UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

TREBALL DE FI DE GRAU

TÍTOL DEL TFG: GNSS Signal Spoofing Detection

TITULACIÓ: Double bachelor's degree in Aerospace Systems Engineering and Telecommunications Systems Engineering

AUTORS: Eric Sánchez González

DIRECTOR: Luis Esteve Elfau

DATA: October 24, 2022

Títol: Detecció de suplantació d'identitat en senyals GNSS
Autors: Eric Sánchez González

Director: Luis Esteve Elfau

Data: 24 d'octubre de 2022

Resum

En aquesta tesi es treballa la implementació d'algunes tècniques de detecció de suplantació d'identitat per a senyals Global Positioning Systems (GPS) L1 C/A, tema que està d'actualitat en la comunitat Global Navigation Satellite System (GNSS). L'interès d'aquest tema recau en el fet que, actualment, hi ha un gran nombre d'aplicacions que depenen de les comunicacion basades en GNSS. A més, el caràcter públic dels detalls i especificacions dels senyals han exposat les comunicacions GNSS a agents de suplantació d'identitat. Aquests, amb uns equips relativament econòmics, són capaços de controlar els mòduls de seguiment d'un receptor víctima i manipular la seva solució de temps o navegació.
Davant d'aquesta problemàtica, aquest projecte pretén contribuir a la comunitat implementant, al reconegut codi receptor GNSS de Borre, i validant algunes tècniques de detecció de suplantació d'identitat. Per fer-ho, el projecte s'organitza en tres grans apartats; l'estudi preliminar de l'estat de l'art i del programari que es considerarà com a punt de partida, l'anàlisi de les senyals contaminades amb atacs de suplantació d'identitat i la implementació de les tècniques de detecció escollides, i, finalment, l'avaluació dels resultats.

Title : GNSS Signal Spoofing Detection
Authors: Eric Sánchez González

Advisor: Luis Esteve Elfau

Date: October 24, 2022

Overview

This thesis elaborates on the implementation of spoofing detection techniques for GPS L1 C/A signals, topic which is up to the minute in the GNSS community. The interest of this topic has its origin on the fact that, currently, there is a large number of applications relying on GNSS communications. Moreover, the public character of the communication details and specifications have exposed the communications to spoofing agents, which, with a relatively cheap equipment, are capable of controlling the tracking loops of a victim receiver and, as a result, manipulate the its timing or navigation solution.
In front of this issue, this project aims to contribute on the spoofing detection community by implementing, in the recognized Borre's GNSS receiver software, and testing some techniques. To do so, the project is organized in three sections; the preliminary study of the state of the art and the software that will be considered as the starting point, the spoofing signal analysis and the implementation of the selected spoofing detection techniques, and the result's evaluation.

”Fortune favours the bold.”
-Latin proverb

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ADC	Analog-to-Digital Converter. 8, 10, 11
AODO	Age Of Data Offset. 22
APT	Auxiliary Peak Tracking. x, xii, xiii, 35, 55, 65–74, 76, 80–89
AUC	Area Under The Curve. 79, 80, 84, 85
CDMA	Code-Division Multiple Access. 7, 8
CPCFSS	Code Phase and Carrier Frequency Serial Search implementation. 13
DLL	Delay-Locked Loop. xi, 16–19, 23
DSA	Digital Signature Algorithm. 34
ECDSA	Elliptic Curve Digital Signature Algorithm. 34
FPR	False Positive Rate. 79
FRFS	Fine Resolution Frequency Search. 42, 44
GLRT	Generalized Likelihood Ratio Test. 12
GNSS	Global Navigation Satellite System. iii, v, 5, 7–9, 29, 37–39, 89
GPS	Global Positioning Systems. iii, v, 5–9, 12, 17, 20, 23, 27, 29, 32–36, 39, 40, 42, 44, 53, 54, 58, 87, 89
IODC	Issue Of Data, Clock. 22, 32
IODE	Issue Of Data, Ephemeris. 22, 32
LHCP	Left-Hand Circularly Polarized. 9
LSR	Legitimate Signal Retriever. ix, 38
NAVI	Navigation message inspector. x, xii, 65, 74–78, 87–89
NCO	Numerically-Controlled Oscillator. 36
NMCT	Navigation Message Correction Table. 22, 23
PCFS	Parallel Carrier Frequency Search implementation. 13
PCPS	Parallel Code Phase Search implementation. 13, 14, 44
PCSS	Phase-Coherent Signal Synthesizer. 32
PLL	Phase-Locked Loop. xi, 16, 19

PRN Pseudorandom Noise. 5, 7, 12, 13, 16, 17, 34, 36, 44, 45, 55, 58

PVT Position Velocity Time Estimation. ix, 20, 23, 30–33, 37, 42, 86

RAM Random Access Memory. 34

RF Radio Frequency. ix, 8, 9, 12, 19

RHCP Right-Hand Circularly Polarized. 9

ROC Receiver Operating Characteristics. 79, 80, 82–85, 88

RSA Rivest, Shamir, and Adleman algorithm. 34

SBAS Satellite-Based Augmentation Systems. 39

TESLA Timed Efficient Stream Loss-Tolerant Authentication. 34

TEXBAT Texas Spoofing Test Battery. xi, xiii, 42, 53–55, 58, 61, 64, 87, 88

TOA Time Of Arrival. 23

TOW Time-Of-Week. x, 21, 74–76

TPR True Positive Rate. 79

URA User Range Accurancy. 22

UTC Coordinated Universal Time. 23

UTM Universal Transverse Mercator. 50

WGS84 World Geodetic System 1984. 50

WN Week number. 22

ACKNOWLEDGEMENTS

First and foremost, I leverage this section to express my gratitude to Professor Luis Esteve for his guidance, help and patience along the project's development. During these months, he has helped with the project through uncountable meetings and on-line consultations even in his holidays or in some gaps his busy schedule had. His goodwill and passion to teaching are inspiring and motivating.

In addition, it is also a good opportunity to thank my family and friends for the unconditional support not only along the project development but throughout my entire academic journey, his support has been fundamental especially in the tough moments.

INTRODUCTION

Nowadays it is substantial and seamless the growth of the applications which its services rely on GPS. Moreover, most of these applications involve safety features which require a high level of integrity and availability. A few examples are civil/military positioning and navigation, personnel tracking, emergency rescue, atmospheric analysis, mining and exploration, power grids, etc.

This capability of being used in a vast variety of applications has led to a definition of a system free from authentication and the publication of the signal structure (data structure, modulation schemes, Pseudorandom Noise (PRN) spreading codes, etc.). However, this implementation presents a significant drawback as its transparency and accessibility leave the receivers exposed to spoofing attacks. Moreover, this vulnerability is aggravated by the low power that the legitimate GPS signals reach the receivers, facilitating the attackers to eclipse or even jam these legitimate signals.

As a result, the GNSS technology is vulnerable to spoofing attacks, in which a malicious agent mimics genuine GNSS signals in order to infer a false location to a target user. This has been proven in several studies, such as (1), which states that the radio navigation department from The University of Texas diverted a $80M\$$ yacht's course using a custom-made GPS device. Another paperwork proving the discussed point is (2), where it is demonstrated that the required equipment to perform an spoofing attack may not exceed the 300 euros.

Counterfeiting detection and mitigation has been an issue of significant interest to the GNSS community for decades. Recent developments have led to architectures of GPS receivers that are able to identify and reject falsified signals, allowing the receiver to obtain estimates of the actual location of the receiver even during an attack forgery.

Motivations

Regarding my personal motivations, mention that since I started my Double Bachelor degree I always had in mind to work with satellite based communications related projects, as I found in these the perfect mix of the aerospace and telecommunications systems engineerings. Nevertheless, it has not been until the last year that I actually took a course covering this field. As I had believed, the course syllabus delighted and motivated me to put my time and effort, fact that enabled me to enjoy the course as I had never done before.

However, I still wanted to delve deeper into the field, so I asked the course professor for a thesis project regarding the satellite-based communications. Fortunately, he came up with a fantastic idea which is the implementation of anti-spoofing techniques on GPS signals, topic that also involved safety and defense, which always caught my attention.

Objectives

The project main aim is to implement several promising spoofing detection techniques for GPS L1 C/A signals. Nevertheless, this main goal can be divided in several complementary objectives:

- Perform a state of the art research on the current developed spoofing detection techniques.
- Get familiar with an open-source software receiver.
- Find and analyse signals containing spoofing attacks.
- Implement several spoofing detection techniques able to detect the attacks of the available spoofed signals.
- Analyse the performance of these techniques on the studied spoofed signals.
- Obtain conclusions.

Contents

In order to carry out the stated objectives, the project has been organized as detailed in this section.

The first section is dedicated to introduce the GPS satellite-based communications fundamentals necessary to understand the applied work discussed in this project.

Next, it is required to carry out a study of the state of the art in the subject of detection of 'spoofing' signals so as to be able to have an overview of the current proposed spoofing techniques.

Afterwards, it is analyzed the architecture, settings and work principles of the source code in which it will be implemented the spoofing detection techniques.

It is necessary to study and analyze the set of recorded signals that contain spoofing attacks, as the developed spoofing techniques will be tested with them.

Once the previous point are accomplished, it is presented the developed spoofing detection techniques developed and its performance will be tested with the stated signals containing spoofing attacks.

Finally, the results of the validation tests are presented and are extracted some conclusions.

CHAPTER 1. GPS FUNDAMENTALS

GPS is the most common and used GNSS which is conformed by 31 satellites orbiting at 20.000 km of altitude. These satellites, which can operate in three frequency bands and are equipped with high-precision atomic clocks, broadcast perfectly time-synchronized Code-Division Multiple Access (CDMA) signals using only its in-phase component or data component. The stated frequency bands are:

GPS bands		
Name	Central frequency [MHz]	Bandwidth [MHz]
L1	1575.42	15.345
L2	1227.6	11
L5	1176.45	12.5

Table 1.1: GPS frequency bands

However, out of these three bands, this project is designed to work within the L1 frequency band due to its longevity and its widespread use.

In addition, it is used spreading codes so-called ranging codes or PRN codes which, while on the one hand allows the signal synchronization on the receiver to improve the estimation accuracy of the arrival time, encrypt the signal and allow the transmission of several signals at the same time and frequency, on the other hand they require more bandwidth.

These codes are almost orthogonal, which means that the cross-correlation value will only be high if both codes are the same and are perfectly aligned, enabling the CDMA signals to be below the noise floor. In particular, in GPS are the employed the Gold PRN codes.

Regarding the transmitted signal, it can be modelled as follows:

$$s_i(t) = \sqrt{2P}C_{I,i}(t)\,D_{I,i}(t)\,cos\,(2\pi f_{L1}t) + \sqrt{2P}C_{Q,i}(t)\,D_{Q,i}(t)\,sin\,(2\pi f_{L1}t) \tag{1.1}$$

Being $C(t)$ the spreading PRN code at 1.023 Msps; $D(t)$ the data (in-phase) or the pilot (quadrature) component at 50 bps.

The data transmitted is the navigation message, which is divided in groups of 1500 bits, the so-called frames, which are comprised by 5 subframes. The navigation message contains the following information:

- Health status of the transmitting satellite
- Ephemeris data (Keplerian elements) of the transmitting satellite
- Almanac providing a reduced version of the Keplerian elements and health status of the other satellites that conform the constellation
- Clock parameters of the transmitting satellite
- Ionosphere and troposphere model parameters

GPS users receive in their antenna the following signal, which is the attenuated, code-phase shifted, phase shifted and noisy transmitted signal:

$$r_i(t) = \alpha_i(t)\sqrt{2P}C_i(t-\tau_i(t))D_i(t-\tau_i(t))\,cos\left(2\pi f_{L1}t + \phi_{Doppler,i}(t)\right) + n(t) \quad (1.2)$$

Being α_i the amplitude of the corresponding received signals, τ_i the code-phase delay related to the temporal delay due to the signal spatial propagation, $f_{Doppler,i}(t) = \frac{1}{2\pi}\frac{d\phi_{Doppler,i}(t)}{dt}$ the Doppler frequency shift due to the relative motion between the satellite and GNSS receiver and $n(t)$ the thermal noise, which is a quantification of the random processes that occur in the nature and is a function of both the temperature and the noise bandwidth. Moreover, thermal noise presents a constant power spectral density throughout the frequency domain.

As it has been stated, henceforth the pilot's component will not be written as in GPS L1 C/A it does not transmit data.

Generally, a GPS receiver consists of the following four blocks:

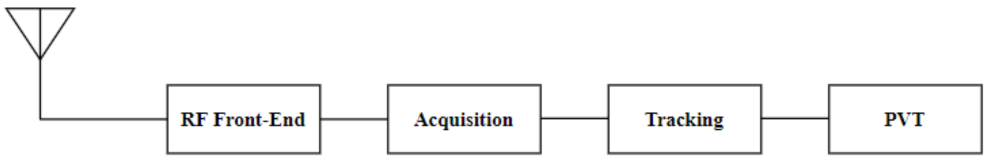


Figure 1.1: GPS receiver blocks (Own source)

1.1. RF Front-end

In broad terms, the RF front-end block is responsible for receiving the incoming signal, down-convert as a conditioning measure for the Analog-to-Digital Converter (ADC) and finally digitize it.

Regarding the signal reception, this initial block receives the incoming signal (1.2) which has propagated through space from the transmitting satellite to the GNSS receiver antenna. Although this signal reaches the antenna at a very low power, in fact below the thermal noise floor, it can be obtained as it uses the code division multiple access (CDMA) acquisition techniques based on the correlation properties.

Afterwards, due to the fact that ADC operate at lower frequencies, the signal shall be down-converted into an intermediate frequency in order to be properly digitized. The down-conversion process is performed by the mixing of the incoming signal and the local oscillator generated tone.

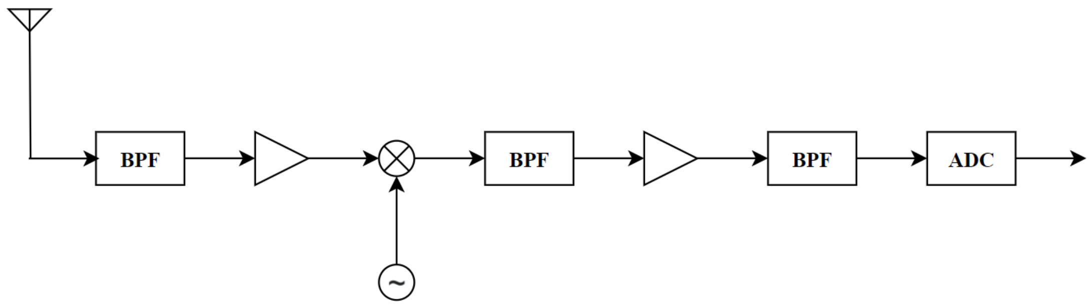


Figure 1.2: RF Front-end blocks (Own source)

However, this stated frequency down-conversion and signal conditioning pre-digitizing is also implemented with amplifiers and filters. Next, it is detailed the RF Front-end components and its main features:

- Antenna
  Particularly in reception, an antenna is a transductor which converts electromagnetic waves propagating through space in electric signals flowing in metal conductors.
  - Antenna pattern:
    It measures the capture capability of the antenna when receiving incoming signals. As the transmitting satellites are allocated above the GPS receiver, the antenna pattern is designed in order to maximize the receiving gain in elevation angles larger than positive 10° or 20° of elevation. Thus, it minimizes the multipath reflections, which usually reaches the antenna at a low elevation angles. In the azimuth plane, no direction is prioritized as the satellites can be distributed in all azimuth directions.
  - Polarization:
    It applies for the electric field orientation of the electromagnetic wave. As GNSS are Right-Hand Circularly Polarized (RHCP) signals, it is convenient that the receiver antenna so does. By doing so, it is achieved a remarkable mitigation of the Left-Hand Circularly Polarized (LHCP) signals and it is clearly beneficial since, when a GNSS RHCP signal is reflected from an object, it shifts its polarization from RHCP to LHCP.
  - Bias-tee feeding
    In GNSS links, it is important for the receivers to amplify in the antenna the received signal which has been remarkably debilitated due to the free-space losses involved in such a large link. Therefore, this amplification entails the incorporation of an active amplifier, which is fed with a three-port device so-called bias-tee feeding circuit.

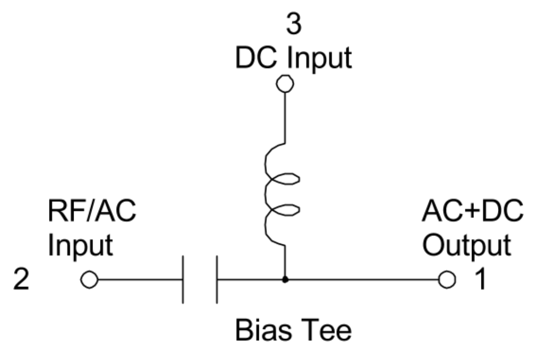


Figure 1.3: Bias-tee circuit, extracted from (10)

- Filters
  Are frequency discriminators devices that, on the one hand, attenuate frequency intervals whereas, on the other hand, do not affect other frequency intervals. Actually, they attenuate all the frequency components as they are passive devices, however, the attenuation of the eliminated band is enormous whilst the attenuation of the pass-band is almost nonexistent. The filters main objectives are:
  - Reduce noise
    In the frequency band that does not contain the GNSS signals entails thermal noise, which shall be eliminated in order to improve the signal-to-noise ratio. Nevertheless, the noise located at the GNSS signal frequencies will inevitably leak into the receiver and, consequently, deteriorate the signal-to-noise ratio.
  - Eliminate the image frequencies
    When down-converting the high frequency received signal, previous to the temporal multiplication with the local oscillator, other non-desired signals known as image frequencies shall be eliminated. Otherwise, these image frequencies can be down-converted to the same frequency as the desired received signal and cause irremediable interference.
- Amplifiers
  Active device that shall be fed which function is to increase the inputs signal's amplitude. In signal reception, it is important to amplify the incoming signal thus adjusting and leveraging the entire ADC dynamic range.
  Moreover, according to the Friis formula, in order not to spoil the overall noise figure of the front end chain, it is important that the first chain element holds a significant gain and a low noise figure. This statement is exposed in the Friis formula shown below:

$$F_{total} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \dots \frac{F_N - 1}{G_1 G_2 \dots G_N} \tag{1.3}$$

- Local oscillator and Mixer
  The local oscillator is an electronic device that generates a tone or sinusoidal signal centered at the frequency denoted as f_{LO}, which can be either:
  - $f_{LO} = f_c$:
    Being f_c the received signal carrier frequency. In this case, the incoming RF

signal is down-converted to base band if no Doppler effect is assumed. In a realistic case where it is considered the Doppler effect, then it is down-converted to $f_{Doppler}$.

– $f_{LO} = f_c - f_{IF}$:
Being f_{IF} the so-called intermediate frequency. In this second case, the received signal is down-converted to an intermediate frequency f_{IF} if no Doppler effect is taken into account.

When multiplying in the time domain both the local oscillator tone and the received signal via the mixer, it is achieved a down-conversion which will be detailed in future sections. The combination of both components enable the frequency reduction of the incoming signal into the ADC performing extent.

- Analog-to-Digital Converter (ADC)
ADC converts the analog signal into digital samples sampled at the sampling frequency (f_s) . As the signal will be processed by software, it involves that the independent variable shall not be real. Consequently, a sequence of the signal shall be captured.
The analog-to-digital conversion can be modelled as the multiplication of both the received analog signal $(r_i(t))$ and $(p(t))$, which is a train of deltas temporary separated the sampling period $(T_s = \frac{1}{f_s})$. The result can be expressed as:

$$r_{i,digital}(t) = r_i(t)p(t) = r_i(t)\sum_{k=-\infty}^{\infty}\delta(t-kT_s) = \sum_{k=-\infty}^{\infty} r_i(kT_s)\delta(t-kT_s) \qquad (1.4)$$

Being $r_i[k] = r_i(kT_s)$ the equivalent discret sequence and k an integer value (0,1,2...).

Note that one of the main front-end features it to down-convert the incoming RF-signal to at least an intermediate frequency. That is due to the fact that, according to the Nyquist criterion, in order to properly sample a continuous signal, the sampling frequency shall be bigger (or equal) than the double of the signal's bandwidth.

$$f_s > 2BW \qquad (1.5)$$

Therefore, without a front-end module reducing the incoming signal's frequency before the digitizing process, processors should be able to operate at a more than 3GHz rate, feature that nowadays is not feasible.
Finally, apart from the signal sampling, the ADC also performs the quantization process. Although the resolution is increased with the bit number, it has been proved that with only 2 bits it is possible to obtain position.

1.2. Acquisition

The acquisition signal model is the output of the RF Front-end and, assuming that the intermediate frequency is zero, can be expressed as:

$$x[k] = \sum_{i=1}^{N}\alpha_i(t)C_i\big(t-\tau_i(t)\big)D_i\big(t-\tau_i(t)\big)e^{j2\pi f_{Doppler,i}t} + n(t)\ |_{t=kT_s} \qquad (1.6)$$

Where $\mathbf{x} = [x[1], x[1], ..., x[K]]$ represents a K-samples-long vector of the RF front-end output sampled signal, N the number of visible satellites and $n(t)$ the noise. Furthermore, as the bit duration $(20ms/bit)$ is larger than the chip duration $(1ms/1023chips \rightarrow 0.9810^{-3}ms/chip)$, during the $1ms$-long acquisition process (in case of GPS L1 C/A) can be simplified as (assuming no bit transition occurs):

$$x[k] = \sum_{i=1}^{N} \alpha_i(t) C_i\big(t - \tau_i(t)\big) e^{j2\pi f_{Doppler,i} t} + n(t) \mid_{t=kT_s} \tag{1.7}$$

Considering the block functions, the acquisition block is in charge of:

- Determining which satellites are visible to the receiver.
- For each visible satellite, determine a coarse estimation of both the Doppler frequency shift $f_{Doppler,i}$ and the code-phase delay τ_i. It is important to highlight that, although in 1.2 these parameters are time-dependent, during the acquisition process, as it takes from 1ms in GPS L1 C/A to 4ms in Galileo, can be considered as constants.

So as to do that, it is defined an hypothesis test in order to compare its result with a specific threshold. Depending whether the test value is higher or lower than the threshold, it is decided either H_1 or H_0 respectively. H_1 denotes the presence of the satellite and H_0 otherwise. In particular, the test function is the so-called Generalized Likelihood Ratio Test (GLRT) and, for a receiving sampled signal vector $\mathbf{x}$, can be mathematically defined as:

$$T_{GLRT}(\mathbf{x}) = argmax_{f_{Doppler},\tau}\{\frac{|\hat{R}_{x,d}(f_{Doppler},\tau)|^2}{\hat{P}_x}\} \tag{1.8}$$

Where $\hat{R}_{x,d}$ is an estimate of the cross-correlation between the signal received at the antenna once sampled and the local replica of the searched satellite sampled vector for a specific code-phase and Doppler frequency shift $d(t,\tau,f_{Doppler}) = C_{SAT}(t-\tau)e^{-j2\pi f_{Doppler} t}$. This correlation is maximum (takes a value of 1 as it has been normalized) when the PRN code used in the local replica corresponds to a visible satellite and both the Doppler frequency and code-phase employed in the local replica correspond to the actual Doppler frequency and code-phase of the respective visible satellite.

$$\hat{R}_{x,d}(f_{Doppler},\tau) = \frac{1}{K}\sum_{k=1}^{K} x[k] d(t,\tau,f_{Doppler}) \mid_{t=KT_s} \tag{1.9}$$

However, the two-dimensional search for the maximum value of the test can not be solved analytically, thus leading to a numerical approach. Therefore, for a static receiver, a search grid is designed with the following boundaries and resolution:

	Search boundaries		**Search resolution**	
	Code-phase	**Doppler frequency**	**Code-phase**	**Doppler frequency**
GPS L1 C/A	[0,1ms]	[-5KHz,5KHz]	$T_{chip}/2$	$2/(3T_{acquisition})$
Galileo E1	[0,4ms]	[-5KHz,5KHz]	$T_{chip}/4$	$2/(3T_{acquisition})$

Table 1.2: Acquisition search parameters

Being T_{chip} the duration of a PRN code chip and $T_{acquisition}$ the duration of the samples vector used in the acquisition process (i.e. $T_{acquisition} = KT_s$).
In the following figures 1.4 and 1.5 , it can be seen the search results of both scenarios, the first one containing the searched satellite and the second one not.

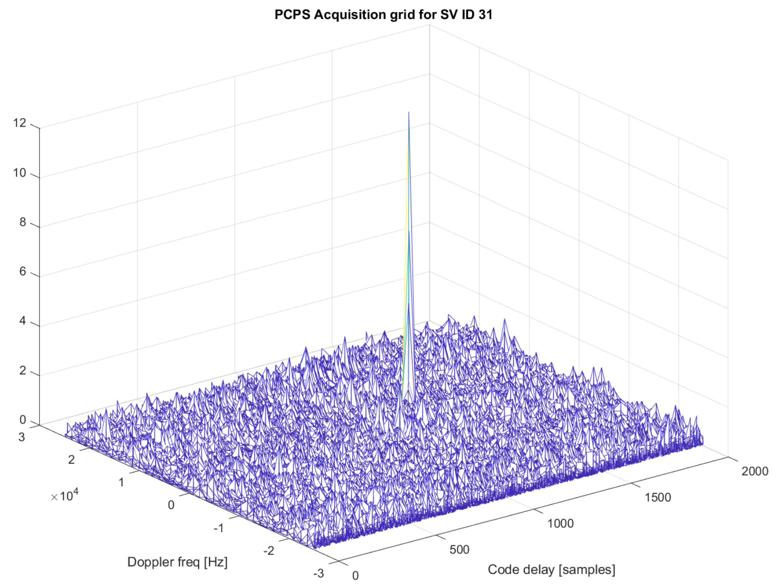


Figure 1.4: Acquisition search grid: present satellite (Own source)

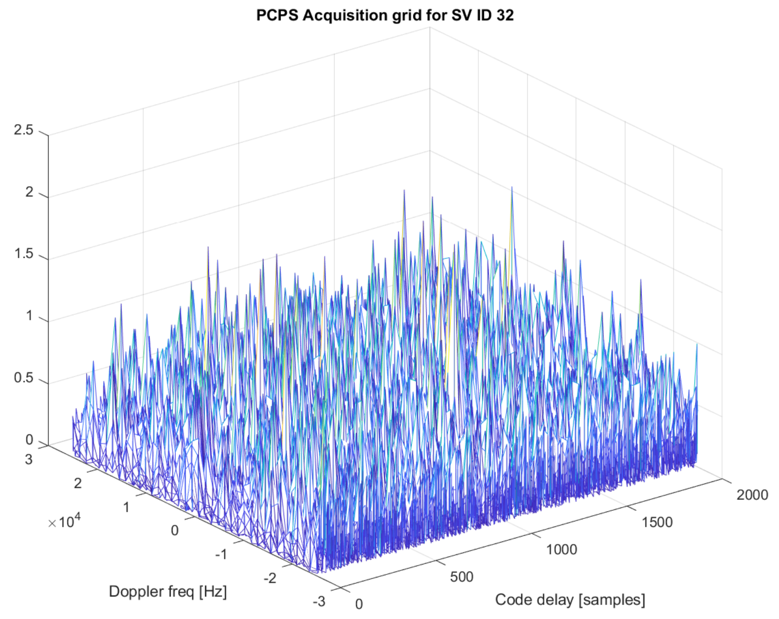


Figure 1.5: Acquisition search grid: absent satellite (Own source)

In order to perform this search, there are several implementations such as the Code Phase and Carrier Frequency Serial Search implementation (CPCFSS), the Parallel Carrier Frequency Search implementation (PCFS) and the Parallel Code Phase Search implementation (PCPS). The first one is the more expensive computationally wise as it computes the test for every point of the search grid. PCFS manages to only examine the code-phase axis by performing the Fourier transform on each code-phase value and, in case the determined satellite is present, it appears a peak at the corresponding $f_{Doppler}$. Finally, the most computationally efficient technique (as it only goes through the less segmented axis), the PCPS, achieve the same purpose by examining the Doppler frequency axis. In order to do that, it leverages the following cross-correlation property, which exposes an alternative method to compute it by using the Fourier's transform:

$$R_{xy}(\tau) = \frac{1}{K}\sum_{k=1}^{K} x(kT_s)y^*(kT_s-\tau) = \mathcal{F}^{-1}\{\mathcal{F}\{x(\tau)\}\mathcal{F}\{y(\tau)\}^*\} \tag{1.10}$$

In the figure below (figure 1.6) it is shown the PCPS implementation scheme:

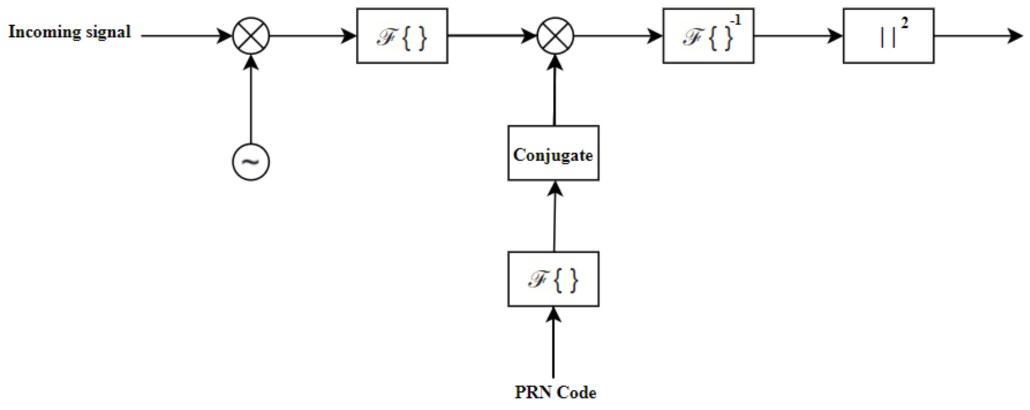


Figure 1.6: PCPS acquisition implementation (Own source)

1.3. Tracking

The tracking fundamental purposes are:

- To refine the acquisition's coarse estimation of both the code-phase and Doppler frequency shift.
- To perform a tracking or following of this parameters over time as they are time dependent.
- To demodulate the signal to base-band.
- To proportionate the navigation bits.

At this stage, the signal model in its complex form can be mathematically expressed as:

$$r_i(t) = \alpha_i(t)D(t-\tau(t))C(t-\tau(t))e^{j\phi(t)} + n(t) \tag{1.11}$$

In which $\phi(t)$ relates to the Doppler frequency as follows:

$$f_{Doppler,i}(t) = \frac{1}{2\pi}\frac{d\phi_{Doppler,i}(t)}{dt} \tag{1.12}$$

In the last formula, $f_{Doppler,i}(0)$ is the coarse Doppler frequency phase shift estimated in the acquisition block.

After the signal acquisition, the tracking module is relentlessly executing as the tracked parameters are time dependent and in case these tracking parameters are lost, a new acquisition process shall be carried out.

In order to obtain the navigation bits, the tracking module shall:

a Down-convert to base band

In the previous expression 1.11 the phase term $e^{j\phi(t)}$ represents the miss-modulation term due to the Doppler frequency effect. In other words, the signal is not exactly frequency-centered at base-band, instead it is centered at the Doppler frequency.

To better understand the down-conversion demonstration, it is analyzed the upper branch of the 1.7 figure, in which the input signal is the real part of the equation 1.11.

$$r_i(t) = \alpha_i(t)D(t-\tau(t))C(t-\tau(t))cos(\phi(t)) + n(t) \tag{1.13}$$

So as to achieve the down-conversion, if the incoming signal is multiplied by a tone generated by the local oscillator ($\hat{\phi}(t)$) at, ideally, the exact same frequency (neglecting noise) it is obtained:

$$r_i(t)cos(\hat{\phi}(t)) = \alpha_i(t)D(t-\tau(t))C(t-\tau(t))cos(\phi(t))cos(\hat{\phi}(t)) \tag{1.14}$$

Applying the following trigonometric identity:

$$cos(A)cos(B) = \frac{1}{2}(cos(A-B) + cos(A+B)) \tag{1.15}$$

- Ideally, when the local oscillator manages to generate a tone at the exact same frequency that the incoming signal $\phi(t) = \hat{\phi}(t)$, the multiplication results in:

$$r_i(t)cos(\hat{\phi}(t)) = \frac{1}{2}(A_iD(t-\tau(t))C(t-\tau(t)) + A_iD(t-\tau(t))C(t-\tau(t))cos(2\phi(t)) \tag{1.16}$$

  Which, by low-pass filtering, can be obtained a perfectly down-converted signal:

$$r_i(t)cos(\hat{\phi}(t)) \mid_{filtered} = \frac{1}{2}A_iD(t-\tau(t))C(t-\tau(t)) \tag{1.17}$$

- In a realistic scenario, the local oscillator tone deviates from the incoming signal's frequency $\phi(t) \neq \hat{\phi}(t)$, therefore the multiplication is:

$$\begin{aligned} r_i(t)cos(\hat{\phi}(t)) = \frac{1}{2}(&A_iD(t-\tau(t))C(t-\tau(t))cos(\phi_e(t)) \\ &+ A_iD(t-\tau(t))C(t-\tau(t))cos(\phi(t)+\hat{\phi}(t)) \end{aligned} \tag{1.18}$$

Being $\phi_e(t) = \phi(t) - \hat{\phi}(t)$.
Which, by low-pass filtering it is obtained:

$$r_i(t)cos(\hat{\phi}(t)) \mid_{filtered} = \frac{1}{2}A_i D(t-\tau(t))C(t-\tau(t))cos(\phi_e(t)) \tag{1.19}$$

The element responsible for continuously down-converting the acquired signal to baseband is the PLL, which provides a $\hat{\phi}(t)$ estimation of the constantly changing Doppler frequency shift via the Carrier Loop Discriminator.

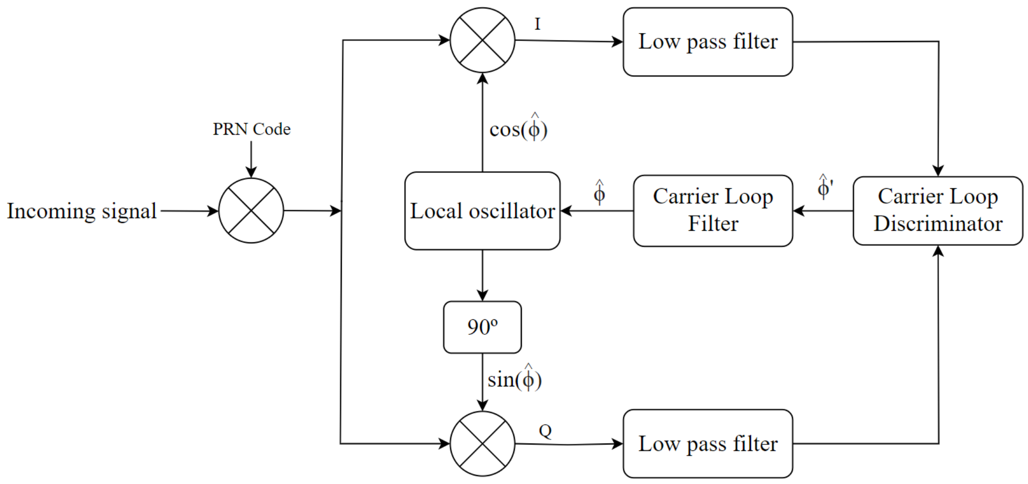


Figure 1.7: PLL (Own source)

Assuming the local generated PRN code to be perfectly aligned with the incoming signal, the Carrier Loop Discriminator, at a certain time, receives as inputs the in-phase and quadrature prompt components:

$$\begin{aligned} I_p &= \frac{1}{2}A_i D_i cos(\phi_e) \\ Q_p &= -\frac{1}{2}A_i D_i sin(\phi_e) \end{aligned} \tag{1.20}$$

With this inputs, the Carrier Loop Discriminator shall estimate the phase $\hat{\phi}'$ in order to minimize ϕ_e.

Due to the fact that the Carrier Loop Discriminator estimation is considerably noisy, it shall be filtered by the carrier loop filter. Thus, the filtered phase estimation is $\hat{\phi}$, from which is generated the tone responsible for the down-conversion.

b Despreading code $C(t-\tau(t))$

From the 1.11 it also shall be removed the signal PRN code, process which is also known as despreading the signal. The responsible for that is the DLL, whose objective is to correlate the incoming signal by the corresponding locally generated PRN code aligned in time. By doing that, assuming no bit transitions, the despreading procedure can be modelled as:

$$\begin{aligned}
R_{x,C_{local}}(t,\tau,\hat{\tau}) \mid_{t=kT_s} &= \sum_{k=1}^{K} x(t)C_{local}(t-\hat{\tau}) \mid_{t=kT_s} \\
&= \sum_{k=1}^{K} \frac{1}{2}A_i D(t-\tau(t))C(t-\tau(t))C_{local}(t-\hat{\tau}) \mid_{t=kT_s} \\
&= \frac{1}{2}A_i D(t-\tau(t)) \mid_{t=kT_s} \sum_{k=1}^{K} C(t-\tau(t))C_{local}(t-\hat{\tau}) \mid_{t=kT_s} \\
&= \frac{1}{2}A_i D(t-\tau(t)) \mid_{t=kT_s} R_c(\tau-\hat{\tau}) \\
&= \frac{1}{2}A_i D(t-\tau(t)) \mid_{t=kT_s} R_c(\tau_{error})
\end{aligned} \tag{1.21}$$

$$R_c(\tau_{error}) = \begin{cases} K, & \text{if } \tau_{error} = 0 \\ \approx 0, & \text{otherwise} \end{cases}$$

Being $x(t)$ an ideal down-converted sampled signal. In addition, in the previous mathematical expression (equation 1.21), the navigation bits term $D(t-\tau(t))$ can be considered as a constant in the integration time since the bit period is larger than the integration period (assuming no bit transition).

After having explained the theoretical concept, the real challenge is to generate a locally generated PRN code continuously aligned in time, as τ changes with time evolution. To do so, the DLL implements a PRN code generator, which locally generates three (five in Galileo E1) code replicas time-displaced $C(t-\tau_E), C(t-\tau_P), C(t-\tau_L)$, so-called early, prompt and late code local replicas, where:

$$\begin{aligned}
\tau_P &= \hat{\tau} \\
\tau_E &= \hat{\tau} - \varepsilon \\
\tau_L &= \hat{\tau} + \varepsilon
\end{aligned} \tag{1.22}$$

Being $\varepsilon = \frac{T_{chip}}{2}$ in GPS.
The incoming signal is cross-correlated with these three code replicas and, according to the early, prompt and late correlation values and knowing that ideally (when the estimated code-phase matches the actual code-phase i.e. $\hat{\tau} = \tau_{actual}$) the prompt correlator shall be larger than the early and late correlators, the Code Discriminator outputs a code phase estimation.

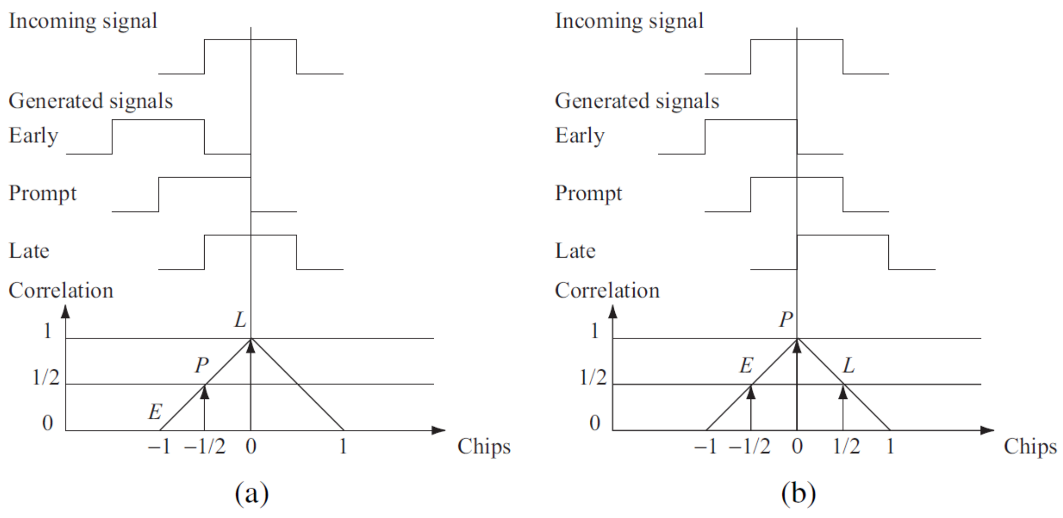


Figure 1.8: Early, prompt and late output correlator values as a function of $\tau_{error} = \tau_{actual} - \hat{\tau}$, extracted from (17)

In the previous figure 1.8 it is represented in (a) an specific scenario where $\tau_{error} = -\frac{1}{2}$, consequently meaning that the code phase estimated $(\hat{\tau})$ shall be decreased. Whereas in (b) it is depicted an scenario where the DLL is perfectly synchronized as $\tau_{error} = 0$ meaning that the prompt correlator value is the highest and the early and late correlator values are similar.

The Code Discriminator function is, based on the early, prompt and late correlator outputs, to estimate the code phase in order to minimize the code phase error (i.e $\tau_{error} = \tau_{actual} - \hat{\tau} = 0$).

A complete DLL block diagram is:

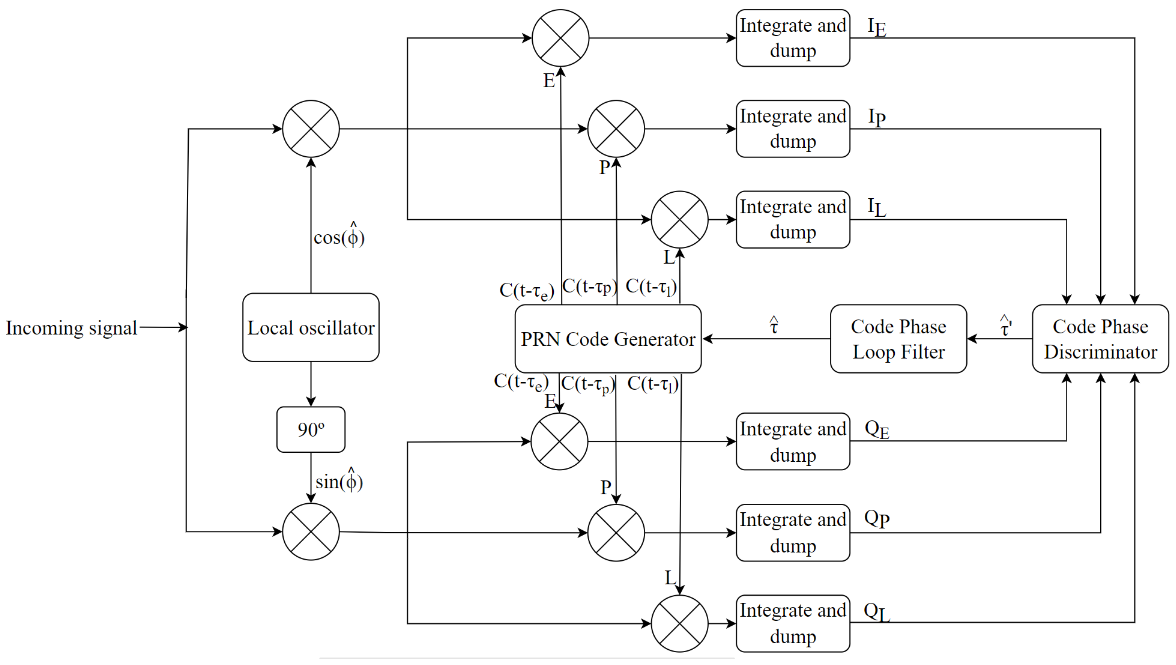


Figure 1.9: DLL (Own source). Note that $\hat{\tau} = \tau_p$

Finally, a possible implementation of the whole tracking channel including both PLL and DLL could be:

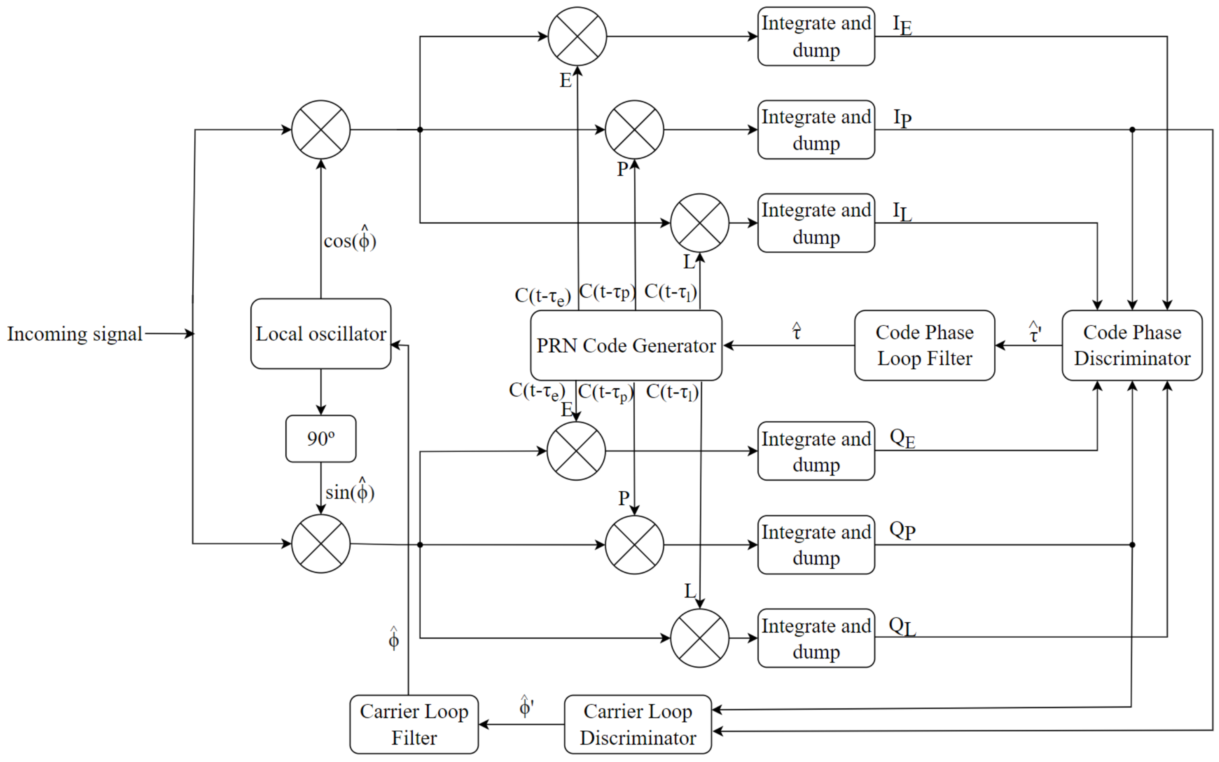


Figure 1.10: Tracking channel: PLL and DLL integration (Own source). Note that $\hat{\tau} = \tau_p$

Once the received signal is processed by the RF front-end, acquisition and tracking blocks, it is obtained the navigation bits from the In-phase correlator output I_p as can be seen in the following figure:

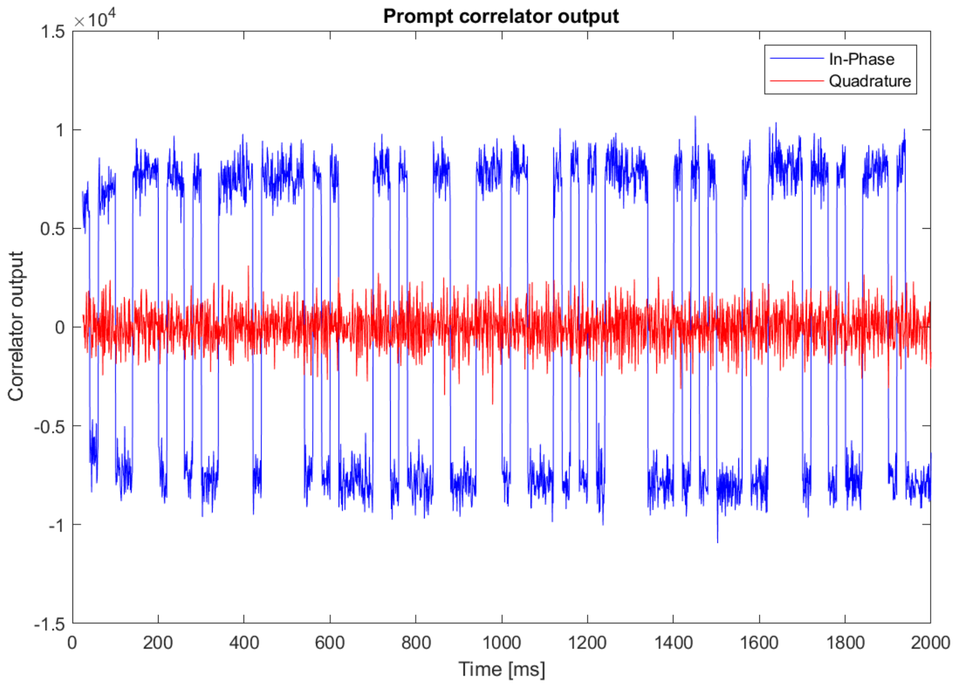


Figure 1.11: Navigation bits (Own source)

1.4. PVT

Finally, at this point the receiver has obtained the navigation bits, which will provide the receiver the required information to compute its position such as the visible satellites locations and transmission times. Nevertheless, a preliminary and necessary step is to examine the navigation message structure in order to understand how to decode it.

1.4.0.1. Navigation message structure

In this section are presented the parameters included in the GPS L1 C/A navigation message and the organization of them.

The navigation message consist of the succession of 30-bit words, which a set of 10 words constitute a subframe. The last 6 bits of each word are reserved for parity check so as to detect errors. Similarly, 5 subframes compose a frame and 25 frames form the whole message. Subframes 4 and 5 are 25 pages-long, hence, as in every subframe it is transmitted a single page, it takes 25 subframes to fully transmit the entire navigation message. Regarding the 50 bps bit rate, the durations are the following ones:

Item	**Number of bits**	**Duration [s]**
Word	30	0.6
Subframe	300	6
Frame	1500	30
Navigation message	37500	750

Table 1.3: Navigation message structure lengths and durations, extracted from (16)

In the next figure it is shown the frame structure and the parameters the 5 subframes contain, which will be detailed down below.

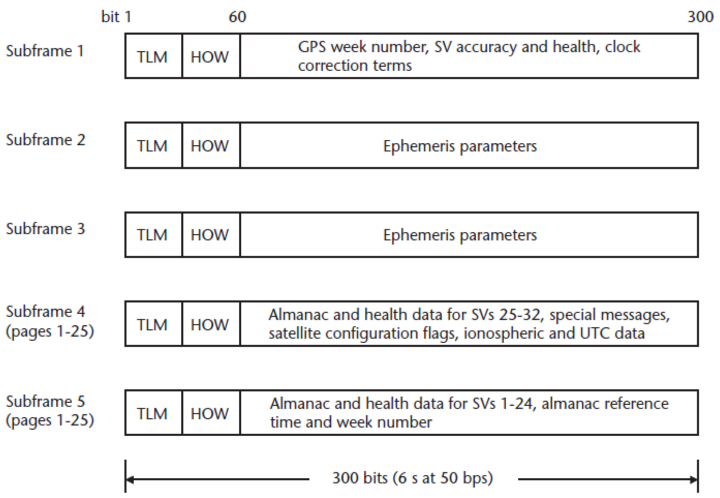

bit 1
60
300
Subframe 1
TLM
HOW
GPS week number, SV accuracy and health, clock correction terms
Subframe 2
TLM
HOW
Ephemeris parameters
Subframe 3
TLM
HOW
Ephemeris parameters
Subframe 4 (pages 1-25)
TLM
HOW
Almanac and health data for SVs 25-32, special messages, satellite configuration flags, ionospheric and UTC data
Subframe 5 (pages 1-25)
TLM
HOW
Almanac and health data for SVs 1-24, almanac reference time and week number
300 bits (6 s at 50 bps)

Figure 1.12: Navigation message structure, extracted from (16)

- TLM
  Telemetry data begins with an 8-bit fixed preamble (10001011) that indicates the beginning of the subframe. The remaining 16 bits (recall that the last 6 bits are reserved for parity check) are dedicated to authorized users and to the control segment.

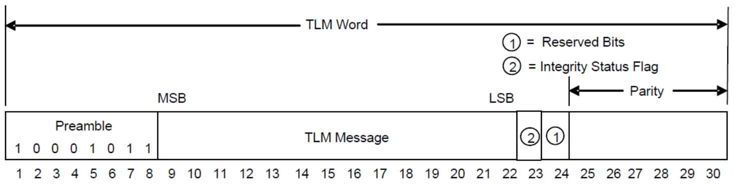


Figure 1.13: TLM word, extracted from (23)

- HOW
  Informs of the TOW modulo 6 seconds of the next subframe start. Moreover, it employs two bits to inform of anti-spoofing activation and a bit to denote poor signal quality.

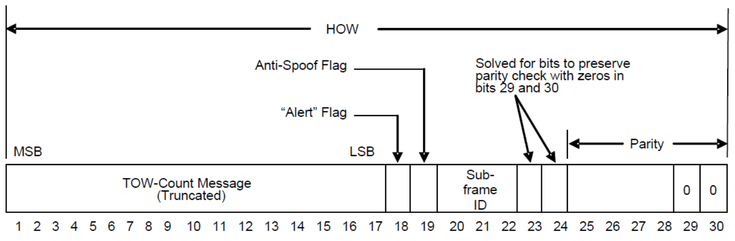


Figure 1.14: HOW word, extracted from (23)

- Subframe 1: Satellite clock and health information

  Provides the parameters regarding the satellite transmission time and the health, which is a parameter that indicates the data validity.

  - Week number (WN) (10 bits)
    Inform of the number of weeks modulo 1024 transcurred from 05/01/1980.
  - Clock correction terms (62 bits)
    Provide information to compensate the miss-synchronization between transmitter and receiver.
  - Issue Of Data, Clock (IODC) (10 bits)
    Dedicated to exclusively identify the navigation data.
  - SV Health (6 bits)
    Is a health indicator informing of the operating conditions of the broadcasting satellite.
  - User Range Accurancy (URA) (4 bits)
    Pseudorange error estimation.
  - Group delay correction T_{gd} (8 bits)
    Provides information to the single-frequency receivers in order to estimate the group delay introduced by the ionosphere.
  - L2 code indicator (2 bits)
    Indicate the active L2 code: either large precision code P(Y) or short coarse/acquisition C/A code.
  - L2 P data flag (1 bit)
    To report the navigation data modulation in the L2 P(Y) code.

- Subframe 2 and 3: Satellite ephemeris
  Informs of the Keplerian elements which describe the satellite orbit. With that information, the exact satellite location can be computed.

  - Ephemeris (358 bits)
    Ephemeris are composed by a set of Keplerian elements that unequivocally describe a satellite orbit and a set of corrections (see appendix A).
  - Fit interval flag (1 bit)
    Reports whether the ephemeris ajust corresponds to a 4 hour interval or a larger interval.
  - Age Of Data Offset (AODO) (5 bits)
    Indicates the navigation message age and it is used when applying the navigation message correction table (Navigation Message Correction Table (NMCT)).
  - Issue Of Data, Ephemeris (IODE) (8 bits)
    It allows the detection of the navigation message variations. It shall coincide with the 8 least significant bits of IODC.

- Subframe 4 and 5: Almanac data
  Almanac data includes information of the entire satellite constellation as well as coarse data of the other satellites in order to ease the signal acquisition. This information is transmitted throughout a total of 50 subframes, which is equivalent to 12.5 minutes.

- Almanac (166 bits)
  In pages 2,3,4,5,7,8,9,10 of subframe 4 and in pages from 1 to 25 of subframe 5 it is allocated the almanac data (see appendix B). Among others, almanac comprises: the satellite ID, the six Keplerian elements, health parameters and clock correction parameters which enable the computation of the satellite clock offset:
  $$dk^k = a_{f0} + a_{f1}(t - t_{0a}) \tag{1.23}$$
- NMCT (182 bits)
  Included in page 13 of the forth subframe.
- Single-frequency receivers ionospheric corrections (32 bits)
  Allocated in page 18 of subframe 4.
- Parameters relating GPS and Coordinated Universal Time (UTC) times (104 bits)
  Allocated in page 18 of subframe 4.

Knowing the navigation message structure, so as to obtain the receiver's position from these navigation bits, the following processes shall be performed:

1.4.0.2. Bit synchronization

Is the process of adapting the bit rate coming from the tracking module to the PVT module. Due to the noise and weak power of the signal, a correlator output value shall be averaged among other consecutive values in order to mitigate the noise effect. In particular, 20 successive correlator output values are averaged when deciding a PVT value. As a result of replacing 20 samples by a single one, the sample rate is reduced from 1KSPS (1 sample each ms) to 50SPS.

1.4.0.3. Preamble identification

As each one of the navigation message subframes starts with the same 8-bit word (10001011), the so-called preamble, the starting sample of a subframe can be detected by correlating the incoming bits (with 1 or -1 values) with the well-known 8-bit word preamble also in 1 and -1 form. Consequently, when the preamble is detected, the correlation outputs a value of 8. In addition, it is carried out an extra check since, as every subframe has a duration of 6 seconds, a preamble only can be detected in a 6 second interval.

At this point, it is possible to properly detect the navigation message beginning, therefore, the receiver position can be obtained as it is detailed in the next section.

1.4.0.4. Position computation

The receiver position computation is based on the Time Of Arrival (TOA) principle. Knowing the transmission time (included in the navigation message) and the reception time (estimated by the DLL), it can be calculated the signal's propagation time. Moreover, as electromagnetic signals emitted by the satellites propagate through space at the well-known light speed, it is possible to infer the distance satellite-receiver. Nevertheless, with a single satellite signal, the receiver can not calculate its position as it can only assume that it is within a circle of radius equal to the distance computed. Therefore, in order to overcome the position ambiguity, it is necessary at least three different satellites.

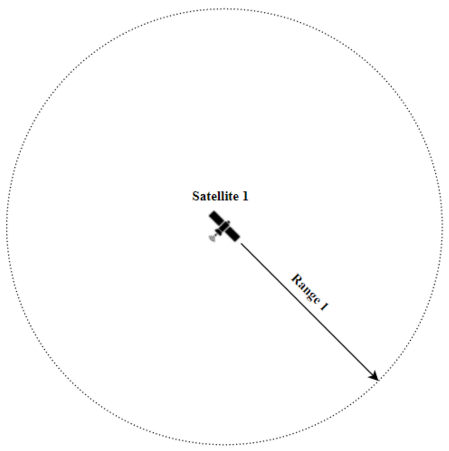


Figure 1.15: One satellite range ambiguity (Own source)

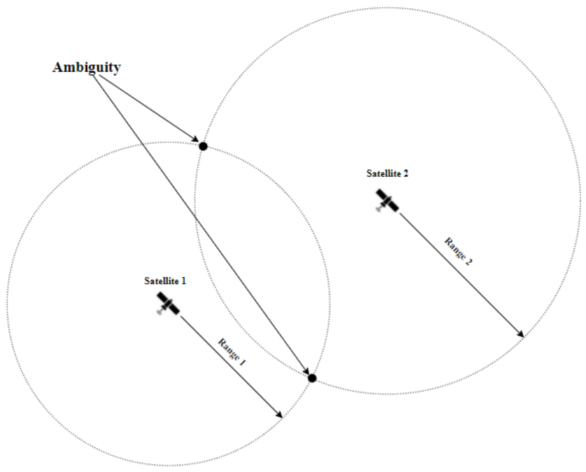


Figure 1.16: Two satellites range ambiguity (Own source)

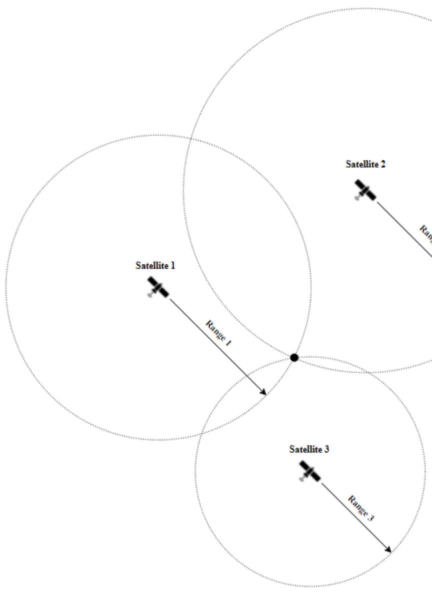


Figure 1.17: Three satellite range ambiguity (Own source)

In addition, the broadcasting satellites clocks and the receivers clocks are not precisely synchronized, leading to an error in the computed pseudoranges. As the transmitting satellites are equipped with atomic clocks, they are perfectly synchronized among them and, consequently, the timing offsets the receiver measures are identical. That is why it is needed a forth satellite in order to figure out this extra unknown.

Regarding the pseudorange computation taking into account the clock's miss-synchronization it is presented the following figure (1.18). $t_{TX,SYS}$ is the system time when the satellite transmits the signal. $\delta_{satellite}$ and $\delta_{receiver}$ are, respectively, the satellite and receiver time offsets with respect to the system time. Finally, $t_{RX,SYS}$ is the system time when the receiver captures the signal.

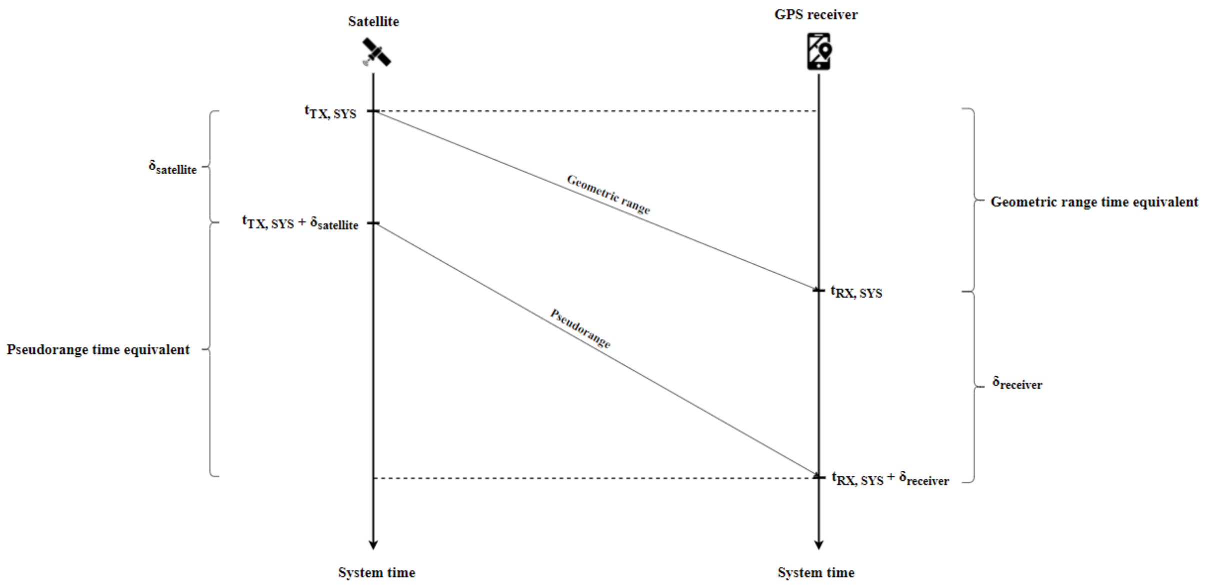


Figure 1.18: Satellite and receiver timing offsets representation (Own source)

As it can be seen, the geometric range corresponds to:

$$r = c(t_{RX,SYS} - t_{TX,SYS}) \tag{1.24}$$

And the computed pseudorange as follows:

$$\rho = c\big((t_{RX,SYS} + \delta_{receiver}) - (t_{TX,SYS} + \delta_{satellite})\big) \tag{1.25}$$

Therefore, pseudorange can be modelled as the geometric range plus a range which depends on the miss-synchronization:

$$\rho = r + c(\delta_{receiver} - \delta_{satellite}) \tag{1.26}$$

On the other hand, considering the following figure (1.19), the geometric range can be expressed as a function of u, which represents a user receiver's position with in the ECEF coordinate system origin, and s, which represents the position of the satellite relative to the coordinate origin and is computed using ephemeris data broadcast by the satellite. It can be modelled as follows:

$$|r| = |\vec{s} - \vec{u}| \tag{1.27}$$

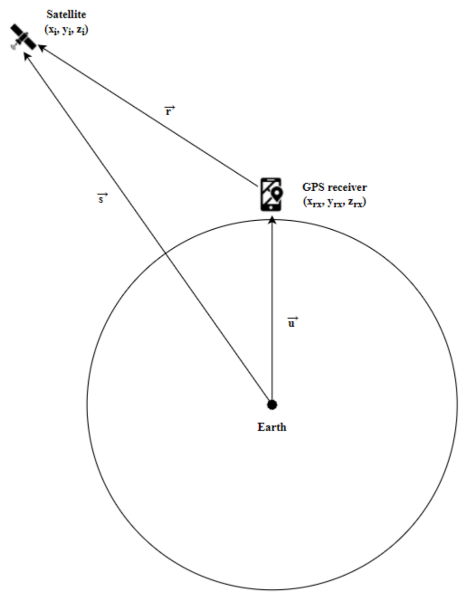


Figure 1.19: Vector representation of the user's position (Own source)

Combining 1.27 and 1.26 we obtain the following set of 4 pseudorange equations, one for each visible satellite:

$$\begin{cases} \rho'_1 = \sqrt{(x_1 - x_{rx})^2 + (y_1 - y_{rx})^2 + (z_1 - z_{rx})^2} + c(\delta_{rx} - \delta_{satellite_1}) \\ \rho'_2 = \sqrt{(x_2 - x_{rx})^2 + (y_2 - y_{rx})^2 + (z_2 - z_{rx})^2} + c(\delta_{rx} - \delta_{satellite_2}) \\ \rho'_3 = \sqrt{(x_3 - x_{rx})^2 + (y_3 - y_{rx})^2 + (z_3 - z_{rx})^2} + c(\delta_{rx} - \delta_{satellite_3}) \\ \rho'_4 = \sqrt{(x_4 - x_{rx})^2 + (y_4 - y_{rx})^2 + (z_4 - z_{rx})^2} + c(\delta_{rx} - \delta_{satellite_4}) \end{cases} \tag{1.28}$$

In the previous equation there are 8 unknowns: $x_{rx}, y_{rx}, z_{rx}, \delta_{rx}, \delta_{satellite_1}, \delta_{satellite_2}, \delta_{satellite_3}, \delta_{satellite_4}$. Recall that x_i ,y_i and z_i are the satellite location coordinates, which are included in the navigation message. Nonetheless, the GPS ground segment is able to estimate the different $\delta_{satellite_1}, \delta_{satellite_2}, \delta_{satellite_3}, \delta_{satellite_4}$ and, consequently, can obtain clock corrections which are sent to the broadcasting satellites so that these corrections can be included in the navigation message. Thus, GPS receivers can compensate the satellite time offset. Therefore, the satellite time offset with respect to the system time is not an unknown.

As a consequence, as there are four unknowns it will be required four equations, each equation derived from a visible satellite.

$$\begin{cases} \rho_1 = \sqrt{(x_1 - x_{rx})^2 + (y_1 - y_{rx})^2 + (z_1 - z_{rx})^2} + c\delta_{rx} = f_1(x_{rx}, y_{rx}, z_{rx}, \delta_{rx}) \\ \rho_2 = \sqrt{(x_2 - x_{rx})^2 + (y_2 - y_{rx})^2 + (z_2 - z_{rx})^2} + c\delta_{rx} = f_2(x_{rx}, y_{rx}, z_{rx}, \delta_{rx}) \\ \rho_3 = \sqrt{(x_3 - x_{rx})^2 + (y_3 - y_{rx})^2 + (z_3 - z_{rx})^2} + c\delta_{rx} = f_3(x_{rx}, y_{rx}, z_{rx}, \delta_{rx}) \\ \rho_4 = \sqrt{(x_4 - x_{rx})^2 + (y_4 - y_{rx})^2 + (z_4 - z_{rx})^2} + c\delta_{rx} = f_4(x_{rx}, y_{rx}, z_{rx}, \delta_{rx}) \end{cases} \tag{1.29}$$

As it can be seen, the equations are not linear. A method to solve the equation's system is by applying a linearization process, which assume an approximation of the receiver's true location. This stated process starts with expressing the pseudodistance satellite-receiver measurement as:

$$f(x_{rx}, y_{rx}, z_{rx}, \delta_{rx}) = f(\hat{x}_{rx} + \Delta x_{rx}, \hat{y}_{rx} + \Delta y_{rx}, \hat{z}_{rx} + \Delta z_{rx}, \hat{\delta}_{rx} + \Delta\delta_{rx}) \tag{1.30}$$

Meaning that the true location of the receiver is the the combination of the initial estimated location $(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})$ and a position displacement $(\Delta x_{rx}, \Delta y_{rx}, \Delta z_{rx}, \Delta\delta_{rx})$.

Expanding the previous expression 1.29 by the Taylor series method and only considering the first-order derivatives:

$$f(\hat{x}_{rx} + \Delta x_{rx}, \hat{y}_{rx} + \Delta y_{rx}, \hat{z}_{rx} + \Delta z_{rx}, \hat{\delta}_{rx} + \Delta\delta_{rx}) = f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx}) + \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{x}_{rx}} \Delta x_{rx} +$$

$$\frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{y}_{rx}} \Delta y_{rx} + \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{z}_{rx}} \Delta z_{rx} + \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{\delta}_{rx}} \Delta\delta_{rx} \tag{1.31}$$

Where:

$$\begin{aligned} \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{x}_{rx}} &= -\frac{x_i - \hat{x}_{rx}}{|\hat{r}|} \\ \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{y}_{rx}} &= -\frac{y_i - \hat{y}_{rx}}{|\hat{r}|} \\ \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{z}_{rx}} &= -\frac{z_i - \hat{z}_{rx}}{|\hat{r}|} \\ \frac{\partial f(\hat{x}_{rx}, \hat{y}_{rx}, \hat{z}_{rx}, \hat{\delta}_{rx})}{\partial \hat{\delta}_{rx}} &= c \end{aligned} \tag{1.32}$$

Being:

$$|\hat{r}| = \sqrt{(x_i - \hat{x}_{rx})^2 + (y_i - \hat{y}_{rx})^2 + (z_i - \hat{z}_{rx})^2} \tag{1.33}$$

Knowing that the estimated pseudorange is the pseudorange from the satellite to the estimated position:

$$\hat{\rho}_i = \sqrt{(x_i - \hat{x}_{rx})^2 + (y_i - \hat{y}_{rx})^2 + (z_i - \hat{z}_{rx})^2} + c\hat{\delta}_{rx} \tag{1.34}$$

The pseudorange to the real receiver position could be approximated as:

$$\rho_i = \hat{\rho}_i - \frac{x_i - \hat{x}_{rx}}{|\hat{r}|}\Delta x_{rx} - \frac{y_i - \hat{y}_{rx}}{|\hat{r}|}\Delta y_{rx} - \frac{z_i - \hat{z}_{rx}}{|\hat{r}|}\Delta z_{rx} + c\Delta\delta_{rx} \tag{1.35}$$

For convenience, the following new variable is introduced. In fact, this new variable corresponds to a unitary vector from the estimated receiver position to the satellite i:

$$\vec{a}_i = (a_{x,i}, a_{y,i}, a_{z,i}) = (\frac{x_i - \hat{x}_{rx}}{|\hat{r}|}, \frac{y_i - \hat{y}_{rx}}{|\hat{r}|}, \frac{z_i - \hat{z}_{rx}}{|\hat{r}|}) \tag{1.36}$$

So that the equation 1.35 can be written as the pseudorange difference:

$$\Delta\rho_i = \hat{\rho}_i - \rho_i = a_{x,i}\Delta x_{rx} + a_{y,i}\Delta y_{rx} + a_{z,i}\Delta z_{rx} - c\Delta\delta_{rx} \tag{1.37}$$

From the forth visible satellites we obtain four following lineal equations. Recall that the objective is to obtain the four unknowns $\Delta x_{rx}, \Delta y_{rx}, \Delta z_{rx}, \Delta\delta_{rx}$ describing the position displacement:

$$\begin{cases} \Delta\rho_1 = a_{x,1}\Delta x_{rx} + a_{y,1}\Delta y_{rx} + a_{z,1}\Delta z_{rx} - c\Delta\delta_{rx} \\ \Delta\rho_2 = a_{x,2}\Delta x_{rx} + a_{y,2}\Delta y_{rx} + a_{z,2}\Delta z_{rx} - c\Delta\delta_{rx} \\ \Delta\rho_3 = a_{x,3}\Delta x_{rx} + a_{y,3}\Delta y_{rx} + a_{z,3}\Delta z_{rx} - c\Delta\delta_{rx} \\ \Delta\rho_4 = a_{x,4}\Delta x_{rx} + a_{y,4}\Delta y_{rx} + a_{z,4}\Delta z_{rx} - c\Delta\delta_{rx} \end{cases} \tag{1.38}$$

Which can be expressed as a matrix equation:

$$\Delta\rho = H\Delta x \tag{1.39}$$

Being $\Delta\rho$ the data vector of the pseudoranges measurements, H the matrix dependent on the relative positions between the receiver estimations and the satellites and Δx the displacement vectors:

$$\Delta\rho = \begin{pmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \Delta\rho_3 \\ \Delta\rho_4 \end{pmatrix} \tag{1.40}$$

$$H = \begin{pmatrix} a_{x,1} & a_{y,1} & a_{z,1} & 1 \\ a_{x,2} & a_{y,2} & a_{z,2} & 1 \\ a_{x,3} & a_{y,3} & a_{z,3} & 1 \\ a_{x,4} & a_{y,4} & a_{z,4} & 1 \end{pmatrix} \tag{1.41}$$

$$\Delta x = \begin{pmatrix} \Delta x_{rx} \\ \Delta y_{rx} \\ \Delta z_{rx} \\ c\Delta\delta_{rx} \end{pmatrix} \tag{1.42}$$

Isolating the position displacement vector:

$$\Delta x = H^{-1}\Delta\rho \tag{1.43}$$

At this point, recalling the expression 1.30 and knowing the position displacement vector, it can be calculated the true receiver's position. Nevertheless, if the initial position approximation is not good enough according to the receiver's accuracy needs, a iteration process shall commence in order to find a better estimation.

CHAPTER 2. STATE OF THE ART RESEARCH

As nowadays there is being a noteworthy growth in safety applications which rely on this technology, the vulnerability towards the spoofing attacks must be confronted in an effective and reliable manner. To do so, firstly it shall be studied the different methods employed by the attackers so as to detect possible techniques to counteract them.

Lately, many mechanisms has been studied and proposed in order to counteract GNSS spoofing attacks. In this section, it will be detailed the current attacks performed by the spoofers and the possible countermeasures.

2.1. GPS Spoofing attacks

The spoofing attacker's intentions are to mislead the receiver and induce him to compute an incorrect position. So as to do so, the receiver shall lock into the malicious signal, which imitates a legitimate GPS signal, without realizing the current locked signal is malevolent. Once the attacker manages to accomplish this first step, then disposes two main strategies to cause the receiver to miscalculate its position.

Henceforth, if the spoofing attack initially presents a different arriving time regarding the legitimate signal, thus the receiver acquires the signal with a different code phase delay, the attack will be designated as non-synchronous. Conversely, it will be denominated as synchronous.

In addition, depending on whether the attack has modified the content of the navigation message or not, the attack will be classified as non-coherent or coherent respectively.

2.1.1. Take-over classification

The take-over is the process in which the spoofer agent takes control of the victim's receivers tracking module. In order to accomplish that, the spoofer can adopt two main strategies, which can be classified depending on the initial synchronization between the legitimate and spoofing signals in the Doppler frequency and code-phase dimensions.

2.1.1.1. Synchronous attack

These kind of attacks are characterized by being initially synchronized in the Doppler frequency and code-phase axis to the legitimate signal and, once they have successfully performed the take-over, modify either the code-phase, the navigation message or both.

In the following figure it is depicted the take-over procedure performed in this type of attacks, which performed the so-called seamless take-over:

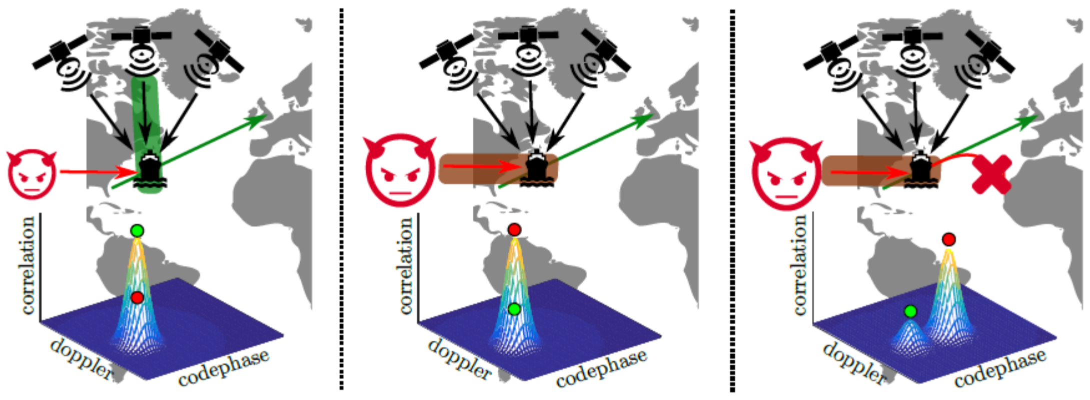


Figure 2.1: Synchronous and non-coherent attack (*seamless take-over attack*) correlation temporal evolution, extracted from (4)

Initially, in the figure located at the left, the spoofer's signal is already perfectly synchronized in both carrier frequency and code-phase parameters. As it can be seen, the malicious signal presents a lower power thus provoking its peak to be unnoticed in the victims receiver's correlation grid.

Afterwards, in the figure in the middle, it can be observed that the spoofer increases its power in order to perform the take-over. Once the take-over is done, the receiver's tracking module has locked into the spoofing signal inadvertently.

Finally, the spoofer proceeds to mislead the victim's receiver by modifying the code-phase, inducing the receiver to incorreclty compute its PVT solutions.

2.1.1.2. Non-synchronous attack

In these attacks, the spoofer agent cannot manage to initially synchronize its malicious signal with the legitimate signal. As a result, the spoofer shall refine the spoofing synchronization by slowly modifying the adjustment of the carrier frequency and the code-phase parameters. Eventually, the spoofer manages to take-over when the receiver unlocks its tracking module of the authentic signals and locks it to the spoofed signals.

As it can be seen in the following picture, the process of trying to synchronize can be detected in the correlation analysis:

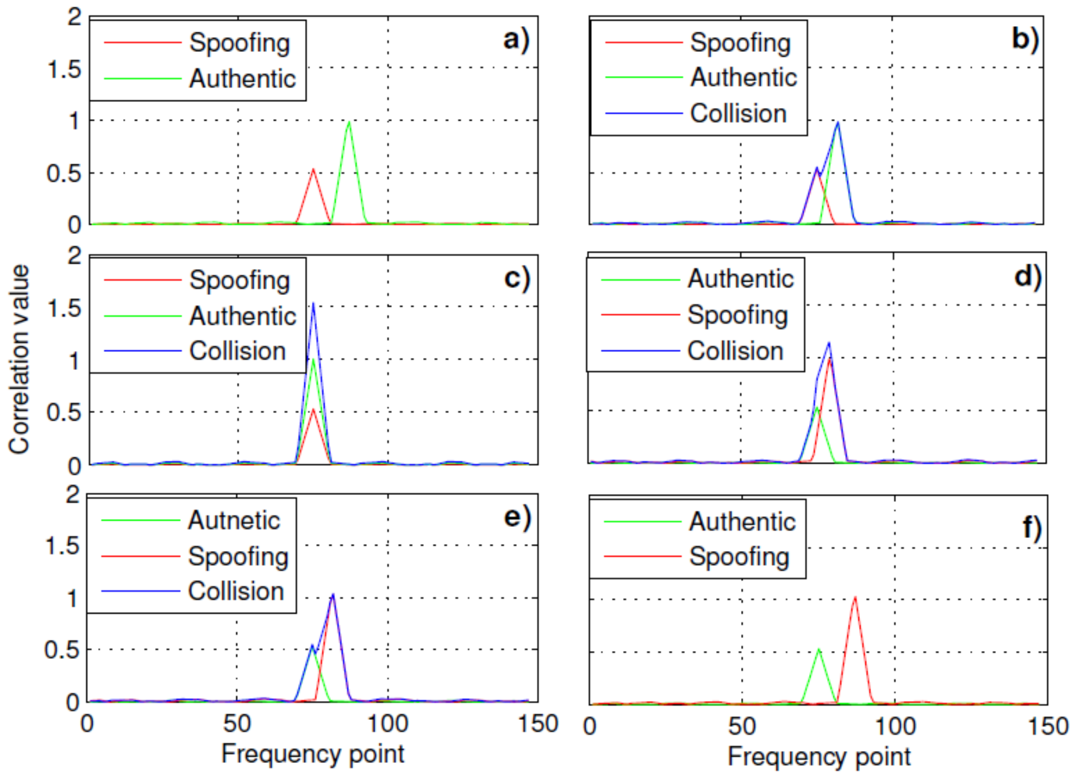


Figure 2.2: Non-synchronous attack correlation temporal evolution, extracted from (24)

In the first situation (case a) it can be seen the initial miss-alignment in the code-phase axis. Consequently, in the correlation measurement it can be appreciated both the authentic and the spoofing signals.

In case b the spoofing signal has refined its code-phase thus reducing the difference with respect to the authentic signal. When both signals peaks are closer than two chips, they start overlapping, causing a distortion in the overall correlation.

In case c the spoofing signal has managed to synchronize its signal with the authentic one, therefore the carrier frequency and the code-phase of the spoofing signal and the authentic signal are equal. At this point, the spoofer increases its power in order to make the malicious peak dominant. When the spoofing peak has a higher value in the receiver correlation measurement, the receiver instantly and inadvertently locks into the spoofing signal.

From cases d to f, the spoofer, as it has managed the locking of the receiver's tracking module, proceeds to deviate the spoofing signal from the authentic signal by changing the code-phase for instance, thus causing an incorrect positioning or timing for the victim receiver.

2.1.2. Drive-away classification

The drive-away manoeuvre is the process the spoofer carries out in order to diverge the spoofer PVT solution from the legitimate one. Mainly, there are two methods.

2.1.2.1. Navigation message modification attack

The first strategy is based on the navigation message content manipulation. By means of altering parameters such as the satellites location and the navigation messages transmission time, the receiver's victim PVT solution can be maliciously manipulated. This attack is also known as non-coherent attack.

Next, it is explained the several implementations of the navigation message manipulations:

- Ephemeris modification attacks:
  The first one consists in modifying the ephemeris parameter $\sqrt{A}$ which represents the square root of the satellite's orbit semi-major axis. In particular, they set this parameter to 0 so as to lead the receiver to believe that this satellite is located at the Earth centre.

  The second one simply replaces the ephemeris data contained in 1-3 subframes with the ephemeris data of another satellite.

- Week number modification attack
  Based on the manipulation of the week number located in subframe 1. For instance, it could be decreased one unit in order to make the receiver assume it is the previous week.

- Date unsynchronization attack
  It simulated rollover events (actually they occur every 20 year approximately) by manipulating the IODC and IODE parameters. This attack damages the receiver permanently if there is no date re-synchronization.

2.1.2.2. Code-phase drive-away attack

The second strategy consist in altering the arriving time by including an offset in the code-phase delay, a crucial parameter to compute the pseudorange. If in this drive-away manoeuvre the navigation message has not been modified, it is also known as coherent attack.

Next, are listed and detailed several spoofing attacks which have been found in the current literature.

Example of a synchronous and non-coherent attack

In the paperwork (3), the authors developed a Phase-Coherent Signal Synthesizer (PCSS). This device receives the broadcasted GPS L1 signals from the in-sight satellites, down-converts these signals and estimates a set of parameters such as the pseudoranges, the Doppler frequency shift, carrier phases and the PVT solution (position, velocity and time). Afterwards, these parameters are used as inputs on the GPS signal synthesizer, which can generate a malicious signal imitating the parameters of the legitimate signal in order to camouflage and mislead the receiver.

Therefore, the PCSS owns the capability to transmit a power-adjustable synchronous (i.e. code-phase coherent) GPS signals and it manages to mislead the target receiver by manipulating the navigation message. In addition, this device is capable of changing the malicious navigation message content in real time.

Example of non-synchronous and non-coherent

In paperwork (5) it is detailed this kind of attack, which is also known as the replay attack. Its fundamental lays on the attacker reception of licit GPS signals and the identical re-transmission of them. In this scenario, it can occur that the attacker relays a signal coming from a satellite which is visible to the attacker but it is not visible for the victim.

To start with, the authors distinguish between the reception and re-transmission at the message/symbol level or at physical level, this latter case meaning that the receiver stores the entire frequency and re-transmits the spread signals.

Another feature discussed in the paper is the selection of the delay of the relay signal t_{replay}, which is the addition of the minimum required time that the receiver spends receiving and generating the relay signal and an additional configurable time τ set by the attacker.

Besides, it is detailed the effect t_{replay} has on the location error of the victim and the time offset or time error, both errors representing the difference between the misled victim's PVT and the actual PVT. In both parameters, the authors state that there exists a proportional relation between the PVT errors and the t_{replay}. In particular, 300 meters error is achieved for each t_{replay} millisecond and less than one millisecond for each t_{replay} second.

2.2. Current anti-spoofing techniques

Depending on its nature, these mechanisms can be classified into cryptographic, physical layer or logical layer solutions. It is important to highlight that each of these solutions owns different capabilities against spoofing attacks, thus some of them being not effective depending on the attacks assets.

2.2.1. Multi-receiver spoofing detection

On the one hand, a multi-receiver can be understood as a multiple single type receivers. On the other hand, it can be seen as the combination of two satellite communication's systems. These two conceptions can be leveraged to detect spoofing attacks.

Regarding the first conception, in the reference (7) it is proposed an anti-spoofing technique based on the cooperation of several synchronized GPS receivers which are aware of the whole GPS receivers network distribution, thus reducing the possible locations to be feasible spoofed by the attacker. This solution solves the issue that a single GPS receiver can be spoofed into a large and diverse number of locations without it realising.

Considering the second stated idea, in reference (4) it is exposed a solution which cross-validates the receiver positions estimated with both GPS and Galileo navigation systems, for instance. Yet, this solution can be overcome if the attacker can spoof both navigation systems.

2.2.2. Cryptographic techniques

This technique is based on the encryption of the GPS signal, which can be implemented at several levels, as the military GPS signal does. Nevertheless, one drawback of this kind of solutions is that it require to modify the current GPS architecture. Furthermore, one crucial asset of GPS communications is the common and public data and PRN codes. Therefore, redesigning this asset would involve re-distribution and management of codes which, considering the vast number of applications that rely on GPS communications, makes this solution unattainable. Another flaw of this sort of solutions is that it does not present protection against signal replay attacks (non-coherent and unmodified message contents).

2.2.2.1. Signatures on navigation messages signal authentication

In reference (14) it is presented a cryptographic solution based on GPS signal authentication. This technique, after studying different possible digital signature protocols such as Timed Efficient Stream Loss-Tolerant Authentication (TESLA)(18), Rivest, Shamir, and Adleman algorithm (RSA)(19), Digital Signature Algorithm (DSA)(19) and Elliptic Curve Digital Signature Algorithm (ECDSA)(20), periodically inserts a 466-bit ECDSA digital signature into the navigation message, leveraging the flexibility of the current GPS civil navigation message that has reserved bits for future applications.

Due to the fact that the navigation messages only permits 238-bit length messages blocks, the signature is divided and transmitted into two navigation messages. Moreover, this two parts of the signature must be comprised between the ephemeris messages (type 10 and 11) and timing message (from type 30 to type 39) in order to fulfill the requirement that ensures a periodicity of 48 seconds of these specific ephemeris and timing messages. Therefore, these insertion of 476 bits (466 signature bits plus 10 extra random bits so-called ’salt’) leads to a periodic five-minutes authentication per channel.

2.2.2.2. Asymmetric security

In reference (15) it is described another cryptographic anti-spoofing mechanism. In this case, it is based on the simultaneous satellite broadcast transmission of the so-called hidden markers, which are rectangular pulses encrypted with a pseudo-random spreading sequences modulated at the specified carrier frequency. This signal possesses a power lower than the noise floor.

Meanwhile, GPS receivers stores in a Random Access Memory (RAM) buffer the encrypted received signal. After a determined time interval, which will be the time period where the system is protected from signal-synthesis attacks, the satellites broadcast a second signal, this time at a higher transmissions power, including the set of parameters that describe the previously sent hidden marker such as the satellite identifier, the satellite location, the transmission time and the key necessary to generate the sequence to despread the hidden marker.

At this point, the receiver can generate the spreading sequence used in the satellite transmitter $s_i(t+\tau)$ and cross-correlate it with the received signal stored in the RAM buffer $B(t)$, thus obtaining the code phase estimation for the cross-correlation values exceeding the determined threshold.

$$C_{i,m}(\tau) = \int_t B(t)\, s_i(t+\tau)dt \tag{2.1}$$

In addition, this paper explains various methods that optimize the data broadcasting. In the general concept previously explained, each satellite would send both a specific hidden message and, subsequently, a message containing the key to decrypt the respective hidden message.

Nevertheless, this method suggests the combination of the second signal describing the hidden markers (each transmitter broadcasts one of these signals) into a single and general signal for all transmitters. Thus there would be one general key which then, by means of applying a function depending on the general key and the satellite identificator, it could be computed the several individual keys that enable the computation of the despreading sequences of each satellite.

2.2.3. Physical layer techniques

2.2.3.1. APT

In reference (4) it is combined two anti-spoofing techniques. The first is the auxiliary peak tracking. Unlike typical GPS receivers where each satellite is acquired, tracked and decoded by a single channel, which keeps track of the highest correlation peak between the specific satellite code replica and the received signal, in this technique it is decided to dedicate more than a single channel per satellite, thus enabling to keep track not only for the highest peak but also for a weaker correlation peaks of a single satellite.

The spoofing attack detection flag is triggered if the code phase difference between peaks exceeds a configurable τ_{max}. As a result, auxiliary peak tracking technique is capable of protecting the GPS receiver from non-synchronized attacks.

Although the attacker could think of eclipsing the authentic GPS signal so that the receiver only detect one correlation peak (the malicious one), apart from transmitting at a higher power, the attacker would need to synchronize to a centimeter level to the real signal in addition to its multi-path components. That entails a lot of complexity and added to the extra complexity it would bring if the receiver was changing its position as the attacker should be able to know the exact trajectory, makes it quite arduous to spoof it.

2.2.3.2. Output correlator analysis (Power level check)

A spoofing detection technique was proposed in the article (11), where they based the detection of the spoofing signal on the analysis of the output correlator. It is stated that, when there is only the authentic signal (plus noise), the distribution of squared amplitude of the output correlator can be modelled as a non-central Chi-Squared.

Nevertheless, when both the spoofing and authentic signals are present, the output correlator does not follow a Chi-Squared distribution anymore. Therefore, there are defined some thresholds on the Chi-squared distribution of the output correlators to identify whether there is a spoofing signal or not.

In the same paper it is presented a solution apart from the detection method. It is a vector-based receiver structure which is able to use the navigation level measurements to drive the Numerically-Controlled Oscillator (NCO) of the spoofed tracking loops after successful detection of the spoofing attack. Therefore, the authentic-spoofing interaction can be bridged by a vector-based receiver if a valid position solution is available from other un-spoofed PRNs or additional positioning sensors.

2.2.3.3. Spatial signal processing

On the other hand, the references (12) and (8) describe an alternative spoofing detection: spatial signal processing. As the aim is to figure out the direction of arrival of the signals and the direction of arrival corresponds to the phase delay differences of signals at the output of receiving antennas (antenna array), this technique is based on the comparison between phase delays.

As spoofing signals usually are transmitted from a single source, they arrive to the receiver with the same direction. Unlike spoofing signals, authentic GPS signals arrive from different directions. Therefore, GPS spoofing is triggered when several received signals have very similar directions of arrivals.
To mitigate spoofing signals, spatial filtering is presented and tested as a robust solution as, by optimally weighing the array, a high attenuation of the undesired signals can be achieved.

2.2.3.4. Doppler Shift validation

According to (5), another physical anti-spoofing technique can be based on the Doppler frequency shift, that is the frequency deviation from the transmitted frequency due to the relative motion between the transmitter (i.e, the satellite) and the receiver. As the navigation message include the necessary information so as to enable the receiver to compute the satellite trajectory and velocity, the receiver is capable to predict the future Doppler shift. In particular, the following general formula is used:

$$f_{rx} = f_{tx}\left(1 + \frac{v_{rx} - v_{tx}}{c}\right) \tag{2.2}$$

Consequently, the locked signal Doppler shift and the predicted Doppler shift is continuously compared and, in case they exceed the maximum deviation threshold, a spoofing attack is reported.

Approximately, in normal conditions the maximum Doppler shift time rate caused by the relative motion of a satellite and a vehicle is 40Hz/min and the Doppler shift about ±5KHz or ±50KHz, depending whether the receiver is static or mobile. As the attacker is either an static station or a low-speed station (compared to the satellite speed), it will not be able to produce a Doppler shift unless the attacker modified the emitted frequency in order to emulate the Doppler shift the receiver would obtain from the actual satellite.

Yet, in this paper they reflect that, in both cases, it is detected a Doppler shift during a specific time which differs from the predicted frequency shift.

2.2.4. Logical layer techniques

Based on the inspection and monitoring of the decoded navigation message aimed to detect inconsistencies or abnormalities among the receiver's observables and PVT solutions. There are attackers that change the navigation's message contents to deceive the receiver.

Currently, there are proposed several logical layer techniques which will be briefly explained below.

2.2.4.1. Position checks

a Satellite orbital position validation

Firstly, in (4) it is proposed a technique that relies on the validation of the satellite orbital positions, which in practise it is implemented by the monitoring and validation of the orbital parameters. Apart from the satellite orbital positions, they also monitor the almanac, ephemeris and ionospheric model data which combined with the comparison of the same data delivered by a reliable third-party assure that there are no unexpected alteration.

b Inertial sensors validation

In (5) is explained this technique whose fundamentals lay on estimating the position using a non-GNSS dependent system in order to compare it with the GNSS position solution. In case of a discrepancy larger than a certain threshold, the alarm will be triggered. In this specific technique, this independent system consist of sensors as altimeters, speedometers and odometers, for instance. Nonetheless, one significant drawback of this method is that sensors lose accuracy following an exponential function as time goes by.

2.2.4.2. Clock consistency checks

In the same paper work (4) it is implemented a logical layer technique regarding the clock consistency by inspecting the time of week or transmission time. Considering the navigation message structure and taking advantage of the fact that each of the five subframes broadcasted with a period of 6 seconds by each satellite contains a shortened version of the particular subframe transmission time, the implemented solution stores the time of week of the previous subframes and checks whether the time has been increased exactly 6 seconds or not. In this latter case, it triggers a flag reporting the inconsistency probably owing to a spoofing attack.

Finally, apart from the previous explained techniques, it could be possible to implement logical solutions based on the static position checks, position jumps, velocity consistency and abnormal position checks by storing previous values and setting suitable thresholds that detected disparities.

2.3. Current spoofing mitigation techniques

Next, it is detailed a method that, once the anti-spoofing detection is triggered, it attempts to overcome the attack and recovering the authentic GNSS signal.

2.3.1. LSR

In the paperwork (6), in addition to the exposed spoofing detection technique, it is also presented this spoofing mitigation technique. In the stated reference, it is assumed a synchronous and non-coherent attack, scenario in which the receiver actually collects the addition of the legitimate and spoofing signals:

$$s_{rx} = s_{legitimate} + s_{spoofing} \tag{2.3}$$

The LSR technique consists in, when an spoofing attack is detected, generating a replica of the spoofing signal in order to recover the legitimate GNSS signal by subtracting the spoofing signal replica to the received signal:

$$s_{rx} = s_{legitimate} + s_{spoofing} - s_{spoofing} = s_{legitimate} \tag{2.4}$$

Nevertheless, so as to achieve generating the spoofing replica, it is necessary to consider the different parameters that define the received GNSS signal:

- Amplitude
  To estimate the amplitude of the spoofing signal it is necessary to compute the cross-correlation of the received signal and the local replica. At the peak, where the squared magnitude of the correlation search grid (1.9) is maximized, it occurs:

  $$\left|\hat{R}_{x,d}(f_{Doppler}, \tau)\right|^2 = |a|^2 K^2 \tag{2.5}$$

  Being a the amplitude of the received signal and K the number of correlated samples. Therefore, as the spoofing signal's amplitude is significantly higher than the authentic GNSS signal, the spoofing signal's amplitude can be estimated as:

  $$|a| = \frac{\sqrt{\left|\hat{R}_{x,d}(f_{Doppler}, \tau)\right|}}{K^2} \tag{2.6}$$

- Code phase delay
  Estimated in the acquisition module.

- Doppler frequency shift
  Estimated in the acquisition module.

- Navigation bits
  Obtained in the tracking module.

One weak point of this implementation is the ineffectiveness towards synchronous and coherent attacks, also known as stealthy attacks, as the navigation bits are the same.

CHAPTER 3. BORRE'S SOFTWARE RECEIVER IMPLEMENTATION

In reference (17) it is included a complete GPS software receiver implemented in MATLAB able to perform acquisition, code and carrier tracking, navigation bit extraction, navigation data decoding, pseudorange estimation, and position computations. These software, as well as the book itself, is considered a reference in the satellite-communications environment, thus making it ideal to set our basis on it.

In the book, it is explained that the reasons to have chosen MATLAB coding language are; the wide-spread use in universities of this programming language, the flexibility it presents, the relative low difficulty involved in it and the remarkable graphic tools to represent results.

Regarding the GNSS software receiver provided in the book, it is worth mentioning that it has been chosen to be a single-frequency receiver using the C/A code on L1 for GPS for the following reasons; for the fact that it simplifies the receiver architecture, the remarkable accuracy of the L1 band, when combined with augmentation systems (Satellite-Based Augmentation Systems (SBAS)), compared to the dual frequency GNSS receivers and the compatibility with the European GNSS Galileo which enhances availability.

In particular, it has been decided to take as initial code the latest version of the GPS L1 C/A Borre's software included in the repository (25) as it included facilitating features regarding the processing of signals containing complex samples.

3.1. Code structure

At a high level point of view, the code structure is based on the scheme presented in figure 3.1. The front-end block is not actually implemented as the input signals are already down-converted at an intermediate frequency.

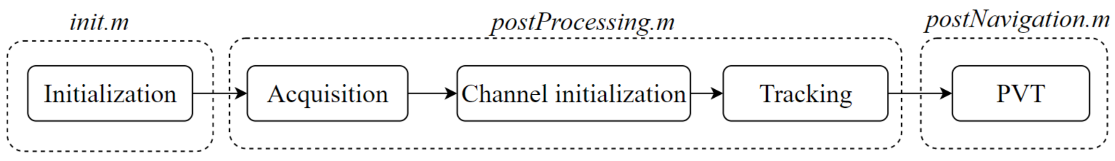


Figure 3.1: Borre's high level architecture (Own source)

Firstly, an initialization process is carried out, in which it is set the global parameters of the receiver and several plots of the incoming data are display into the user screen.

Subsequently, the acquisition process is performed in order to detect the presence of any GPS signal and, if so, it provides a coarse estimation of the Doppler frequency and the code phase for each detected signal.

The next step is to initialize the channels which will serve as an input for the tracking block.

Afterwards, the tracking block is initiated and it goes through all channels and tracks several parameters such as the Doppler frequency, the code phase, the output correlators, etc.

Finally, once the navigation bits are extracted by the tracking function, the PTV block is called so as to compute the signal transmission time and the pseudoranges, necessary to obtain the position.

In this section, names followed by .m (for instance, *acquisition.m*) indicate that it is a Matlab file either a function or a script, which will be specified in each statement.

3.1.1. Initialization: init.m

The initialization process is carried out in the init.m Matlab script. Its main tasks to perform are the following ones:

- Reset the workspace:
  Delete variable values from previous executions and close all previous figures.
- Include folders:
  Regarding the software receiver functions and additional common functions necessary for the several computations.
- Display a welcoming and disclaimer message in the command window
- Create the *settings* structure
  settings is a global structure available to access by for all receiver blocks contain all the variables that characterize a GPS signal-processing chain. This approach enables a centralized and flexible management of the software.

  The different parameters are sorted out by its nature and are presented in the following table (3.1):

Structure	Section	Parameter	Units/Values
settings	Processing parameters	msToProcess	[ms]
		numberOfChannels	Natural number
		skipNumberOfBytes	Natural number
	Signal file parameters	fileName	Char
		dataType	'int8' or 'int16'
		fileType	'1' or '2'
		IF	[Hz]
		samplingFreq	[Hz]
		codeFreqBasis	[Hz]
		codeLength	[chips]
	Acquisition parameters	skipAcquisition	'0' or '1'
		acqSatelliteList	Natural number's vector
		acqSearchBand	[KHz]
		acqNonCohTime	Natural number
		acqThreshold	Positive double
		acqSearchStep	[Hz]
		resamplingThreshold	[Hz]
		resamplingflag	'0' or '1'
	Tracking parameters	dllDampingRatio	[-]
		dllNoiseBandwidth	[Hz]
		dllCorrelatorSpacing	[chips]
		pllDampingRatio	[-]
		pllNoiseBandwidth	[Hz]
		intTime	[s]
	PVT parameters	navSolPeriod	[ms]
		elevationMask	[°]
		useTropCorr	'0' or '1'
	Plots parameters	plotAcquisition	'0' or '1'
		plotTracking	'0' or '1'
		plotNavigation	'0' or '1'
	Constants	c	[m/s]
		startOffset	[ms]
	CN0 parameters	accTime	[s]
		VSMinterval	[ms]

Table 3.1: *settings* structure parameters

- Display some initial plots of the incoming signal:
  The function *probeData.m* is responsible for displaying initial plots of the incoming signal's raw data such as the time domain plot, the frequency domain plot and the histogram. In figure 3.2 it is shown an example:

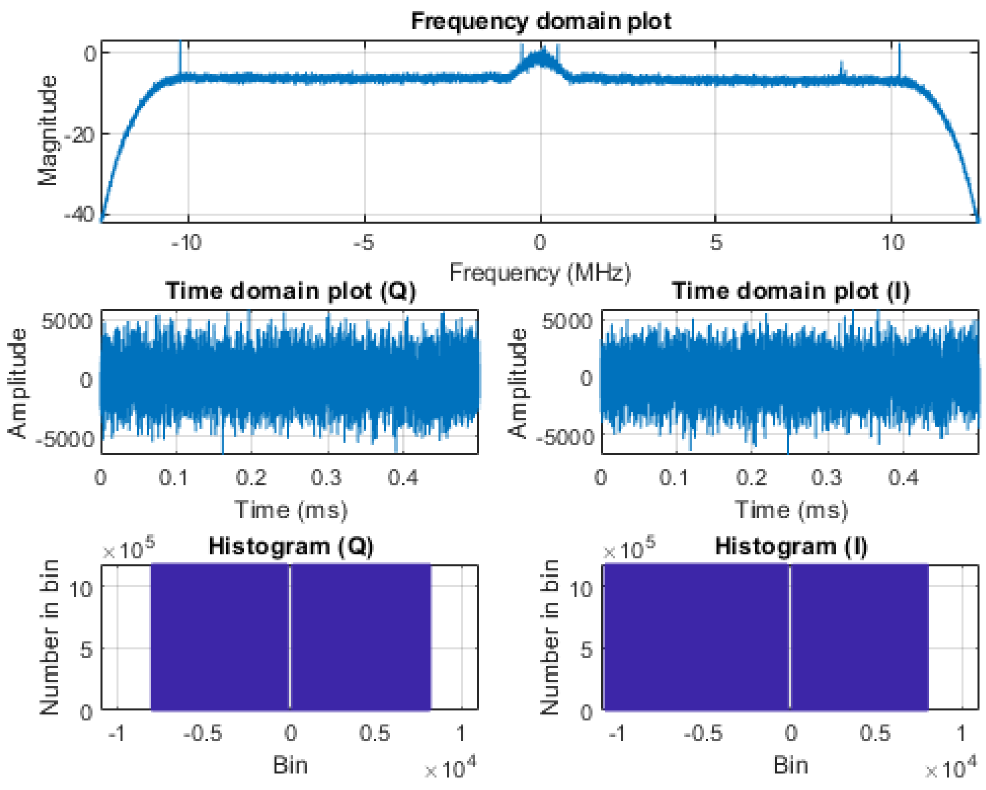


Figure 3.2: Output plots of the *probeData.m* function for the TEXBAT signal ”Clean Static” (Own source)

- Launch next block (*postProcessing.m* Matlab function)

3.1.2. Acquisition and tracking: *postProcessing.m*

In this Matlab script takes place the entire signal processing. It includes, the acquisition, the channel initialization and the tracking processes. Furthermore, it calls the last block responsible for obtaining the PVT solutions.

3.1.2.1. Acquisition

The acquisition takes place at the *acquisition.m* function.
On the one hand, it receives as input 42ms of the incoming signal from the RF front-end and the settings structure. This function searches for the GPS signals of all satellites included in the settings structure parameter *acqSatelliteList*, which in normal conditions is a vector listing all 32 GPS L1 C/A satellites. Besides, once it detects a GPS signal, it refines the Doppler frequency estimation in the so called Fine Resolution Frequency Search (FRFS) in order to facilitate the tracking procedure.
On the other hand, this function returns a structure called *acqResults*, which gathers the estimated Doppler frequencies, the code phases and the statistic metrics of the detected signals.

In the following flowchart (figure 3.3) it is represented the acquisition process:

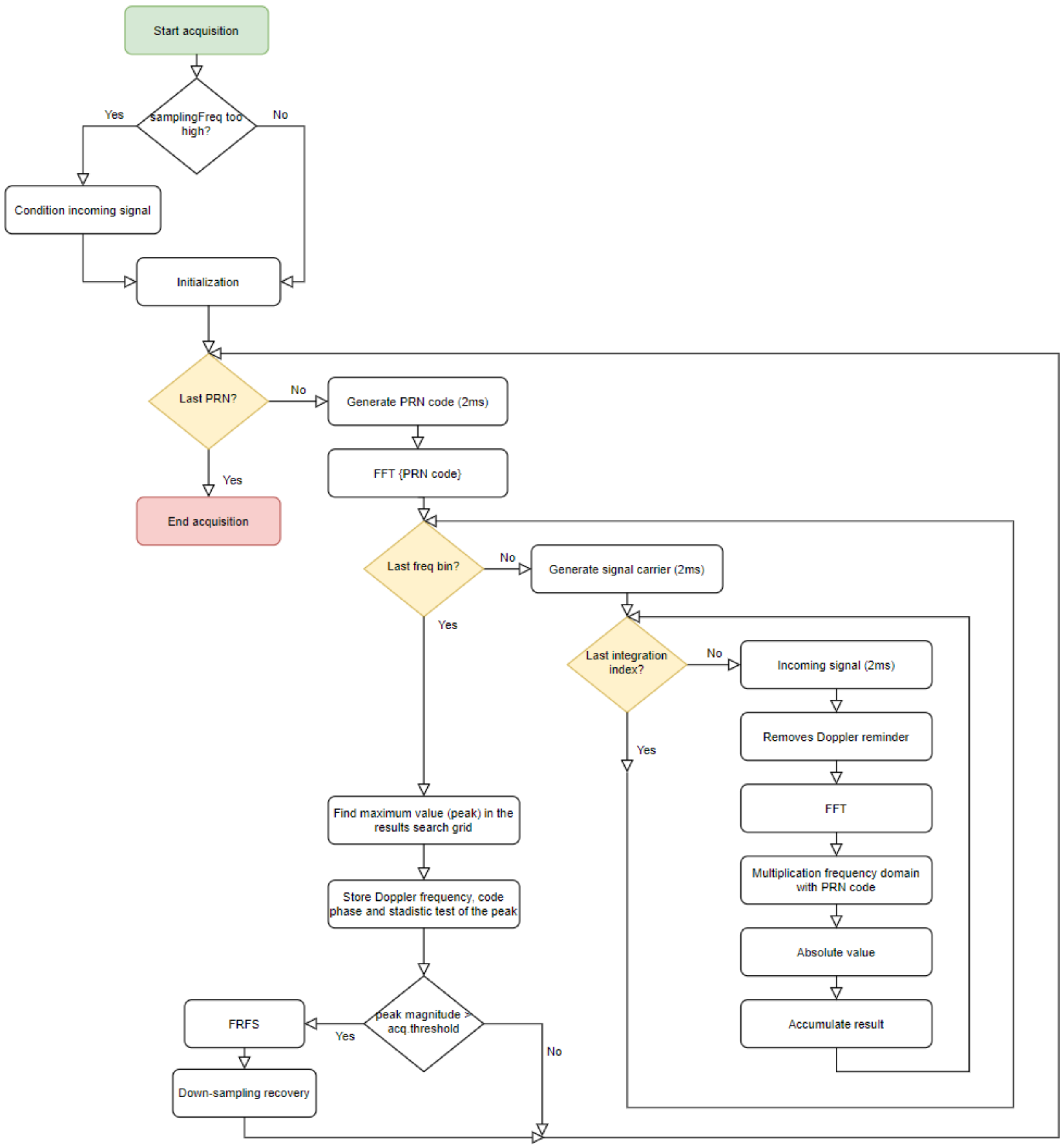


Figure 3.3: Acquisition flowchart (Own source)

The first process is to apply a re-sampling strategy if the incoming signal's intermediate frequency is too high. By doing that, it speeds up the acquisition. In order to launch this re-sampling strategy, it is necessary that both the sampling frequency exceeds an adjustable re-sampling threshold and that the re-sampling flag is set to 1. All this parameters are configurable in the settings structure.
Secondly, an initialization of several variables needed for the acquisition, the *acqResults*

and other variables concerning the FRFS.
Subsequently, it goes through all possible GPS L1 C/A satellites and proceeds to compute the test statistic in all points of the search grid by following the PCPS acquisition implementation strategy (see section 1.2. in particular figure 1.6 for more details). It is worth mentioning that the test statistic has been computed as a ratio between the integrated results and a signal power estimation (and also multiplied by the integration times).

$$Test_{statistic} = \frac{N_{int} \sum_{i=1}^{N_{int}} \hat{R}_{x,d}(f_{Doppler}, \tau)}{\hat{P}_x} \tag{3.1}$$

Moreover, the acquisition implementation includes two special features worth mentioning:

- Bit transition resilience:
  This feature allows the proper acquisition even in the case that the fragment of the signal selected to perform the acquisition contains a bit transition in the navigation bits.

  The problems arises as, in case of bit transition, it equivalent as if the part of the receiver's code after the transition was inverted, as it is multiplied by the navigation bits. Consequently, due to the fact that both codes from the received signal and the local replicas differ, the correlation will not result in a high value (peak) although the code-phase estimation is accurate. To sum up, without this feature, if the acquired signal millisecond contained a bit transition, it would not appear a peak in the correlation even if the satellite was present.

  The solution to this issue lays on performing the acquisition of a longer signal (2 milliseconds instead of 1 millisecond). It leverages the fact that there cannot be bit transition in two adjacent milliseconds (1 bit lasts 20 milliseconds). By adopting this measure, in case of bit transition in one of the two acquired milliseconds, it can be ensured that at least in one millisecond it will detected the present satellite.

  As a consequence of taking 2 milliseconds of the incoming signal, both the local replicas of PRN and signal carrier shall coincide in dimensions, therefore, local replicas shall contain the samples equivalent to 2 milliseconds as well. In particular on the PRN local replica, it is implemented by adding to the 1 millisecond PRN code a zero-padding extra millisecond.

- Search grid integration:
  In order to minimize the acquisition estimations noise it is performed an integration or accumulation of the results of several milliseconds (as a maximum of 42 ms as it is the signal fragment that inputs the function). The stated process takes groups of two consecutive signal milliseconds, performing the PCPS on them and then accumulating it to the search grid results.

Once it has acquired a certain satellite, it proceeds to refine the Doppler frequency acquisition, thus facilitating the tracking's follow-up. This process is called FRFS and it can be understood with the flowchart of the figure 3.4:

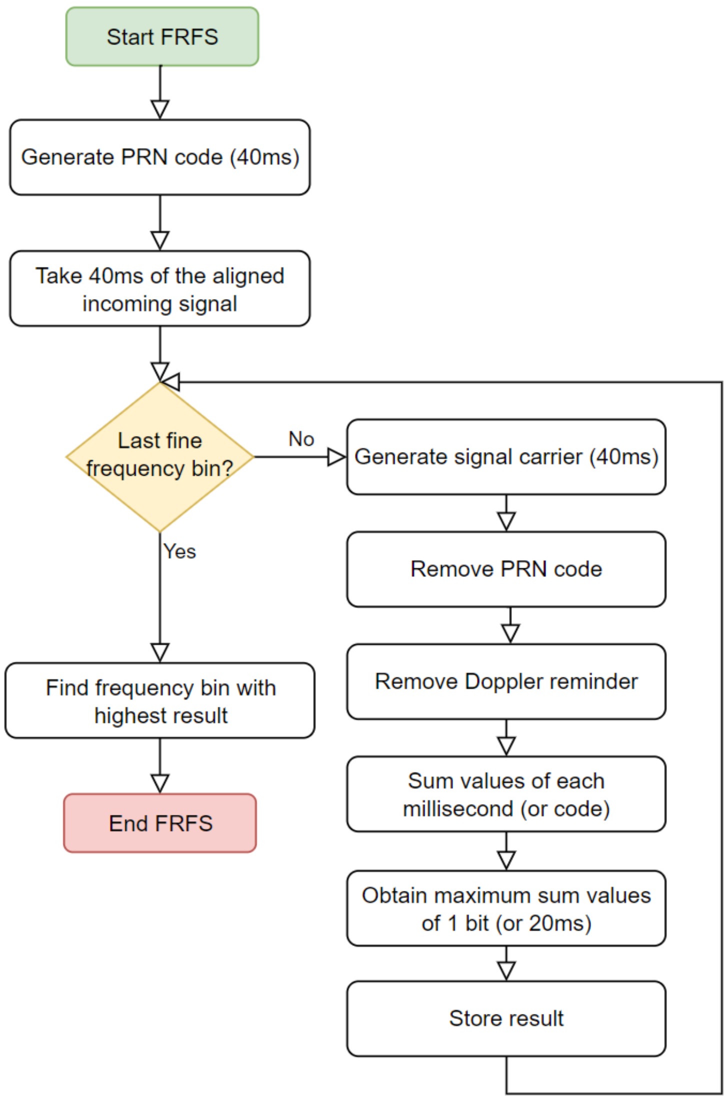

Start FRFS
Generate PRN code (40ms)
Take 40ms of the aligned incoming signal
Last fine frequency bin?
No
Generate signal carrier (40ms)
Remove PRN code
Remove Doppler reminder
Sum values of each millisecond (or code)
Obtain maximum sum values of 1 bit (or 20ms)
Store result
Yes
Find frequency bin with highest result
End FRFS

Figure 3.4: FRFS flowchart (Own source)

Next, some details about the previous flowchart (figure 3.4) are presented.

Firstly, on the one hand it generates a 40-milliseconds-long code local replica by concatenating 40 PRN codes of the corresponding satellite.

On the other hand, it takes 40 milliseconds of the aligned incoming signal. It is aligned as the starting point of these incoming signal fragment starts at the code-phase coarse estimation determined in the previous coarse acquisition procedure. Recall that the code-phase indicates the time displacement at which 1-millisecond of the received signal is aligned with the local code replica. Hence, if the first signal millisecond is aligned with the 1-millisecond-long code local replica, the following 39 signal milliseconds will be aligned with the 39 concatenated PRN codes.

The following step is to traverse all frequency fine bins (which are centered at the coarse Doppler frequency estimation and are spaced closer than the frequency coarse bins) and generate different local 40-millisecond signal carriers.

The PRN code removal or despreading code process is carried out by multiplying the

40-milliseconds-long incoming signal with the 40-milliseconds-long code local replica. In section b it can be found more details about its fundamentals.

After having despread the code, the next step it to remove the Doppler reminder. In order to do that, it is multiplied the despread incoming signal with the 40-millisecond signal carrier. More details about the signal down-convertion can be found in section a. The aim is to match the local signal carrier with the actual Doppler frequency, in fact, if the perfect matching is managed, it would lead to the perfect extraction of the navigation bits (ideally belong to real domain). Being $r_d(t)$ the despread incoming signal:

$$\begin{aligned} Bits_{Navigation} &= r_d(t)e^{-j\hat{\phi}_{Doppler}(t)} \\ &= D(t)e^{j\phi_{Doppler}(t)}e^{-j\hat{\phi}_{Doppler}(t)} \\ &= D(t) \end{aligned} \tag{3.2}$$

However, if the local signal carrier is not perfectly matched, the navigation bits are complex:

$$Bits_{Navigation} = D(t)e^{j(\phi_{Doppler}(t)-\hat{\phi}_{Doppler}(t))} \tag{3.3}$$

The following code is based on the idea that, the better the fine Doppler frequency estimation matches the actual Doppler frequency, the lower the complex phases of the navigation bits samples are (ideally the navigation bits will be real instead of complex). In order to detect the frequency bin that produces the navigation bits samples with lower complex phases, it is performed the following steps for each frequency bin:

- Sum values of each millisecond (or code):
  It is stored in a vector the sum of the 40 individual milliseconds that comprise the base-band signal. Nonetheless, in this 40 milliseconds of signal, bit transitions have occurred, as the bit lasts 20 milliseconds. Hence, if the bit transitions from ”1” to ”-1” or from ”-1” to ”1”, the sum of the samples of this millisecond containing the bit transition will be lower than the sum of a millisecond which does not contain the bit transition.

- Obtain maximum sum values of 1 bit (or 20 ms):
  It is considered the previous vector that contains the sum of the 40 individual milliseconds comprised in the base-band signal and it is examined with the objective to find the 20 consecutive milliseconds (or bit) with the higher absolute value of the overall sum. This 20 consecutive milliseconds ensure that no bit transition is present, and therefore it is stored the value of the samples of an entire bit, which for sure it does not contain a bit transition.

Once it has been computed the sum of the samples of a bit (containing no bit transition) for each frequency bin, it is selected the frequency bin that has led to the higher sum value, which is the frequency bin that that offers the best Doppler frequency estimation.

Ideally, if a frequency bin perfectly matches the actual Doppler frequency (i.e. $f_{Doppler} = \hat{f}_{Doppler}$), the base band signal is equal to $D(t)$, which is real. Therefore, the absolute value of the sum of the samples will be higher than if a frequency bin that does not perfectly match the Doppler frequency, as $D(t)$ will contain complex values.

At this point, the acquisition function has been able to detect the present satellites signals and estimate its main unknowns, both the Doppler frequency and the code phase.

With this information it is created the structure *acqResults*, from which allows to both plot the acquisition results and create the channel structure, needed to initialize the tracking module.

- Plot acquisition
  The acquisition plotting is performed in the *plotAcquisition.m* function and displays a graph containing the entire satellite constellation statistic test determined in the acquisition module.
  Next, in figure 3.5 it is shown an example of the presented plot:

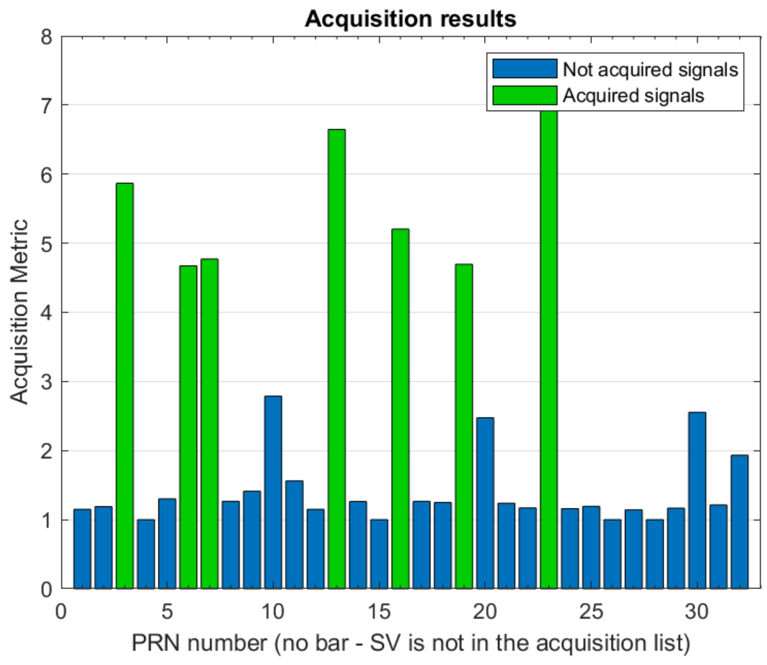


Figure 3.5: Acquisition results plot (Own source)

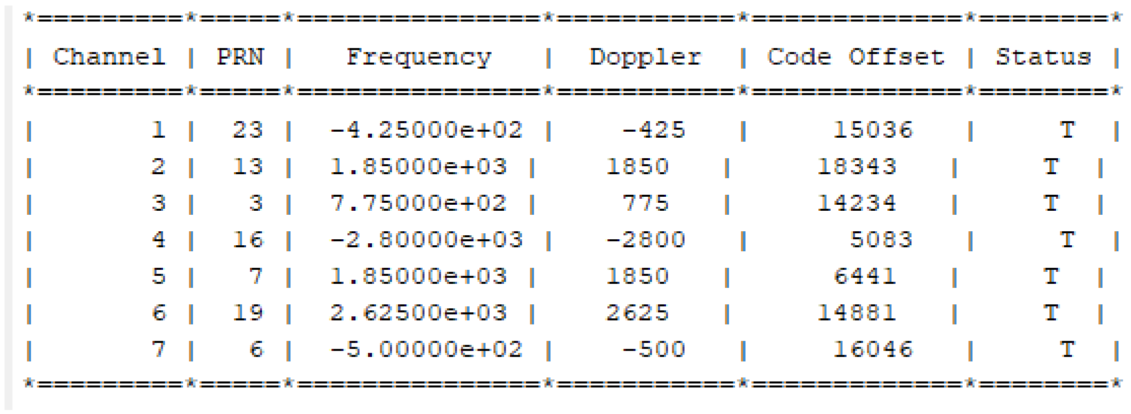
In this particular example, satellites with PRN 3, 6, 7, 13, 16 19 and 23 have been acquired.

3.1.2.2. Channel initialization

In Borre’s original code, the channel initialization process is expressed in the creation of the *channel* structure. This process takes place at the *preRun.m* function, which receives as an input the *acqResults* structure and returns the channel structure needed for the tracking module launching.

As the structure name suggest, this function initialize the tracking channels from the acquisition data. The channel allocation is based on the acquired signal strength, reserving the first channels for the strongest acquired signals. The other parameters the channel structure stores are the acquisition results estimations and the channel status. Regarding the channel status, it can only adopt the *’T’* value for tracking purposes and the *’-’* for not specified purposes.

An example of the structure can be seen in figure 3.6:



```
*=========*=====*===============*===========*=============*========*
| Channel | PRN |   Frequency   |  Doppler  | Code Offset | Status |
*=========*=====*===============*===========*=============*========*
|       1 |  23 | -4.25000e+02 |    -425   |    15036    |     T  |
|       2 |  13 | 1.85000e+03 |   1850    |   18343    |     T  |
|       3 |   3 | 7.75000e+02 |    775    |   14234    |     T  |
|       4 |  16 | -2.80000e+03 |   -2800   |     5083    |     T  |
|       5 |   7 | 1.85000e+03 |   1850    |    6441    |     T  |
|       6 |  19 | 2.62500e+03 |   2625    |   14881    |     T  |
|       7 |   6 | -5.00000e+02 |    -500   |    16046    |     T  |
*=========*=====*===============*===========*=============*========*
```

Figure 3.6: *channel* structure example (Own source)

3.1.2.3. Tracking

Once the acquisition has managed to obtain the acquisition results and has created the channel structure, the tracking module can be initialized. The tracking process occurs at the *tracking.m* function and its main objective is to perform the code phase and carrier tracking for all channels. As mentioned before, it receives as input the channel structure and returns a new structure called *trackResults*, in which all the tracking results are stored.

The entire tracking process is depicted in the following flowchart (figure 3.7):

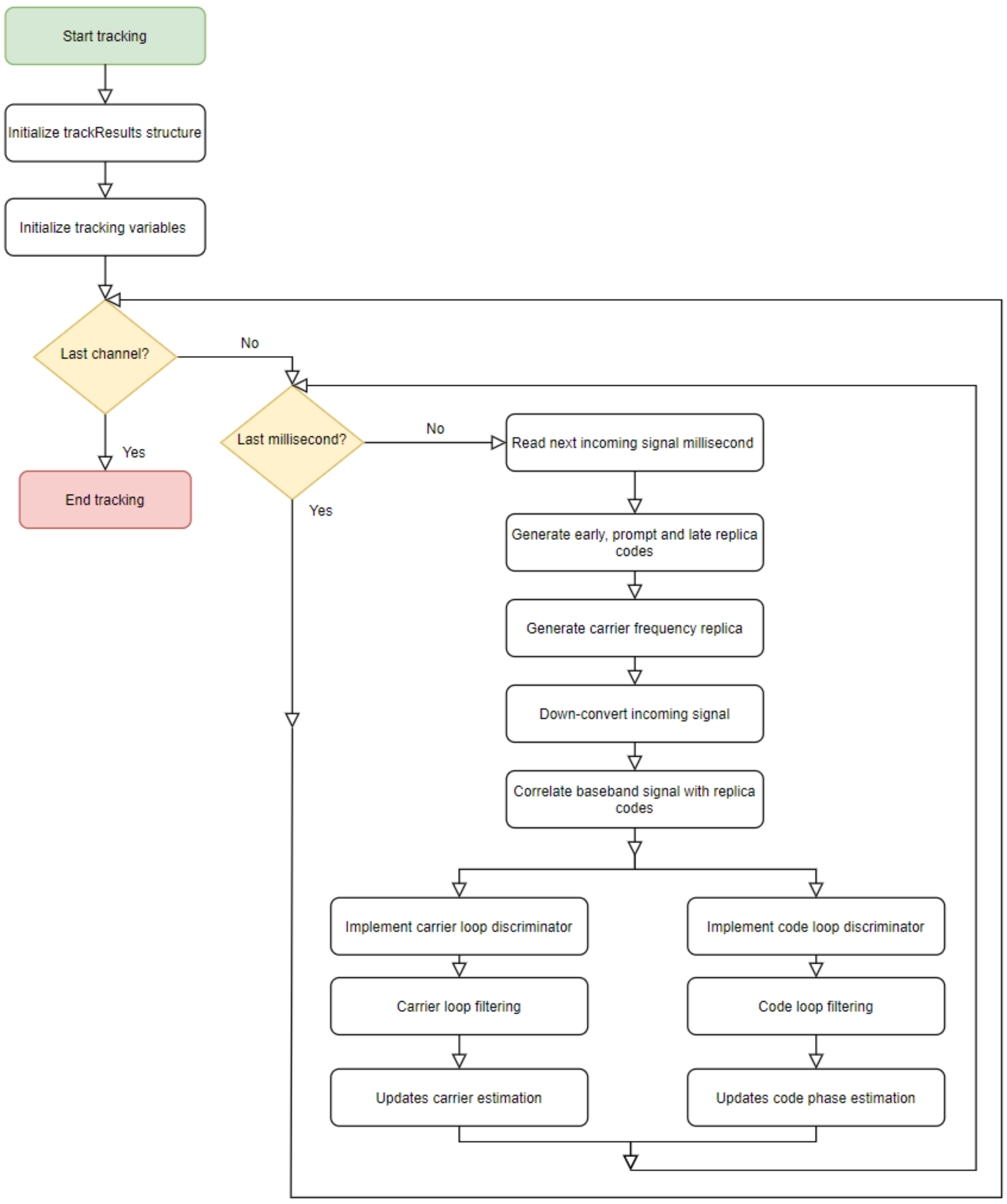

Start tracking
Initialize trackResults structure
Initialize tracking variables
Last channel?
No
Yes
End tracking
Last millisecond?
No
Yes
Read next incoming signal millisecond
Generate early, prompt and late replica codes
Generate carrier frequency replica
Down-convert incoming signal
Correlate baseband signal with replica codes
Implement carrier loop discriminator
Implement code loop discriminator
Carrier loop filtering
Code loop filtering
Updates carrier estimation
Updates code phase estimation

Figure 3.7: Tracking flowchart (Own source)

As the flowchart indicates, the tracking module goes through all channels comprised in the input channel structure and, for each channel, it processes the incoming signal millisecond per millisecond.

For each millisecond, it down-converts the incoming signal by multiplying it by the local carrier signal and generates three outputs correlators (early, prompt and late) by multiplying the previous down-converted signal with the three code replicas. The three output correlators will the be input of the carrier and code discriminators.

Regarding the discriminators, this software implements the following ones:

- Carrier Loop Discriminator
  It is sensitive to bit transitions and it is defined as:

$$\phi_e = ATAN2\frac{Qp}{Ip} \tag{3.4}$$

- Code Phase Loop discriminator
  It is a version of a non-coherent normalized early minus late power discriminator that presents a remarkable estimation when the code phase error exceeds half a chip. It is defined as:

$$\tau_{error} = \frac{\sqrt{{I_E}^2 + {Q_E}^2} - \sqrt{{I_L}^2 + {Q_L}^2}}{\sqrt{{I_E}^2 + {Q_E}^2} + \sqrt{{I_L}^2 + {Q_L}^2}} \tag{3.5}$$

3.1.3. Navigation solutions: *Navigation.m*

The last software main block is the responsible for computing navigation solutions for the receiver. To do so, it obtains pseudoranges, satellite positions and finally performs a coordinate conversion from the World Geodetic System 1984 (WGS84) system to the Universal Transverse Mercator (UTM), geocentric or any additional coordinate system.
The following figure (3.8) contains the flowchart depicting its work flow:

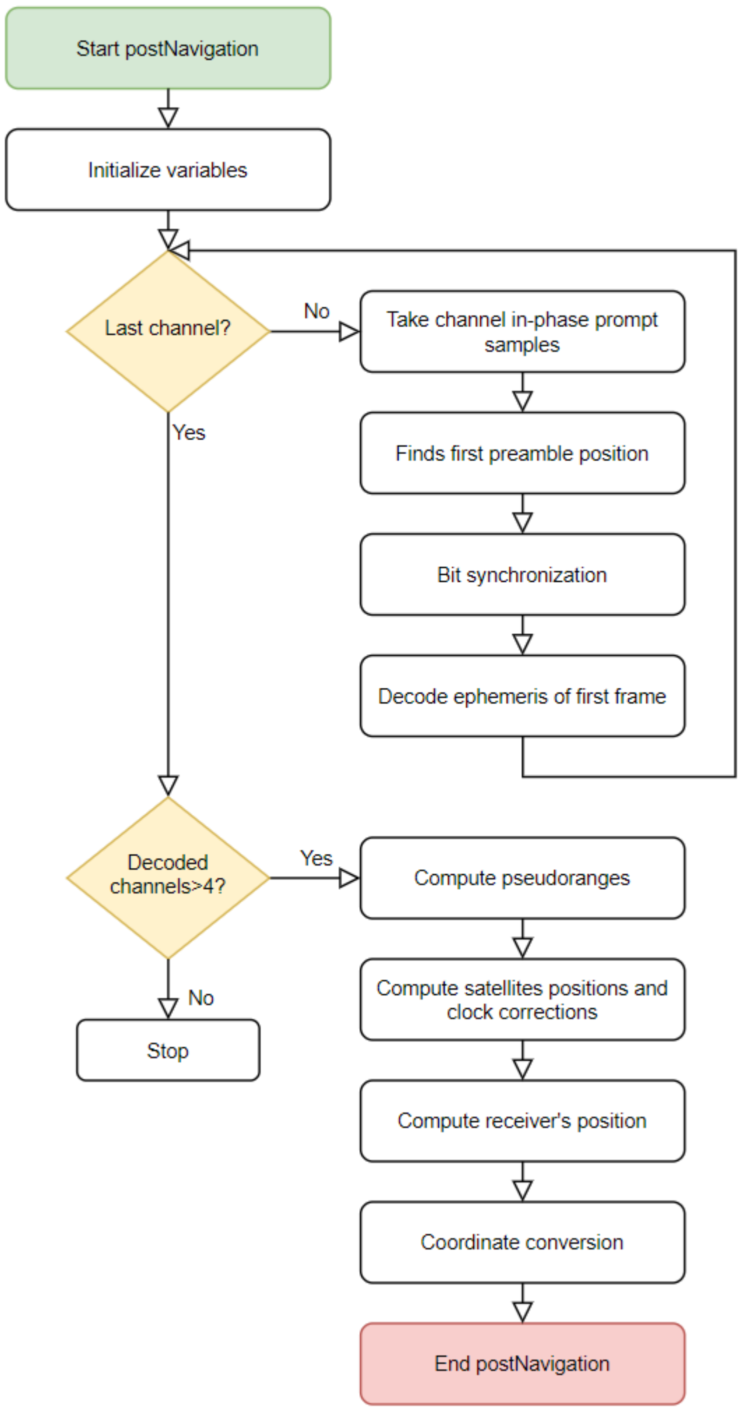

Start postNavigation
Initialize variables
Last channel?
No
Yes
Take channel in-phase prompt samples
Finds first preamble position
Bit synchronization
Decode ephemeris of first frame
Decoded channels>4?
Yes
No
Stop
Compute pseudoranges
Compute satellites positions and clock corrections
Compute receiver's position
Coordinate conversion
End postNavigation

Figure 3.8: Obtaining navigation solution flowchart (Own source)

The flowchart shows that, after initializing some variables, the function goes through all channels with tracking results and decodes the first frame ephemeris. The process of decoding the ephemeris of the first frame requires some previous work, details of which are:

- Take channel in-phase prompt samples:
  The tracking result parameter I_P stores the in-phase prompt samples, which are the received navigation bits.

- Find first preamble position:
  At this point, it is important to recall that the tracking module processes the entire incoming signal by processing fragments of 1 millisecond, meaning that for each millisecond it obtains a value of each tracking result parameter. Therefore, the in-phase prompt correlator samples are obtained at 1 millisecond sampling period.

  As it is performed the preamble identification before the bit synchronization, the correlation shall be done with an equivalent preamble in samples instead of the 8-bit-long preamble introduced in previous sections. In particular, the used preamble is 160 samples long since each bit of the 8-bit word is now takes 20 samples (1 bit is 20 millisecond-long and each in-phase prompt samples correspond to one millisecond).

  With the stated preamble it is correlated the incoming signal and it is decided to be the start of the subframe when there are two consecutive potential subframe starts separated 6000 samples (equivalent to 6 seconds).

- Bit synchronization:
  In this section is performed the bit rate adaptation explained in the previous section 1.4.0.2.. However, the sampling averaging process is not an arithmetic averaging. Instead, the resulting bit takes as a value the sum of the 20 samples.

- Decode ephemeris of the first frame:
  Finally, the receiver can identify the beginning and the end of a frame, therefore, following (23), can decode the navigation message. In this software version, it is only decoded the first, second and third subframe, thus neglecting the almanac information transmitted in the forth and fifth subframes.

Once it has decoded the navigation message of the acquired and tracked satellites, if there are at least four of them, it can compute the receiver's position. This processes are not detailed as the project does not involve anything of this part, however, some details are provided in section 1.4.0.4.

CHAPTER 4. SIGNAL ANALYSIS

4.0.1. Documentation

The studied anti-spoofing techniques will be tested under the TEXBAT, a public database elaborated in the Texas University radionavigation laboratory which contains different GPS L1 C/A signals affected with several types of spoofing attacks. As held in (21), the TEXBAT is considered adequate enough to certify civil GPS receivers as spoof resistant. Consequently, many papers analysing anti-spoofing techniques have based their evaluation on the TEXBAT signal's repository.

The first TEXBAT release contained a set of six spoofed signals named as from ds1 up to ds6. These signals were based on two un-spoofed or spoof-free (also referred to as clean) signals, one where the receiver was static and other where the receiver's antenna was mounted on a vehicle. These clean scenario signals are also included in the database. Subsequently, two more signals (ds7 and ds8) representing different attacks were added to the database.

The recorded spoofed signals contain the contribution of both the authentic GPS and the spoofed GPS signals. The authentic GPS signals are not directly captured from the receiver antenna but it comes from the initially recorded clean signals. The clean static signal was replayed through the National Instruments vector signal generator to serve as the authentic signal stream for TEXBAT scenarios 1-4 whereas the clean dynamic signal was used similarly for scenarios 5 and 6.

All eight TEXBAT spoofing scenario signals contain base band complex samples at a rate of 25MSPS, which, with the high-quality front-end filtering employed, provides a frequency response over a 20-MHz bandwidth around the L1 band. Each signal is approximately 7 minutes (420 seconds) long. In the first 100 seconds, no spoofing signals are present, thus allowing receivers to acquire authentic signals and, consequently, obtaining real navigation and timing solutions.

Regarding the spoofer signal generation, the University of Texas (UT) spoofer receives authentic civil GPS L1 C/A and GPS L2C signals and generates counterfeit GPS L1 C/A signals that are closely code-phase aligned with their authentic counterparts. The spoofer is incapable of generating spoofer signals carrier-phase aligned with the authentic signals at the location of a target receiver as it would require a cm-level knowledge of the receiver-spoofer's relative position. Therefore, there are two alternatives regarding the phase-carrier, which can be differentiated in two modes:

- Unlocked (default) mode: the rate of change of its signals' carrier phase is proportional to the rate of change of the corresponding code phase.
- Locked mode: The spoofer preserves the initial phase offset between the authentic and spoofed signal.

Apart from properly aligning the spoofer signal code-phase and carrier-phase wise, it is also required to align the spoofer simulated navigation data bits to the authentic navigation data bits. Instead of reading the authentic navigation data bits and replying them immediately, the spoofer leverages the high predictability of the navigation data that modulate the

GPS L1 C/A signals. Over the course of a 12.5-minute navigation data superframe, the spoofer collects the data bits corresponding to each tracked GPS satellite. Afterwards, the spoofer compensates for its approximately 5-ms processing delay due to geometrical and cable delays by predicting the value of the navigation data stream slightly more than 5 ms in advance.

At this point, the spoofer can decide the dimension to maliciously modify, either position or time.

In the following table there is a summary of the different scenarios and their features:

Scenario Designation	**Spoofing Dimension**	**Platform Mobility**	**Power Advantage (dB)**	**Frequency Lock**	**Noise Padding**	**Size (GB)**
ds1	N/A	Static	N/A	Unlocked	Enable	43
ds2	Time	Static	10	Unlocked	Disable	42.5
ds3	Time	Static	1.3	Locked	Disable	42.6
ds4	Position	Static	0.4	Locked	Disable	42.6
ds5	Time	Dynamic	9.9	Unlocked	Disable	38.9
ds6	Position	Dynamic	0.8	Locked	Disable	38.9

Table 4.1: TEXBAT scenario summary, extracted from (22)

The following point is to describe each scenario in detail:

- ds1: Static switch
  It represents an instantaneous transition from the initial situation when it is only received an authentic signal to the situation when it is only received a spoofing signal. That kind of switch can occur if the spoofer either blocks the authentic signal or cable-switches the received signal (i.e. it has physical access to the receiver's antenna). As the spoofing signal contains less power than the authentic signal, the switch can be observed when monitoring the normalized received signal power (after 100s from the signal's beginning).

- ds2: Static Overpowered Time Push
  The spoofer performs a timing attack with a 10dB power advantage over the authentic signal. That feature, due to the receiver's automatic gain control, causes authentic signals to be masked by the noise floor. Thus, reducing the interaction between the authentic and spoofing signals. This fact can be reflected in the correlation signature, as the appearence of the spoofing signal goes unnoticed. Nevertheless, as in ds1 scenario, in attacks where there is a remarkable power miss-match, they can be detected by monitoring the received input power.

- ds3: Static Matched-Power Time Push
  This scenario is almost identical to the previous, nonetheless, the spoofer intends to camouflage its attack by matching the power with the authentic signals. By doing so, it goes almost unnoticed in the power's monitoring.

- ds4: Static Matched-Power Position Push Same attack as in Scenario 3 but with even a closer power matching. However, in this case the spoofer attack modify the position instead of the time.

- ds5: Dynamic Overpowered Time Push Scenario 5 is similar to Scenario 2, however, in this case the receiver is mounted on a dynamic platform and the carrier-phase is in locked mode. The dynamic receiver provokes remarkable fluctuations in the indicators used to detect the spoofing such as the C/N0 and the phase trauma, thus complicating the spoofing detection as the variation can be attributed to the natural phenomena induced by the movement such as the multipath and the natural fading.

- ds6: Dynamic Matched-Power Position Push Analogously, Scenario 6 is similar to Scenario 4 but the receiver is dynamic.

- ds7: Static Matched-Power Time push: An enhanced version of the attack presented in Scenario 3 as the spoofing signals are carrier-phase aligned with respect to the authentic signals.

- ds8: The ds8.bin spoofing scenario is identical to the ds7.bin scenario except that the spoofer treats every received navigation data bit as if it were an unpredictable low-rate security code and attempts to guess the value of the data bit in real time.

4.0.2. Signal analysis

Although there are several candidate signals to examine with the developed anti-spoofing techniques, the intention of the APT anti-spoofing techniques implemented is to test it towards the following spoofing attacks:

- Non-synchronous and coherent
  For evaluating this kind of attack it has been taken into consideration the TEXBAT scenario 3. Performing a monitoring in the correlation grid in the PRN 13, it has been identified this type of attack.

  In the following pictures it is presented this analysis:

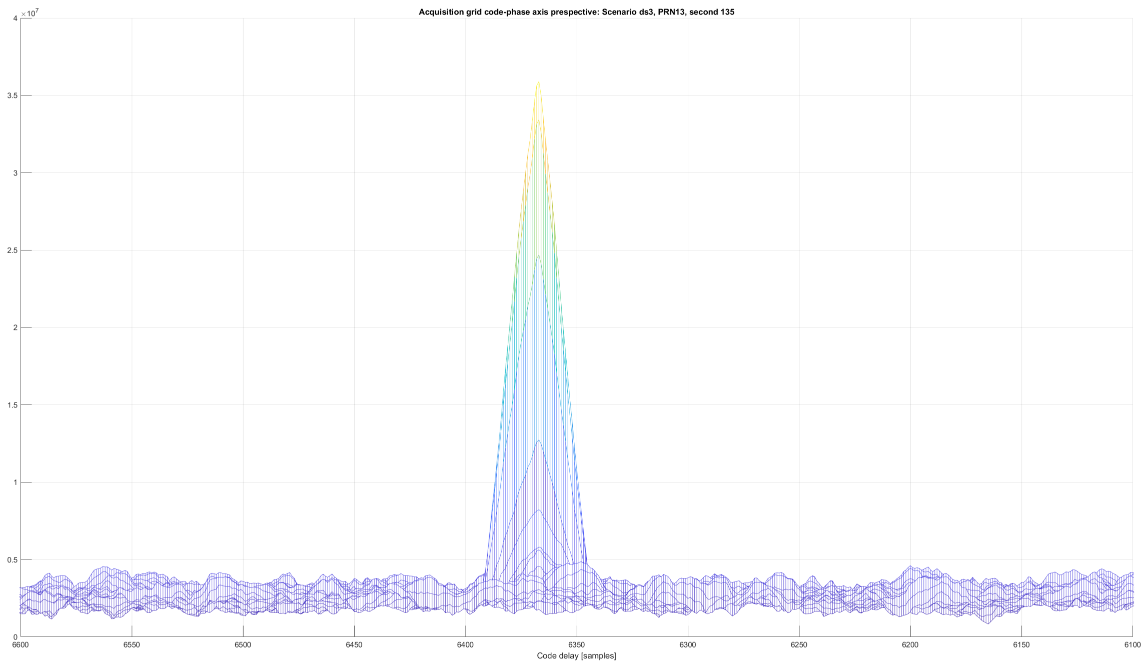


Figure 4.1: Acquisition grid code-phase axis perspective: Scenario ds3, PRN13, second 135 (Own source)

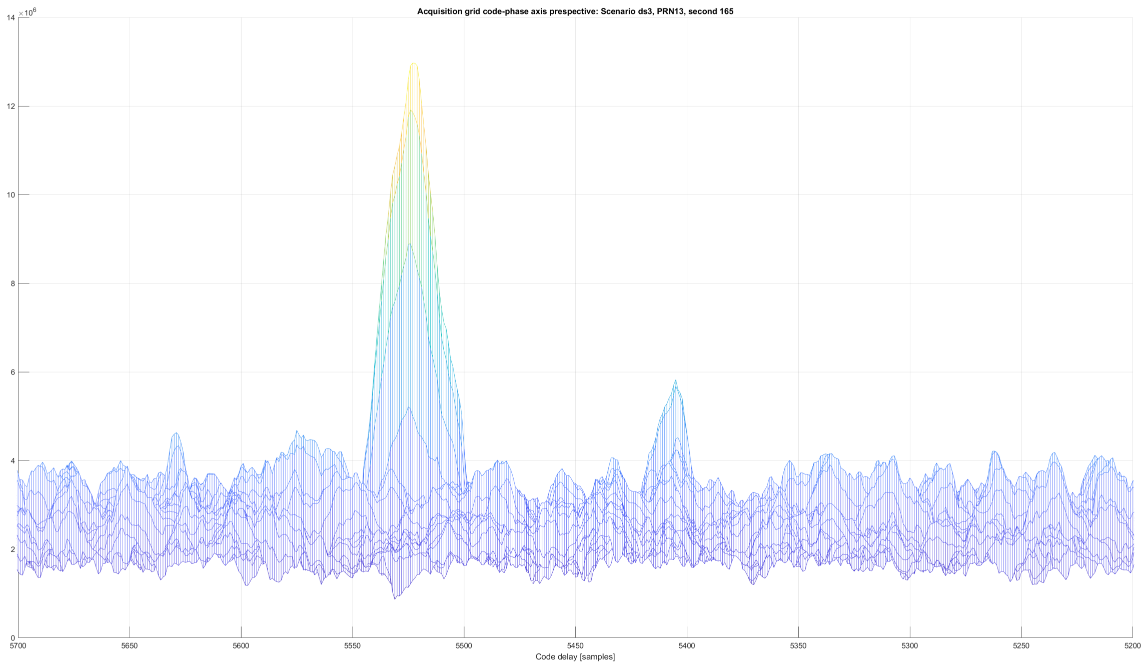


Figure 4.2: Acquisition grid code-phase axis perspective: Scenario ds3, PRN13, second 165 (Own source)

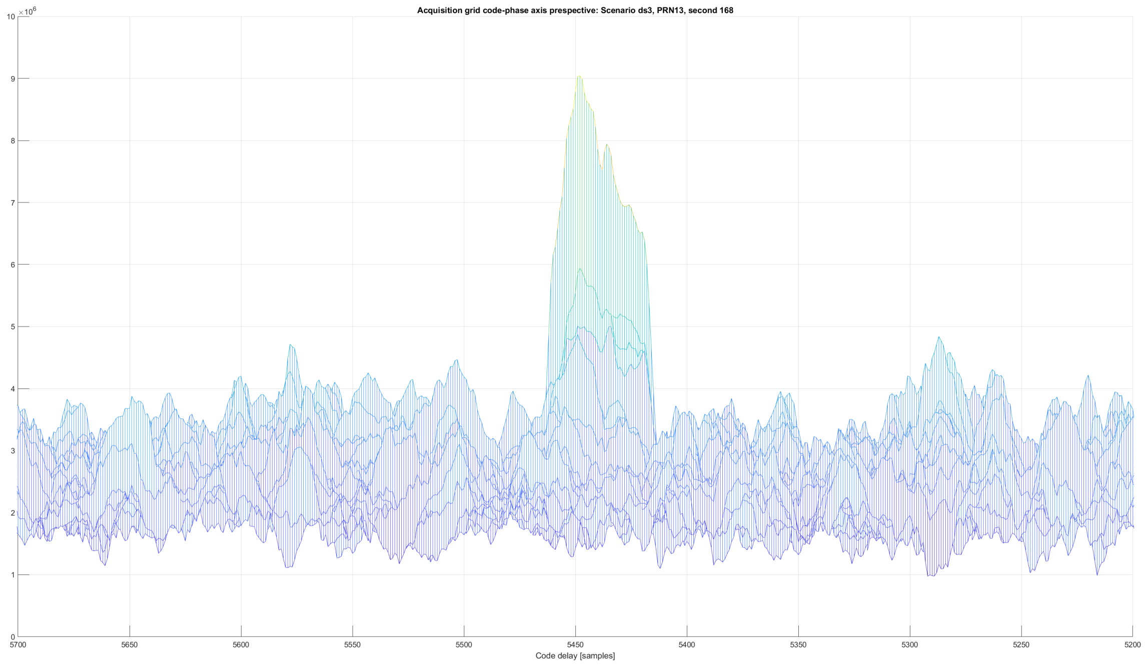


Figure 4.3: Acquisition grid code-phase axis perspective: Scenario ds3, PRN13, second 168 (Own source)

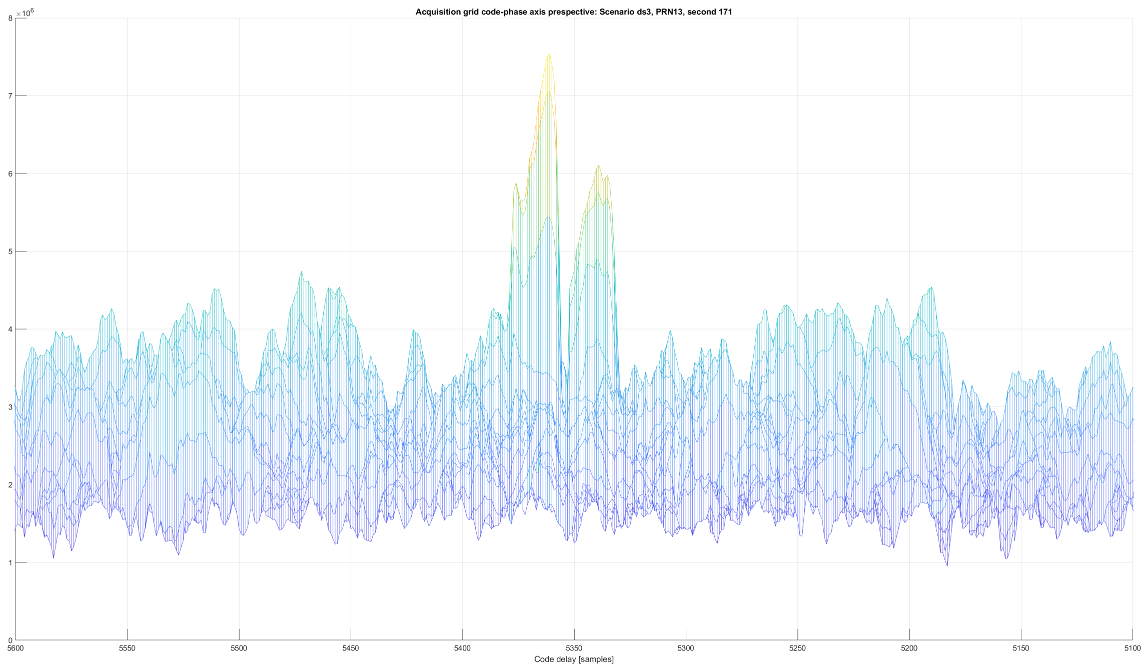


Figure 4.4: Acquisition grid code-phase axis perspective: Scenario ds3, PRN13, second 171 (Own source)

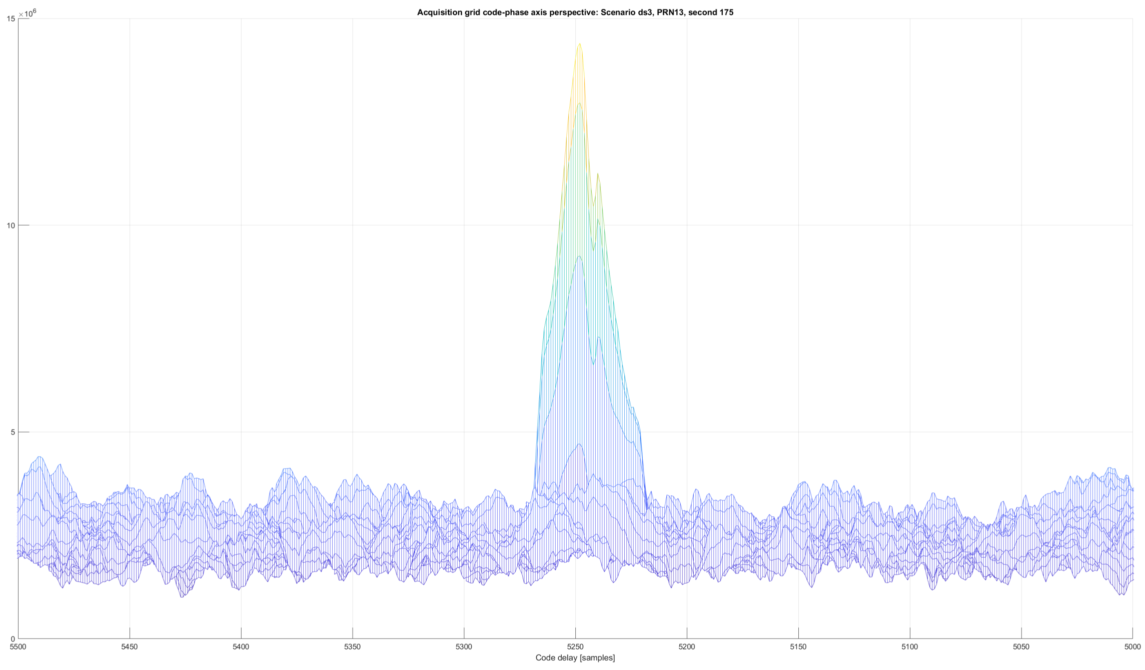


Figure 4.5: Acquisition grid code-phase axis perspective: Scenario ds3, PRN13, second 175 (Own source)

In figure 4.1 it is represented the correlation grid before the attack is being detected. As in normal conditions, it only appears the peak corresponding to the legitimate GPS signals.

Nonetheless, in figure 4.2 it can be detected a suspicious second peak with lower amplitude. Moreover, the floor level has risen remarkably.

In the following two figures (4.3 and 4.4), it can be seen how the secondary peak caused by the spoofing signal approaches and tries to synchronize in the code-phase axis with the legitimate correlation peak.

Finally in the 4.5 it can be seen how the spoofing signal has synchronized with the legitimate signal and has increased its power in order to perform the take-over.

- Synchronous and coherent
  This kind of attack has been identified in different scenarios, which will be presented and detailed in this section. In this section are sorted out the attacks in which the take-over is seamless, meaning that the spoofing signal has managed to synchronize to the legitimate signal from the beginning. Therefore, in this attacks it only can be detected both peaks when the malicious peak has already locked into the victim receiver's tracking module and maliciously driven the spoofing peak away. At this point, it can be spotted both peaks, the one coming from the spoofing signal and the one coming from the legitimate signal.

  – Coarse seamless take-over
    Surprisingly, in the same TEXBAT scenario 3 but in another signal coming from another satellite it has been identified another kind of attack. Particularly in the PRN 23. In the following pictures depict the most relevant events occuring in this attack:

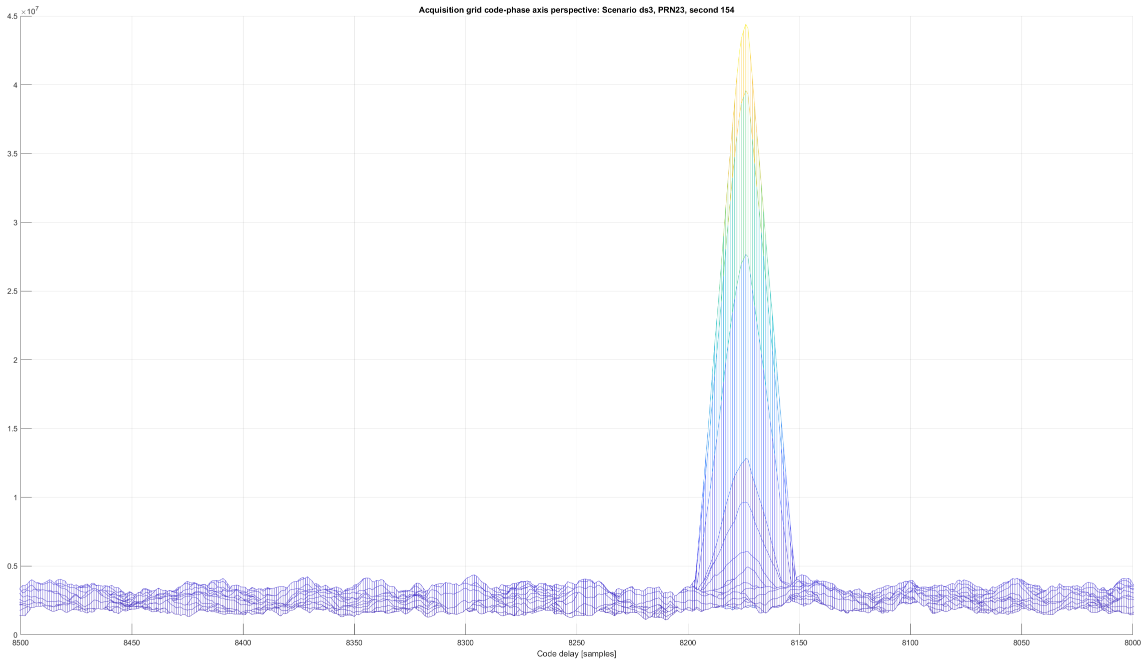


Figure 4.6: Acquisition grid code-phase axis perspective: Scenario ds3, PRN23, second 154 (Own source)

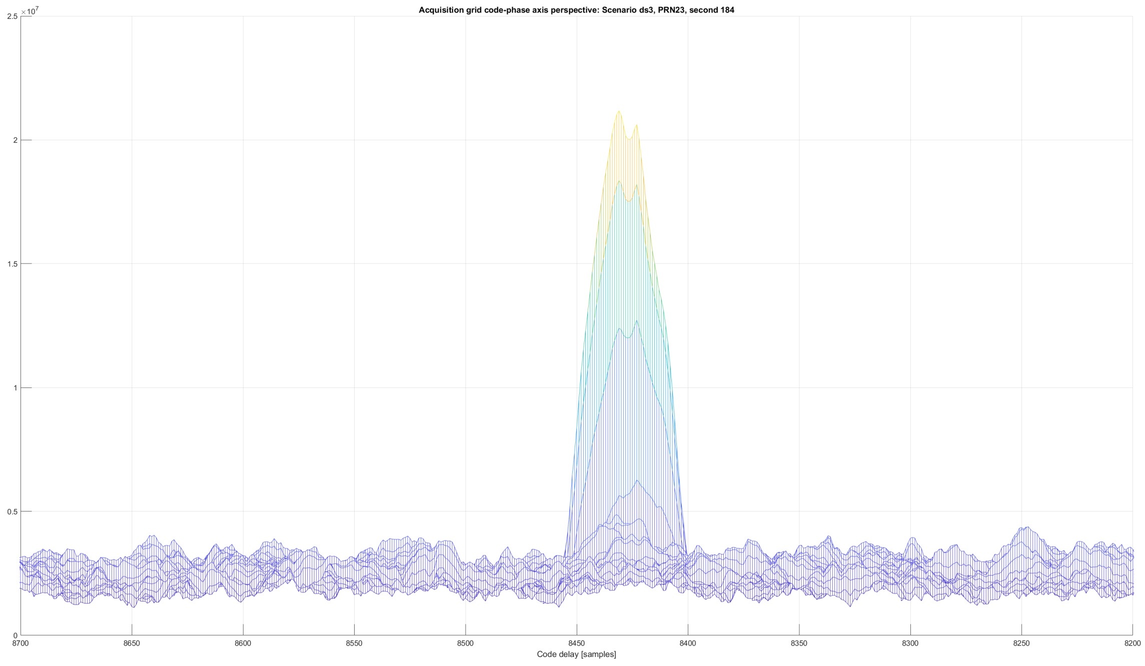


Figure 4.7: Acquisition grid code-phase axis perspective: Scenario ds3, PRN23, second 184 (Own source)

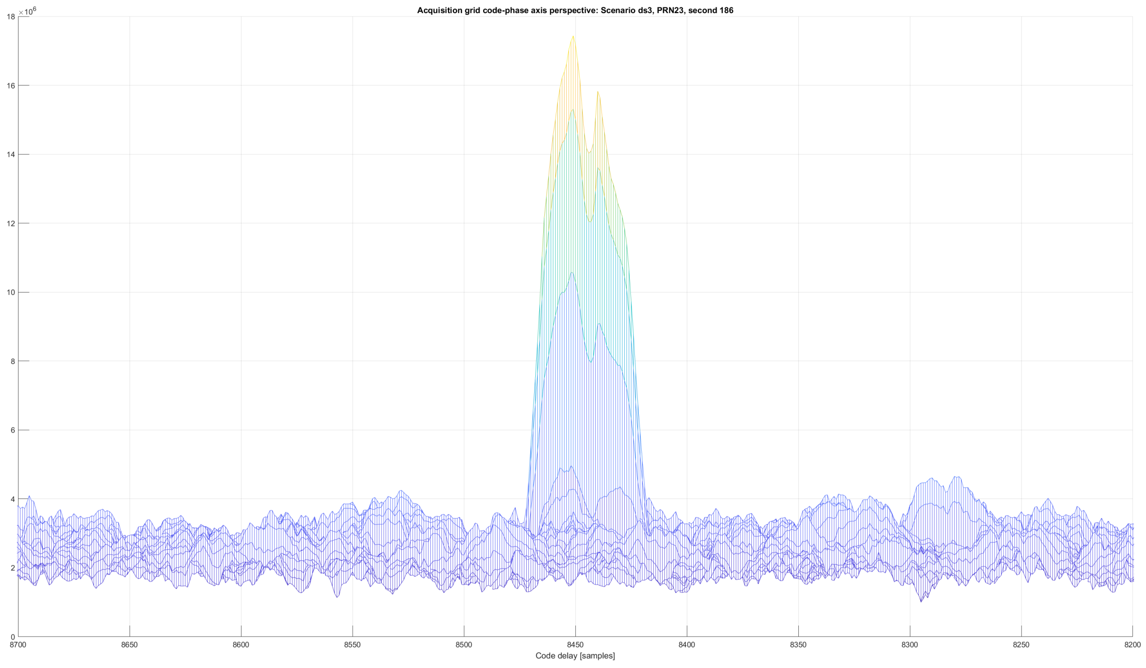


Figure 4.8: Acquisition grid code-phase axis perspective: Scenario ds3, PRN23, second 186 (Own source)

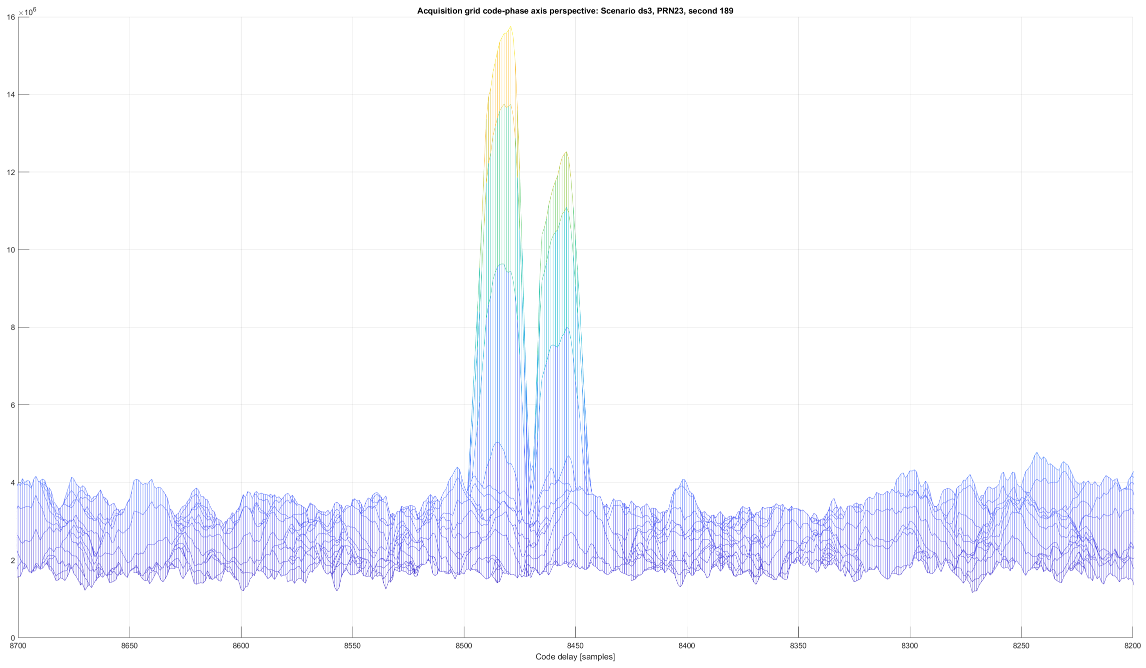


Figure 4.9: Acquisition grid code-phase axis perspective: Scenario ds3, PRN23, second 189 (Own source)

The first figure (figure 4.6) intends to show the initial state before any secondary peak is acquired.

In figure 4.7 can be seen the appearance of the secondary peak almost synchronized to the legitimate signal. That is why it is considered a synchronous attack.

In the following two figures (4.8 and 4.9) it is shown the process of de-syncrhonization once the malicious signal has locked into the victim receiver's tracking module. Nonetheless, it can be seen that this process leads to an increase of the noise floor and that the malicious signal performs it without remarkably increasing its power thus increasing its peak's amplitude. Consequently, once it separates a bit of the legitimate signal, the correlation peak of the legitimate signal presents the same order of magnitude. Ideally, the spoofer would desire not to leave this trace as it allows the detection of the spoofing attack.

– Fine seamless take-over

  In TEXBAT spoofing scenario number 7 (*ds7*) it is presented the most sofisticated, therefore challenging spoofing attack. It is characterized by a perfeclty synchronized non-coherent attack that, once has managed to lock to the victim's receiver, it drives it away from the legitimate signal and, in addition, remarkably masks the legitimate signal into the noise floor.

  Next, the following figures depict the most relevant events carried out in this attack:

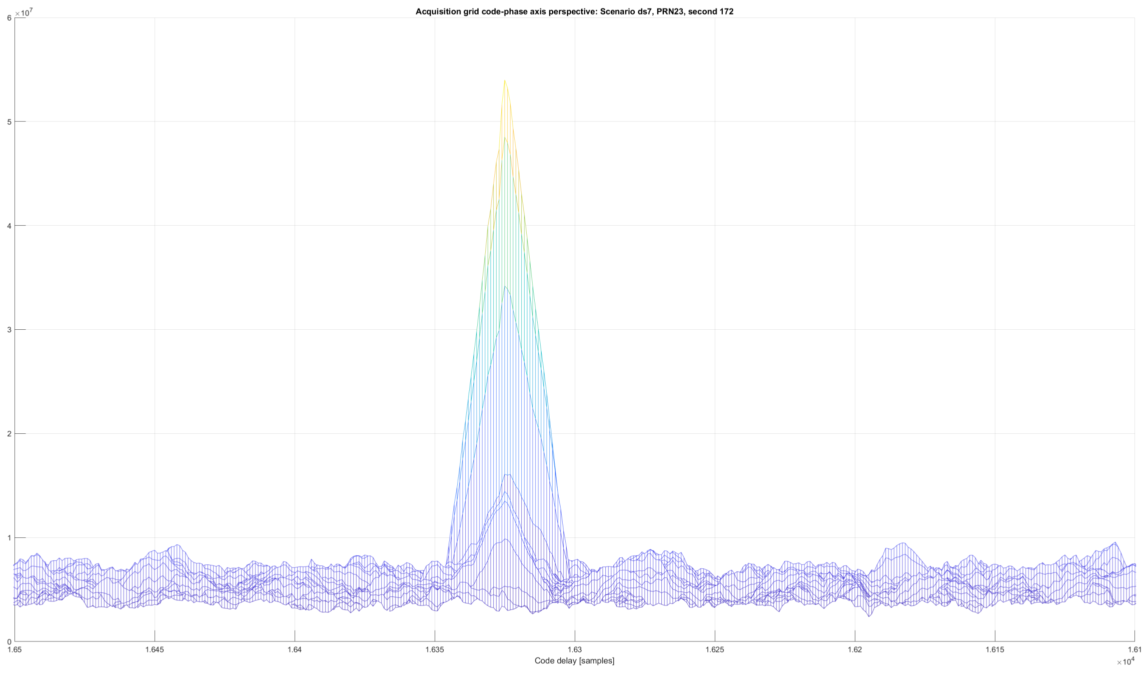


Figure 4.10: Acquisition grid code-phase axis perspective: Scenario ds7, PRN23, second 172 (Own source)

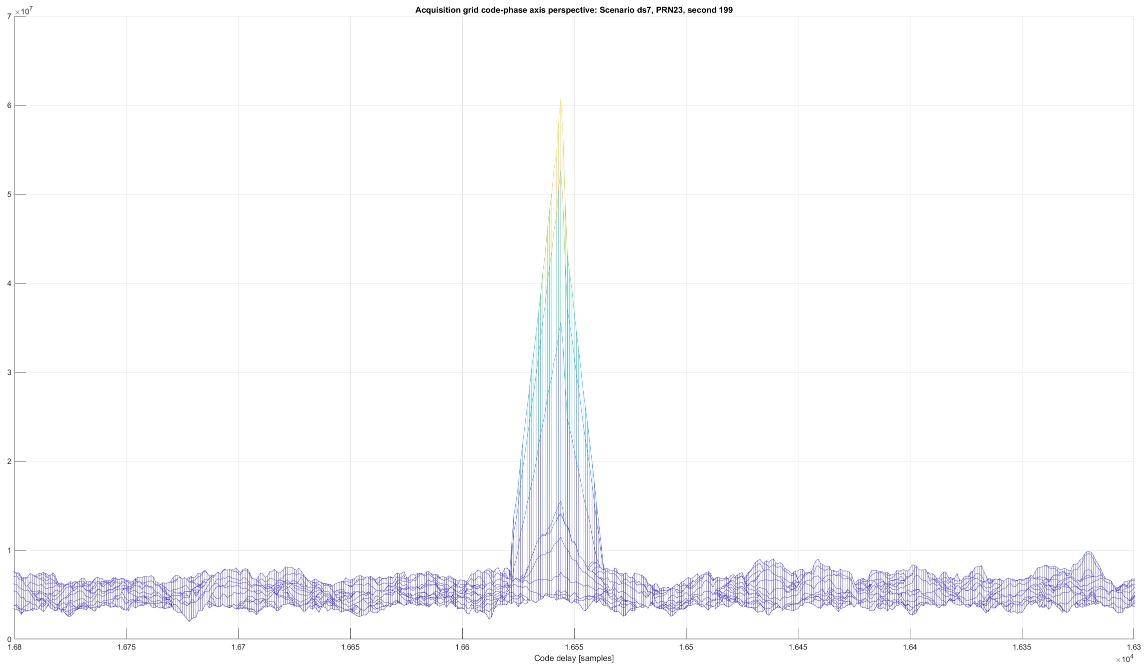


Figure 4.11: Acquisition grid code-phase axis perspective: Scenario ds7, PRN23, second 199 (Own source)

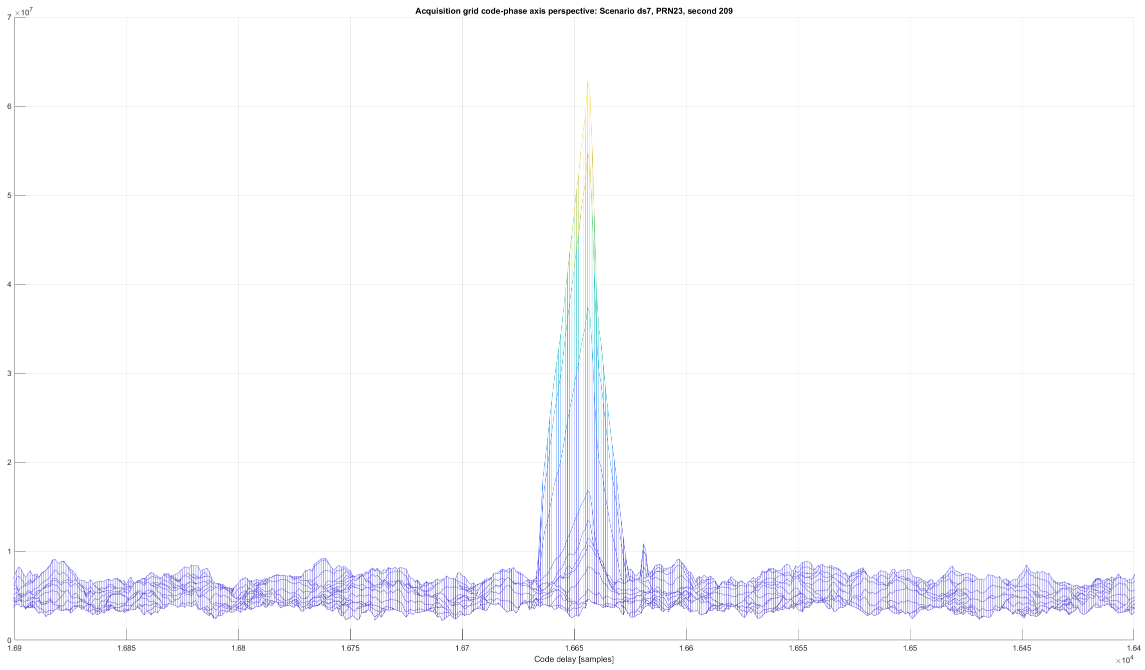


Figure 4.12: Acquisition grid code-phase axis perspective: Scenario ds7, PRN23, second 209 (Own source)

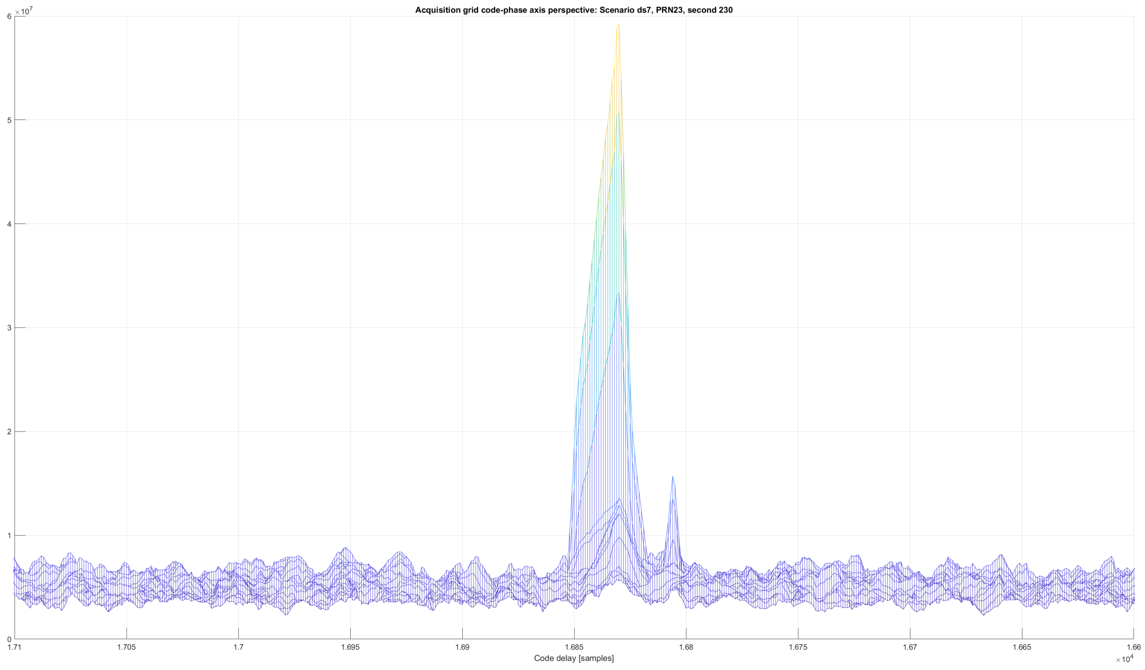


Figure 4.13: Acquisition grid code-phase axis perspective: Scenario ds7, PRN23, second 230 (Own source)

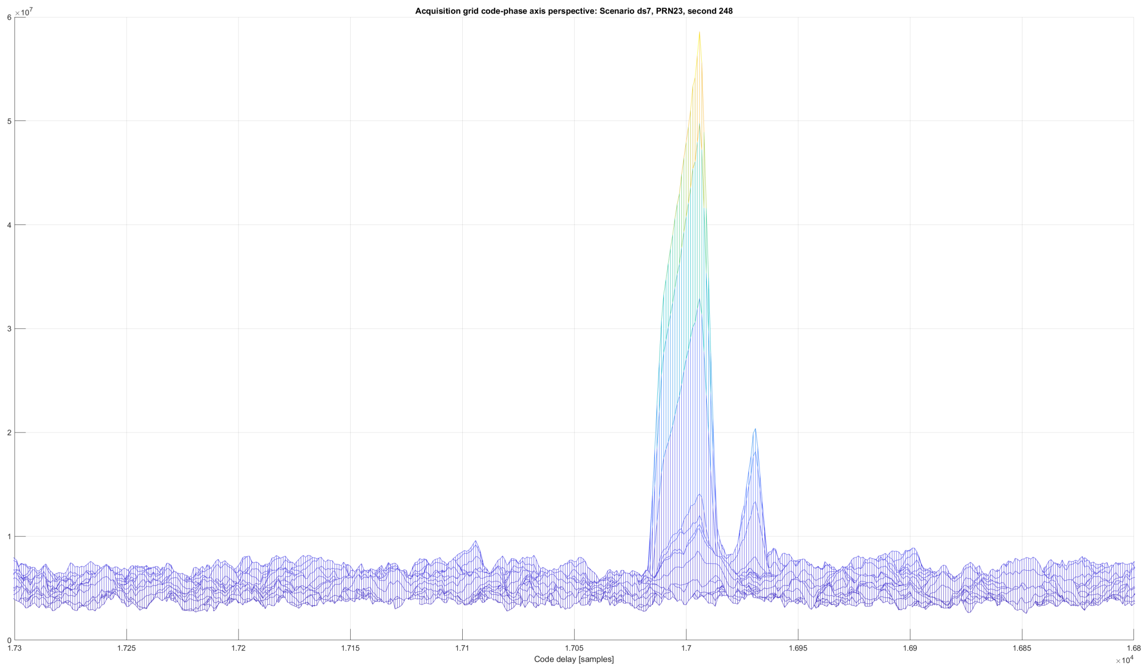


Figure 4.14: Acquisition grid code-phase axis perspective: Scenario ds7, PRN23, second 248 (Own source)

Following the same strategy when explaining the spoofing attacks, the first figure (4.10) intends to visualize the acquisition grid in an instant free from spoofing.

In the figure 4.11 it can be noticed a slight distortion in the correlation pattern, mainly in the right part, distortion which can be caused by either the auxiliary peak or either natural fading or multi-path effect. At this point, it is not possible to alert from the presence of a spoofing attack despite the fact that the TEXBAT documentation informs of it.

Looking the figure 4.12 it can be noticed a secondary peak that starts to form at the right of the main peak. Nonetheless, at first sight can not be identified as a spoofing attack.

After some seconds (figures 4.13 and 4.14), definitely can be spotted a secondary peak, which actually is the legitimate peak. The malicious peak has already taken-over and increased its power, therefore becoming the main peak in the acquisition grid. However, as this seamless take-over attack is so sophisticated, it is only possible to detect it in the acquisition grid once it has driven the receiver's tracking module some chips away the legitimate signal.

CHAPTER 5. ANTI-SPOOFING TECHNIQUES IMPLEMENTATION

In this section are detailed the different anti-spoofing techniques that we have implemented in the Borre's software. It has been decided to implement one technique on the physical layer (APT) and another on the logical layer (NAVI).

The repository containing the developed techniques is (be careful with the underscore when copying the link):

https://github.com/ericsg1999/TFG_antiSpoofingTechniquesAptNavi.git

On the one hand, the physical layer technique has been considered to be the most promising one as, depending on the attack nature, it can detect the spoofing attack even before the spoofer takes over the control of the victims receiver. On the other hand, the logic layer technique NAVI is a reactive countermeasure as it is capable to detect the spoofing attack once the spoofer has managed to successfully take over the control of the victim's receiver and has maliciously modified its parameters.

Although the NAVI logical detection technique may look it delivers worst performance, it has been decided to implement it as it is perfectly compatible with the the physical layer technique APT. The combination of a physical and logical layer techniques provide more redundancy to the overall detection system.

5.1. APT

Several modifications to the original software were required in order to implement the APT spoofing detection technique. At a high level point of view, the main difference is that, in order to perform this new feature, the receiver shall allocate more than one channel to each tracked satellite, one for the traditional tracking purposes and the second (also called secondary channel) for spoofing detection purposes.

This change in the architecture mainly affects the tracking module and the acquisition module. In this section are explained the different modifications and its functionalities.

5.1.1. *init* structure

New managing parameters shall be added in order to define the several parameters related to the APT configuration. All these parameters are sorted together and its name starts with the *Apt* distinctive. The following table exposes a summary:

Structure	Section	Parameter	Units/Values
settings	APT parameters	AptActive	’0’ or ’1’
		AptPeriod	[ms]
		AptPlots	’0’ or ’1’
		AptShowPlots	’0’ or ’1’
		AptSavePlots	’0’ or ’1’
		AptNumberChannelsPerSat	Natural number
		AptThreshold	Positive double

Table 5.1: *init* structure new parameters regarding APT feature (Own source)

- *AptActive*
  ’0’ disables APT, ’1’ enables APT.

- *AptPeriod*
  Number of milliseconds between APT detection checks.

- *AptPlots*
  ’0’ disables APT plots, ’1’ enables APT plots. With this parameter disabled, is no possible to neither show nor save the generated plots.

- *AptShowPlots*
  ’0’ does not display the generated APT plots, ’1’ automatically displays the generated APT plots.

- *AptSavePlots*
  ’0’ does not save the APT plots, ’1’ saves the APT plots in the specified directory.

- *AptNumberChannelsPerSat*
  Indicates how many channels are dedicated for each satellite. The first channel is assumed to be for tracking purposes, consequently, if it is desired to perform APT functions, at least it is required to allocate 2 channels per satellite (being the second for APT purposes).

- *AptThreshold*
  Indicates how many times the potential spoofing peak found in the acquisition search grid shall be bigger than the third highest peak in order to trigger the APT detection alarm.

5.1.2. Acquisition function

Next are listed and explained the different modifications that the acquisition function has suffered in order to incorporate the APT feature.

- *acqMode* parameter:
  The acquisition function has been modified in order to accept a new input parameter, which will define among which satellites the function will perform the acquisition. It has been called *acqMode* and can adopt two different scenarios or modes:

  - Normal:
    Acquisition in normal mode looks for the presence of all possible satellites in the received signal. This mode is used in the initial acquisition, when the receiver has still not acquired any satellite, therefore still does not know which satellites are present.

    The objective of this acquisition mode is to determine the visible satellites and provide an estimation of the Doppler frequency and code-phase parameters.

  - APT:
    Acquisition in APT mode only performs the acquisition of the previously acquired satellites. In this mode, the acquisition function does not look for satellites that were declared as not visible in the initial acquisition. By restricting the number of satellites to acquire, the computational load is remarkable reduced as it is assumed that the spoofer will only perform the attack to the satellites that are visible for the victim receiver.

    The objective of this acquisition mode is, for the satellites that previously were considered as visible, monitor the acquisition search grid in order to detect the presence of an intruder peak. In fact, it does not monitor the entire search grid but the frequency bin of the legitimate peak as it has been assumed that the spoofer will perform the attack with a perfect Doppler frequency alignment.

- Statistic test: The acquisition function uses different statistic tests depending on the acquisition mode (*acqMode*):

  - Normal mode:
    The original Borre's software used a test statistic which took into account the signal power. Nonetheless, in this case, due to the fact that some spoofing attacks involve an increase in the noise floor thus affecting the signal power, it was decided to use a different test statistic which did not consider an estimation of the signal power.

    $$Test_{statistic} = \frac{Peak_{1st}}{Peak_{3rd}} \tag{5.1}$$

    The third highest peak value is a manner to quantify the noise floor, actually, it assumes a pessimistic scenario.

  - APT mode:
    This new feature uses the following statistic test, which is a ratio of the second highest peak and third highest peak obtained in the acquisition search grid.

    $$Test_{statistic} = \frac{Peak_{2nd}}{Peak_{3rd}} \tag{5.2}$$

    This is the statistic test performed in order to detect the presence of a spoofing peak.

    When there is no spoofing attack, in the acquisition search grid it should only be one peak, the one corresponding to the legitimate signal. Therefore, the statistic test should be similar to one.

    Nevertheless, when there is an spoofing attack, the secondary peak should be observed in the acquisition search grid leading to an statistic test remarkably higher than 1.

The following flowchart (figures 5.1 and 5.2) reflects the new acquisition.m function work's principle. Down below are presented several details about it.

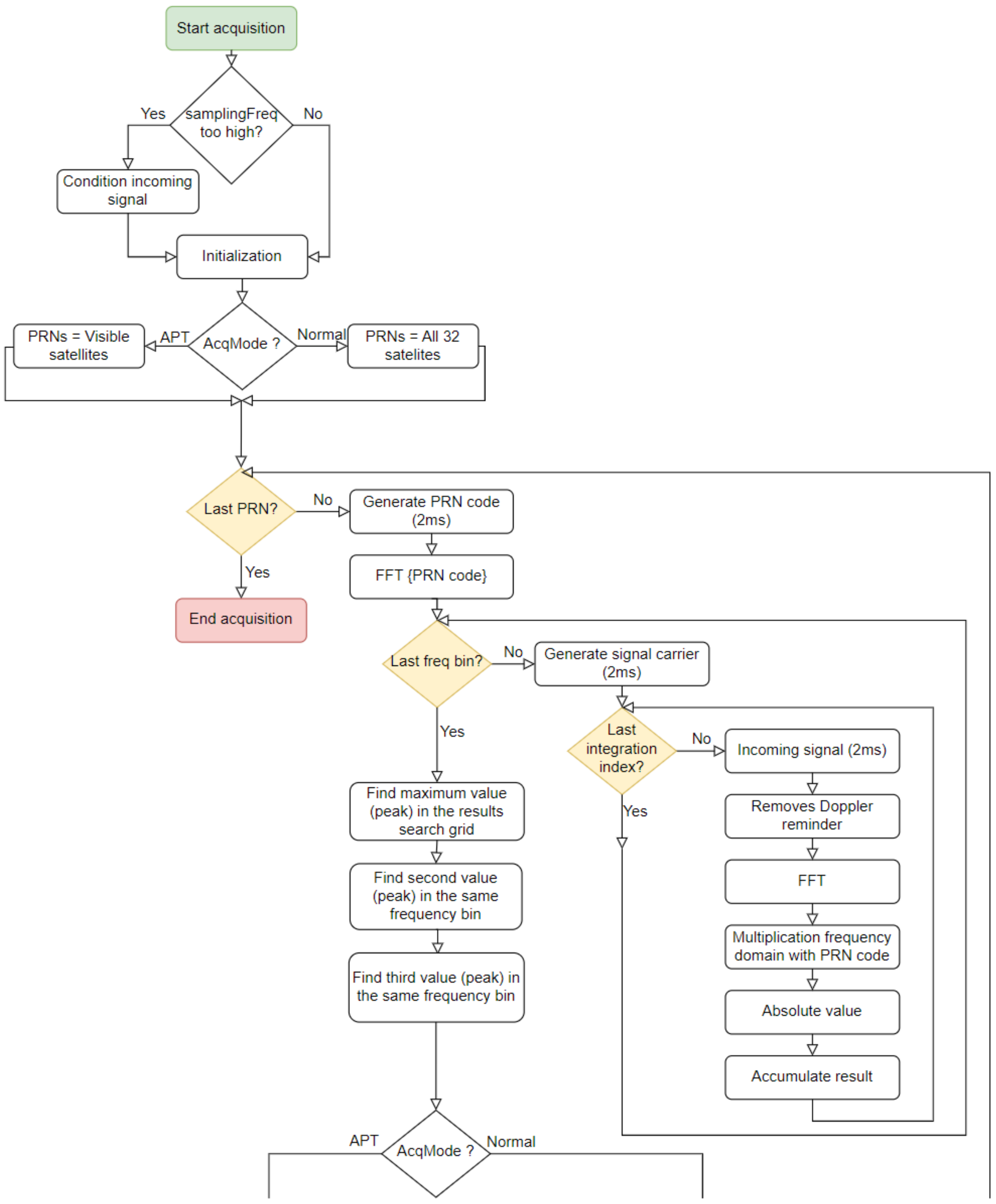


Figure 5.1: Acquisition with APT feature flowchart part 1 (Own source)

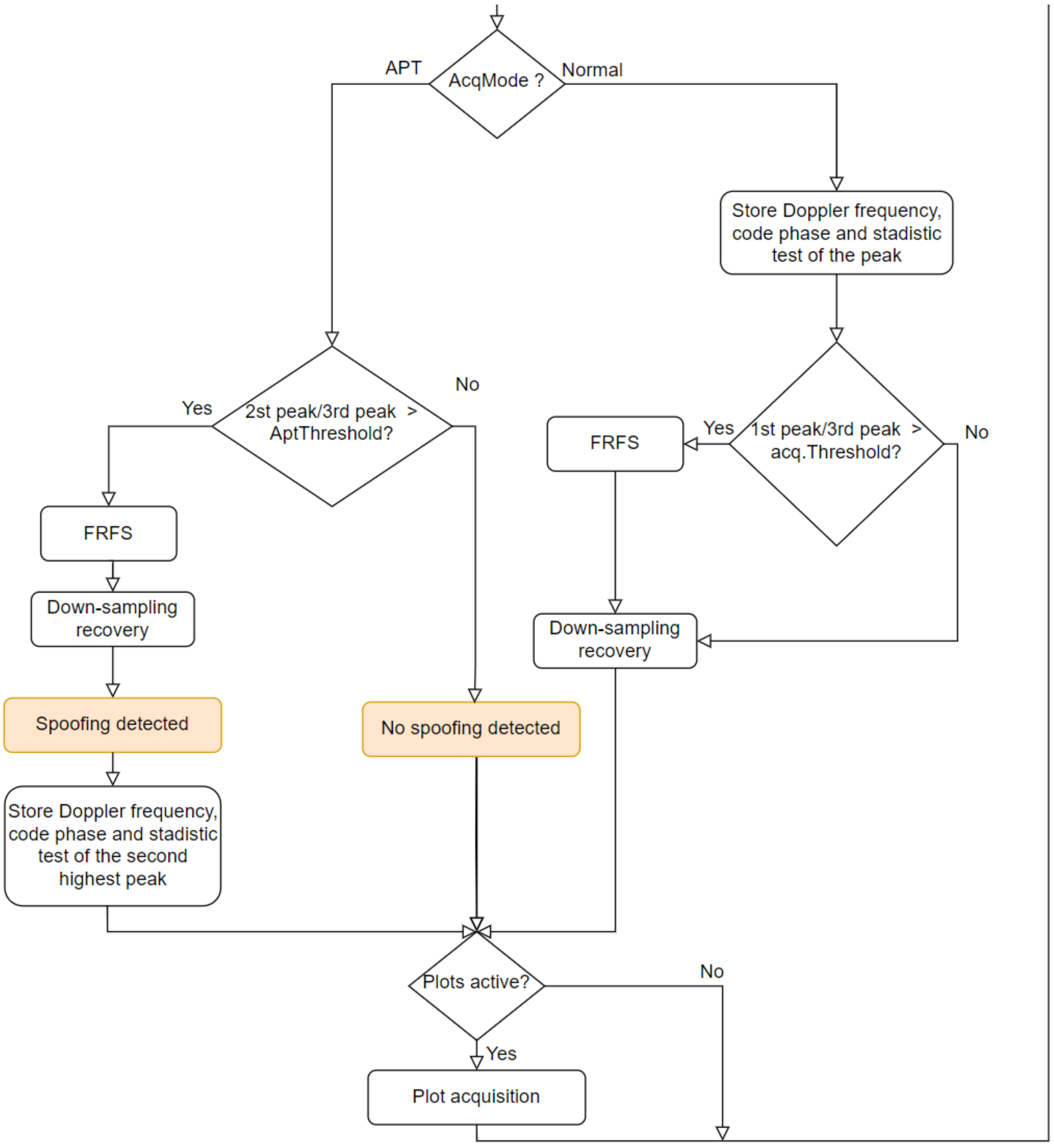

APT
AcqMode ?
Normal
Store Doppler frequency, code phase and stadistic test of the peak
2st peak/3rd peak > AptThreshold?
Yes
No
1st peak/3rd peak > acq.Threshold?
Yes
No
FRFS
FRFS
Down-sampling recovery
Down-sampling recovery
Spoofing detected
No spoofing detected
Store Doppler frequency, code phase and stadistic test of the second highest peak
Plots active?
No
Yes
Plot acquisition

Figure 5.2: Acquisition with APT feature flowchart part 2 (Own source)

With respect to the original acquisition flowchart, the main differences are:

- Channels to perform acquisition:
  Depending on the acquisition mode, it performs the acquisition on either all satellites (Normal mode acquisition) or only the visible satellites (APT mode acquisition).

- Test statistic metric:
  The test statistic metric has been changed in both acquisition modes. As now, depending on the mode, it is based on the ratio between the magnitudes of the highest

or second highest and third highest peak, as opposed to the original *acquisition.m* function, it is necessary to perform the search of the third highest peak magnitude.

In order to do that, it is carried out a search process in the same frequency bin as the highest peak, due to the fact that most of the spoofing attacks present a perfect alignment in the Doppler frequency parameter. Once it has located the highest peak, it finds the second highest peak in the same frequency bin which is no closer than 1 chip to the highest peak. Afterwards, also in the same frequency bin, it searches for the third highest peak which is no closer than 1 chip to the previously found peaks.

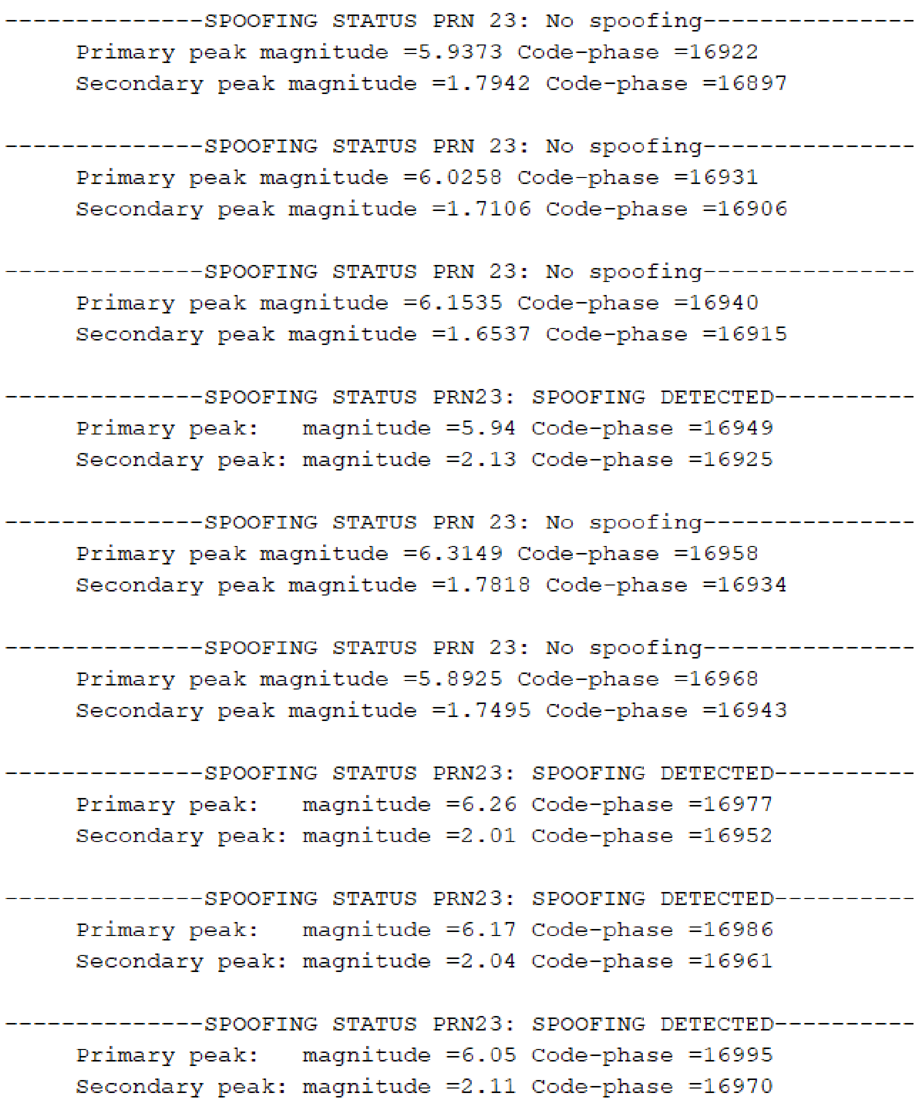
- Spoofing detection new feature:
  In APT mode, it checks if the second highest peak is remarkably higher than the third highest peak. The decision criterion is adjustable through the *acq.AptThreshold*.

  The test behaviour can be modelled as follows:

$$\begin{cases} \text{if } T_{APT} \geq \gamma \rightarrow \text{it is decided } H_1 \\ \text{if } T_{APT} < \gamma \rightarrow \text{it is decided } H_0 \end{cases} \tag{5.3}$$

  Where T_{APT} is the test statistic defined in equation 5.2, γ the *AptThreshold*, H_1 the test hypothesis indicating the presence of a spoofing attack and H_0 the test hypothesis indicating the absence of a spoofing attack.

  The spoofing detection displays a console message informing of the alert status as well as some information of both primary and secondary peaks such as the statistic tests and the code-phases.



```
--------------SPOOFING STATUS PRN 23: No spoofing---------------
     Primary peak magnitude =5.9373 Code-phase =16922
     Secondary peak magnitude =1.7942 Code-phase =16897

--------------SPOOFING STATUS PRN 23: No spoofing---------------
     Primary peak magnitude =6.0258 Code-phase =16931
     Secondary peak magnitude =1.7106 Code-phase =16906

--------------SPOOFING STATUS PRN 23: No spoofing---------------
     Primary peak magnitude =6.1535 Code-phase =16940
     Secondary peak magnitude =1.6537 Code-phase =16915

--------------SPOOFING STATUS PRN23: SPOOFING DETECTED----------
     Primary peak:   magnitude =5.94 Code-phase =16949
     Secondary peak: magnitude =2.13 Code-phase =16925

--------------SPOOFING STATUS PRN 23: No spoofing---------------
     Primary peak magnitude =6.3149 Code-phase =16958
     Secondary peak magnitude =1.7818 Code-phase =16934

--------------SPOOFING STATUS PRN 23: No spoofing---------------
     Primary peak magnitude =5.8925 Code-phase =16968
     Secondary peak magnitude =1.7495 Code-phase =16943

--------------SPOOFING STATUS PRN23: SPOOFING DETECTED----------
     Primary peak:   magnitude =6.26 Code-phase =16977
     Secondary peak: magnitude =2.01 Code-phase =16952

--------------SPOOFING STATUS PRN23: SPOOFING DETECTED----------
     Primary peak:   magnitude =6.17 Code-phase =16986
     Secondary peak: magnitude =2.04 Code-phase =16961

--------------SPOOFING STATUS PRN23: SPOOFING DETECTED----------
     Primary peak:   magnitude =6.05 Code-phase =16995
     Secondary peak: magnitude =2.11 Code-phase =16970
```

Figure 5.3: APT spoofing detection window command example (Own source)

- Plot acquisition search grid:
  For the sake of simplicity, in the flowchart it has only been indicated the plot acquisition functionality. Nevertheless, depending on the *settings* structure, the two following new functionalities can be used:
  - Show APT acquisition plots
  - Save APT acquisition plots in the specified directory.
    Both features are independent, meaning that can be both enabled, both disabled or one enabled whereas the other remains disabled.

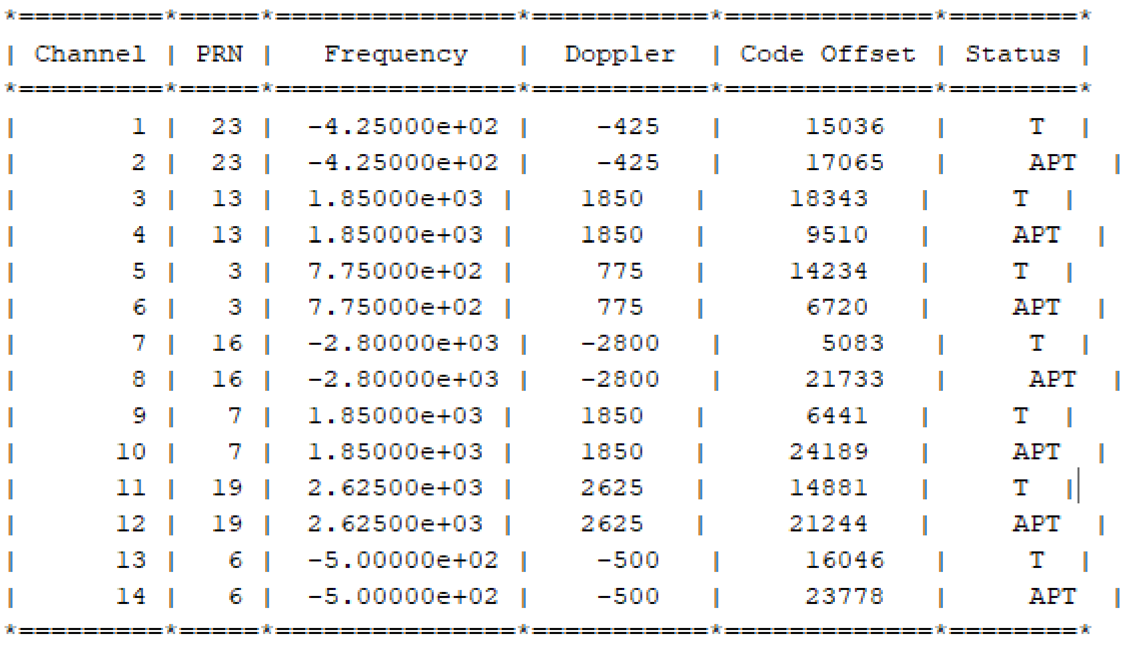
  Nonetheless, the *settings* parameter *AptPlots* activates/deactivates this whole functionality.

5.1.3. Channel initialization

In this section, which in particular is coded in the *preRun.m* function, it is implemented the initial idea of allocating more than one channel per satellite. The primary channel is responsible for the traditional tracking process whereas the secondary, third, etc. channels, referred as auxiliary channels, attempt to detect a spoofing signal by launching the acquisition module.

Therefore, apart from the channels responsible for tracking the signals with the estimated values in the acquisition, the channel initialization shall contain as well the channels dedicated to detect the presence of any spoofing signal in form of a secondary peak in the acquisition search grid. In order to do that, depending on the *settings* parameter *AptNumberChannelsPerSat*, more than one channel are allocated for each acquired signal. These auxiliary channels are set with the tracking status *'APT'*.

An example of the modified *channel* structure is the following one:



```
*=========*=====*===============*===========*=============*========*
| Channel | PRN |   Frequency   |  Doppler  | Code Offset | Status |
*=========*=====*===============*===========*=============*========*
|       1 |  23 | -4.25000e+02 |   -425    |    15036    |    T   |
|       2 |  23 | -4.25000e+02 |   -425    |    17065    |    APT   |
|       3 |  13 | 1.85000e+03 |   1850    |   18343    |    T   |
|       4 |  13 | 1.85000e+03 |   1850    |    9510    |    APT   |
|       5 |   3 | 7.75000e+02 |    775    |   14234    |    T   |
|       6 |   3 | 7.75000e+02 |    775    |    6720    |    APT   |
|       7 |  16 | -2.80000e+03 |  -2800    |     5083    |    T   |
|       8 |  16 | -2.80000e+03 |  -2800    |    21733    |    APT   |
|       9 |   7 | 1.85000e+03 |   1850    |    6441    |    T   |
|      10 |   7 | 1.85000e+03 |   1850    |   24189    |    APT   |
|      11 |  19 | 2.62500e+03 |   2625    |   14881    |    T   |
|      12 |  19 | 2.62500e+03 |   2625    |   21244    |    APT   |
|      13 |   6 | -5.00000e+02 |   -500    |    16046    |    T   |
|      14 |   6 | -5.00000e+02 |   -500    |    23778    |    APT   |
*=========*=====*===============*===========*=============*========*
```

Figure 5.4: *channel* structure with the APT feature example (Own source)

5.1.4. Tracking function

The tracking function has also been modified as, in the original software, is where the code goes through all the incoming signal and processes it. The APT detection process presents the same nature as it goes through all signal. However, whereas the tracking processes the incoming signal millisecond per millisecond, the APT detection processes fragments of 42 milliseconds separated *AptPeriod* milliseconds.

The following flowchart summarizes the work flow of the receiver including the APT new feature.

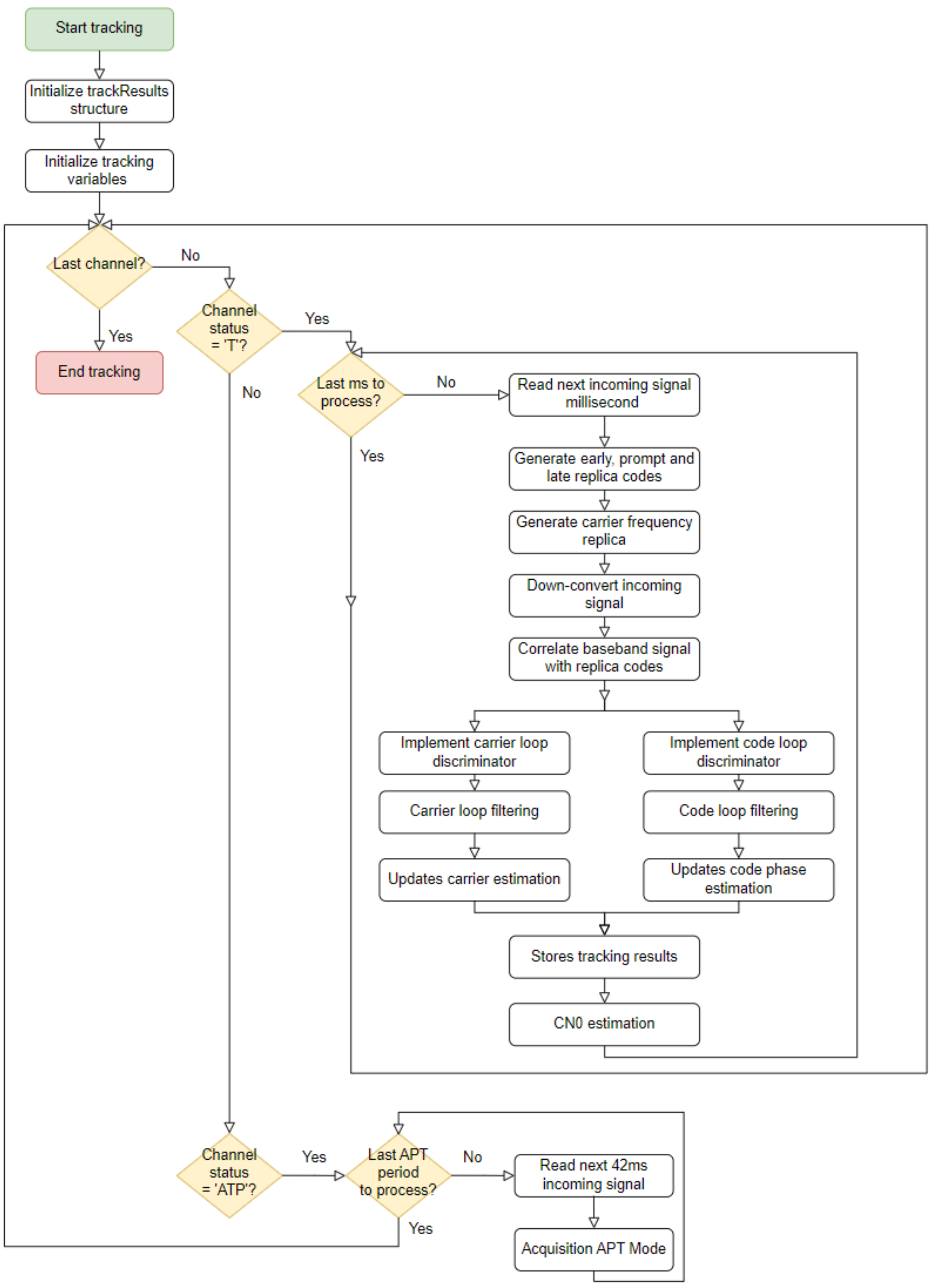

Start tracking
Initialize trackResults structure
Initialize tracking variables
Last channel?
No
Yes
End tracking
Channel status = 'T'?
Yes
No
Last ms to process?
No
Yes
Read next incoming signal millisecond
Generate early, prompt and late replica codes
Generate carrier frequency replica
Down-convert incoming signal
Correlate baseband signal with replica codes
Implement carrier loop discriminator
Carrier loop filtering
Updates carrier estimation
Implement code loop discriminator
Code loop filtering
Updates code phase estimation
Stores tracking results
CN0 estimation
Channel status = 'ATP'?
Yes
Last APT period to process?
No
Yes
Read next 42ms incoming signal
Acquisition APT Mode

Figure 5.5: Tracking with APT feature flowchart (Own source)

As the original code, the tracking function still examines all channels which receives as input. However, this time, it examines the channel status and depending on it, it performs either the original tracking process or the new APT detection process. In the latter case, it reads 42 milliseconds of the incoming signal located *AptPeriod* milliseconds away from the previous APT check and performs an acquisition in *'APT'* mode. The acquisition function itself will be responsible for looking for an spoofing peak in the acquisition search grid and trigger an alarm in case it exceeds the established APT threshold.

5.2. NAVI

So as to implement the new NAVI's feature, the original software required some adaptations. Regarding the overall architecture, the principal difference is that the receiver needed to increase the scope of the navigation message processing.

Initially, the receiver only decoded the ephemeris of one frame (i.e. five subframes, which are equivalent to 1500 bits or 30 seconds of signal). Nonetheless, in order to properly monitor the TOW parameter, it was necessary to make the code able to decode all frames contained in the entire navigation message. Recall that the objective is to compare the TOW among consecutive subframes in order to ensure the TOW coherence.

This change in the architecture is translated into modifications in the *postNavigation.m* function and its sub-functions. Moreover, it has been defined a new function called *naviTowSpoofingDetection.m* which had the specific task of checking the TOW coherence among consecutive subframes.

In this section are detailed the several modifications the code has suffered in order to incorporate the new NAVI's feature.

5.2.1. *ephemeris* structure

The initial structure *ephemeris* was an object which stored in its attributes the *ephemeris* parameters decoded of a single frame. As the NAVI's feature required to process, thus decode, more than one frame, the *ephemeris* structure also needed to be modified.

In particular, the structure had to be be able to store the *ephemeris* of all frames. In order to do that, the attributes of the initial *ephemeris* structure were changed from a single value to a list of values. Consequently, the new ephemeris structure stored each decoded frame *ephemeris* in the same list index (i.e. the *ephemeris* parameters of the frame number *i* are stored in the *i* element of each *ephemeris* attribute list). However, the TOW parameter was stored differently. As each subframe contained its particular TOW, one subframe comprised a set of 5 TOW. As a consequence, the ephemeris actually stored the TOW parameter in a matrix, where the rows indicate the frame's indexes and the columns the subframe's indexes of the stored TOW.

Once it was obtained the list of *ephemeris* structure, it was coded the *naviTowSpoofingDetection.m* function, which received as an input the new *ephemeris* structure and was in charge of checking the TOW parameter consistency among the different frames. For each consecutive subframes it performed the TOW consistency check and displayed in the command window the spoofing detection status.

5.2.2. Navigation message decoding: ***NAVdecoding.m***

For each channel tracked, it has been obtained the navigation bits. Nevertheless, unlike the original code, now it is decoded several frames. Therefore, the code shall go through all decoded frames and decode them. The resulting decoded parameters are stored into the parameters lists of the ephemeris structure.

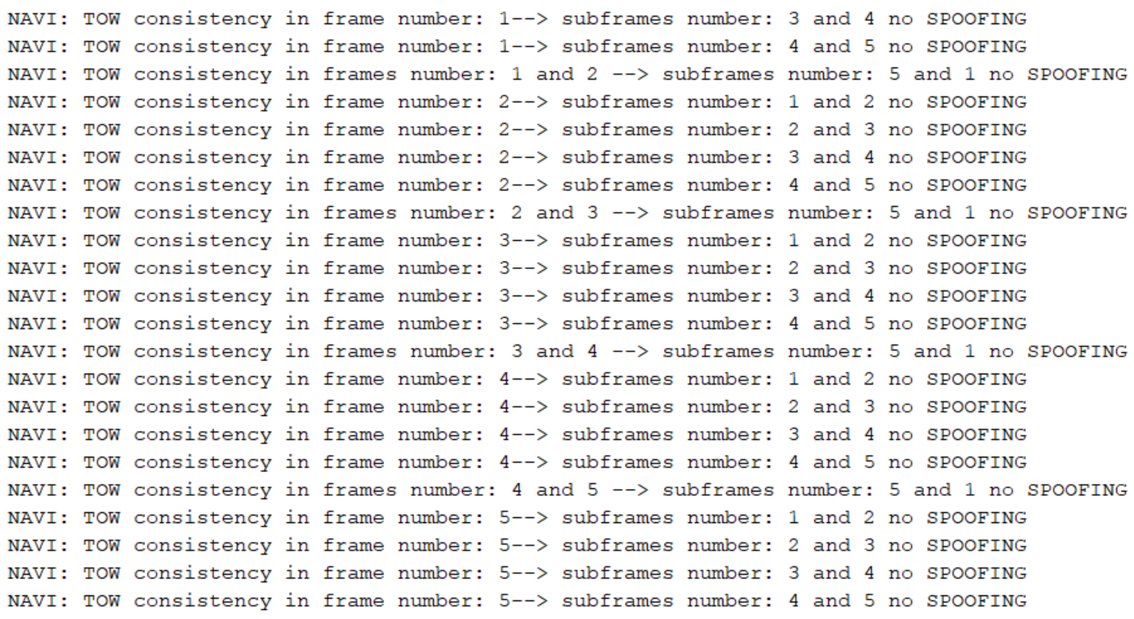
It is important to mention the special processing of the first frame in comparison with the other frames. On the one hand, as the receiver starts tracking the signal at an arbitrary instant, the first decoded frame may be not contain all five subframes. Hence, in the first subframe the first process performed is to identify its subframe's ID and, depending on it, are processed the following consecutives subframes of the first frame. On the other hand, the subsequent frames are decoded in a groups of 5 subframes (the entire frame).

The last step is, in case of active NAVI's feature, call the naviTowSpoofingDetection.m function.

5.2.3. TOW coherence check: ***naviTowSpoofingDetection.m***

The stated function received as input the ephemeris structure and was responsible for performing the TOW coherence check. In order to do that, it went through the entire TOW matrix of the *ephemeris* structure and took the TOWs of two consecutives subframes. Afterwards, it substracted its values and displayed in the Matlab command window the spoofing detection status among the corresponding subframes.

Next it is shown an example of the command window spoofing detection status:



```
NAVI: TOW consistency in frame number: 1--> subframes number: 3 and 4 no SPOOFING
NAVI: TOW consistency in frame number: 1--> subframes number: 4 and 5 no SPOOFING
NAVI: TOW consistency in frames number: 1 and 2 --> subframes number: 5 and 1 no SPOOFING
NAVI: TOW consistency in frame number: 2--> subframes number: 1 and 2 no SPOOFING
NAVI: TOW consistency in frame number: 2--> subframes number: 2 and 3 no SPOOFING
NAVI: TOW consistency in frame number: 2--> subframes number: 3 and 4 no SPOOFING
NAVI: TOW consistency in frame number: 2--> subframes number: 4 and 5 no SPOOFING
NAVI: TOW consistency in frames number: 2 and 3 --> subframes number: 5 and 1 no SPOOFING
NAVI: TOW consistency in frame number: 3--> subframes number: 1 and 2 no SPOOFING
NAVI: TOW consistency in frame number: 3--> subframes number: 2 and 3 no SPOOFING
NAVI: TOW consistency in frame number: 3--> subframes number: 3 and 4 no SPOOFING
NAVI: TOW consistency in frame number: 3--> subframes number: 4 and 5 no SPOOFING
NAVI: TOW consistency in frames number: 3 and 4 --> subframes number: 5 and 1 no SPOOFING
NAVI: TOW consistency in frame number: 4--> subframes number: 1 and 2 no SPOOFING
NAVI: TOW consistency in frame number: 4--> subframes number: 2 and 3 no SPOOFING
NAVI: TOW consistency in frame number: 4--> subframes number: 3 and 4 no SPOOFING
NAVI: TOW consistency in frame number: 4--> subframes number: 4 and 5 no SPOOFING
NAVI: TOW consistency in frames number: 4 and 5 --> subframes number: 5 and 1 no SPOOFING
NAVI: TOW consistency in frame number: 5--> subframes number: 1 and 2 no SPOOFING
NAVI: TOW consistency in frame number: 5--> subframes number: 2 and 3 no SPOOFING
NAVI: TOW consistency in frame number: 5--> subframes number: 3 and 4 no SPOOFING
NAVI: TOW consistency in frame number: 5--> subframes number: 4 and 5 no SPOOFING
```

Figure 5.6: NAVI command window spoofing detection status example (Own source)

In the previous figure (figure 5.6) it is executed the TOW coherence check among all consecutive subframes decoded in the entire navigation message. However, in some situations it may be excessive to have this level of rigour as it is detrimental to the computational load.

Due to the existence of this trade-off between the rigour/exactitude of the NAVI coherence checks and the computational load, it was decided to incorporate the *NaviTowPeriodBits* parameter, which was included in the *settings* structure.

This parameter enabled the parameterization of the trade-off as indicated, as the name suggests, the period measured in bits among TOW coherence checks. Thus, depending on the situation, the user could relax the computational load by renouncing on the security accuracy or vice versa. One of this situations could be if it is preferred to use the NAVI's detection method algorithm altogheter with the APT, as the latest consumes many resources.

After having detailed the different modifications, in the next flowgraphs (figures 5.7, 5.8) are represented the updated work's principle with the new feature:

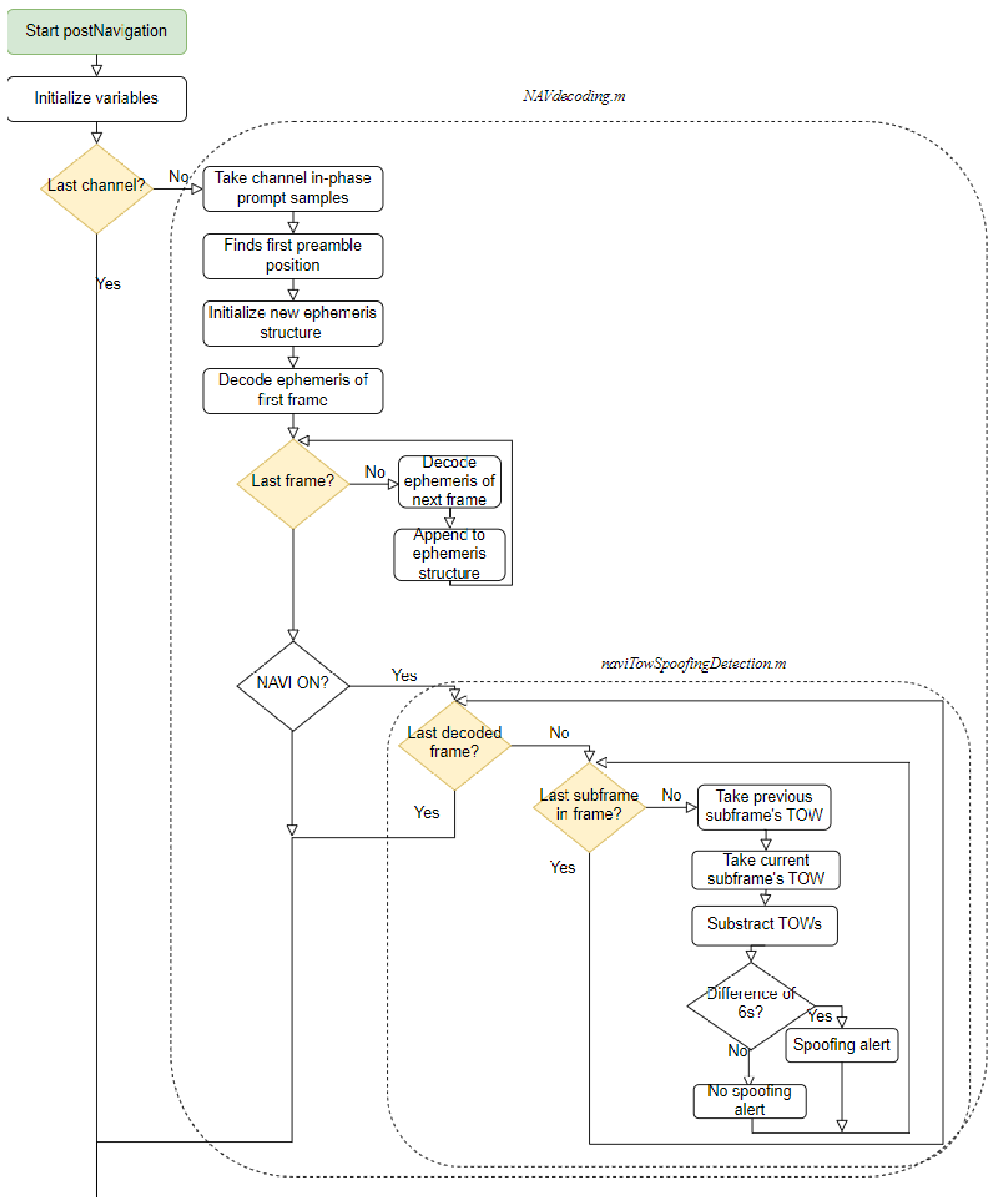

Start postNavigation
Initialize variables
NAVdecoding.m
Last channel?
No
Take channel in-phase prompt samples
Finds first preamble position
Initialize new ephemeris structure
Decode ephemeris of first frame
Yes
Last frame?
No
Decode ephemeris of next frame
Append to ephemeris structure
NAVI ON?
Yes
naviTowSpoofingDetection.m
Last decoded frame?
No
Yes
Last subframe in frame?
No
Take previous subframe's TOW
Take current subframe's TOW
Substract TOWs
Yes
Difference of 6s?
Yes
Spoofing alert
No
No spoofing alert

Figure 5.7: NAVI feature flowchart part 1 (Own source)

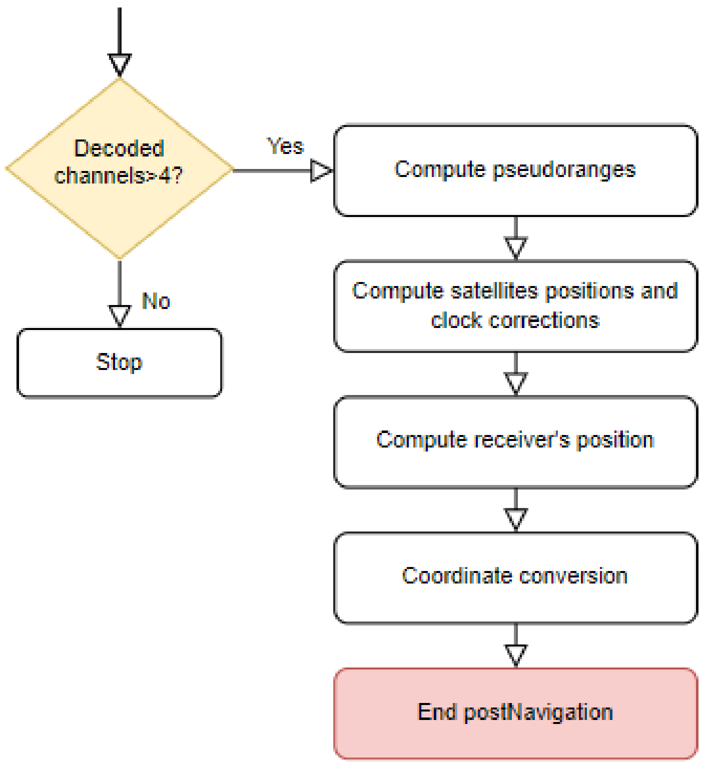


Figure 5.8: NAVI feature flowchart part 2 (Own source)

CHAPTER 6. VALIDATION AND RESULTS

This chapter elaborates on the tests carried out in order to evaluate the performance of the implemented spoofing detection techniques.

In particular, the method chosen to evaluate the performance is the Receiver Operating Characteristics (ROC) and the Area Under The Curve (AUC).

ROC is one of the most important evaluation metrics for validating classifications of model's performance. Actually, it is a graphical plot that represents the diagnostic ability of a binary classifier system as its discrimination threshold is varied. In other words, is the resulting curve after plotting the so-called True Positive Rate (TPR) as a function of False Positive Rate (FPR) for different threshold values. On the one hand, the TPR is also known as sensitivity, recall or probability of detection, whereas, on the other hand, FPR is also known as probability of false alarm.

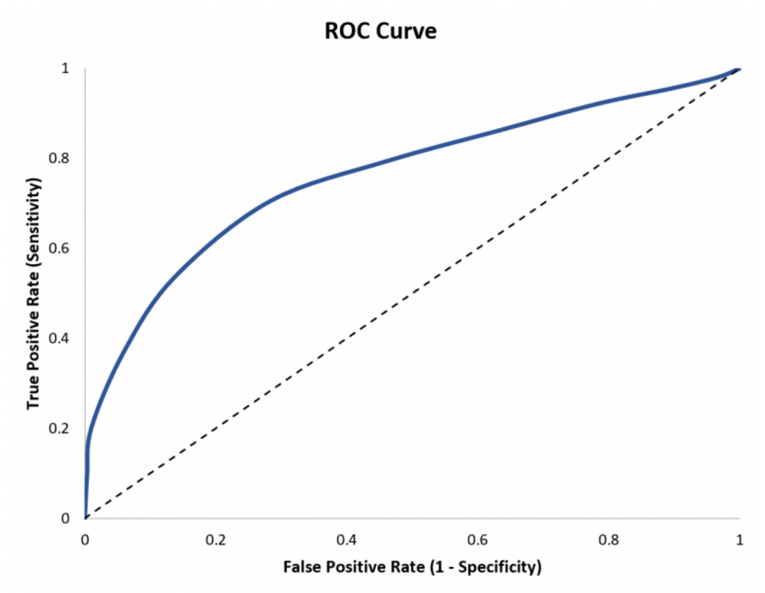


Figure 6.1: ROC example, extracted from (26)

Considering the AUC, by definition is a single scalar value corresponding to the entire area under the ROC curve. This indicator measures the overall performance of a binary classifier as, in fact, it represents the degree of separability, meaning that determines the capability of the model to distinguish between scenarios. Another manner to interpret the AUC is as the probability that the model will rank a random positive example higher than a random negative example.

Concerning separability, an AUC equal to 1 is computed from a perfect classifier model. Contrary, a model with an AUC equal to 0 means it is reciprocating the result. Besides, an AUC equal to 0.5, represents a model whithout any separation capacity, also known as a random classifier.

Thanks to the fact that the AUC computation takes into account the entire ROC curve (consequently involving all discrimination thresholds), the AUC is a robust measurement to evaluate the performance of score classifiers.

Some advantages of the AUC measurement are the followings:

- Presents invariability with respect to the scale, meaning that measures how well predictions rank, rather than their absolute values.
- Is non-dependant from the classification threshold, meaning that measures the quality of the model's predictions, regardless of the chosen classification threshold.

Regarding the spoofing detection issue, it can be considered as a binary detection problem as the objective of the implemented techniques is to conclude the presence of any spoofing attack. Hence, this two hypothesis can be defined:

- H_0 if there is no spoofing attack
- H_1 if there is spoofing attack

From that, it can be defined two main probabilities. On the one hand, the probability of false alarm, which is the probability that it is decided the hypothesis H_1 considering a spoofing attack when, in fact, there is no spoofing attack. On the other hand, the probability of detection, which is the probability that the hypothesis of the presence of a spoofing attack is accepted when in reality there is spoofing attack.

In the performed experiments, the presented probabilities have been estimated as:

$$P_{fa} = \frac{\#(H_1|H_0)}{N} \tag{6.1}$$

$$P_d = \frac{\#(H_1|H_1)}{N} \tag{6.2}$$

Being $\#()$ is the total number of satisfied argument conditions.

Hence, the probability of false alarm and probability of detection are a ratio between the number of times the test statistic exceeds the given threshold and the number of experiment realizations N. In particular, for these set of experiments N is 300.

The objective was to evaluate the overall performance of the spoofing detection technique APT and to test it among different scenarios so as to see if the technique is functional in different situations.

For the three scenarios presented in section 4.0.2., it has been performed the computation of both probabilities (probability of detection and probability of false alarm) for different *AptThresohld* (see table 5.1 for more details).

Next, are presented the ROC graphs for the three different scenarios and are extracted some conclusions out of them.

6.0.1. Non-synchronous and coherent: *DS3 PRN13*

In this scenario, the spoofer could not manage to perfectly synchronize from the beginning to the legitimate signal. Moreover, it entailed a considerable increase in the noise floor.

Next, it is presented the obtained ROC of the experiment:

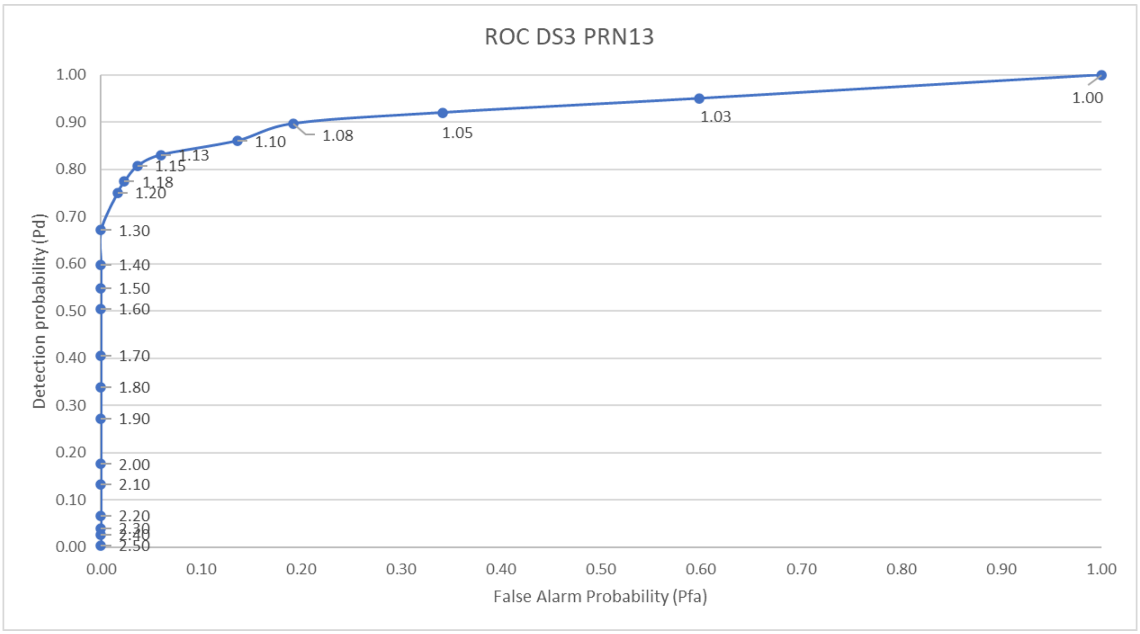


Figure 6.2: ROC of the APT spoofing detection technique in scenario DS3 PRN13 (Own source)

In the previous figure 6.2 it can be seen the performance of APT technique towards this specific attack. In particular, it can be noticed that, for a threshold higher than 1.3, the APT presents a probability of false alarm near to zero. However, there is no threshold that manages to detect the spoofing presence perfectly. In fact, a perfect detection is produced when the *AptThreshold* is 1, threshold which is not useful as all checks are considered as detection scenarios, leading to a probability of false alarm of 1.

Consequently, depending on the criteria of either wanting to ensure no false positive or wanting to maximize the probability of detection, the following two thresholds are decided to be the more optimal for this scenario:

- Criterion aiming to maximize the probability of detection:

  Contrastingly, it shall admit some probability of false alarm. The optimal threshold would be 1.08 as it presents a high probability of detection (0.9) and a considerably low probability of false alarm (0.2).

- Criterion aiming to minimize the false alarm:

  For this criterion, the best threshold is the one that provides the maximum probability of detection assuring a probability of false alarm of approximately zero. In this case, the *AptThreshold* is 1.3 which corresponds to a probability of detection of 0.68.

However, with these two criteria aiming to optimize one specific probability, the probability which has not been optimized results in a poor value. Hence, the *AptThreshold* that provides the most balanced performance in this scenario is 1.13, as presents a probability of detection of 0.83 and a probability of false alarm of 0.07.

The non-ideality of the graph is due to the fact that there are moments of oscillation when the spoofing attack occurs, since it involves an increase in the noise floor especially at the beginning and during the take over. Oscillations in this context are understood as, due to the randomness of the noise, the appearance and disappearance in some instants of the malicious peak. In other words, in some instants the malicious peak is hidden by the noise floor one and in other instants it is exposed. Although this phenomena complicates detection process, the detection of the spoofing attack can be considered effective enough.

6.0.2. Synchronous and coherent, coarse seamless take-over: *DS3 PRN23*

In this scenario, the spoofer could manage to perfectly synchronize from the beginning to the legitimate signal. However, when it tried to lead the receiver's victim away from the legitimate signal it could not manage to hide the legitimate signal, leaving behind traces that facilitated the spoofing detection.

The next figure depicts the results of the ROC for this particular spoofing scenario:

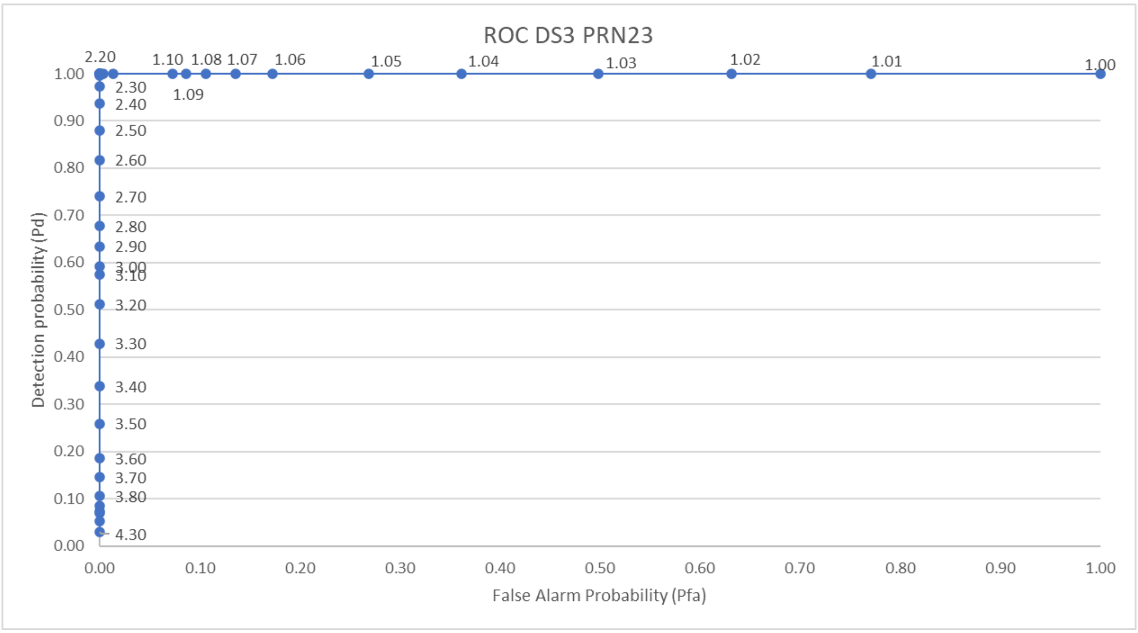


Figure 6.3: ROC of the APT spoofing detection technique in scenario DS3 PRN23 (Own source)

In front of this attack, the APT technique presents an ideal case, which results in a perfect separability of both events for an *AptThreshold* of 2.2. Therefore, the APT is able to perfectly detect the presence of the spoofing attack.

This ideality can be attributed to the fact that the attack is coarse and there is no rise in the noise floor that helps the spoofer to camouflage the legitimate peak. As coarse in this context it is understood the attack that does not increases its power once it has synchronized with the legitimate peak in the code-phase axis, thus making the malicious peak dominant with respect to the legitimate peak. As a result, when the spoofer manages to perform the take over in the receiver's victim tracking module and deviates the malicious peak from the legitimate peak, both peaks are exposed. Consequently, the clear exposure of both peaks facilitate the spoofing detection as it has been seen in the previous figure (6.3).

In addition, the number of experiment realizations also limits its precision. With a larger number of experiment it could be found that values are not exactly 0 nor 1.

Although the spoofing detection in this scenario seems to be ideal, it is important to consider that the detection of a synchronous attack is always more difficult than the detection of a non-synchronous attack. In a synchronous attack, the detection the APT can provide is a detection post-take-over, whereas in a non-synchronous attack, the detection the APT can be even before the take-over.

Hence, in this synchronous coarse take-over, the detection is ideal but the detection has been done once the spoofer has already taken the control of the victim's receivers control. Thanks to the ideal detection of the APT, the spoofing attack is detected so rapidly that the spoofer does not manage to induce a wrong position to the victim.

6.0.3. Synchronous and coherent, Fine seamless take-over: *DS7 PRN23*

In this scenario, the spoofer could manage to perfectly synchronize from the beginning to the legitimate signal. In addition, when it tried to lead the receiver's victim away from the legitimate signal it could remarkably hide the legitimate signal. However, it does not manage to hide the legitimate peak completely, so it also leaves behind traces that facilitated the spoofing detection. Compared to the previous synchronous coarse seamless take-over attack, the traces are not so obvious.

In the following figure appears the experimentally obtained ROC curve of the *DS7 PRN23* attack:

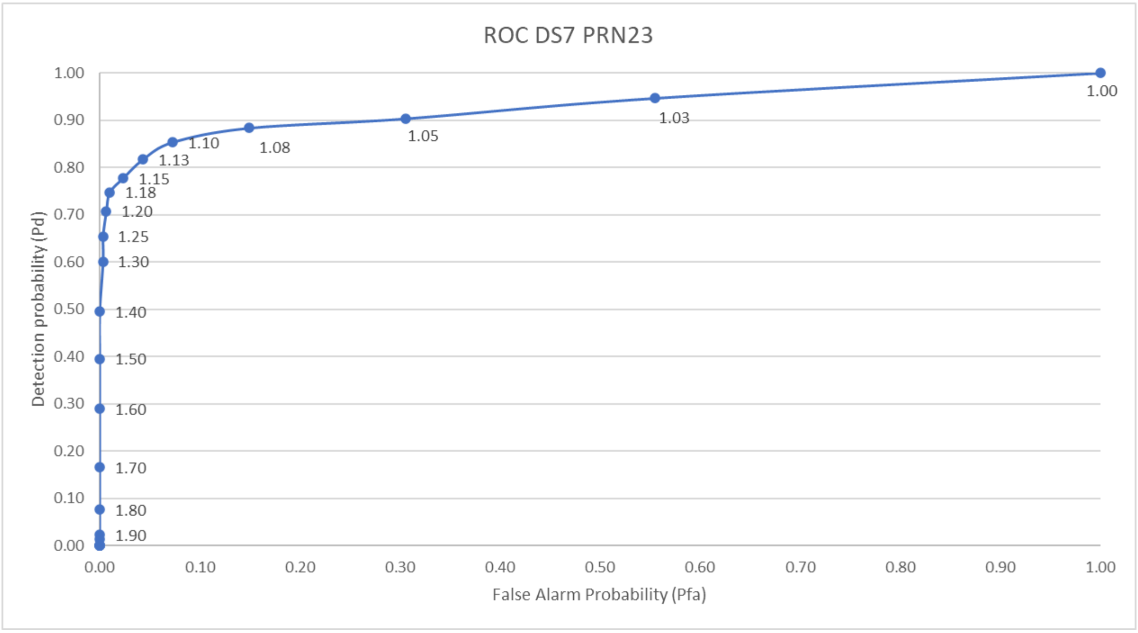


Figure 6.4: ROC of the APT spoofing detection technique in scenario DS7 PRN23 (Own source)

In the former figure 6.4 it can be seen that, in this attack in particular, for a threshold higher than 1.4, the APT presents a probability of false alarm close to zero. However, analogous the *DS7 PRN13* scenario, there is no threshold that manages to detect the spoofing presence perfectly apart from a threshold equal to 1, which has been discussed not to be relevant in this matter.

As a result, depending on the desired criteria, the optimal thresholds for this scenario are:

- Criterion aiming to maximize the probability of detection:

  The optimal threshold would be 1.08 as it presents a high probability of detection (0.89) and a considerably low probability of false alarm (0.15).

- Criterion aiming to minimize the false alarm:

  The best threshold is the one that provides the maximum probability of detection assuring a probability of false alarm close to zero. In this case, the *AptThreshold* that accomplishes that is 1.4 and entails a probability of detection of 0.5.

  Nonetheless, the *AptThreshold* 1.18 could be more interesting as, despite the fact that the probability of false alarm is not strictly 0, it provides a remarkably higher probability of detection 0.75 compared to the 0.5 probability of detection that the *AptThreshold* 1.4 presents.

For this attack, the most balanced *AptThreshold* is 1.1, since it corresponds to a remarkably high probability of detection (0.86) and a relatively low probability of false alarm (0.08) In this scenario, the non-ideality of the graph is caused by the fact the malicious miss-leading of the receiver's victim is careful and precise. Once the spoofer has managed to synchronize with the legitimate signal and starts driving away the receiver's victim tracking module, it performs the variation gently. As a result, at the beginning of the drive-off manoeuvre, for the APT is especially complicated to detect the attack. Moreover, the spoofer, by increasing its power, manages to partially sink the legitimate signal into the noise floor, which also contributes to complicate the APT detection purposes.

Even tough the sophistication of this attack, the detection of the spoofing attack can be considered effective enough as well.

6.0.4. Overall results conclusions

This section aims to discuss the overall performance of the APT spoofing detection technique. First of all, are gathered together in a single table the results of the different ROC experiments. Secondly, it is graphically compared the different ROC by plotting the curves in the same graph. Next, it is compared for the three experiments the AUC and it is interpreted. Finally, out of the different measurements are extracted final conclusions regarding the overall APT performance.

Down below, in table 6.1 are gathered the previous results of the different performed experiments:

Next, in figure 6.5 are plotted in the same graph the three obtained ROC in order to properly compare them:

Scenario	Criterium	AptThreshold	Pd	Pfa	Effectiveness
DS3 PRN13	Maximum Pd	1.08	0.9	0.2	Good
	Maximum Pfa	1.3	0	0.68	
	Balance	1.13	0.83	0.07	
DS3 PRN123	Maximum Pd	2.2	1	0	Good
	Maximum Pfa	2.2	1	0	
DS7 PRN23	Maximum Pd	1.08	0.89	0.15	Good
	Minimum Pfa	1.18	0.75	0.02	

Table 6.1: ROC experiments summary table

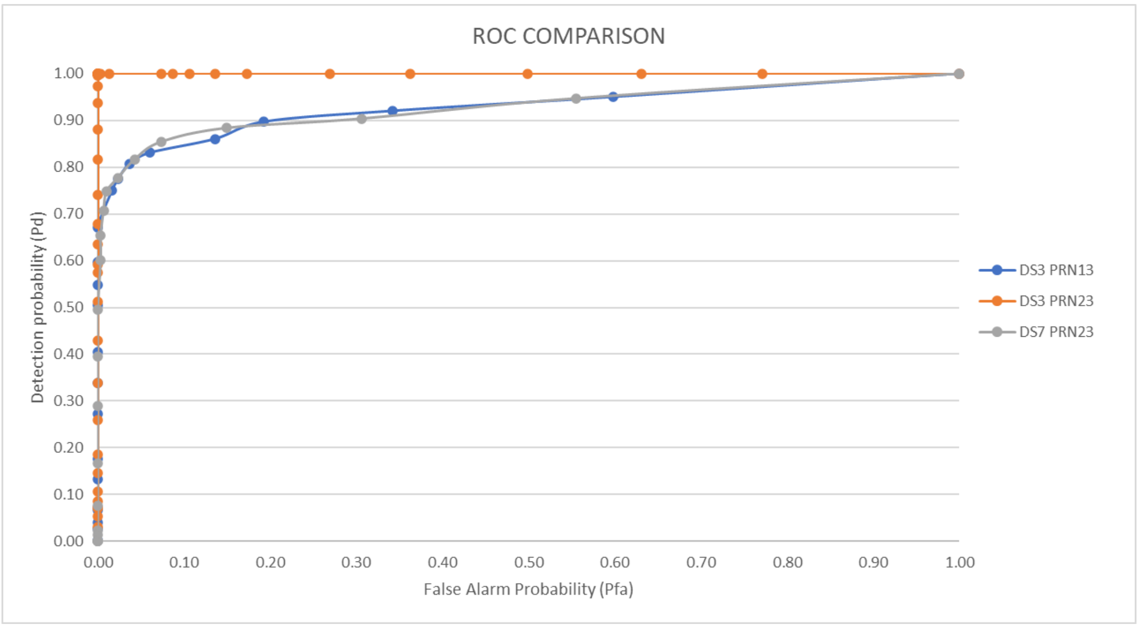


Figure 6.5: ROC of the APT spoofing detection technique for all scenarios (Own source)

From the ROC curves it can be computed the AUC for each spoofing scenario, the next table 6.2 summarizes the AUC results:

Scenario	AUC
DS3 PRN13	0.9277
DS3 PRN23	1
DS7 PRN23	0.9273

Table 6.2: AUC table resume

The average AUC for the three scenarios is 0.9517, which is really close to the ideal scenario. Therefore, it can be concluded that the APT detection technique is a good classifier, meaning that it successfully manages to identify the presence of an spoofing attack.

After having analyzed a particular case of a non-synchronous attack, it can be extracted the following generalized conclusion about this kind of attacks. Although this attack's typology seems to be the least sophisticated as it is not able to synchronize from the beginning, it presents a special danger. This hazard is due to the fact that, due to the high noise it involves when performing the take-over manoeuvre, it is easy that the receiver's victim either does not manage to detect the appearance of the malicious peak or even loses the tracking's lock thinking that the satellite is no longer visible. This latter case it is also dangerous for the victim receiver's as, when the take over is finished, the acquisition module may acquire the malicious signal. Moreover, once it manages to sync up, it has been observed that this kind of attack presents the capability of partially hiding the legitimate peak.

One of the measures against non-synchronous attacks that involve an increase of the noise floor would be to discard the channels that suddenly present an increase in the noise floor as long as there are enough visible satellites that enable the receiver to calculate the PVT solutions. If in doubt, dismiss noisy acquired satellites signals as there may present an spoofing attack.

Regarding the synchronous attacks, it has been seen that although the coarse scenario (DS3 PRN23) is more sophisticated than the non-synchronous attack since it is capable of perfectly synchronizing with the legitimate peak from the beginning, once it deviates from the legitimate peak it is exposed to spoofing detection.

In this case, the detection is triggered later as the APT does not have the capability to prevent the attack before it takes control of the victim receiver but it can be detected very effectively and quickly once the spoofer tries to cause a change that is detrimental to the correct obtantion of the PVT solution from the victim receiver. Although ideally it would be great to detect the attack before it takes-over the receiver's victim tracking module, an effective and rapid detection of the attack once it is detrimental to the receiver can already be considered as a good performance of the detection method.

After having discussed the overall performance, from the extracted results it can be selected a general *AptThreshold* of 1.08 as, for both scenarios DS3 PRN13 and DS7 PRN13 is the optimal threshold for the criterium maximizing the probability of detection and, for the DS3 PRN23, an *AptThreshold* of 1.08 only entails an probability of false alarm of 0.12 meanwhile it maintains a perfect probability of detection.

CONCLUSIONS AND FUTURE WORK

Conclusions

The initially defined project main objective was to implement some of the most promising spoofing techniques. However, this general objective has been divided in several tasks.

The first step was to refresh all the satellite communications basics previously studied, as well as summarizing them in order to include a theory fundamentals in the memory.

The second step was to perform an study of the art research so as to discover the current studied and developed techniques so far. In addition, an evaluation of the analysed techniques was required in order to properly decide which spoofing detection techniques were implemented. The main conclusion extracted from this tasks was the decision of discarding the promising cryptographic solutions due to the fact that involved changes in the current GPS architecture, which clashed with the fact that the GPS is spread and currently used world-wide in a large number of applications. Instead, it was concluded to implement a combination of two compatible techniques, the APT and NAVI, as each technique belonged to a different layer (the physical and logical respectively).

The next step was to get familiar with the software in which the work was intended to be implemented. For this purpose, it was decided to sketch different flowgraphs that depicted the logical behaviour of the code and helped understand the work flow. The Borre's software enabled the scalability, addition and modification of new blocks without the need to drastically change the overall architecture of the original code.

Once the code was understood, the following step was to find and analyze a set of spoofed signals for which it could be tested the implemented techniques. From the state of the art research tasks, it could be seen that the TEXBAT spoofed signals set was highly recognized among the community, in fact, it was considered to be the standard when testing spoofing detection techniques. Nevertheless, one of the main difficulties the entire project presented was the proper analysis of these signals due to two main reasons. The first challenge was found when initially the Borre's software was not able to properly process complex samples data sets when the initial reading time was different from 0, therefore it required signals reading adaptation tasks. The second obstacle was the fact that, although the TEXBAT signals were widely recognized by the community, no proper description of the spoofing scenarios was useful for the project purposes. As a consequence, a wide analysis of the data sets was required in order to find the scenarios that fitted the most to properly test the implemented spoofing techniques.

At this point, after having finished the preliminary tasks, the subsequent task to carry out was the implementation of the spoofing detection techniques. The preliminary tasks facilitated this tasks since, at this moment, I had very clear the code's functioning and the spoofing detection technique requirements. Moreover, I felt very comfortable with the Matlab programming language as in several bachelor courses involved projects coded in this particular programming language.

Nevertheless, the main challenge presented in the APT implementation was to optimize the APT detection checks reading, as the initial developed version required an excessive execution time. This issue was related to the initial incompetence of the software in reading signal's fragments that its starting reading point was different from the initial file.

Regarding the main NAVI's implementation challenges, it is worth mentioning the difficulties decoding the first frame as in some cases was incomplete. Finally it was decided to decode it separately from the other complete frames.

Eventually, the last task was to obtain a metric experiment in order to test the performance of the implemented spoofing techniques. The selected metric experiment was to obtain the ROC curve. With this metric, it has been possible to discuss the performance of the detection techniques in some scenarios and extract some interesting conclusions which were not obvious at first sight.

In these results, one of the main conclusions that can be drawn is that a non-synchronous but high-noise attack can be surprisingly dangerous. This is due to the fact that, even though this attacks do not present a high sophistication degree compared to the synchronous attack, therefore the malicious peak has to synchronize little by little, the noise manages to camouflage it. Besides, once the takeover is done, it also manages to camouflage the legitimate peak.

Thus, proposed countermeasure against non-synchronous attacks would be:

- To discard channels that suddenly present an increase in the noise floor, especially if the application presents demanding safety requirements.
- To discard channels that experiment a loss of lock even tough the lock could be subsequently recovered, as the loss can be attributed to an spoofing attack with the floor noise increase feature.

Consequently, for the spoofing detection techniques is also crucial the abundant presence of visible satellites.

Finally, from the results, it can also be concluded the proper operation of the APT detection technique in the several studied scenarios, which was the main objective of the project.

Future work

Apart from the work carried out in this project, in this section are proposed some ideas for further exploration in this field, exploration which could take as starting point the work developed in this project:

The first idea would be to test the already developed algorithms with future TEXBAT data sets. The University of Texas from time to time publish new and more sophisticated spoofing scenarios and, if in the future they do so, it would be interesting to see the performance of the developed techniques towards the new spoofing strategies. For instance, in this project, induced by the fact that all TEXBAT signals set made the same assumption, it has been assumed that the spoofing attacker could manage to perfectly align the signals in the Doppler frequency axis. However, may be in the future the spoofers try to perform either the take-over or the drive-away process in the Doppler axis as well. Hence, it would be a new possible feature for the APT to keep track of secondary peaks in both Doppler frequency and code-phase axis. It would be more demanding computationally-wise but, with the current implementation of the APT, it would be feasible and easy to implement.

The second proposal would be to test altogheter the APT and NAVI detection techniques and evaluate if the overall detection performance is improved. Nevertheless, it would be necessary to have spoofing signals that comprised both spoofing attacks, which we did not have access to.

Another future work proposal lies on the idea of implementing the APT and NAVI techniques on a real time software defined radio, such as GNSS-SDR (27), which is an open-source project that implements a global navigation satellite system software-defined receiver in C++. The Borre's software is a great tool for teaching purposes but it is limited for real time applications.

In GNSS-SDR software, it also processes other GNSS systems such as Galileo E1, Galileo E6, GPS L2 C/A an much more. Hence, it also be appealing to adapt the developed spoofing detection techniques to these other GNSS systems.

As well, this project could be expanded by implementing some other spoofing detection techniques such as the other methods detailed in the state of the art research 2.2.

Finally, it also would be interesting to perform a deep-learning-based approach so as to analyze the physical characteristics of GNSS signals and detect abnormal sequences.

BIBLIOGRAPHY

[1] UT Austin Researchers Successfully Spoof an 80 dollar million Yacht at Sea. http://news.utexas.edu/2013/07/29/ut-austin- researchers-successfully-spoof-an-80-million-yacht-at- sea. 5

[2] Hacking A Phone's GPS May Have Just Got Easier. https://www.forbes.com/sites/parmyolson/2015/08/07/gps-spoofing-hackers-defcon/ 5

[3] Nighswander, T., Ledvina, B. M., Diamond, J.,Brumley, R., and Brumley, D. "GPS software attacks." *In Proceedings of the ACM Conference on Computer and Communications Security*. (2012) 32

[4] Ranganathan, A., Ólafsdóttir, H. and Capkun, S. "SPREE: A Spoofing Resistant GPS Receiver".*Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. 348–360. (2016) xi, 30, 33, 35, 37

[5] Papadimitratos, Panagiotis, and Aleksandar Jovanovic. "GNSS-based positioning: Attacks and countermeasures." *MILCOM 2008-2008 IEEE Military Communications Conference. IEEE*, (2008) 33, 36, 37

[6] Sathaye, H., LaMountain, G., Closas, P., and Ranganathan, A. "Semperfi: A spoofer eliminating GPS receiver for uavs". *arXiv preprint arXiv:2105.01860*. (2021) 38

[7] Tippenhauer, N. O., Popper, C., Rasmussen, K. B., and Capkun, S. "On the requirements for successful GPS spoofng attacks". *In Proceedings of the 18th ACM Conference on Computer and communications security* (2011) 33

[8] Daneshmand, S., Jafarnia-Jahromi, A., Broumandon, A., and Lachapelle, G. "A low-complexity GPS anti-spoofing method using a multi-antenna array". *In Proceedings of the 25th international technical meeting of the satellite division of the institute of navigation*. 1233–1243. (2012) 36

[9] Google Summer of Code llc. https://harshadsathaye.com/gsoc21/ftnt2https

[10] Cadence, AWR Microwave Office Element Catalog https://awrcorp.com/download/faq/english/docs/Elements/biastee.htm xi, 10

[11] Jafarnia-Jahromi, A., Lin, T., Broumandan, A., Nielsen, J., and Lachapelle, G. "Detection and mitigation of spoofing attacks on a vector based tracking GPS receiver".*In Proceedings of the 2012 International Technical Meeting of the Institute of Navigation*. 790–800. (2012) 35

[12] Magiera, J. and Katulski, R. "Detection and mitigation of GPS spoofing based on antenna array processing". *Journal of applied research and technology*. **13**(1), 45–57. (2015) 36

[13] Semanjski, S., Semanjski, I., De Wilde, W., and Muls, A. "Use of supervised machine learning for GNSS signal spoofing detection with validation on real-world meaconing and spoofing data—Part I".*Sensors*. **20**(4), 1171. (2020)

[14] Wesson, K., Rothlisberger, M., and Humphreys, ”T. Practical cryptographic civil GPS signal authentication”. *Journal of Navigation*. (2012) 34

[15] Kuhn, M. G. ”An asymmetric security mechanism for navigation signals”. *In Information Hiding*. (2005) 34

[16] E.D. Kaplan and C.J. Hegarty, (Ed), ”Understanding GPS: Principles and Applications”, 2ndEd., *Boston: Artech House*. (2006) xi, xiii, 20, 21, 95

[17] Borre, Kai, et al. A software-defined GPS and Galileo receiver: a single-frequency approach. Springer Science Business Media. (2007) xi, xiii, 18, 39, 95, 97

[18] Adrian, P., Ran, C., Tygar, J. D., Dawn, S. ”The TESLA broadcast authentication protocol.” RSA CryptoBytes 5. (2002) 34

[19] Schneier, B. ”Applied Cryptography, 2nd edn. Hoboken.” (1996) 34

[20] Paar, Christof, and Jan Pelzl. ”Understanding cryptography: a textbook for students and practitioners”. Springer Science Business Media. (2009) 34

[21] Humphreys, Todd. ”Statement on the vulnerability of civil unmanned aerial vehicles and other systems to civil GPS spoofing.” University of Texas at Austin (July 18, 2012) (2012): 1-16. 53

[22] Humphreys, Todd E., et al. ”The Texas spoofing test battery: Toward a standard for evaluating GPS signal authentication techniques.” Radionavigation Laboratory Conference Proceedings. (2012) xiii, 54

[23] Fyfe, Peter, and Karl Kovach. ”Navstar GPS space segment/navigation user interfaces (public release version)”. RESEARCH CORP FOUNTAIN VALLEY CA. (1991) xi, 21, 52

[24] Wang, Wenyi, Na Li, Renbiao Wu and Pau Closas. ”Detection of induced gnss spoofing using s-curve-bias.” Sensors 19.4 (2019): 922. xi, 31

[25] https://github.com/gnsscusdr/CU-SDR-Collection/tree/main/GPS/GPS_L1CA

[26] https://www.statology.org/interpret-roc-curve/ 39

[27] https://gnss-sdr.org/ xii, 79

APÈNDIXS

APPENDIX A. KEPLERIAN ELEMENTS

	Ephemeris data element	Scale Factor	Units	Bits
t_{0e}	Reference time of ephemeris	2^{4}	s	16
$\sqrt{a}$	Semimajor axis square root	2^{-19}	$\sqrt{m}$	32
e	Eccentricity	2^{-33}	-	32
i_0	Inclination angle	2^{-31}	semicircle	32
Ω_0	Ascending node longitude	2^{-31}	semicircle	32
ω	Perigee argument	2^{-31}	semicircle	32
M_0/μ_0	Mean anomaly	2^{-31}	semicircle	32
$\dot{i}$	Inclination angle rate of change	2^{-43}	semicircle/s	14
$\dot{\Omega}$	Ascending node longitude rate of change	2^{-43}	semicircle/s	24
Δn	Mean motion correction	2^{-43}	semicircle/s	16
C_{uc}	Amplitude of cosine correction to argument of latitude	2^{-29}	rad	16
C_{us}	Amplitude of sine correction to argument of latitude	2^{-29}	rad	16
C_{rc}	Amplitude of cosine correction to orbital radius	2^{-5}	m	16
C_{rs}	Amplitude of sine correction to orbital radius	2^{-5}	m	16
C_{ic}	Amplitude of cosine correction to inclination angle	2^{-29}	rad	16
C_{is}	Amplitude of sine correction to inclination angle	2^{-29}	rad	16
IODE	Issue Of Data, Ephemeris	-	-	8

Table A.1: Ephemeris data elements [table extracted from (16) and (17)]

APPENDIX B. ALMANAC DATA

	Keplerian element	Scale Factor	Units	Bits
SV$_{ID}$	Satelite identification	1	-	7
$\sqrt{a}$	Semi-major axis square root	2^{-11}	$\sqrt{m}$	24
e	Eccentricity	2^{-21}	-	16
i_0	Orbit inclination	2^{-19}	semicircle	16
ω	Perigee argument	2^{-23}	semicircle	24
Ω_0	Ascending node longitude	2^{-23}	semicircle	24
$\dot{\Omega}$	Ascending node longitude rate of change	2^{-38}	semicircle/s	16
M_0/μ	Mean anomaly	2^{-23}	semicircle	24
a$_{f0}$	Clock correction parameter	2^{-20}	s	15
a$_{f1}$	Clock correction parameter	2^{-38}	s/s	11
SV$_{SHS}$	Satellite's signal health	-	-	5
SV$_{DHS}$	Satellite's navigation data health	-	-	3
Data$_{ID}$	Applicable navigation data structure	-	-	2
IODA	Almanac set idenfitication	-	-	2
t$_{0a}$	Reference time	4096	s	8
WN$_a$	Week number	1	week	8

Table B.1: Almanac elements [table extracted from (17)]