0.1; %s
test_skip = 20; %s

% Get IQ samples from the binary file
fid = fopen(output_file_path, 'r');
status = fseek(fid, round(test_skip*Fs*4), 'bof');
IQ = fread(fid, [2, round(test_duration*Fs)], 'int16');
I_samples = IQ(1,:);
Q_samples = IQ(2,:);
IQ = I_samples + 1i*Q_samples;

% Signal Parameters Definition
T_int = 6.8e-3; % s/frame
tone_duration = 2.6e-3; % s
tone_period = 6.8e-3; %s
Num_samlpes_per_frame = round(Fs*T_int);
Num_samples_tone_duration = round(Fs*tone_duration);
sim_time = 10*60; % s
T_frames = 6.8e-3; % s/frame
Num_frames = floor((sim_time-test_duration-test_skip)/T_frames);

% Find the initial Doppler Shift Estimate
[pxx, f] = pwelch(IQ, 500, 50, 2^15, Fs);
[pxx_max, index] = max(pxx);
fD_init_estim = f(index);

% PLL Initial Parameters
Out_fD(1) = fD_init_estim; % Doppler shift initial estimate
Out_phase(1) = 0; % Signal phase initial estimate
tVec_samples = (0:Num_samlpes_per_frame-1)/Fs; % time vector for every frame
tVec_frames = (0:Num_frames-1)*T_int; % time vector for all frames
```

```
% PLL loop filter Initialization
N_percentage = 0.1;
N_phase_error = floor((N_percentage/100)*Num_frames);
PD_phase(1) = 0;
Bp_initial = 13;
[A, B, C, D, Bp_out] = Loop_Filter_Adaptive_Bandwidth(Bp_initial, T_int, PD_phase);

% Loop Filter Initialization (Steady state initial condition)
I = eye(size(A));
H = [(A-I);C];
b = [0;0;Out_fD(1)*2*pi];
X(:,1) = H\b;

% Signal Timing Shift Tracking
N_window = Num_samples_tone_duration;
d_0 = 2.2533e+06;
N_pulse_start = (d_0/3e8)*Fs;
N_pulse_start = mod(round(N_pulse_start),Num_samlpes_per_frame);

% PLL Tracking Loop
for n = 2:Num_frames
    % Generate wipe off sine wave
    signal = exp(-1i*(2*pi*Out_fD(n-1)*tVec_samples + Out_phase(n-1)));
    Current_Frame_Raw = fread(fid, [2, round(Num_samlpes_per_frame)], 'int16');
    for nn=1:Num_samlpes_per_frame
        Current_Frame(nn) = Current_Frame_Raw(1,nn) + 1i*Current_Frame_Raw(2,nn);
    end
    % multiply the current frame by the wipe off sine signal
    signal = signal.*Current_Frame;

    % Signal Timing Shift Tracking
    if(mod(n,(1/T_int)) == 0)
        i_slide = 0;
        for in=1:(Num_samlpes_per_frame-N_window+1)
            P_avg(in) = (1/N_window)*sum(abs(signal(in:N_window+i_slide)).^2);
            i_slide = i_slide + 1;
        end
        [P_max,N_pulse_start] = max(P_avg);
        N_pulse_start = mod(round(N_pulse_start),Num_samlpes_per_frame);
    end
    signal = signal(N_pulse_start:N_pulse_start+Num_samples_tone_duration-1);

    Int_signal = mean(signal); % integrate the resulting signal
    I_message(n) = real(Int_signal); % Store the transmitted message
    Q_message(n) = imag(Int_signal);
    PD_phase(n) = atan2(imag(Int_signal), real(Int_signal)); % Find the current frame phase

    % Discretized State-Space Model of the Loop Filter
    X(:,n)  = A*X(:,n-1) + B*PD_phase(n);
    Out_FL(n-1) = C*X(:,n-1) + D*PD_phase(n);
    Out_fD(n) = Out_FL(n-1)/(2*pi); % Current Doppler shift
    Out_phase(n) = 2*pi*Out_fD(n-1)*T_int + Out_phase(n-1); %  current frame and next frame

    % Discretized State-Space Model of the Loop Filter
    X(:,n)  = A*X(:,n-1) + B*PD_phase(n);
    Out_FL(n-1) = C*X(:,n-1) + D*PD_phase(n);
    Out_fD(n) = Out_FL(n-1)/(2*pi); % Current Doppler shift
    Out_phase(n) = 2*pi*Out_fD(n-1)*T_int + Out_phase(n-1); %  current frame and next frame

    if(mod(n,N_phase_error) == 0)
        [A, B, C, D, Bp_out] = Loop_Filter_Adaptive_Bandwidth(Bp_out, T_int, PD_phase(n-N_phase_error+1:n));
    end

    % Update the pulse start of the next frame
    N_pulse_start = N_pulse_start - (Out_fD(n-1)/fc)*tone_period*Fs;
    N_pulse_start = mod(round(N_pulse_start),Num_samlpes_per_frame);
end

fclose(fid);
```

B.2. Adaptive Bandwidth Algorithm for the Loop Filter



```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Adaptive Bandwidth Algorithm for the Loop Filter
%%%%%

function [A, B, C, D, Bp_out] = Loop_Filter_Adaptive_Bandwidth(Bp_in, T_int, PD_phase)
a = 0.9999;
b = 0.0001;

% Compute Signal Phase Error Variance
VAR = var(PD_phase);

% Update Filter Loop Bandwidth
if(VAR > pi)
    Bp = a*Bp_in + b*pi;
elseif(VAR < -pi)
    Bp = a*Bp_in + b*(-pi);
else
    Bp = a*Bp_in + b*VAR;
end

% Update Filter Coeficients
B_1p = 0.1*(Bp/0.82)^3;
B_2p = 1.12*(Bp/0.82)^2;
B_3p = 2.4*(Bp/0.82);

% Update State Space Matrices
A = [1, T_int; 0, 1];
B = [T_int^2/2;T_int];
C = [B_1p, B_2p];
D = B_3p;
Bp_out = Bp;
end
```

APPENDIX C. ORBITS PREDICTOR MATLAB CODES

C.1. Main of the Satellite Orbits Predictor

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Satellite Orbits Predictor
%%%%%

clear all
clc
close all
fclose('all');

% Define the current time (UTC = PST + 7h)
month = 6;
day = 30;
hour = 4;
mins = 15;
secs = 11;

% Express the current time in seconds from the start of the Epoch
current_time = floor(mmddyyyy2Fractional(month, day, hour, mins, secs)*24*3600);

% Define the simulation duration
Sim_time = 100.4; % minutes

file = 'IridiumNext_15_May_2020.txt';
[Eph] = Read_TLE_file(file);

 Select_Sat = 3; % Select a satelite to plot its Ground Track
 ID_sat = Eph(Select_Sat,1); % Satellite ID
 Num_sat = Eph(Select_Sat,2); % Satellite number
 Epoch = Eph(Select_Sat,3); % Get the Epoch
 Epoch = floor(Epoch*24*3600); % TLE reference time [s]

 % Define Keplerian Parameters for the selected satellite
 i_0 = Eph(Select_Sat,4); % Orbit inclination [rad]
 RAAN = Eph(Select_Sat,5); % Right Ascension of the Ascending Node (RAAN) [rad]
 e = Eph(Select_Sat,6); % Orbit eccentricity
 w = Eph(Select_Sat,7); % Argument of the perigee at time ToA (rad)
 M = Eph(Select_Sat,8); % Mean Anomaly at time ToA [rad]
 n = Eph(Select_Sat,9); % Mean Motion [rad/s]
 a = Eph(Select_Sat,10); % Major Semi Axis [m]
 change_RA = Eph(Select_Sat,11); % Rate of change of RAAN [rad/s]
```

```
% Compute LLA Coordinates every 5 seconds from current time
ii = 1;
latitude = [];
longitude = [];
for t = current_time:5:current_time+(Sim_time*60)
    [x,y,z] = Compute_ECEF_Coordinates(t, Epoch, a, n, e, i_0, RAAN, change_RA, w, M);
    [lat,lon,h] = ECEF_to_LLA(x,y,z);
    latitude(ii) = lat;
    longitude(ii) = lon;
    ii = ii + 1;
end

%Plot the Earth Map on Figure 1 from a point file in .txt format
figure();
hold on
data = importdata('world_110m.txt');
world_x = data(:,1);
world_y = data(:,2);
scatter(world_x,world_y,8,'blue','diamond','filled')

%Plot the Ground Track
scatter(longitude,latitude,8,'red','diamond','filled')
text(longitude(end) + 2, latitude(end) + 1, sprintf('%d', ID_sat));
scatter(longitude(end),latitude(end),50,'red','o');
hold off;

%Define axis legend for figure 1
axis([-180 180 -90 90])
grid on;
xlabel('Longitude (°)');
ylabel('Latitude (°)');
stringTitle = sprintf('Iridium Next Satellite %d ground track', ID_sat);
title(stringTitle);

%Plot the Earth Map on Figure 2 from a point file in .txt format
figure();
hold on;
data = importdata('world_110m.txt');
world_x = data(:,1);
world_y = data(:,2);
scatter(world_x,world_y,8,'blue','diamond','filled')

num_Sats = size(Eph,1); % Get the number of satelites in Eph
nn = 1;
latitude = [];
longitude = [];
colour = 1;
for ID = 1:1:num_Sats
    % Define the Keplerian Parameters for each satelite
    Epoch = Eph(ID,3); % Get the Epoch of the Ephemeris
    Epoch = floor(Epoch*24*3600); % TLE reference time [s]
    i_0 = Eph(ID,4); % Orbit inclination [rad]
    RAAN = Eph(ID,5); % Right Ascension of the Ascending Node (RAAN) [rad]
    e = Eph(ID,6); % Orbit eccentricity
    w = Eph(ID,7); % Argument of the perigee at time ToA (rad)
    M = Eph(ID,8); % Mean Anomaly at time ToA [rad]
    n = Eph(ID,9); % Mean Motion [rad/s]
    a = Eph(ID,10); % Major Semi Axis [m]
    change_RA = Eph(ID,11); % Rate of change of RAAN [rad/s]
    % Compute LLA Coordinates every 15 seconds from current time
    for i = current_time:15:(current_time+8*60)
        [x,y,z] = Compute_ECEF_Coordinates(i, Epoch, a, n, e, i_0, RAAN, change_RA, w, M);
        [lat,lon,h] = ECEF_to_LLA(x,y,z);
        latitude(nn) = lat;
        longitude(nn) = lon;
        nn = nn + 1;
    end

    switch(colour)
        case 1
            stringColor = 'red';
            colour = colour + 1;
        case 2
            stringColor = 'green';
            colour = colour + 1;
        case 3
            stringColor = 'cyan';
            colour = colour + 1;
        case 4
            stringColor = 'magenta';
            colour = 1;
    end

    %Plot Ground track for the next hour for each satelite
    scatter(longitude, latitude, 8, stringColor, 'diamond', 'filled')
    text(longitude(end) + 2, latitude(end) + 1, sprintf('%d', Eph(ID, 1)));
    scatter(longitude(end), latitude(end), 50, stringColor,'o');

    %Reset variables
    nn = 1;
    latitude = [];
    longitude = [];
end
hold off;

%Define axis legend for Figure 2
axis([-180 180 -90 90])
grid on;
xlabel('Longitude (°)');
ylabel('Latitude (°)');
title('Map of all Iridium Next Satellite ground tracks');
```

C.2. ECEF Coordinates Computer

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: ECEF Coordinates Computer
%%%%%

function[x,y,z] = Compute_ECEF_Coordinates(current_time, Epoch, a, n, e, i_0, RAAN, change_RA, w, M)

    % Get the time between the Epoch and the current time
    t = current_time;
    t_k = t - Epoch;

    % Compute the Mean Anomaly
    M_k = M + n*t_k;

    % Find the eccentric anomaly
    E_k_p=[];
    E_k_p(1)  = M_k;
    E_k_p(end+1)  = M_k+e*sin(E_k_p(end));
    while(abs(E_k_p(end)-E_k_p(end-1))>1e-8)
        E_k_p(end+1)  = M_k+e*sin(E_k_p(end));
    end
    E_k = E_k_p(end);

    % Find the true anomaly
    Img = (sqrt(1-e^2)*sin(E_k))/(1-e*cos(E_k));
    Real = (cos(E_k)-e)/(1-e*cos(E_k));
    Cmplx_Aux = complex(Real,Img);
    v_k = angle(Cmplx_Aux);

    % Find the Argument of Latitude
    u_k = v_k+w;

    % Find the Orbit Radius
    r_k = a*(1-e*cos(E_k));

    % Find the current Longitude of the ascending node
    Ang_speed_Earth = 7.2921151467e-5; %angular speed of the Earth rotation
    Lon_AN_k = RAAN + change_RA*t_k - Ang_speed_Earth*t;

    % Get x coordinate within the orbit plane
    x_p = r_k*cos(u_k);

    % Get y coordinate within the orbit plane
    y_p = r_k*sin(u_k);

    % FINAL ECEF Coordinates
    x = x_p*cos(Lon_AN_k)-y_p*cos(i_0)*sin(Lon_AN_k);
    y = x_p*sin(Lon_AN_k)+y_p*cos(i_0)*cos(Lon_AN_k);
    z = y_p*sin(i_0);
end
```

C.3. LLA Coordinates Computer

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: LLA Coordinates Computer
%%%%%

function[lat,lon,h] = ECEF_to_LLA(x,y,z)

    a = 6378137; %Major axis of the Earth orbit
    e = sqrt(0.00669437999014); %Earth Orbit
    c = e*a;
    b = sqrt(a^2 - c^2);
    e_p = c/b;

    %Step 1:
    r = sqrt(x^2+y^2);

    %Step 2:
    F = 54*b^2*z^2;

    %Step 3:
    G = r^2 + (1-e^2)*z^2-e^2*c^2;

    %Step 4:
    C = (e^4*F*r^2)/(G^3);

    %Step 5:
    s = (1+C+sqrt(C^2+2*C))^(1/3);

    %Step 6:
    P = F/(3*(s+1/s+1)^2*G^2);

    %Step 7:
    Q = sqrt(1+2*e^4*P);

    %Step 8:
    r_0 = -P*e^2*r/(1+Q)+sqrt(0.5*a^2*(1+1/Q)-(P*(1-e^2)*z^2)/(Q*(1+Q))-0.5*P*r^2);

    %Step 9:
    U = sqrt(z^2+(r-r_0*e^2)^2);

    %Step 10:
    V = sqrt(z^2*(1-e^2)+(r-r_0*e^2)^2);

    %Step 11:
    z_0 = b^2*z/(a*V);

    % FINAL LLA Coordinates
    h = U*(1-b^2/(a*V));
    lat = atan2(z+e_p^2*z_0,r)*180/pi;
    lon = atan2(y,x)*180/pi;
end
```

C.4. TLE Files Reader

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: TLE Files Reader
%%%%%

function [Eph] = Read_TLE_file(file)
fid = fopen(file);

line_0 = fgetl(fid);
line_1 = fgetl(fid);
line_2 = fgetl(fid);

% Constants
G = 6.67384e-11; % Gravitational constant
M = 5.972e24; % Earth Mass
u = G*M; % Standard gravitational parameter [m3/s2]
J_2 = 1.08262668e-3; % Earth Spaceflight constant
R_e = 6378137; % Earth Radius [m]

i = 1;
while ischar(line_2)
    Eph(i,1) = str2double(line_0(9:11)); % Satellite ID
    Eph(i,2) = str2double(line_1(3:7)); % Satellite number
    Eph(i,3) = str2double(line_1(21:32)); % Epoch of the Ephemeris
    Eph(i,4) = str2double(line_2(9:16))*(pi/180); % Orbit inclination [rad]
    Eph(i,5) = str2double(line_2(18:25))*(pi/180); % Right Ascension of the Ascending Node (RAAN) [rad]
    Eph(i,6) = str2double(line_2(27:33))/1e7; % Orbit eccentricity
    Eph(i,7) = str2double(line_2(35:42))*(pi/180); % Argument of the perigee [rad]
    Eph(i,8) = str2double(line_2(44:51))*(pi/180); % Mean Anomaly [rad]
    Eph(i,9) = str2double(line_2(53:63))*((2*pi)/86400); % Mean Motion [rad/s]
    Eph(i,10) = (u^(1/3))/(Eph(i,9)^(2/3)); % Major Semi Axis [m]
    Eph(i,11) = -1.5*J_2*(R_e/(Eph(i,10)*(1-Eph(i,6)^2)))^2*Eph(i,9)*cos(Eph(i,4)); % Rate of RAAN [rad/s]

    line_0 = fgetl(fid);
    line_1 = fgetl(fid);
    line_2 = fgetl(fid);
    i = i + 1;
end

fclose(fid);
end
```

C.5. Fractional Day Converter

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Fractional Day Converter
%%%%%

function [fract_time] = mmddyyyy2Fractional(month, day, hour, minutes, seconds)

for i=1:12 % Set up the array of days for each month of the year
    Day_months(i) = 31;
    if i == 2
        Day_months(i)= 29;
    end
    if i == 4 || i == 6 || i == 9 || i == 11
        Day_months(i)= 30;
    end
end

% Find the day of the Epoch
if(month == 1)
    int_time = day;
elseif(month == 2)
    int_time = 31 + day;
else
    int_time = 31;
    for n=2:month-1
        int_time = int_time + Day_months(n);
    end
    int_time = int_time + day;
end

% Compute Fractional Time
fract_time = seconds/60 + minutes;
fract_time = fract_time/60 + hour;
fract_time = fract_time/24;
fract_time = fract_time + int_time;
end
```

APPENDIX D. EKF MATLAB CODES

D.1. Main EKF

```matlab
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Extended Kalman Filter
%%%%%

clear all
clc
close all
fclose('all');

% LLA Receiver's Location [33.643301; -117.837981; 40]
r_rec = [-2482107.57446003 -4700181.05692380 3513599.39891489];

load('Sat_1_positions.mat')
load('Sat_1_velocities.mat')
r_sat_1 = rs; % Satellite Positions
v_sat_1 = vs; % Satellite Velocities
d_sat_1 = 5*rand(1); % Satellite clock drift

load('Sat_2_positions.mat')
load('Sat_2_velocities.mat')
r_sat_2 = rs; % Satellite Positions
v_sat_2 = vs; % Satellite Velocities
d_sat_2 = 4*rand(1); % Satellite clock drift


F = eye(5);
P_0_elements = [1e7 1e7 1e7 10 20]; % std of estimarion error
P_0 = diag(P_0_elements); % Estimation error covariance
e = 1e-12;
Q = e*eye(5); % Process noise covariance
R_elements = [0.1^2 0.2^2]; % measurement noise covariance
R = diag(R_elements); % measurement noise covariance

x = [r_rec d_sat_1 d_sat_2].'; % States Matrix
x0 = mvnrnd(x, P_0);

x_kk = x0.';
P_kk = P_0;
save_x = x_kk;
save_P = diag(P_0);
N = length(r_sat_1); % Number of iterations = number of measurements
for k=1:N-1
    % Time-Update Equations
    x_k1k = F*x_kk;
    P_k1k = F*P_kk*F + Q;

    z_k1_sat = Get_z(x, r_sat_1(:,k+1), v_sat_1(:,k+1), r_sat_2(:,k+1), v_sat_2(:,k+1), R);
    z_k1_hat = Get_zhat(x_k1k, r_sat_1(:,k+1), v_sat_1(:,k+1), r_sat_2(:,k+1), v_sat_2(:,k+1));
    V_k1 = z_k1_sat - z_k1_hat; % Innovation Vector

    H = Get_H(x_k1k, r_sat_1(:,k+1), v_sat_1(:,k+1), r_sat_2(:,k+1), v_sat_2(:,k+1)); % Measurement Jacobian
    S_k1 = H*P_k1k*H.' + R;
    K_k1 = P_k1k*H.'*inv(S_k1);

    % Update Equations
    x_k1k1 = x_k1k + K_k1*V_k1;
    P_k1k1 = (eye(5) - K_k1*H)*P_k1k;
    P_k1k1 = (P_k1k1 + P_k1k1')/2;
    x_kk = x_k1k1;
    P_kk = P_k1k1;

    save_P(:,k+1) = diag(P_kk);
    save_x(:,k+1) = x_kk;

end
```

D.2. Measurements Computer

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Measurements Computer
%%%%%

function [z] = Get_z(x, r_sat1, v_sat1, r_sat2, v_sat2, R)

r_rec = x(1:3); % Estimated position of the receiver
d_sat1 = x(4); % Estimated clock drift of the satellites
d_sat2 = x(5);
R_sat1 = R(1,1); % Measurement noise covariances
R_sat2 = R(2,2);
z_sat1 = (v_sat1.'*(r_rec - r_sat1))/norm(r_rec - r_sat1) + d_sat1 + sqrt(R_sat1)*rand(1);
z_sat2 = (v_sat2.'*(r_rec - r_sat2))/norm(r_rec - r_sat2) + d_sat2 + sqrt(R_sat2)*rand(1);

z = [z_sat1 z_sat2].';
end
```

D.3. Measurement Estimates Computer

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Measurement Estimates Computer
%%%%%

function [zhat] = Get_zhat(xhat, r_sat1, v_sat1, r_sat2, v_sat2)

r_rec_e = xhat(1:3); % Estimated position of the receiver
zhat_sat1 = (v_sat1.'*(r_rec_e - r_sat1))/norm(r_rec_e - r_sat1) + xhat(4);
zhat_sat2 = (v_sat2.'*(r_rec_e - r_sat2))/norm(r_rec_e - r_sat2) + xhat(5);
zhat = [zhat_sat1 zhat_sat2].';

end
```

D.4. Measurement Jacobian Computer

```
%%%%%
% OPPORTUNISTIC NAVIGATION WITH IRIDIUM NEXT LEO SATELLITES
% Author: Carlos Acebes Cebrian
% Email: carlos.acebes@estudiant.upc.edu / cacebesc@uci.edu
% Center: UPC - EETAC / UCIrvine

% File Description: Measurement Jacobian Computer
%%%%%

function [H] = Get_H(xhat, r_sat1, v_sat1, r_sat2, v_sat2)

r_k1k = xhat(1:3); % Estimated position of the receiver
dh_dr_sat1 = (v_sat1)/norm(r_k1k - r_sat1) - (r_k1k - r_sat1)*((v_sat1.'*(r_k1k - r_sat1))/norm(r_k1k - r_sat1)^3);
dh_dr_sat2 = (v_sat2)/norm(r_k1k - r_sat2) - (r_k1k - r_sat2)*((v_sat2.'*(r_k1k - r_sat2))/norm(r_k1k - r_sat2)^3);
H = [dh_dr_sat1.' 1 0;dh_dr_sat2.' 0 1];

end
```