Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Centre de Tecnologia Aeroespacial

TIGHT HYBRIDISATION FOR POSITIONING WITH GPS AND WLAN

by

Daniel Fernández Fernández

Advisors: Marc Ciurana and Francisco Barceló

Barcelona, 2010

This project has been developed in the Navigation Group at CTAE (Centre de Tecnologia Aeroespacial) in conjunction with the Departament de Telemàtica of the UPC (Universitat Politècnica de Catalunya).

ACKNOWLEDGEMENTS

First of all I want to thank Marc Ciurana and Francisco Barceló for their time and patience. It was very easy to work with them. They were always willing to help me.

I want to thank all the CTAE team for their support. They made me feel at home and so it was less hard to wake up early all the mornings.

Finally, I want to thank all my friends and family. Without them, I wouldn't be the person I am.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 MOTIVATION AND OBJECTIVES

In the last few years there has been an incrementing interest on LBS (Location Based Services) and ITS (Intelligent Transport Systems), and therefore location systems able to calculate user's position have become crucial. GNSS (Global Navigation Satellite Systems), especially GPS (Global Positioning System), has been offering this kind of services for 20 years, but it doesn't work properly in all environments.

Despite the latest advances in GNSS high-sensitivity receivers for harsh environments, signal blocking caused by buildings makes it unfeasible to consider GNSS as an overall positioning solution to cover outdoor urban areas entirely. Since this can limit the deployment of LBS and ITS in urban zones, numerous proposals exist for augmenting GNSS availability with terrestrial signals. Most of them employ public cellular networks like GSM (Global System for Mobile Communications)/GPRS (General Packet Radio Service), UMTS (Universal Mobile Telecommunications System), etc. and are therefore constrained in terms of flexibility, due to operator dependency, and accuracy. Another alternative considered is the use of pseudolites [1], but they are not suitable for widespread use due to legislation and cost reasons.

Figure 1.1: AP on the top of a traffic light

In the last years local authorities are deploying WLAN (Wireless Local Area Network) access points (WiFi, IEEE 802.11) on the streets of many cities in order to provide municipal services or wireless internet access (e.g. Figure 1.1, an AP located on the top of a traffic light in the city of Barcelona), while the integration of GNSS and WLAN in a single device is now very common. The main objective of this project is to take advantage of these existing WLAN infrastructures to increase urban positioning coverage (i.e. availability) of GNSS by means of a flexible and cost-effective solution, thus achieving a practical positioning system able to combine WLAN and GNSS signals, keeping the location accuracy as close as possible to that provided by GNSS alone.

1.2 RELATED WORK

Hybrid systems can be divided in two big groups: loose coupling and tight coupling. In the first one, an estimated position is calculated with each technology composing the system. Afterwards, a combination of both positions is calculated to provide a final position. This approach is simple, but it has a drawback that the final position cannot be determined if any of the previous estimated positions cannot be calculated. In the second one, raw measurements with each technology are performed and then combined, providing a final position. In the case of non-synchronized hybridisation, it is assumed that there is no time synchronization between the terrestrial and satellite networks; while in synchronized hybridisation terrestrial and satellite networks are synchronized (i.e. satellites and Base Stations can be seen as belonging to the same network). The drawback of tight coupling is that if not enough measurements are performed, the algorithm that calculates the position can diverge. The advantage is that a position can be determined with few measurements. An extended explanation can be found in [2].

A hybrid system is composed of two or more technologies. One of them is usually GNSS and the others are terrestrial technologies. The most common for GNSS is GPS while for terrestrials are WiFi, UWB (Ultra-Wideband) and GSM/GPRS. In [3], a positioning system is presented hybridising GPS and Galileo with WiFi, UWB and MEMS (Micro-Electro-Mechanical Systems), using loose coupling technique. In [4] and [5], a tight coupled GNSS/UWB positioning system is proposed. The first one is synchronized while the second one is non-synchronized. In [6], a non-synchronized tight coupling system combining GPS and television signals is presented. An important amount of systems have been found, but as far as we know no one hybridising GPS and WiFi using a non-synchronized tight coupling technique.

1.3 GENERAL APPROACH

As said in Section 1.1, the objective of the project is achieving a positioning system able to combine WLAN and GNSS signals. Given the low correlation between the coverage of both contributing systems, a hybridisation at observable level (i.e. tight coupling) seems to be the best option (more

information is available at Section 1.2). The hybrid system will be non-synchronized because it reduces complexity and avoids hardware modifications.

As it is seen in Figure 1.2, the proposed system is divided in 3 subsystems. The first two subsystems are used to calculate the distance to the reference points. In the first one, all the satellites available are detected and their pseudoranges are calculated. In the second one, all the terrestrial APs (Access Points) available are detected and their ranges are calculated. Finally, with this information and the position of the satellites and the APs[1], an iterative multilateration algorithm that handles the combined observation equations calculates the position of the user.

In Chapter 2 it is explained the method to calculate de distance to the terrestrial AP and in Chapter 3 they are explained the changes made to the conventional GPS Positioning Algorithm to convert it to a Hybrid Positioning Algorithm. Special emphasis is dedicated to the phenomenon of positioning ambiguity that appears due to the incorporation of the terrestrial measurements. Although the work has been performed considering WLAN and GPS, most of the presented ideas, procedures and results are also valid for generic tight non-synchronized fusion of GNSS with terrestrial signals [7].

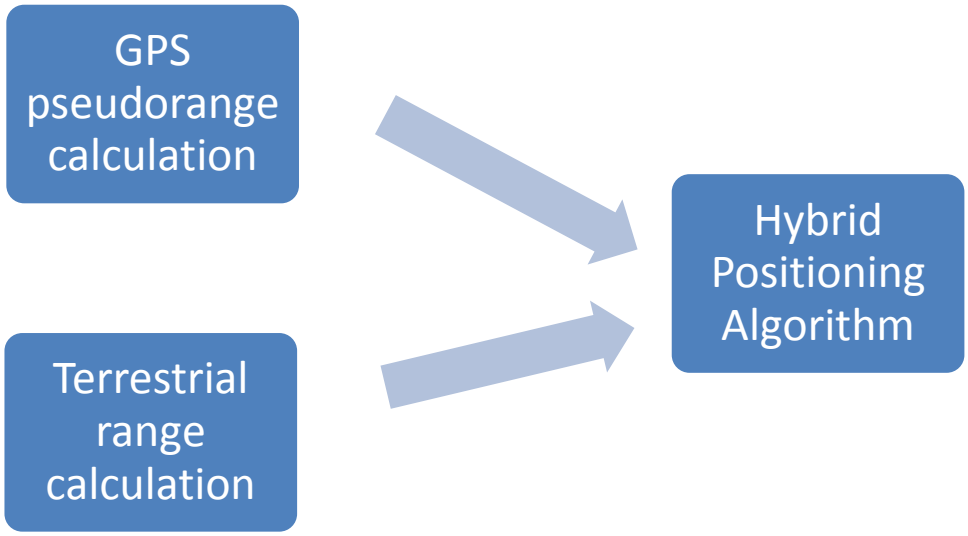

Figure 1.2: Subsystems of the proposed system

1.4 GPS: HOW IT WORKS

As it is said in [8], GPS utilizes the concept of Time of Arrival (TOA) to determine user position. This concept consists on measuring the time it takes for a signal to travel from an emitter (whose position is known) to a receiver. This time, known as propagation time, is then multiplied by the speed of the signal (usually the speed of light) to calculate the distance between the emitter and the receiver. By measuring the propagation time of the signal from multiple emitters at known locations, the receiver can determine its position.

[1] It is assumed that the information of the APs position is available. Otherwise, the algorithm could not calculate an estimated position.

If a two-dimensional position wants to be determined, at least three emitters are required. Assuming that emitters and the receiver are synchronized, with only two emitters there are two possible solutions:

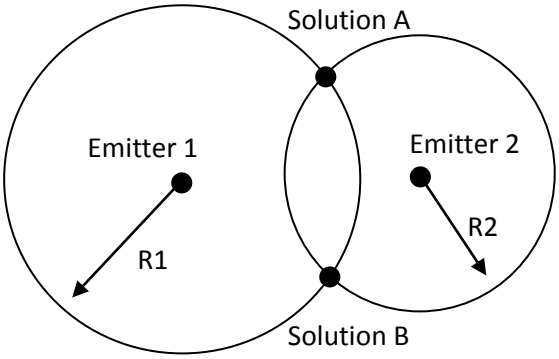

Figure 1.3: Solution ambiguity resulting from using two emitters

The propagation time from the emitter 1 multiplied by the speed of the signal results in the distance R1. The same happen for the emitter 2. It is seen in figure 1.3 that two emitters lead to two possible solutions, both at the same distance to emitter 1 and 2. To solve this ambiguity, another emitter is incorporated:

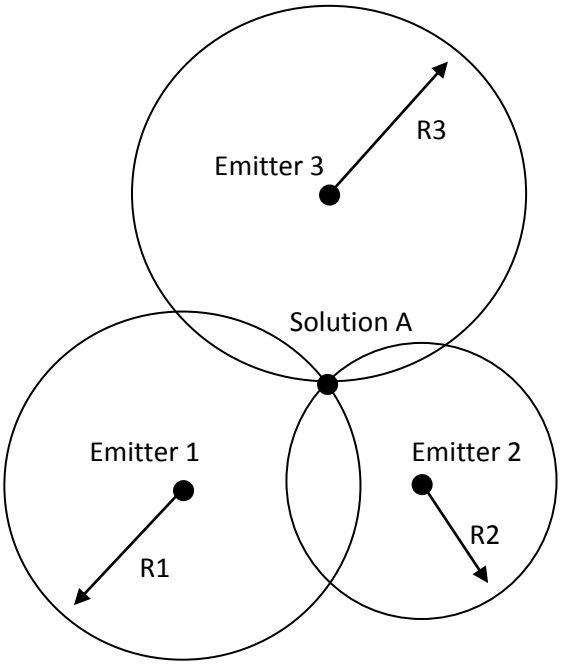

Figure 1.4: Ambiguity resolution incorporating the third emitter

In a GPS system, the emitters are satellites, and so the circles become spheres. By making TOA measurements to multiple satellites, three dimensional positioning is achieved. The intersection of two spheres results in a circumference (Figure 1.5). When a third satellite is considered, the surface of its sphere intersects on two points of the circumference, thus resulting in an ambiguity of

solutions (Figure 1.6). Unlike the two-dimensional positioning case, three emitters are not enough to calculate the position. To solve the ambiguity, a forth satellite is needed.

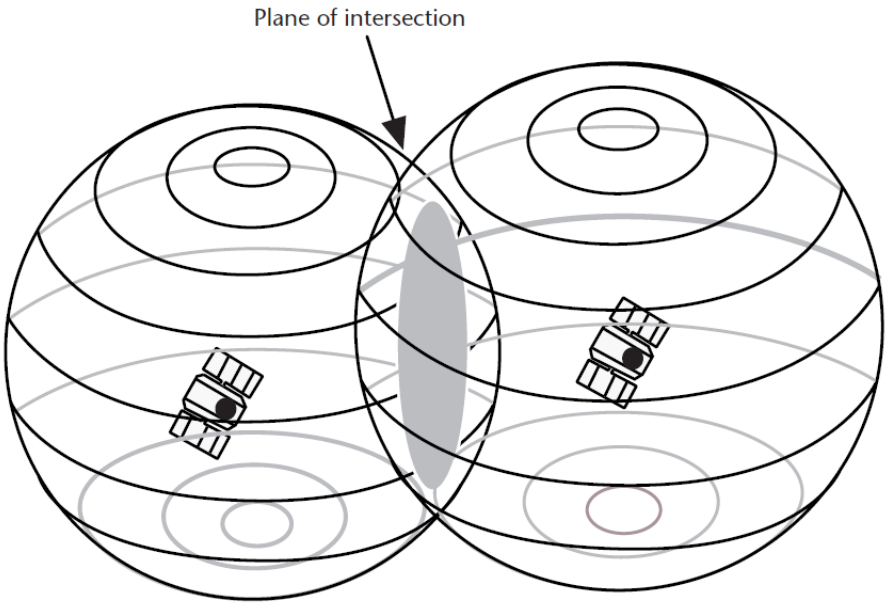

Figure 1.5: Intersection of two spheres [8]

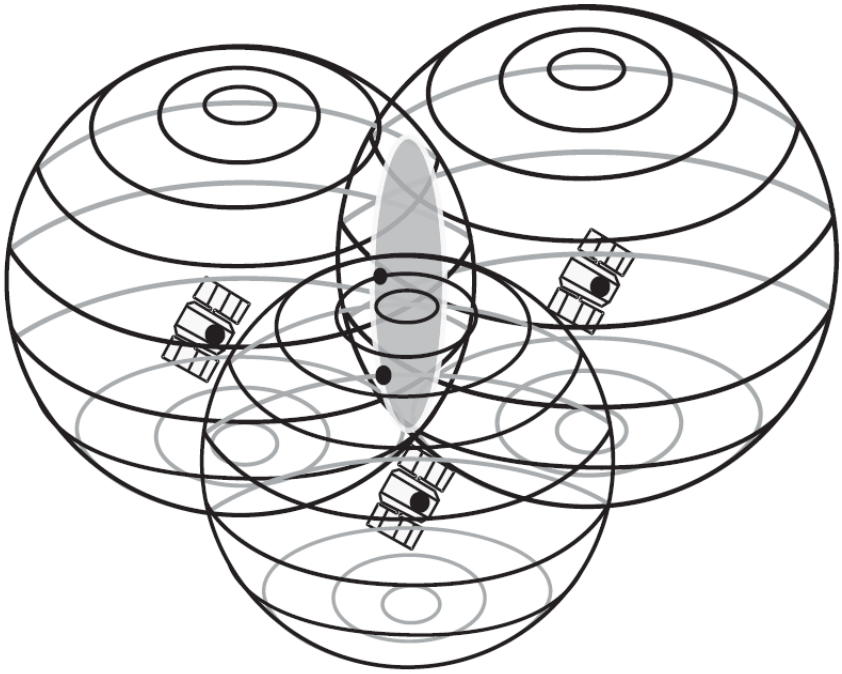

Figure 1.6: Intersection of three spheres [8]

The above considerations are made assuming that emitters and receivers are well synchronized, but that is not the case in real life. The satellite clock and the receiver clock usually have a drift respect

to the GPS system time. That provokes an error when calculating the distance between the user and the satellite. Considering this error, the calculated distance between user and satellite is called pseudorange.

The satellite clock drift is usually much smaller than the receiver clock drift because satellites use atomic clocks, while the receivers use normal clocks. Taking in account this clock drifts, it can be determined the equations of GPS (called observable equations). The times in Figure 1.7 are:

- T_s : System time at which the signal left the satellite
- δt : Offset of the satellite clock from system time
- T_u : System time at which the signal reached the user receiver
- t_u : Offset of the receiver clock from system time

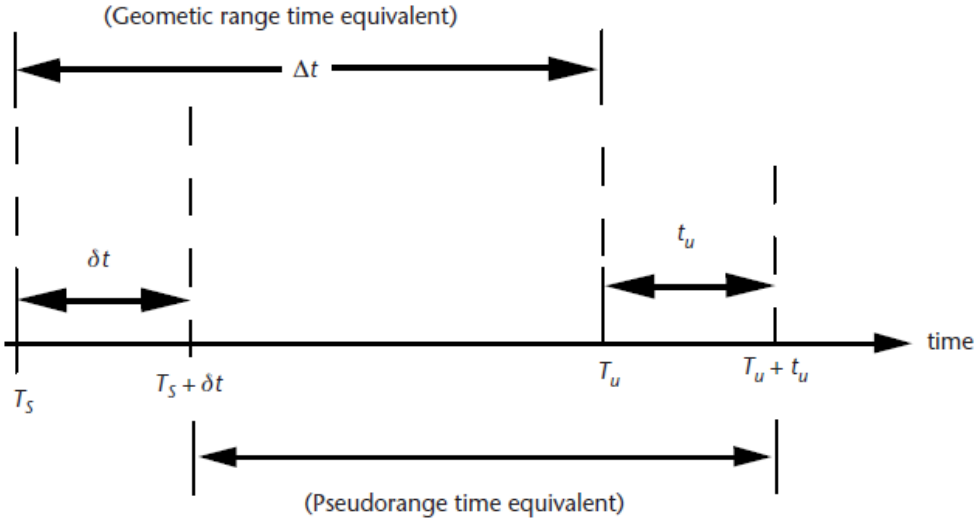

Figure 1.7: Range measurement timing relationship [8]

The geometric range can be expressed as:

- $r = c(T_u - T_s) = c\Delta t$, where c is the speed of light

And the pseudorange as:

- $\rho = c[(T_u + t_u) - (T_s + \delta t)] = c(T_u - T_s) + c(t_u - \delta t) = r + c(t_u - \delta t)$, where c is the speed of light and r is the geometric range

The satellite clock offset from system time, δt, is corrected by the GPS ground-monitoring network, who transmits the corrections to the satellites for rebroadcast to the users. Therefore, it is no longer considered as an unknown. Hence, the pseudorange is expressed as:

- $\rho = r + ct_u$

If r is the distance from GPS satellite to the user, r can be expressed as:

- $r = ||s - u||$, where s represents the satellite coordinates and u the user coordinates

Finally, each observation equation can be expressed as:

- $\rho_i = ||s_i - u|| + ct_u = \sqrt{(x_i - x_u)^2 + (y_i - y_u)^2 + (z_i - z_u)^2} + ct_u$, where x_i, y_i, z_i represent the coordinates from satellite i, and x_u, y_u, z_u, t_u represent user coordinates and clock offset.

2 DISTANCE ESTIMATION PERFORMED BY THE TERRESTRIAL SUBSYSTEM

2.1 INTRODUCTION

Many techniques exist to calculate the distance from a receiver to one or more reference points. Some are based on time and others on power measurements. TOA (Time of Arrival) is based on time. It consists on measuring the time it takes for a signal to travel from the emitter to the receiver. This time is then multiplied by the speed of the signal, achieving the distance. This technique offers a good positioning precision due to its linearity with distance [9], but it is usually expensive to implement in a WLAN. RSSI (Received Signal Strength Indicator) is based on power measurements. It consists on measuring the power received from an emitter. Later, the power is converted into distance by a propagation model. This technique cannot offer as good accuracy as TOA, but it is easier and less expensive to implement than TOA. Although both techniques are considered to test the system, it is decided to work with RSSI because of the facility of implementation.

As said above, RSSI needs a propagation model to convert power into distance. There exist different propagation models that represent the channel path loss, but the most accurate for the project is:

- $\Pr = P(0) - 10 * \alpha * \log(d)$, where $\Pr$ refers to received power (RSSI), $P(0)$ is an initial constant that refers to received power at 1m, α refers to the propagation constant and d refers to the distance between emitter and receiver.

This propagation model will be used in Section 2.6, where α and $P(0)$ are determined, enabling a link between power and distance, thus obtaining a ranging model.

All measurements and tests previous to the measurements are performed with LOS (Line Of Sight) in outdoor scenarios[2], simulating conditions similar to an urban canyon. The technology used is WiFI IEEE 802.11g.

2.2 RELATED WORK

Propagation models can be categorized into three types: empirical, deterministic and stochastic. Empirical models are based on observations and measurements. These models are mainly used to predict path loss. The deterministic models use the laws governing electromagnetic wave propagation to determine the received signal power at a particular location. Stochastic models, on

[2] Except tests performed at CTAE's laboratory.

the other hand, model the environment as a series of random variables. More information can be found at [10].

For this project, it is decided to use an empirical model, whose equation is commented in Section 2.1. In order to check the values of α and $P(0)$, many propagation models for WLAN signals are studied. In [11], outdoor measurements are performed achieving values of α from 2.54 to 3.11. In [12], an α of 1.67 is obtained in an outdoor environment and compared with an α of 2.76 obtained in other experiments. In [13], a novel system is presented that adapts the propagation constant depending on the environment.

2.3 TOOLS USED TO CALCULATE POWER

As it is said in Section 2.1, the method selected to calculate the terrestrial range is RSSI. For that reason, a program that scans the APs available and reports a measure of their power is searched. One of the essential requirements of the program is that it has to be able to export the measurements to a text file, because they are subsequently processed by Matlab calculation program. Network Stumbler is the first tried. When the program is executed, a list of all the APs available and their SNR (Signal to Noise Ratio), RSS (Received Signal Strength) and Noise appears. The units used to show RSS values are dBm, but when they are exported to a text file they don't agree (i.e. they are not dBm). Searching on the Internet, it was found that the values corresponding to the text file are relative values and therefore a conversion table RSSI-dBm is needed. That table is not standardised and each manufacturer defines its particular approach. The table corresponding to the used NIC (Network Interface Card) Intel PRO/Wireless 3945ABG was searched on the Internet, but it was not found. Manufacturers were answered to provide this information, but no response was obtained. Network Stumbler was discarded.

Other similar programs were searched, but no one accomplished all the requirements. Another way of doing measurements was found: *iwlist* command. This command scans all the APs available and gives some information about each of them. The most important characteristic is that the units of the power are dBm. The problem of Network Stumbler was solved, but *iwlist* does not have the option to export the measurements to a text file. It was searched on the Internet the file where Linux store all the power measurements, and it was found to be *proc/network/wireless*. This file is permanently updating, but it only reports measurements if it is connected to a network, and that is not the case. Therefore, it was decided to create a simple script that obtains the information of the *iwlist* command and writes it to a text file. The problem is that the time of execution of the *iwlist* command varies and so a sampling rate cannot be set.

2.4 PREVIOUS TESTS BEFORE MEASUREMENTS

A study of the behaviour of three NICs has been carried out at CTAE's laboratory. Simultaneous static measurements have been made for an hour to assess the behaviour of each NIC respect to the same transmitted power. The used NICs are an Intel PRO/Wireless 3945ABG, a Netgear WG511T and a D-Link DWL-G122. The AP used is a D-Link DWL2100AP. The three NICs have been plugged to a computer, which has been situated three meters from the AP with LOS.

As it is commented above, *iwlist* command execution time varies. For that reason, the comparison of the three NICs cannot be done in number of samples. It is seen in Figure 2.1 that in one hour each NIC has made a different number of samples. That means that the sampling time also depends on the NIC used.

The objective of the study is to determine the measurement time needed at each location in order to obtain a robust channel model and thus a realistic ranging model.

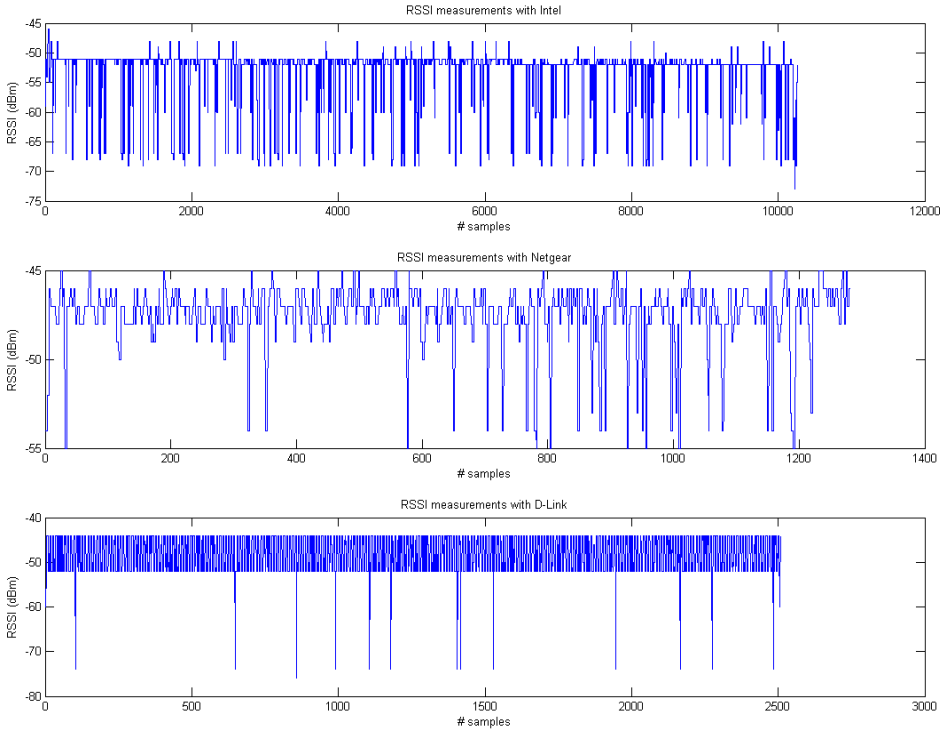

Figure 2.1: RSSI in each NIC

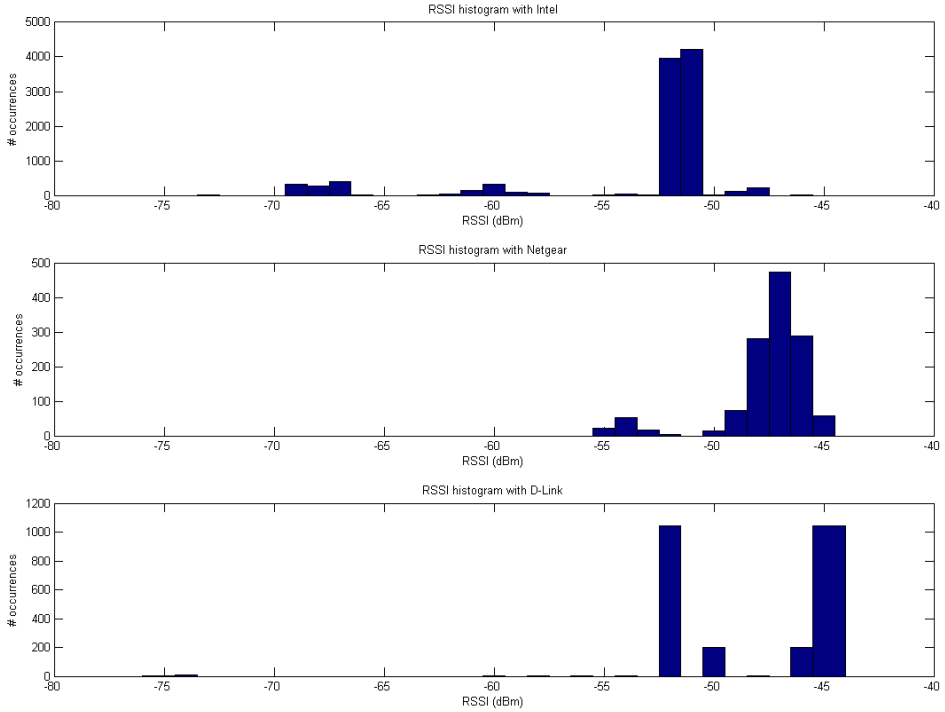

Figure 2.2: RSSI Histogram in each NIC

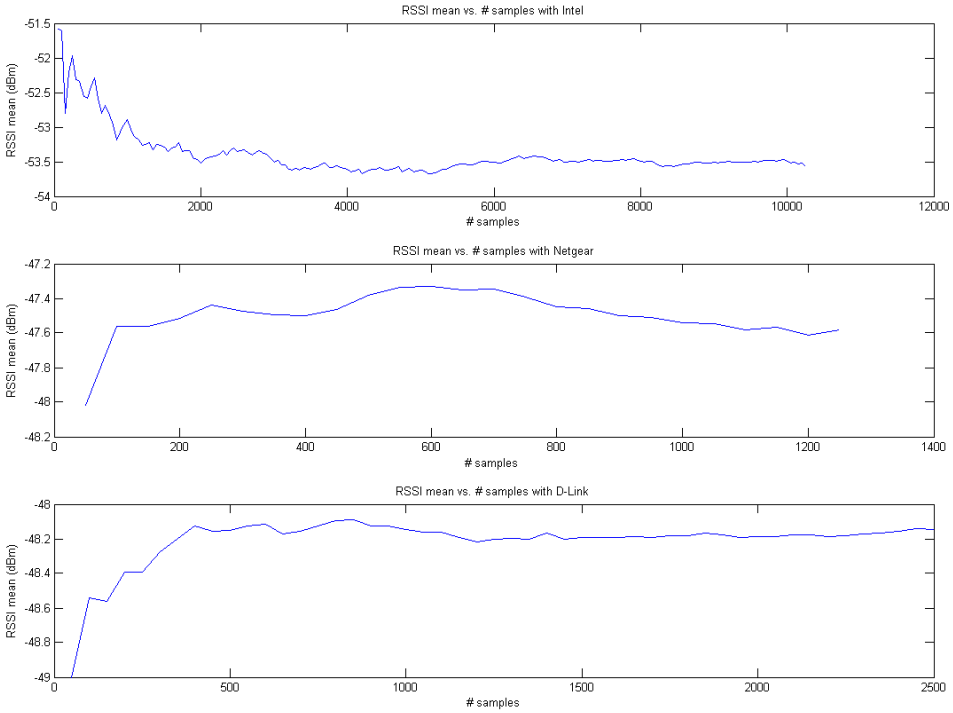

Figure 2.3: RSSI mean as a function of number of samples

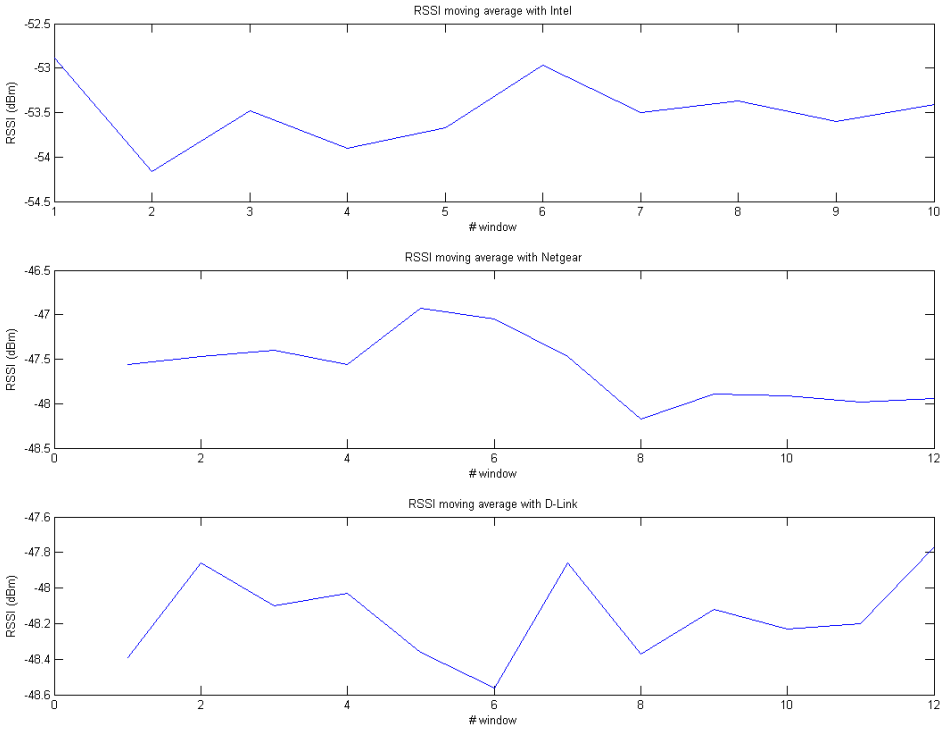

Figure 2.4: RSSI moving average

As it is seen in Figure 2.2, certain periodicity is shown in the RSSI measurements of D-Link's NIC. This behavior is not appreciated in the other NICs. For that reason, it is decided to discard D-Link NIC from the study. Referring to the measurement time, it can be seen in Figure 2.3 that after approximately 15 minutes a stable mean is achieved.

Considering this results, it is decided to perform more measurements at UPC (Universitat Politècnica de Catalunya), where an urban canyon is simulated. Specifically, the measurements take place between the buildings C3 and D3. The AP is situated in the junction of the buildings C2, C3, D2 and D3, and the mobile device (i.e. the computer) is shifted to achieve distances of 5m, 15m, 25m and 35m (the location can be seen in more detail in Figure 2.5). 30 minutes of measurements are performed in each localisation. The equipment used is the Intel and Netgear NICs mentioned above and the Linksys WAP54G Access Point. A stool 70 cm high is used to raise the devices from the floor (see Figure 2.6).

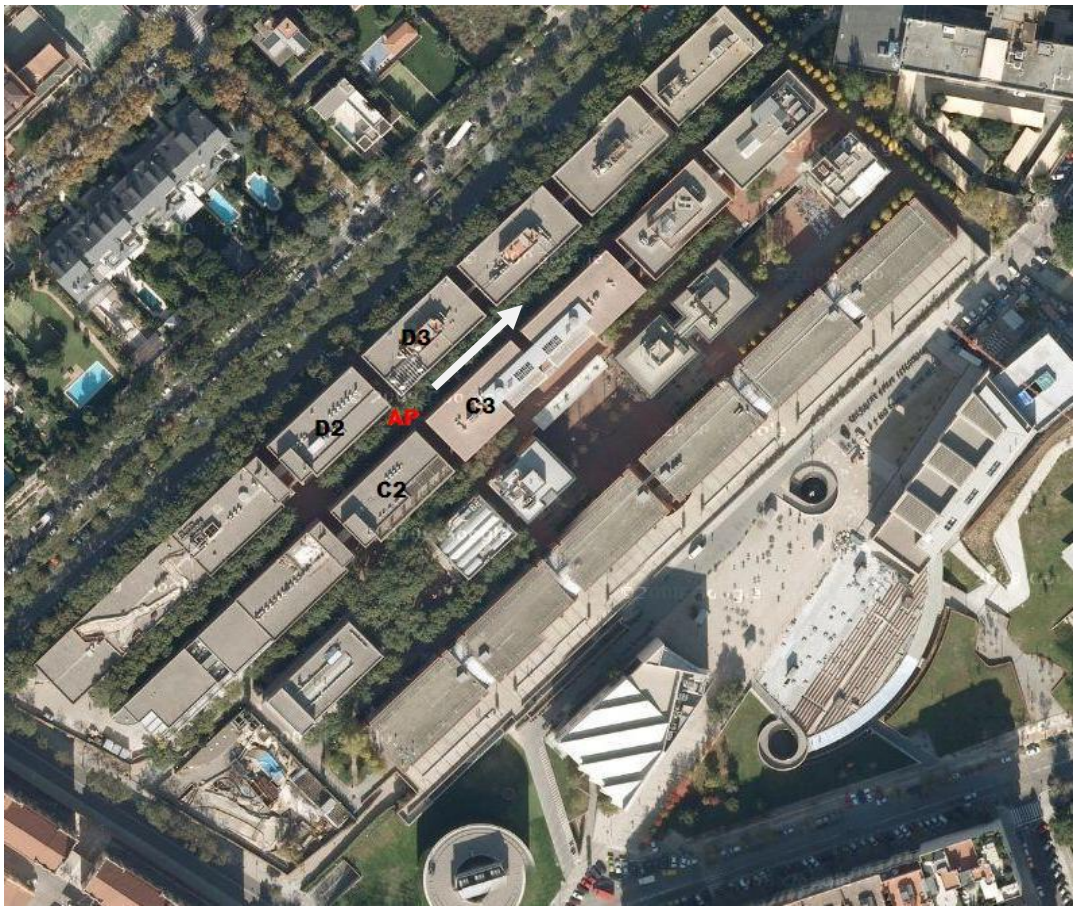

Figure 2.5: Location of the UPC measurements

Figure 2.6: Measurements at UPC

The results obtained are shown in the next figures:

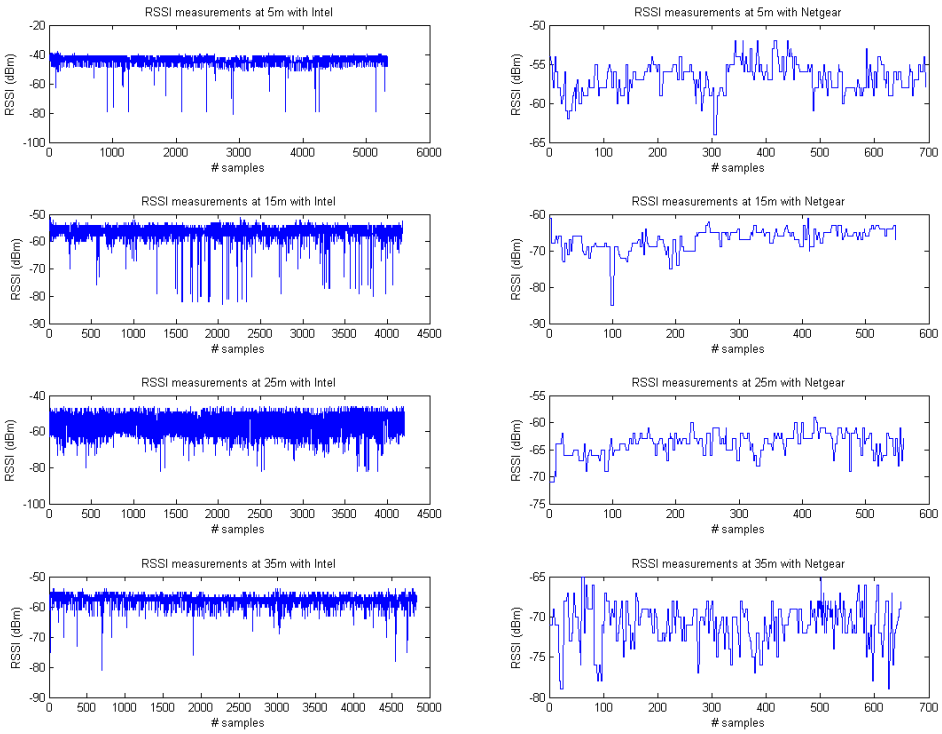

Figure 2.7: RSSI measurements for different distances with each NIC

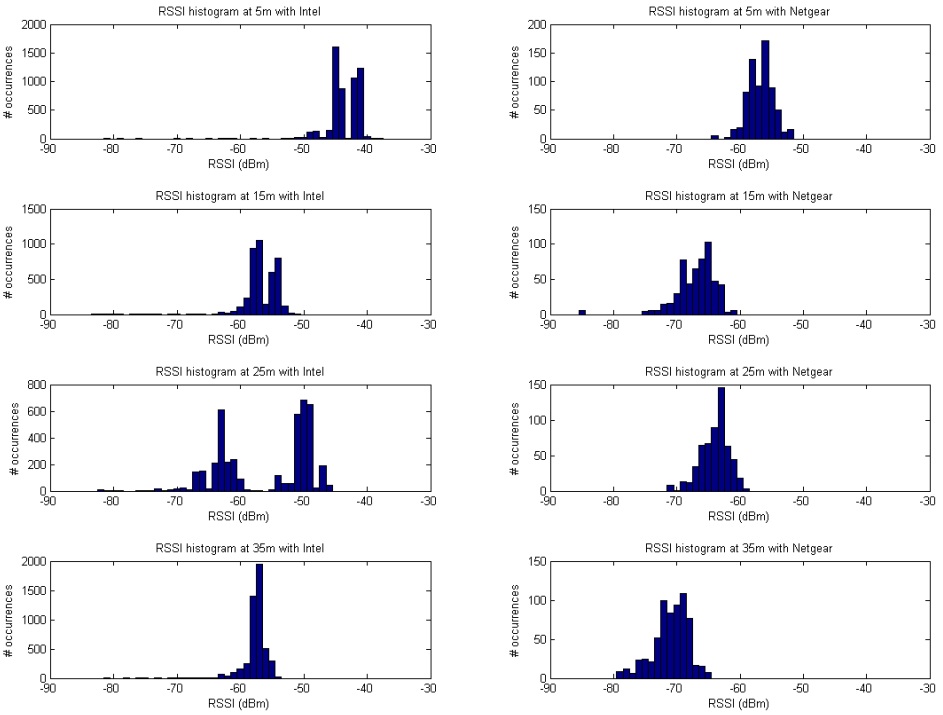

Figure 2.8: RSSI histograms for different distances with each NIC

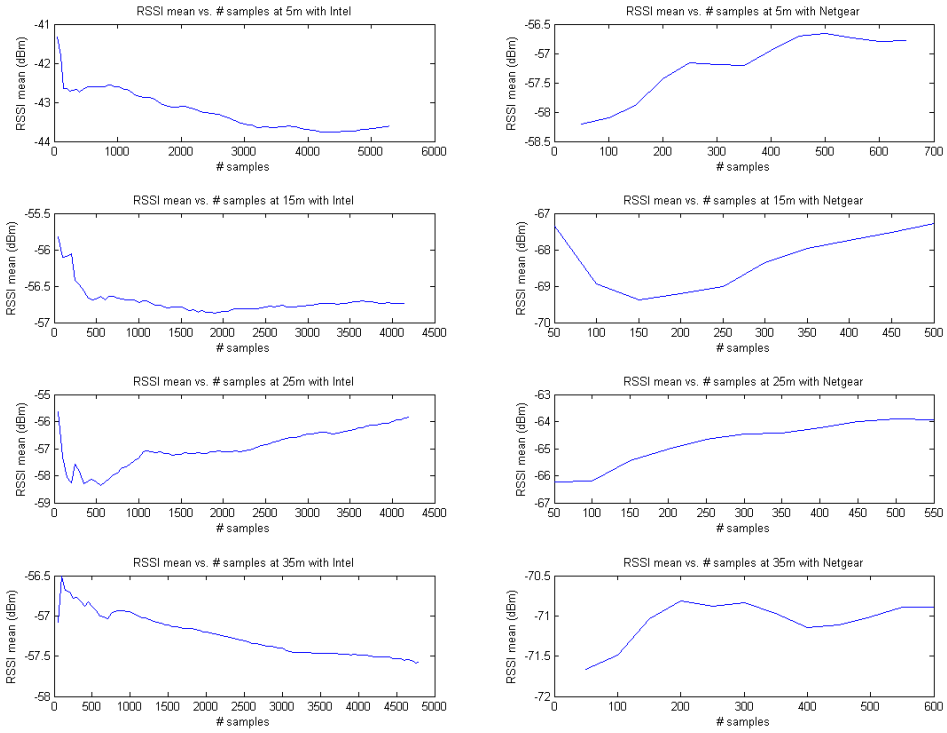

Figure 2.9: RSSI mean as a function of the number of samples for different NICs and localisations

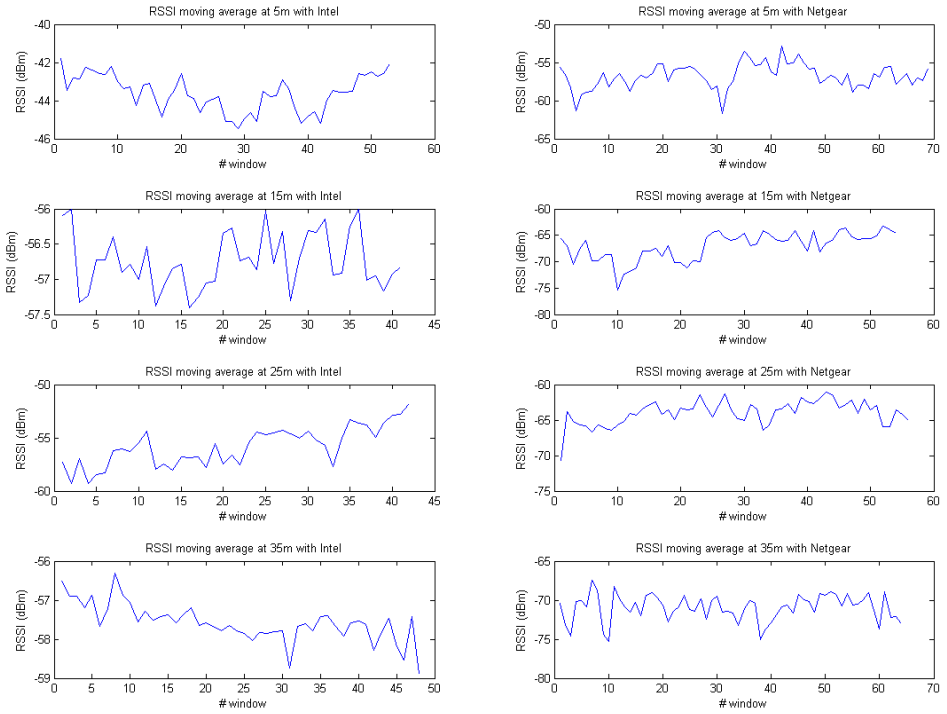

Figure 2.10: RSSI moving average for different NICs and localisations

As can be seen in the histograms in Figure 2.8, the RSSI distribution is generally better if Netgear NIC is used. At 5m, 15m and 25m two principal components are observed with Intel NIC, while Netgear NIC only has one. However, at 35m the Intel NIC distribution is better than the Netgear one because it has a unique component with a smaller variance. Therefore, measurements will be performed with both targets, thus providing more robust results.

Referring to the measurement time, it can be seen in Figure 2.8 that after approximately 20 minutes a stable mean is achieved. Therefore, 20 minutes will be spent for each localisation.

2.5 MEASUREMENTS

In order to obtain robust results, measurements are carried out along two directions. The first one (direction 1) corresponds to the space between the buildings C3 and D3, and between C4 and D4. The second one (direction 2) corresponds to the space between C2 and C3, and between B2 and B3 (see Figure 2.11). In both directions, the distance between consecutive measurements is 5m. In direction 1, measurements are carried out until 100m, while in direction 2 are carried out until 50m. In both scenarios, a stool 70 cm high is used to raise the AP and the mobile device from the floor.

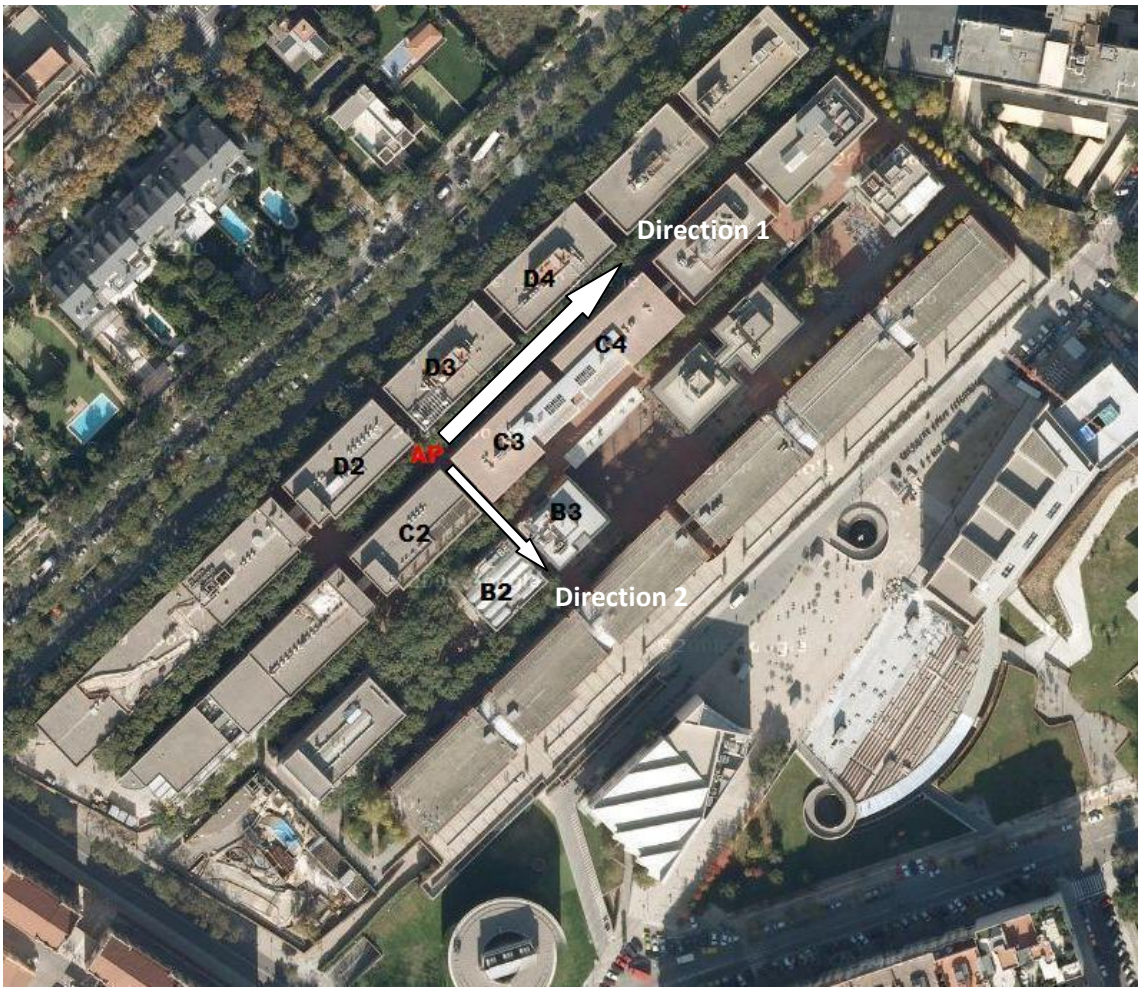

Figure 2.11: Direction of measurements carried out at UPC

The next results are obtained for direction 1:

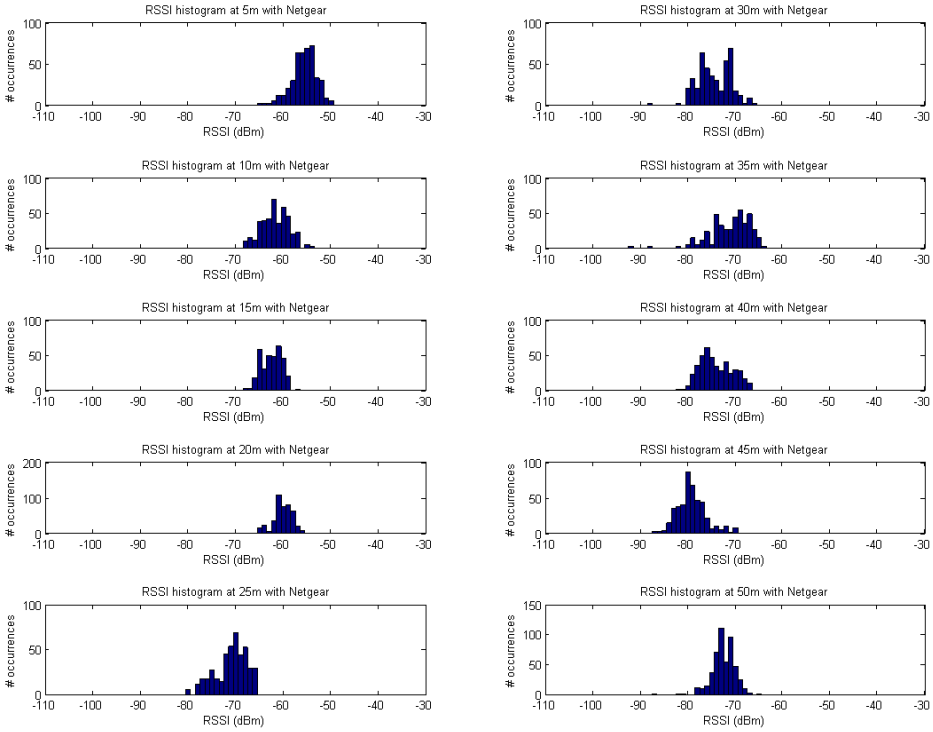

Figure 2.12: RSSI histograms for the first 50m with Netgear NIC in direction 1

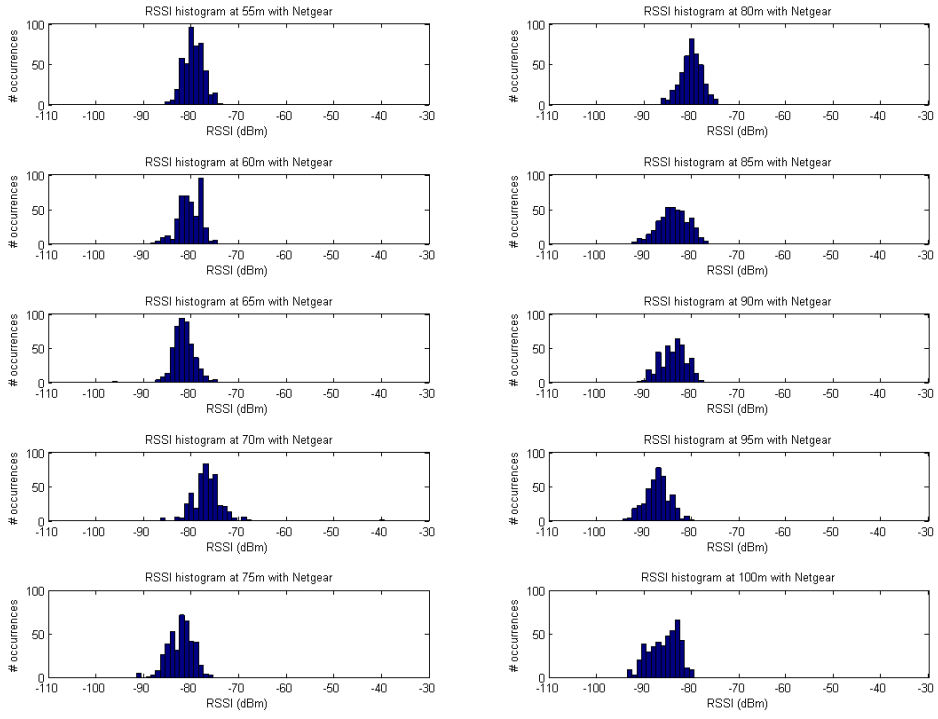

Figure 2.13: RSSI histograms for the last 50m with Netgear NIC in direction 1

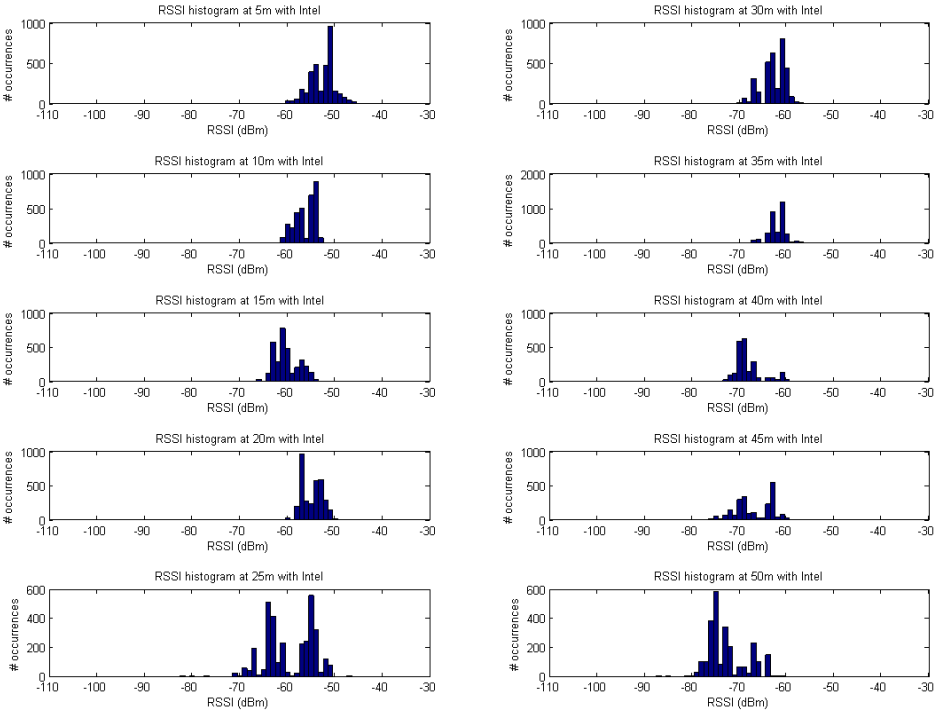

Figure 2.14: RSSI histograms for the first 50m with Intel NIC in direction 1

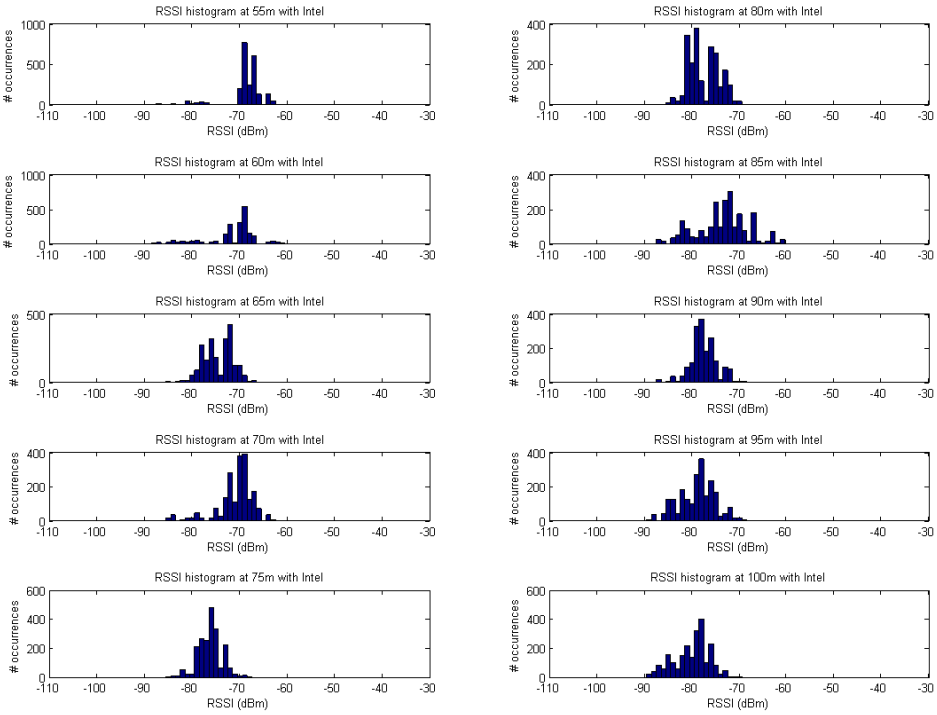

Figure 2.15: RSSI histograms for the last 50m with Intel NIC in direction 1

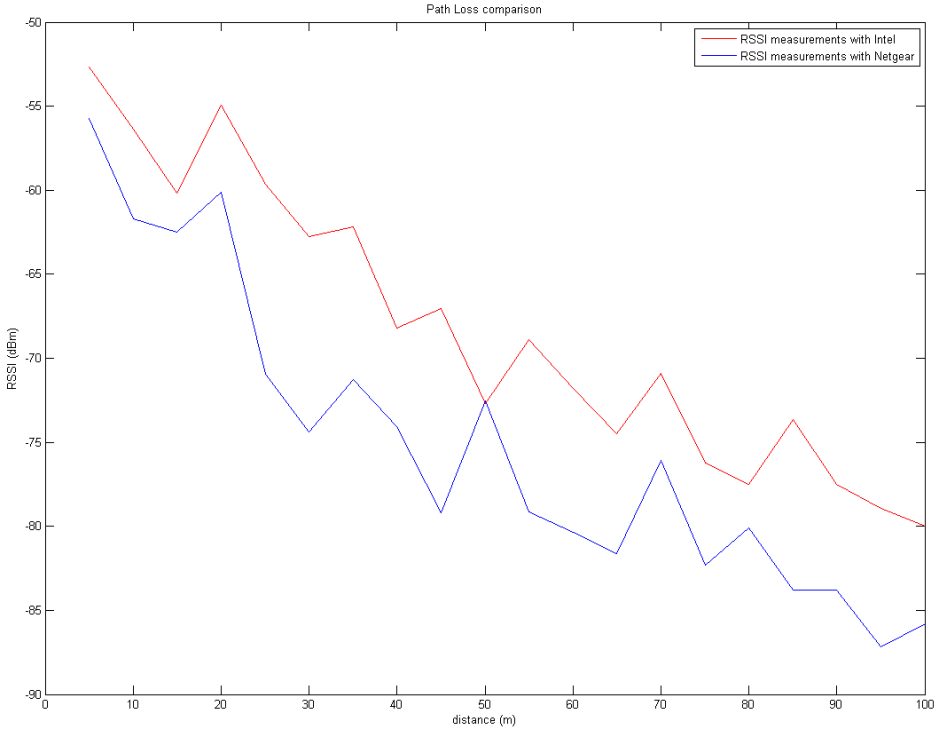

Figure 2.16: RSSI as a function of the distance with both NICs in direction 1

It is seen in Figure 2.16 that RSSI decreases as the distance between the AP and the mobile device increases (as expected). Furthermore, this effect can be appreciated in both NICs, thus resulting in more robust results. The path loss model that fits better and the ranging model are obtained in Section 2.6.

The next results are obtained for direction 2:

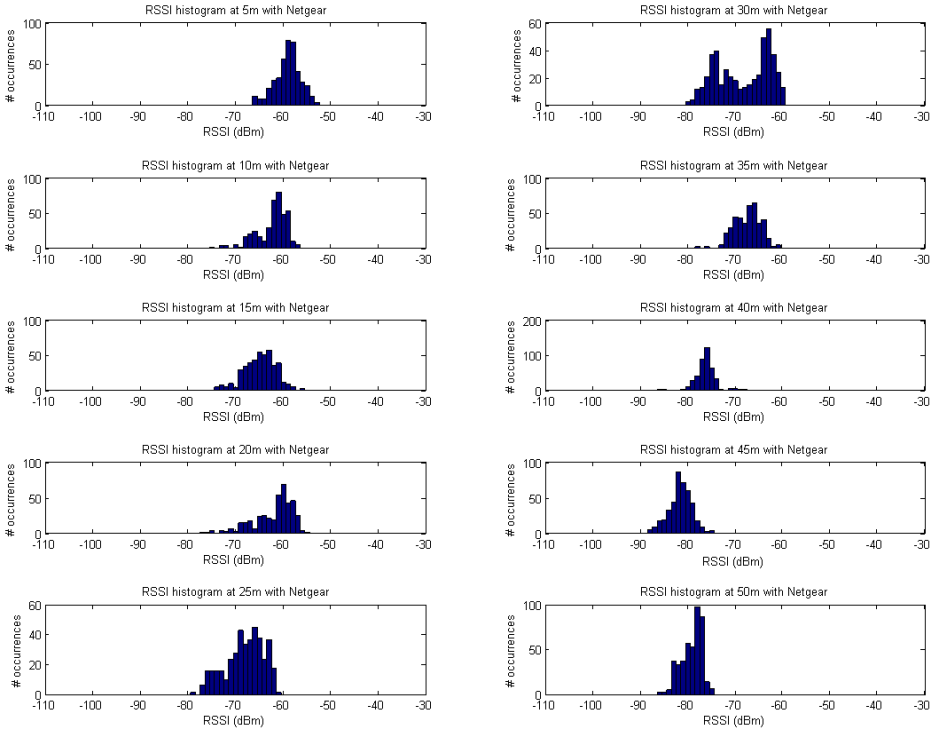

Figure 2.17: RSSI histograms with Netgear NIC in direction 2

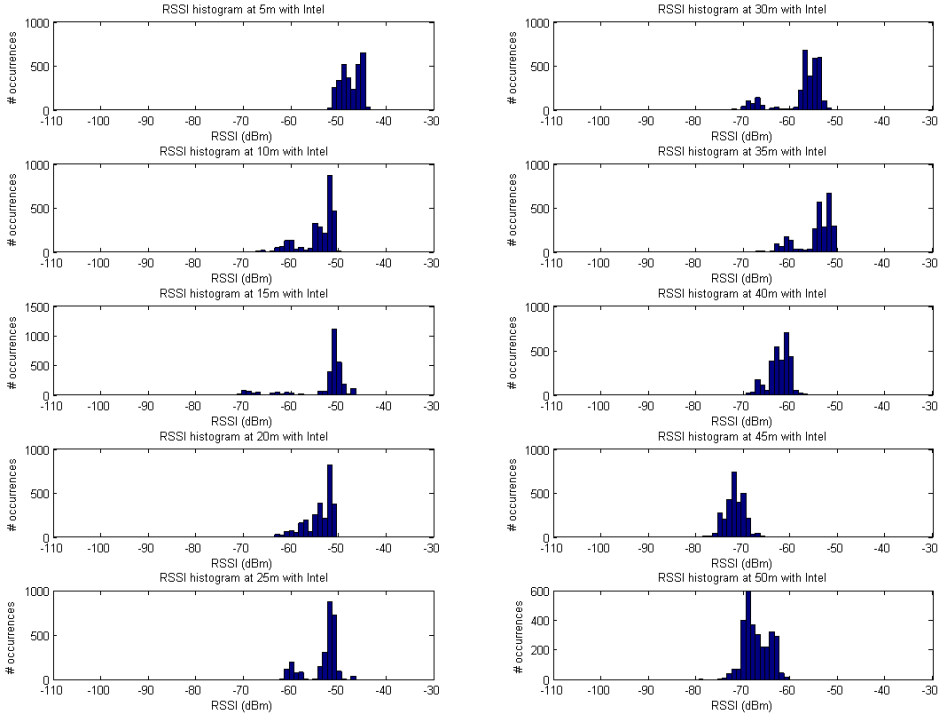

Figure 2.18: RSSI histograms with Intel NIC in direction 2

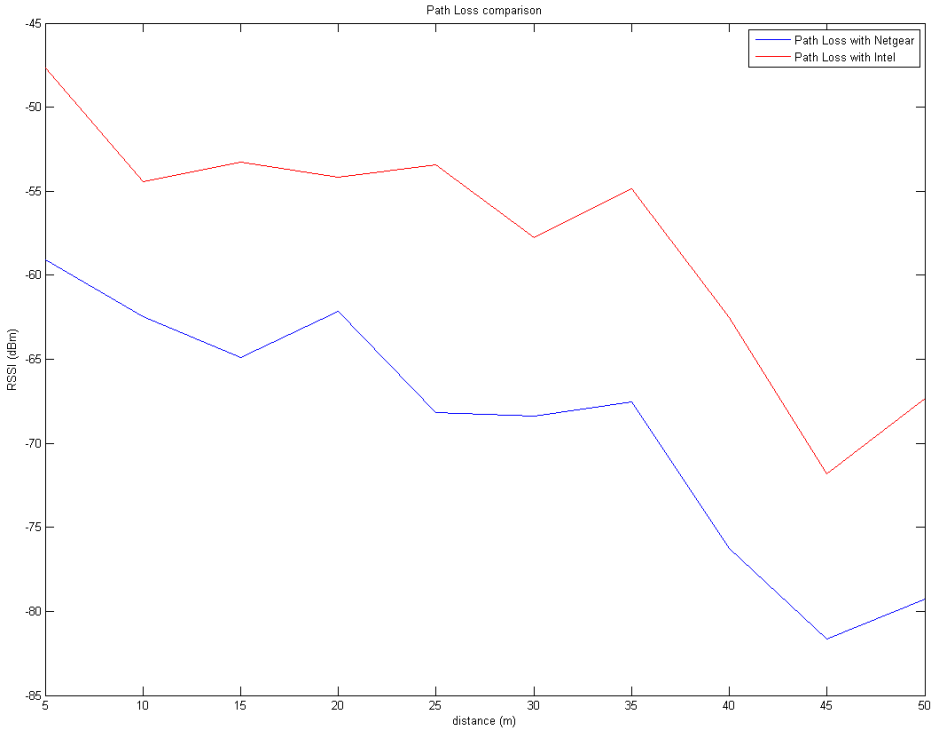

Figure 2.19: RSSI as a function of the distance with both NICs in direction 2

It is seen in Figure 2.19 (as in Figure 2.16) that RSSI decreases as the distance between the AP and the mobile device increases. The behaviour of both NICs is similar to the one commented above for direction 1. Therefore, the results obtained are robust. The propagation model that fits better and the ranging model are obtained in Section 2.6.

2.6 PROPAGATION MODEL AND RANGING MODEL

In this section, a propagation model is obtained. This model enables a relationship between received power and distance, therefore achieving a ranging model.

A propagation model that fits better the results obtained in Section 2.5 is searched. As it is said in Section 2.1, the equation used is $\mathrm{Pr} = P(0) - 10 * \alpha * \log(d)$. To characterize the channel, a single RSSI value is calculated in every location using the mode or the mean, taking into account all the performed measurements. The propagation constant α and the initial constant $P(0)$ are empirically determined. α depends on the environment and $P(0)$ depends on the NIC (for more details see [13]). A program is created that matches the measurements with a regression line and calculates α and $P(0)$. It also provides the quadratic error of the proposed model.

In direction 1, a propagation constant of 2.49 and an initial constant of -34.27 are computed employing the mode as RSSI statistical estimator. The error of that model is 146.58 (see Figure 2.20). With the mean, a propagation constant of 2.44 and an initial constant of -35.61 are computed. The error is 158.04 (see Figure 2.21). As the error with the mode is smaller than the one with the mean, it is decided to work with the mode.

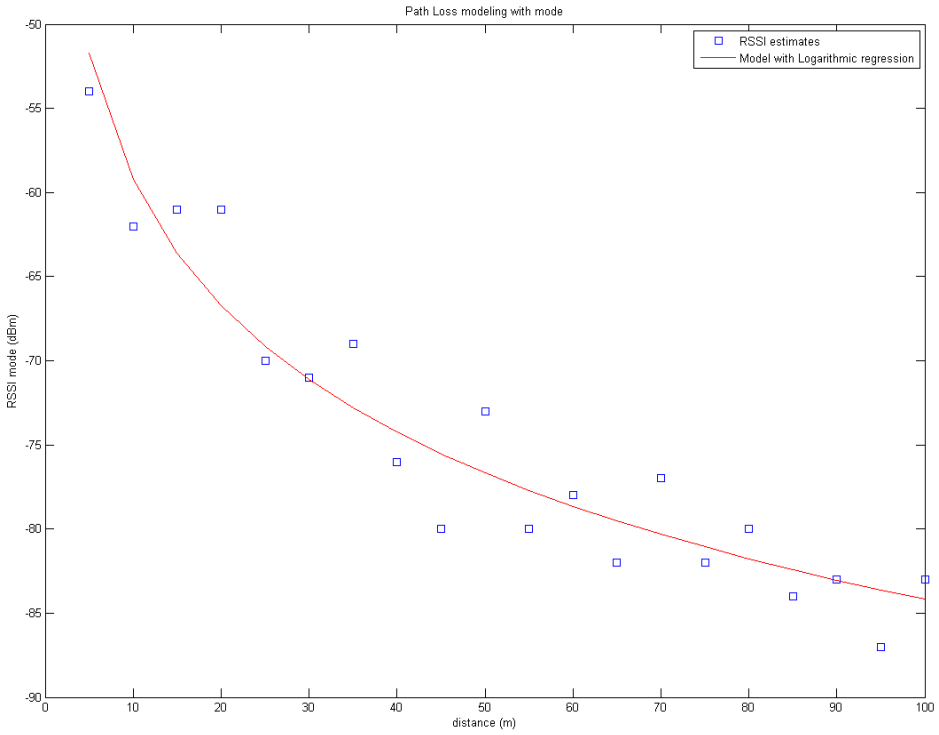

Figure 2.20: Propagation model with mode in direction 1

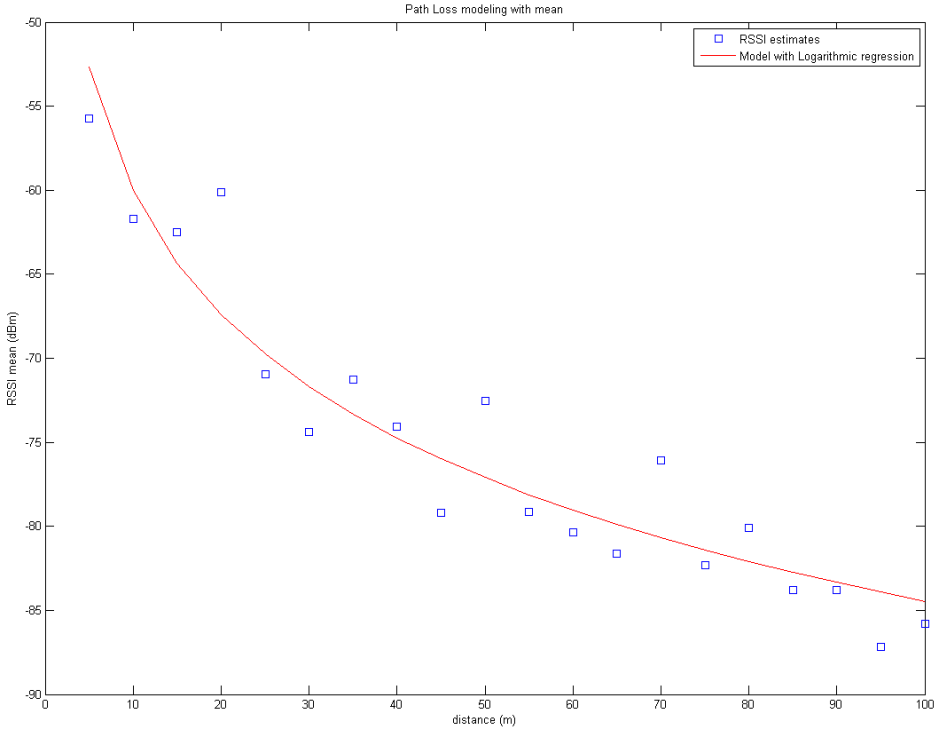

Figure 2.21: Propagation model with mean in direction 1

In direction 2, a propagation constant of 2.00 and an initial constant of -40.32 are computed with the mode. The error is 243.97 (see Figure 2.22). In this case, the error is much bigger than the one obtained with the mode in direction 1. It is seen in Figure 2.22 that the regression line does not match so well as in Figure 2.20. Therefore it is decided to set the propagation constant as 2.49 and the initial constant as -34.27.

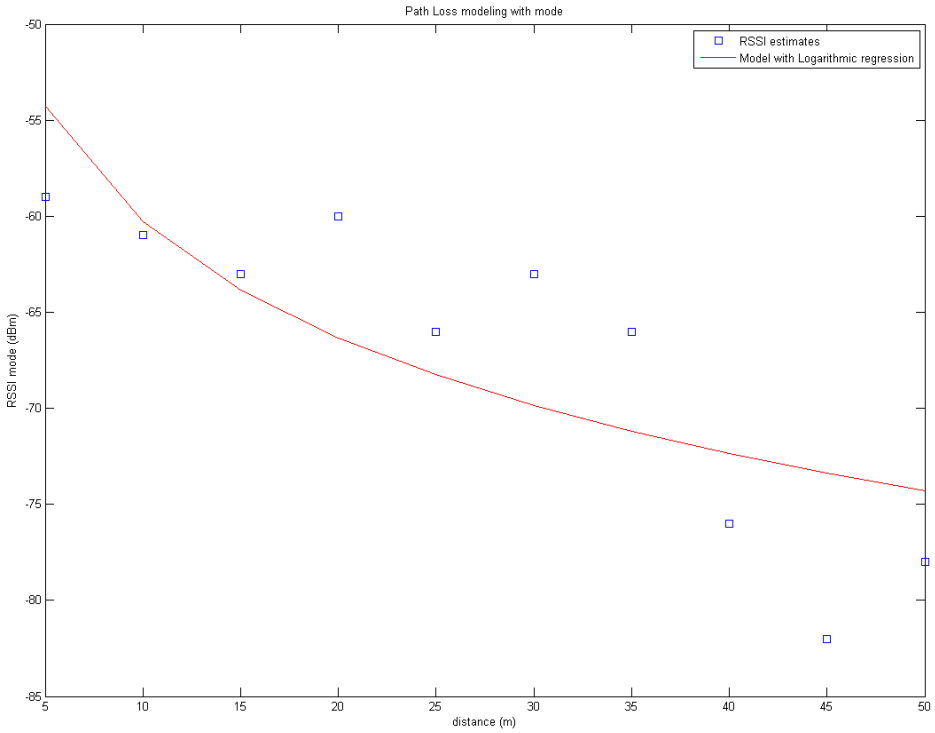

Figure 2.22: Propagation model with mode in direction 2

Once the propagation model is defined, more measurements are performed in order to obtain a ranging model. In each point, measurements are carried out and then divided into groups to calculate the statistics. An array of errors is extracted and thus mean and standard deviation are calculated. The idea is to model the error as a bias plus a random variable. The bias is the difference between an estimator's expectation and the true value of the parameter being estimated, and the dispersion is the variation of the samples in comparison with the mean. As it is seen in Figure 2.23, the regression line does not fit well with all the mean values calculated. Although it is seen in Figure 2.24 that standard deviation can be modelled as a linear expression, the overall error cannot be modelled as a bias plus a random variable. For that reason, it is decided to model the ranging error more generically. It is seen in Figure 2.23 that approximately 60% of the mean values are below 10m and it is seen in Figure 2.24 that 80% of the standard deviation values are below 10m. Considering this, the error is modelled as a normal random variable with a mean of 0m and a dispersion of 10m.

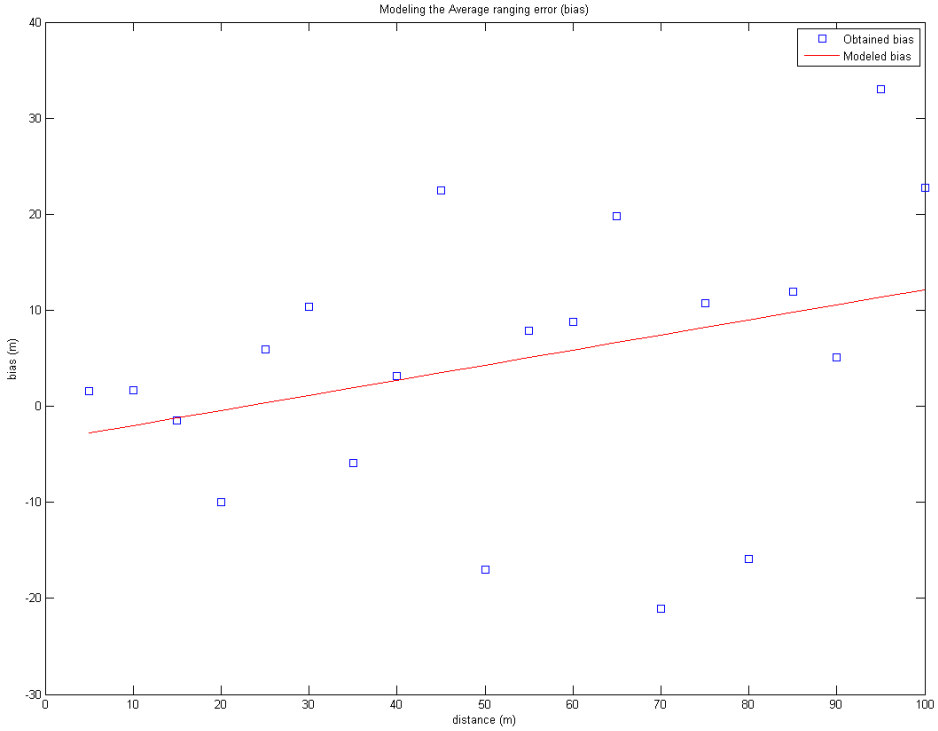

Figure 2.23: Bias of the ranging error

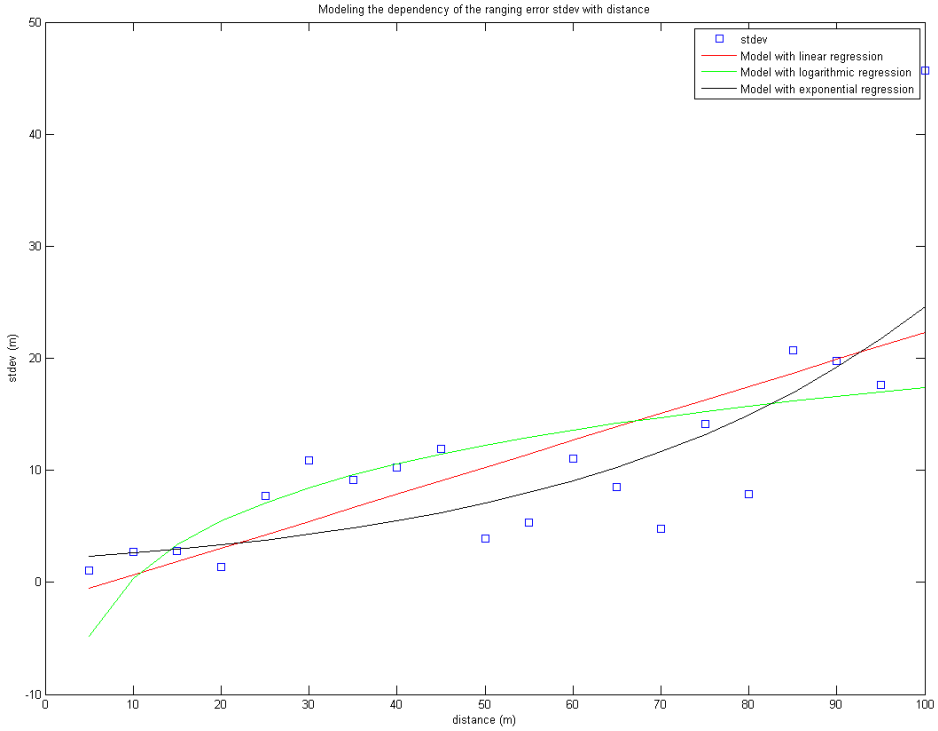

Figure 2.24: Standard deviation of the ranging error

3 HYBRID POSITIONING ALGORITHM

3.1 INTRODUCTION

The GPS observable equations are quadratic (see Section 1.4 for more details) and thus lineal properties cannot be applied. To solve the equations system there are several algorithms, which can be divided in two groups: iterative and non-iterative. The first ones compute the result starting with an initial value adjusted in each iteration until the solution does not improve significantly. The second ones, on the other hand, calculate the solution in only one iteration. Iterative algorithms offer better accuracy than non-iterative ones, but higher computational cost and therefore higher execution time. GPS receivers usually work with iterative algorithms due to its higher accuracy, and in 8 or 10 iterations they usually converge to a solution.

3.2 ALGORITHMS

As said in Section 1.4, each observation equation can be expressed as:

- $\rho_i = \sqrt{(x_i - x_u)^2 + (y_i - y_u)^2 + (z_i - z_u)^2} + ct_u$, where ρ_i represents the measured pseudorange corresponding to satellite i, x_i, y_i, z_i represent the coordinates from satellite i, and x_u, y_u, z_u, t_u represent user coordinates and receiver clock error.

To solve the system and obtain the user position and receiver clock error, at least four observation equations are needed. Many techniques exist to resolve the equations system: linearization, cost function minimization, etc. In the next sections they will be commented.

3.2.1 ITERATIVE ALGORITHMS

As said in Section 3.1, iterative algorithms need an initial value to calculate the result. Initial value plays a very important role in the process of computing the result, as it can change the result.

In OLS (Ordinary Least Squares) [8], linearization is applied to solve the nonlinear equations. If an approximate location of the receiver is known, the true position of the user and the receiver clock error can be expressed as an approximate position and clock error plus an offset:

- $x_u = \hat{x}_u + \Delta x_u$
- $y_u = \hat{y}_u + \Delta y_u$
- $z_u = \hat{z}_u + \Delta z_u$
- $t_u = \hat{t}_u + \Delta t_u$

If the equation in Section 3.2 can be expressed as a function of the unknowns x_u, y_u, z_u, t_u (i.e. $f(x_u, y_u, z_u, t_u)$), the same equation expressed as a function of the approximated parameters results in an approximate pseudorange:

- $\hat{\rho}_i = \sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2} + c\hat{t}_u = f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)$

Therefore, the pseudorange can be expressed as:

- $\rho_i = f(x_u, y_u, z_u, t_u) = f(\hat{x}_u + \Delta x_u, \hat{y}_u + \Delta y_u, \hat{z}_u + \Delta z_u, \hat{t}_u + \Delta t_u)$

This last expression can be expanded using a Taylor series:

- $f(\hat{x}_u + \Delta x_u, \hat{y}_u + \Delta y_u, \hat{z}_u + \Delta z_u, \hat{t}_u + \Delta t_u) = f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u) + \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{x}_u}\Delta x_u + \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{y}_u}\Delta y_u + \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{z}_u}\Delta z_u + \frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{t}_u}\Delta t_u + \cdots$

The expansion is truncated after the first-order partial derivatives in order to eliminate non-linear terms. The partial derivatives are:

- $\frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{x}_u} = -\frac{x_i - \hat{x}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$
- $\frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{y}_u} = -\frac{y_i - \hat{y}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$
- $\frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{z}_u} = -\frac{z_i - \hat{z}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$
- $\frac{\partial f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{t}_u)}{\partial \hat{t}_u} = c$

Substituting the partial derivatives into the pseudorange expression, it is obtained:

- $\rho_i = \hat{\rho}_i - \frac{x_i - \hat{x}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta x_u - \frac{y_i - \hat{y}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta y_u - \frac{z_i - \hat{z}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta z_u + c\Delta t_u$

Rearranging the last equation, the next expression is achieved:

- $\hat{\rho}_i - \rho_i = \frac{x_i - \hat{x}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta x_u + \frac{y_i - \hat{y}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta y_u + \frac{z_i - \hat{z}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}\Delta z_u - c\Delta t_u$

In order to simplify the notation of the last equation, the next variables are defined:

- $\Delta\rho_i = \hat{\rho}_i - \rho_i$
- $a_{xi} = \frac{x_i - \hat{x}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$
- $a_{yi} = \frac{y_i - \hat{y}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$

- $a_{zi} = \frac{z_i - \hat{z}_u}{\sqrt{(x_i - \hat{x}_u)^2 + (y_i - \hat{y}_u)^2 + (z_i - \hat{z}_u)^2}}$

Now, the last equation can be rewritten as:

- $\Delta\rho_i = a_{xi}\Delta x_u + a_{yi}\Delta y_u + a_{zi}\Delta z_u - c\Delta t_u$

As there are four unknowns, range measurements to at least four satellites are needed. The next set of linear equations has to be solved:

- $\Delta\rho_1 = a_{x1}\Delta x_u + a_{y1}\Delta y_u + a_{z1}\Delta z_u - c\Delta t_u$
- $\Delta\rho_2 = a_{x2}\Delta x_u + a_{y2}\Delta y_u + a_{z2}\Delta z_u - c\Delta t_u$
- $\Delta\rho_3 = a_{x3}\Delta x_u + a_{y3}\Delta y_u + a_{z3}\Delta z_u - c\Delta t_u$
- $\Delta\rho_4 = a_{x4}\Delta x_u + a_{y4}\Delta y_u + a_{z4}\Delta z_u - c\Delta t_u$

Expressed in a matrix form, it would be:

- $\begin{bmatrix}\Delta\rho_1\\ \Delta\rho_2\\ \Delta\rho_3\\ \Delta\rho_4\end{bmatrix} = \begin{bmatrix}a_{x1} & a_{y1} & a_{z1} & 1\\ a_{x2} & a_{y2} & a_{z2} & 1\\ a_{x3} & a_{y3} & a_{z3} & 1\\ a_{x4} & a_{y4} & a_{z4} & 1\end{bmatrix}\begin{bmatrix}\Delta x_u\\ \Delta y_u\\ \Delta z_u\\ \Delta t_u\end{bmatrix}$

Defining the next variables:

- $\Delta\boldsymbol{\rho} = \begin{bmatrix}\Delta\rho_1\\ \Delta\rho_2\\ \Delta\rho_3\\ \Delta\rho_4\end{bmatrix}$
- $\boldsymbol{J} = \begin{bmatrix}a_{x1} & a_{y1} & a_{z1} & 1\\ a_{x2} & a_{y2} & a_{z2} & 1\\ a_{x3} & a_{y3} & a_{z3} & 1\\ a_{x4} & a_{y4} & a_{z4} & 1\end{bmatrix}$
- $\Delta\boldsymbol{x} = \begin{bmatrix}\Delta x_u\\ \Delta y_u\\ \Delta z_u\\ \Delta t_u\end{bmatrix}$

The set of linear equations can be expressed as:

- $\Delta\boldsymbol{\rho} = \boldsymbol{J}\Delta\boldsymbol{x}$

In the case of having measurements to four satellites only, the solution is:

- $\Delta\boldsymbol{x} = \boldsymbol{J}^{-1}\Delta\boldsymbol{\rho}$

When more than four ranges are measured, the inverse matrix cannot be calculated, and thus the pseudoinverse is used:

- $\Delta\boldsymbol{x} = (\boldsymbol{J}^{\top}\boldsymbol{J})^{-1}\boldsymbol{J}^{\top}\Delta\boldsymbol{\rho}$

Once the unknowns (i.e. $\Delta x_u, \Delta y_u, \Delta z_u, \Delta t_u$) are computed, the algorithm checks the values of the unknowns. If they are small enough (i.e. the estimated position is a good approximation of the real position), it stops. If not, a new iteration is performed in which the estimated pseudorange is replaced by the previously calculated pseudorange. Finally, the user position and clock offset (i.e. x_u, y_u, z_u, t_u) are calculated.

In GNA (Gauss-Newton Algorithm) [14], a cost function is defined consisting on a sum of squared functions:

- $C(\boldsymbol{x}) = \sum_{i=1}^{m} {r_i}^2(\boldsymbol{x})$, where $\boldsymbol{x}$ represents the unknowns and r_i the functions to minimize

The goal of the algorithm is to minimize the cost function. To achieve it, it starts with an initial value ($\boldsymbol{x}^{(0)}$) that is updated in each iteration:

- $\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \Delta \boldsymbol{x}$

Where $\Delta \boldsymbol{x}$ is called the step and is calculated as:

- $\Delta \boldsymbol{x} = (\boldsymbol{J}^{\top}\boldsymbol{J})^{-1}\boldsymbol{J}^{\top}\boldsymbol{r}$, where $\boldsymbol{r}$ is the vector of functions and $\boldsymbol{J}$ is the Jacobian matrix

In the particular case of GPS, the functions to minimize are the pseudorange differences (i.e. $\Delta \boldsymbol{\rho}$), arriving at the same expression as in OLS.

In SD (Steepest Descent) [14], steps proportional to the negative of the gradient[3] are taken to find the minimum:

- $\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} - \lambda_s \nabla F(\boldsymbol{x})$, where λ_s is a positive scalar and F is the function to minimize.

This algorithm is good at the first steps because it rapidly converges to the point, but once it is near the point, the steps are very slow.

In LMA (Levenberg-Marquardt Algorithm) [14], an interpolation between GNA and SD is done. The step is defined as:

- $\Delta \boldsymbol{x} = (\boldsymbol{J}^{\top}\boldsymbol{J} + \lambda_s \boldsymbol{I})^{-1}\boldsymbol{J}^{\top}\boldsymbol{r}$

Adjusting the damping parameter (i.e. λ_s), the behaviour of the LMA changes. If λ_s is large, the algorithm behaves as SD, while if $\lambda_s$is small, it behaves as GNA. λ_s can be adjusted in each iteration, making the algorithm converge rapidly when it is far from the minimum (like SD) and avoiding jumps when it is near the minimum (like GNA).

[3] In the case of more than one function to minimize, the algorithm calculates the Jacobian instead of the gradient.

3.2.2 NON-ITERATIVE ALGORITHMS

As said in Section 3.1, non-iterative algorithms calculate the solution in only one iteration. To make it possible, they linearize first and then they calculate the result.

In Linear Least Squares, all the reference points (except one) are expressed as a function of one of them (the selected one). The error of the receiver clock is assumed to be corrected and therefore the user's position unknowns (x_u, y_u, z_u) are the only ones to be computed. The algorithm only works well if the measured pseudoranges are corrected (i.e. the receiver clock error is corrected). Only in this circumstance, lineal properties are well applied. More information can be found in [15].

In [16], Bancroft proposes an algebraic solution to the GPS equations. The algorithm calculates two candidate solutions and after, it distinguishes the correct one substituting back into the equations. Only one of them agrees.

In [17], a non-iterative algorithm hybridising GPS pseudoranges and terrestrial ranges is proposed. The algorithm distinguishes between the case of measuring GPS pseudoranges and terrestrial ranges, and it assigns a different equation depending on each case.

As said in Section 3.1, non-iterative algorithms are able to calculate a position with less computational cost, but the accuracy is not as good as the one obtained with iterative algorithms.

3.3 POSITION AMBIGUITY

In a standalone GPS system, at least four satellites are needed to solve the equations system. Each observation equation represents a sphere of points whose centre is the satellite and whose radius is the measured pseudorange. In most cases, the spheres intersect in a unique point, which corresponds to the target's position. Only in the particular case of coplanar GPS satellites the spheres intersect in two points, one located at the Earth's surface (the target's position) and the other at thousands of kilometres from the surface of the Earth (the mirror-symmetric solution [18]), which is geographically inconsistent. Nevertheless, working with an iterative algorithm that uses the centre of the Earth as the initial position guarantees the convergence to the solution located at the Earth's surface.

In the case of a hybrid system, some reference points are on the Earth's surface and their ranges are drastically smaller than the GPS ones. This leads to a completely different geometrical arrangement in which the spheres can intersect in two close points near to the Earth's surface, thus arising a positioning ambiguity difficult to resolve because both are geographically valid. In the particular situation of three GPS pseudoranges and one WLAN range, which is a very common scenario given public WLAN deployments, this phenomenon is essentially always observed. The key fact is that one of these two candidate solutions (the one corresponding to the true target's position) is usually much more accurate than the other, and therefore identifying the best one becomes crucial. This time, working with an iterative algorithm does not guarantee the convergence to the solution

corresponding to the true target's position[4]. In Figure 3.1 a scan of initial positions is performed, and the line dividing initial positions that make the algorithm converge to one solution or the other is represented. Initial positions located on the left of the boundary line leads the algorithm to the green solution, while initial positions on the right leads to blue solution. It can be observed that the estimated position depends entirely on the initial position.

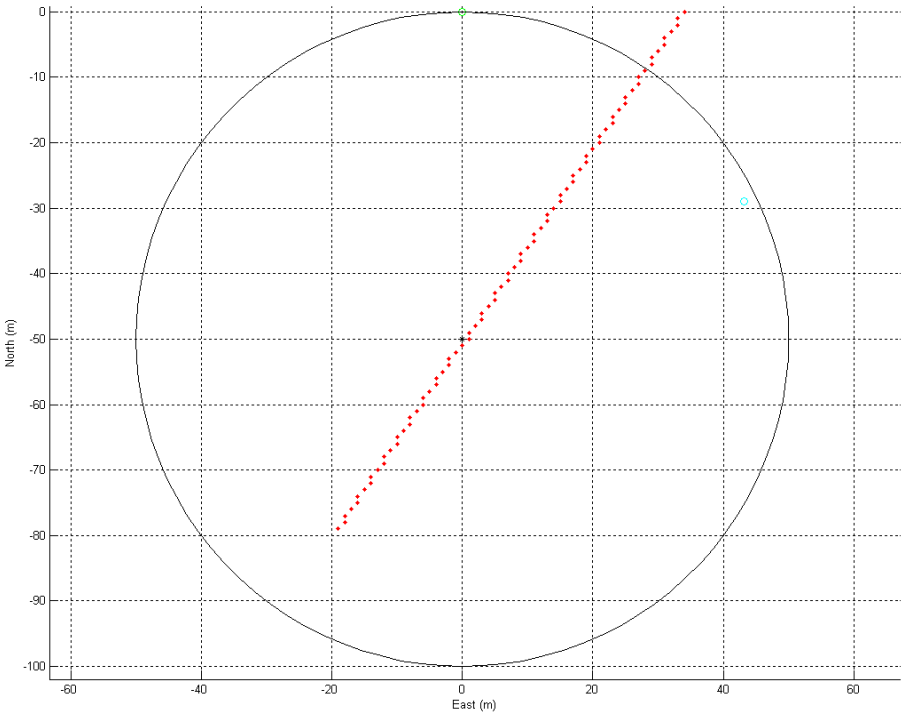

Figure 3.1: Scan of initial positions

If a three-dimension scan is performed, it is seen that the line is in fact a surface, which divides the space of initial positions in two regions. The initial positions located at the left of the surface lead the algorithm to the green solution, while the ones at the right lead to the blue solution. In Figure 3.2, this effect can be appreciated in more detail. The key fact is that points of the region containing the correct position lead to the correct solution, but it is not possible to know a priori whether a starting point belongs to that region.

[4] From now on it will be considered that correct solution corresponds to the true target's position.

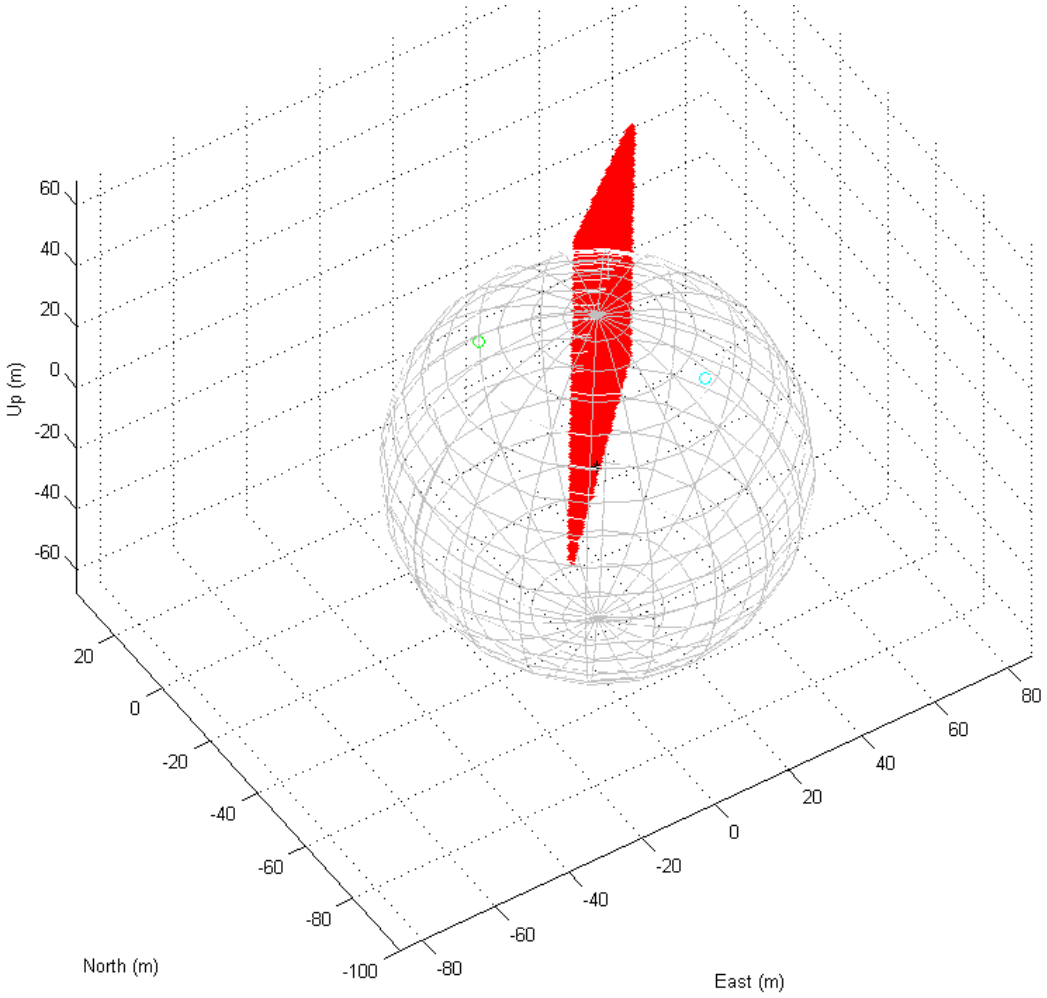

Figure 3.2: 3D Scan of initial positions

3.4 EVALUATED ALTERNATIVES TO SOLVE THE AMBIGUITY

Many tests have been done to try to solve the problem of the position ambiguity commented in Section 3.3. In the next lines the most representative steps will be mentioned.

The first test is trying different iterative algorithms. The algorithm implemented by default is Ordinary Least Squares. Gauss-Newton is tried, but no differences have been found (as it is commented in Section 3.2.1, GNA behaves as OLS when working with GPS). Levenberg-Marquardt is tried, but it is very unstable. There are always two possible solutions, but the configuration parameters (basically the damping parameter) determine the convergence to the “good” or “bad” solution (i.e. for the same epoch, one solution or the other is achieved depending on the value of the damping parameter). Steepest Descent has been tried too but, as Levenberg-Marquardt, depends entirely on the configuration parameters (i.e. the step size).

The second test is trying non-iterative algorithms. Linear Least Squares provides an estimated position far away from the real one because it considers the same clock error in all the equations[5]. Therefore, an error of the order of 10^5 is summed in the terrestrial station equation. If the result of Linear Least Squares is used as initial position for Least Squares, one of both possible solutions is achieved (sometimes the "good" one and others the "bad" one). Bancroft provides two solutions very far from each other. None is good as estimated position because Bancroft considers too that the clock error is the same for each equation. However, if both solutions are used as initial positions for Least Squares, one leads to the "good" position and the other to the "bad" position. Hence, the position ambiguity can be detected executing the Bancroft algorithm. In [17] a non-iterative algebraic algorithm is proposed similar to Bancroft's one, but in this case the clock error only affects some of the equations. It provides two solutions, but none is good as an estimated position. If they are used as initial positions of Least Squares, two positions are not always obtained. For that reason, it cannot be used to detect the position ambiguity.

The third test is taking profit of the residuals to discriminate the correct solution. The idea behind comes from Bancroft's algorithm, where the residuals are used to distinguish between the correct solution and the non-sense solution. In the case of hybridisation, both residuals are too small to distinguish between the "good" and the "bad" solution because both are geographically correct (for more details see Section 3.3).

The fourth test consists on expressing the two possible estimated positions referenced to the terrestrial station and taking the one that has the most similar height. This would be a good method only if it were known that the user usually has the same height as the reference point, but that is not the case.

After performing all this tests without achieving robust results, it is decided to work with a set of epochs in order to have more information. The objective is to study the behaviour of both candidate solutions over time and try to draw conclusions.

The first idea consists on comparing the height of both solutions. In some cases the height of the "good" solution maintains practically constant, whereas the "bad" solution varies. In others, both heights keep practically constant, i.e. the difference between them is not enough to decide which the "good" one is.

Developing a little bit more the last idea, it is concluded that the "good" solution (which corresponds to the actual position of the user) should vary less than the "bad" solution (which is geographically correct, but in practice does not indicate the actual position of the user). Both solutions are on the surface of the sphere that draws the terrestrial AP, but the "bad" solution is more influenced by the movement of the GPS satellites than the "good" solution. As the GPS pseudorange measurements take place at user's actual position, the "good" solution should not be influenced by the movement of the satellites (i.e. theoretically should not move), while the "bad"

[5] When incorporating equations corresponding to terrestrial measurements, it is assumed that receiver clock error is zero, because terrestrial network is not synchronized with GPS network.

solution should move[6]. In practice, the difference between both solutions cannot be appreciated in a clear way because real measurements have an associated error. Nevertheless, if pseudorange measurement errors are small enough and do not mask the effect mentioned above, it can be seen that "bad" solution moves more than "good" solution[7].

In order to decide which of the two candidate solutions is correct, the *slope method* is proposed: information is collected along four minutes and the evolution of each solution respect to its initial position is compared. The positions computed in the first epoch are used as the initial position of solution 1 and the initial position of solution 2. For the next epochs, the estimated positions obtained for each solution are compared to their respective initial positions. The slope of the errors for each solution is calculated and the lower one in absolute value determines the correct position.

Different tests have been performed in order to check the correct working of the *slope method*. The results can be seen in Section 4.3.

3.5 PROPOSED METHOD

As it is commented in Section 3.3, the estimated position depends entirely on the initial position when hybridising. Therefore, the proposed hybrid positioning system will distinguish between two cases. The former one when the system knows a reliable initial position near the user, and the latter when the system has no information about the initial position.

3.5.1 POSITION ESTIMATION KNOWING A RELIABLE INITIAL POSITION

When the system knows a reliable initial position near the user's actual position (e.g. a position calculated with four or more GPS satellites), Least Squares algorithm will be used to calculate the next position. The previous position (the one that is reliable) will be used as initial position and thus the algorithm will converge to the "good" solution[8].

3.5.2 POSITION ESTIMATION WITHOUT KNOWING AN INITIAL POSITION

When information about initial position is not available, the *slope method* (explained in Section 3.4) will be used to obtain the "good" solution. This position will be considered as reliable and therefore the proceedings of Section 3.5.1 will be applied (i.e. the position will be used as initial position of Least Squares in the next position calculation). The next flowchart shows the proceedings of the proposed method:

[6] It is assumed that the target is a static user.
[7] This only happens when the algorithm converges and for the performed testcases.
[8] As said in Section 3.3, points of the region containing the correct position lead to the correct solution.

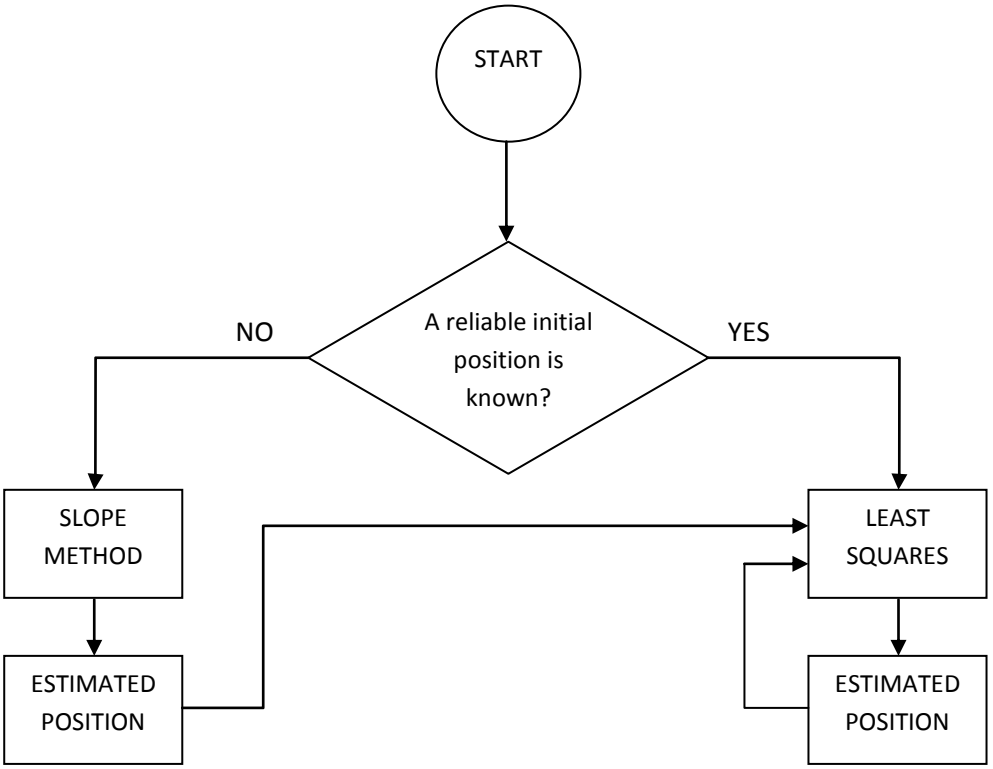

Figure 3.3: Flowchart of the proposed method

4 EVALUATION OF THE PROPOSED METHOD

4.1 TESTBED

In order to evaluate the proposed method, simulations are carried out with *matlab* calculation program. Tests are performed with WLAN estimation distances and real GPS data. The first ones are emulated with ranging models obtained from real WiFi measurements and the second ones are obtained by the antenna situated at the terrace of the CUBIC building. GPS measurements are performed during a whole day, thus obtaining a sizeable RINEX observation file. The size of the file slows down the data processing and therefore it is decided to truncate the file on 2681 epochs, thus reducing the size.

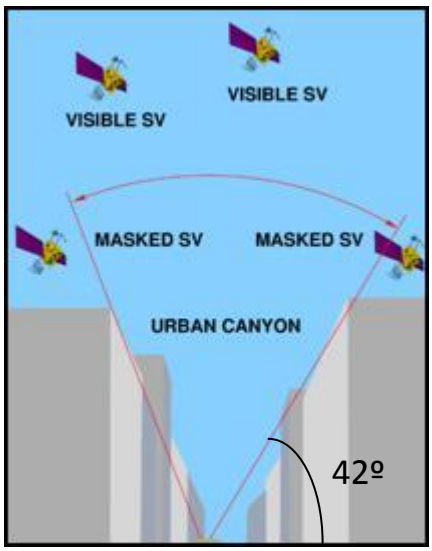

Figure 4.1: Urban canyon simulation

GPS measurements take place in an almost ideal scenario since the CUBIC building is located in an open area, without high buildings around. In these conditions, the minimum number of visible satellites is 8. To simulate an urban canyon scenario, it is decided to select only the satellites whose angle is above 42º (see Figure 4.1). Thereby, two intervals of epochs are obtained in which there are only three visible satellites:

- From 1617 to 2019 (Interval 1)
- From 2361 to 2681 (Interval 2)

These two intervals are used to incorporate the WLAN AP and study the behaviour of the proposed method. In order to facilitate the tests, a different name is assigned to each AP location:

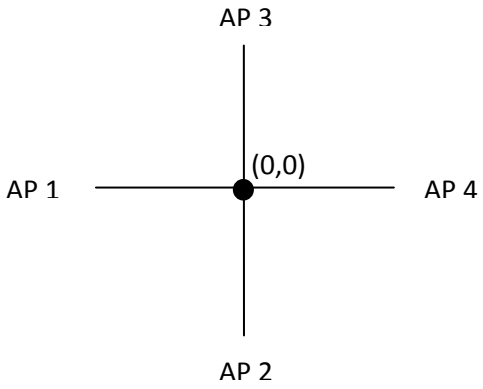

Figure 4.2: AP name assignation depending on the location

In order to know the geometrical arrangement, a skyplot has been performed for each interval. The skyplot allows a view of the reference points position referred to the user's position. The circular lines represent the angle of the reference point. The bigger circumference represents 0º and the point in the centre represents 90º (i.e. just above the user). Number 33 corresponds to the AP (in this case the AP1), and the rest of numbers refer to satellite numbers.

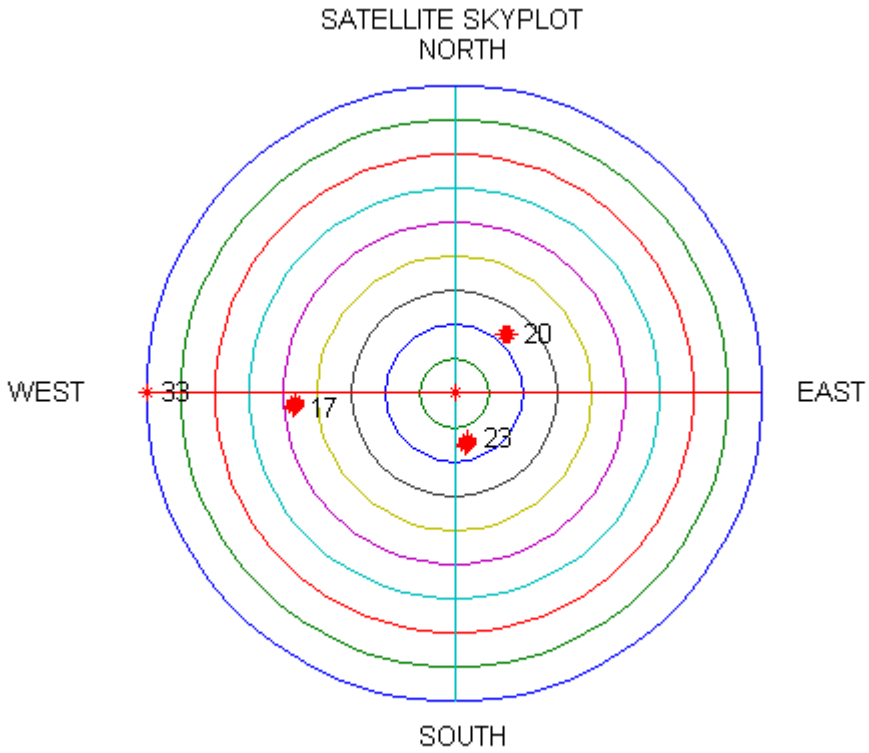

Figure 4.3: Skyplot of interval 1

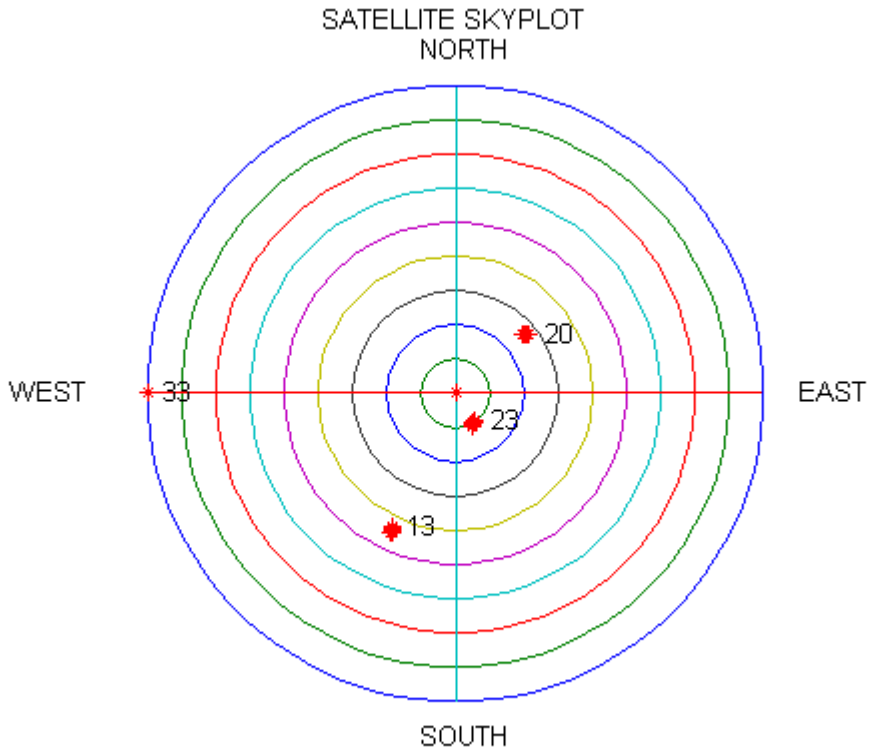

Figure 4.4: Skyplot of interval 2

For each epoch interval, the method will be tested as a function of the AP location and the terrestrial range (which corresponds to the distance between the AP and the target device). As WiFi has an outdoor coverage of 150m approx., it is decided to carry out tests at 20m, 50m and 100m.

4.2 CALCULATION OF THE DURATION OF THE *SLOPE METHOD*

A key point of the proposed method is the duration of the *slope method* because it is the time that the user has to wait to know its location. Usually, all the systems define a TTFF (Time to First Fix), which corresponds to the time needed to initialize and set all the parameters to work properly. In the case of the proposed method, this only occurs in the particular situation of starting the system and having only three GPS satellites and one terrestrial AP available, without knowing a reliable initial position. To set the needed time, the method is tested with neither GPS nor terrestrial errors for different durations. The results are shown in the next table:

Epoch	Terrestrial Range (m)	AP	4 min	3 min	2 min
1700	20	1	OK	OK	OK

to 1940		2	OK	OK	X
		3	OK	OK	OK
		4	Doesn't work (algorithm doesn't converge)	Doesn't work (algorithm doesn't converge)	Doesn't work (algorithm doesn't converge)
	50	1	OK	OK	X
		2	OK	OK	OK
		3	OK	OK	X
		4	OK	X	X
2400 to 2640	20	1	OK	OK	OK
		2	OK	OK	OK
		3	OK	OK	X
		4	OK	OK	OK
	50	1	OK	OK	OK
		2	OK	OK	X
		3	OK	OK	OK
		4	OK	OK	OK

Table 4.1: Test of the slope method depending on the duration

It is seen that the *slope method* does not work well in the case of 2 minutes. However, for the case of 3 minutes, it works well in all the tests except one. For the case of 4 minutes, it always works. Therefore, the case of 2 minutes is discarded.

In order to conclude the duration, the same tests are performed for the cases of 3 and 4 minutes, but this time with terrestrial range errors. The errors are modelled with a bias and dispersion. As said in Section 2.6, the bias is the difference between an estimator's expectation and the true value of the parameter being estimated, and the dispersion is the variation of the samples in comparison with the mean. 50 executions of the *slope method* are carried out in each test, obtaining the percentage of success:

Epoch	Terrestrial range (m)	AP	Success percentage (Bias = 0 / Disp. = 1)		Success percentage (Bias = 0 / Disp. = 10)	
			4 min	3 min	4 min	3 min
1700 to 1940	20	1	74 %	42 %	**42 %**	**46 %**
		2	52 %	52 %	60 %	48 %
		3	84 %	64 %	50 %	48 %
		4	60 %	56 %	64 %	60 %
	50	1	78 %	68 %	64 %	34 %
		2	**56 %**	**58 %**	**68 %**	**76 %**
		3	92 %	84 %	50 %	38 %
		4	70 %	64 %	70 %	68 %
2400 to 2640	20	1	100 %	100 %	**68 %**	**72 %**
		2	76 %	48 %	48 %	48 %
		3	**80 %**	**86 %**	58 %	54 %
		4	**60 %**	**74 %**	**54 %**	**60 %**
	50	1	100 %	100 %	90 %	76 %
		2	100 %	80 %	**54 %**	**62 %**
		3	100 %	98 %	72 %	62 %
		4	100 %	100 %	70 %	52 %

Table 4.2: Test of the slope method with terrestrial errors depending on the duration

In the case of having small errors (bias=0 and disp.=1), it can be appreciated that only in three cases the percentage of success is larger for the case of 3 minutes instead of 4 minutes (marked in bold). In the case of having large errors (bias=0 and disp.=10), it can be seen that the number of cases where the percentage of success is larger in 3 minutes instead of 4 minutes grows up to five (marked in bold too). Nevertheless, the percentage of success is still larger in general for the case of 4 minutes. Therefore, the duration of the *slope method* is fixed to 4 minutes.

4.3 EVALUATION OF THE *SLOPE METHOD*

Once the duration of the *slope method* is fixed, the main *slope method* can be tested. First, it will be tested with neither terrestrial nor GPS errors[9]. Then, it will be assessed with terrestrial errors only. After, only GPS errors will be applied. Finally, it will be evaluated with GPS and terrestrial errors. Thereby, the influence of each error will be studied in the behaviour of the method.

Random Gaussian variables are generated with *randn()* matlab function in order to simulate the measurement errors. A sample of the error random variable is generated in each epoch. As it is different for every epoch, 100 executions of the *slope method* are performed for each case, thus obtaining a robust success percentage.

For the case of terrestrial range errors, two types of errors are distinguished: the first one, considering a small error (bias=0 and disp.=1), and the second one, considering a large error (bias=0 and disp.=10). A better accuracy is obtained with the first one, because if the dispersion is 1m, it is assured that 95.44% of the errors are below 2m. This could correspond to a TOA based system utilizing technologies like WiFi or UWB. With the second one, the accuracy is not so good. As the dispersion is 10m, it is assured that 95.44% of the errors are below 20m. This could correspond to a system based on power measurements, as the one proposed in Section 2.6. TOA based systems are always more expensive than power measurement based systems and thus depending on the considered accuracy-cost trade-off one or the other type of method could be adopted in practice.

For the case of GPS pseudorange errors, a multipath error corresponding to an urban canyon is simulated. It is composed of a bias and dispersion. For each satellite, a different bias and a Gaussian random variable corresponding to 10% of the bias are assigned. As the satellites taken in account have an angle above 42º, it is considered that the pseudorange errors will not overpass 10m [19]. The next biases are assigned to the involved satellites.

- Interval 1:
  - Sat23: 5m
  - Sat17: 10m
  - Sat20: 7m
- Interval 2:
  - Sat13: 9m
  - Sat23: 5m
  - Sat20: 7m

In order to perform the tests and present the results, a *matlab* test program has been created able to configure the type of range error and pseudorange error, calculate the success of the slope method and present the results with different graphics. The code of the program can be seen in Appendix 1.

[9] The error in the GPS measurements performed at CUBIC building is considered negligible.

4.3.1 GPS AND TERRESTRIAL SIGNALS WITHOUT ERROR

Epoch	Terrestrial Range (m)	AP	Success
1700 to 1940	20	1	OK
		2	OK
		3	OK
		4	Doesn't work (algorithm doesn't converge)
	50	1	OK
		2	OK
		3	OK
		4	OK
	100	1	OK
		2	OK
		3	OK
		4	OK
2400 to 2640	20	1	OK
		2	OK
		3	OK
		4	OK
	50	1	OK
		2	OK
		3	OK
		4	OK

	100	1	OK
		2	OK
		3	OK
		4	OK

Table 4.3: Test of the slope method without errors

As it is seen in Section 4.2, the *slope method* always works well if no errors are considered. There is only one case in which the method does not work because the algorithm does not converge.

4.3.2 GPS SIGNAL WITHOUT ERROR AND TERRESTRIAL SIGNAL WITH ERROR

Epoch	Terrestrial range (m)	AP	Success percentage (Bias = 0 / Disp = 1)	Success percentage (Bias = 0 / Disp = 10)
1700 to 1940	20	1	67 %	46 %
		2	46 %	59 %
		3	81 %	37 %
		4	64 %	62 %
	50	1	82 %	41 %
		2	62 %	69 %
		3	84 %	39 %
		4	71 %	67 %
	100	1	97 %	38 %
		2	83 %	67 %
		3	90 %	43 %
		4	83 %	75 %
2400	20	1	100 %	68 %

to 2640		2	68 %	54 %
		3	87 %	61 %
		4	64 %	53 %
	50	1	100 %	88 %
		2	99 %	60 %
		3	100 %	74 %
		4	100 %	70 %
	100	1	100 %	99 %
		2	100 %	79 %
		3	100 %	85 %
		4	100 %	98 %

Table 4.4: Test of the slope method with only terrestrial error

It is seen in Table 4.4 that the behaviour of the *slope method* depends on the location of the AP. In the case of having small errors, the success percentage of the AP1 (located at the west of the user) is always better or equal than the one of the AP2 (located at the south). In Figures 4.5 and 4.6, the two possible positioning solutions for the interval 1 when the range is 50m are represented. One solution is printed on blue and the other one on green. The red star represents the actual user position and the sphere represents the terrestrial range without error. Figure 4.5 corresponds to the AP1, where an 82% of success is achieved, and Figure 4.6 corresponds to the AP2, where a 68% is achieved. It is seen that the *slope method* works better when both candidate solutions are more separated. This effect is observed too when using the same AP location and varying the range. In Figures 4.7, 4.8 and 4.9, the two candidate solutions for the interval 2 when using AP2 are represented. It can be seen that separation increases while the range increases, and that is reflected in the percentage of success. When the range is 20m, the percentage is 68%. When it is 50m, the percentage is 99%. When it is 100m, the percentage is 100%. The larger the separation is, the larger the success percentage. This is a great feature of the slope method, because when the probability of failure is high, the positioning error is small.

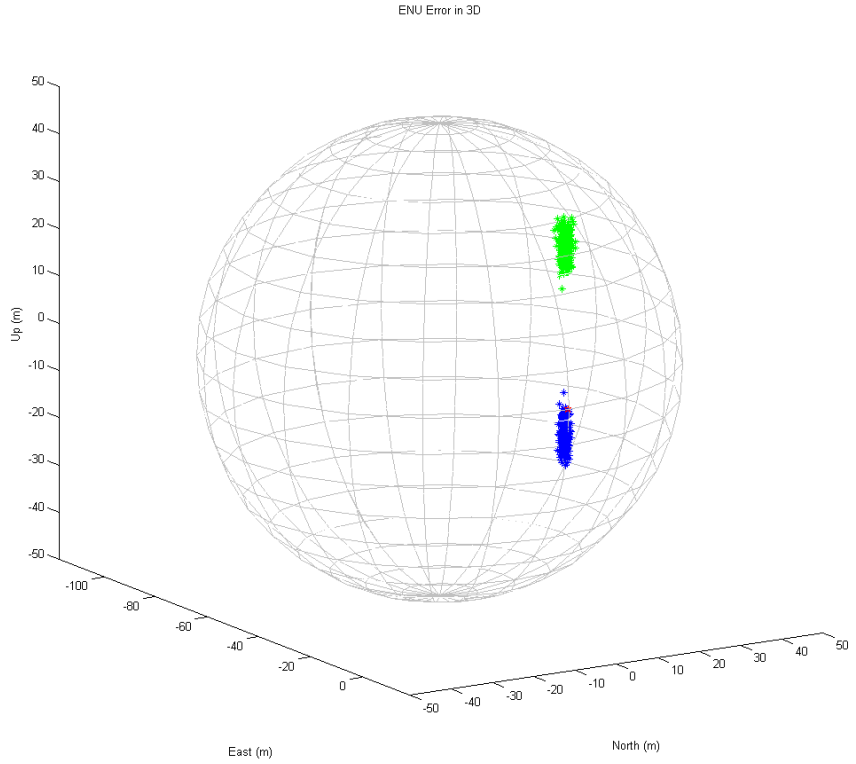

Figure 4.5: Candidate solutions for interval 1 and AP1 with 50m range and small error

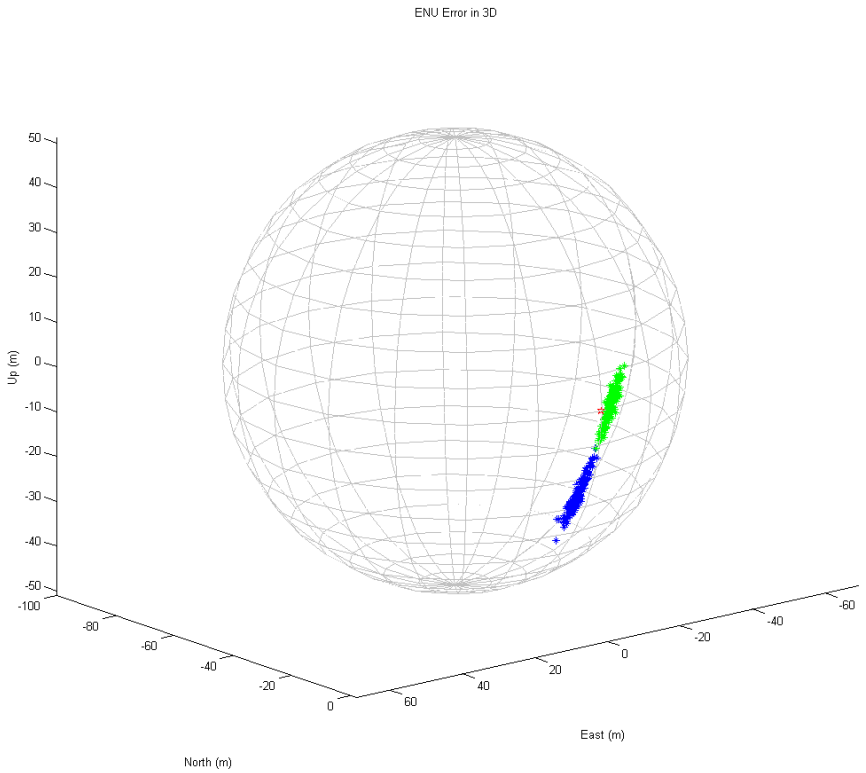

Figure 4.6: Candidate solutions for interval 1 and AP1 with 50m range and small error

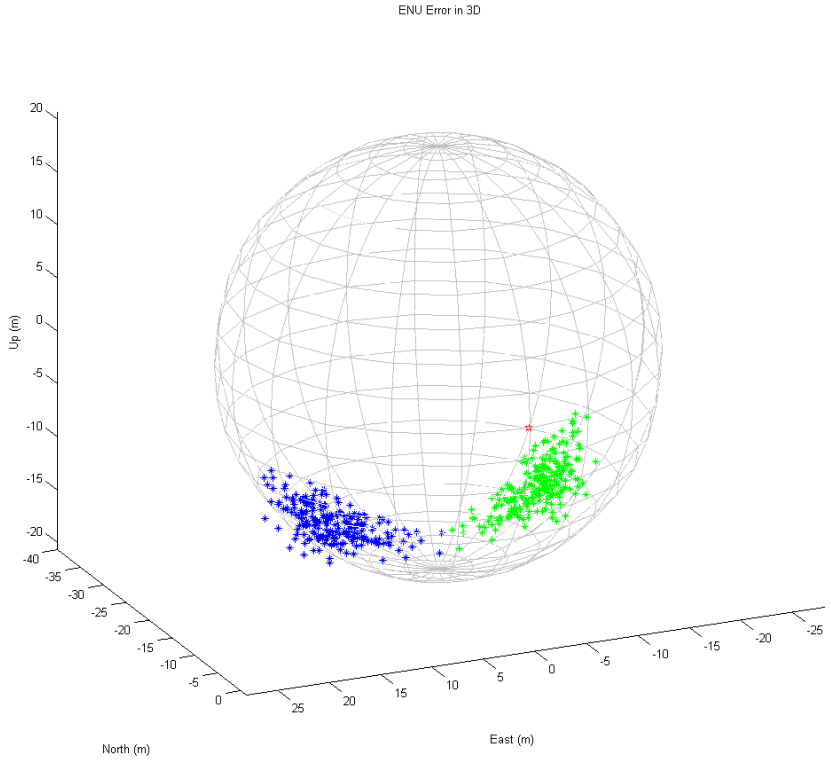

Figure 4.7: Candidate solutions for interval 2 and AP2 with 20m range and small error

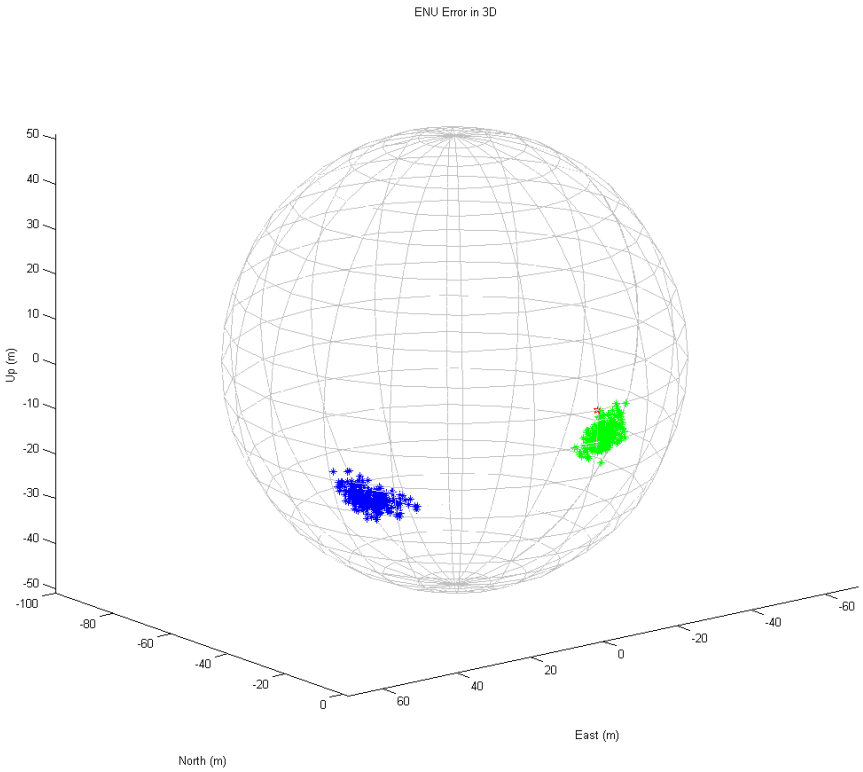

Figure 4.8: Candidate solutions for interval 2 and AP2 with 50m range and small error

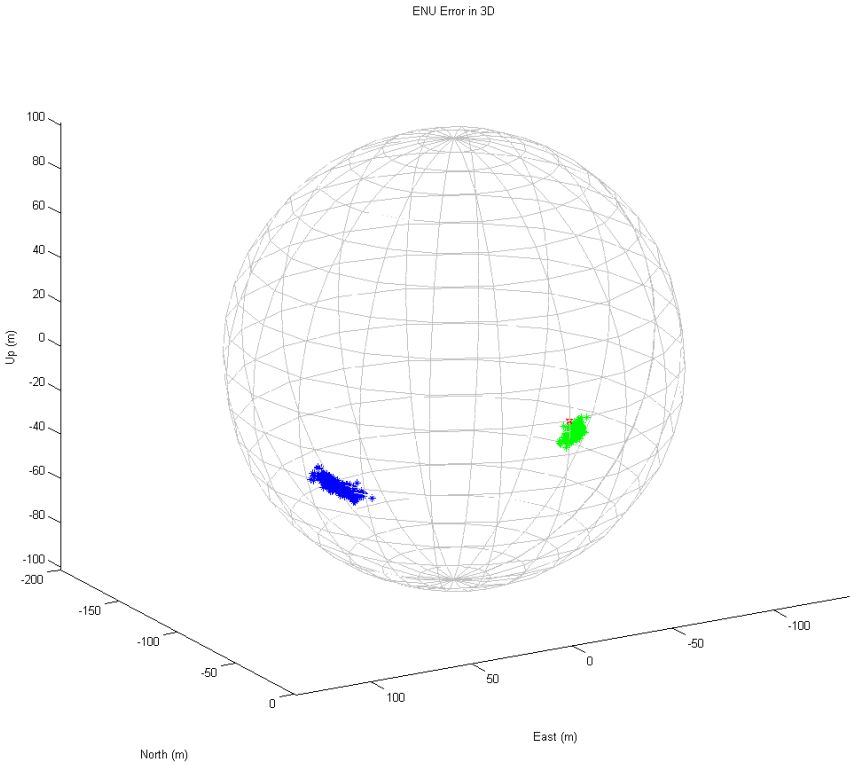

Figure 4.9: Candidate solutions for interval 2 and AP2 with 100m range and small error

In the case of having large errors, the two candidate solutions are not so separated and therefore the *slope method* works worse than before. It is seen in Figure 4.10 (corresponding to interval 1 and AP1 with 50m range) that both solutions are not as separated as in Figure 4.5, where the error is small. In Table 4.4 it can be observed that when the range increases, not always the percentage of success increases. In interval 1 and AP1, the percentage of success decreases while the range increases. For AP2, the percentage of success increases when the range changes from 20m to 50m, but it decreases when the range changes from 50m to 100m. However, in interval 2 for every AP when the range increases, the percentage of success increases. In Figures 4.11, 4.12 and 4.13 it is seen how the separation between the two solutions increases while the range increases. If it is compared with the case of small errors (Figures 4.7, 4.8 and 4.9), it is seen how large range errors make the separation between solutions become smaller. The percentages of success (54% for 20m, 60% for 50m and 79% for 100m) are lower than the ones obtained with small errors (68%, 99% and 100%).

As the range error has a direct impact on the positioning error, the *slope method* works better when simulating small range errors.

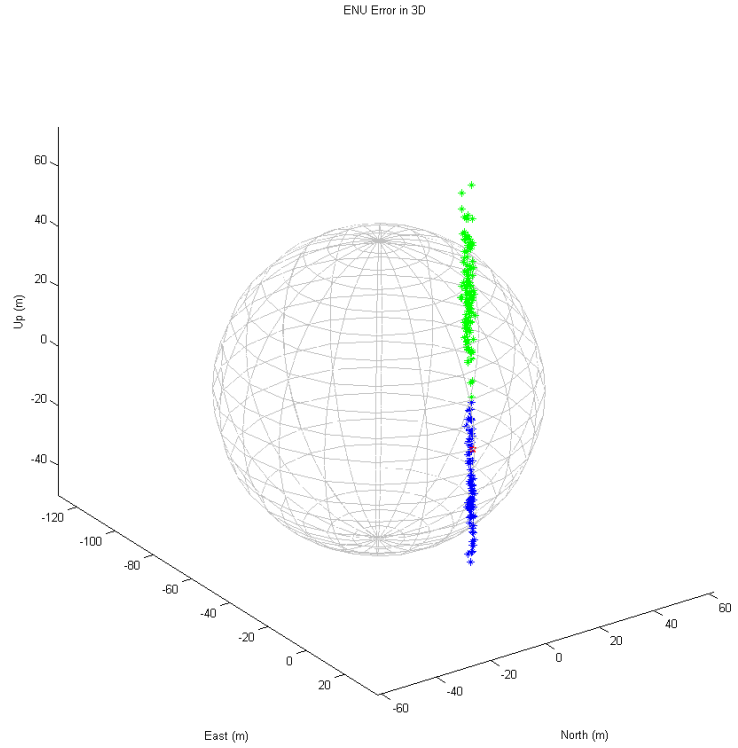

Figure 4.10: Candidate solutions for interval 1 and AP1 with 50m range and large error

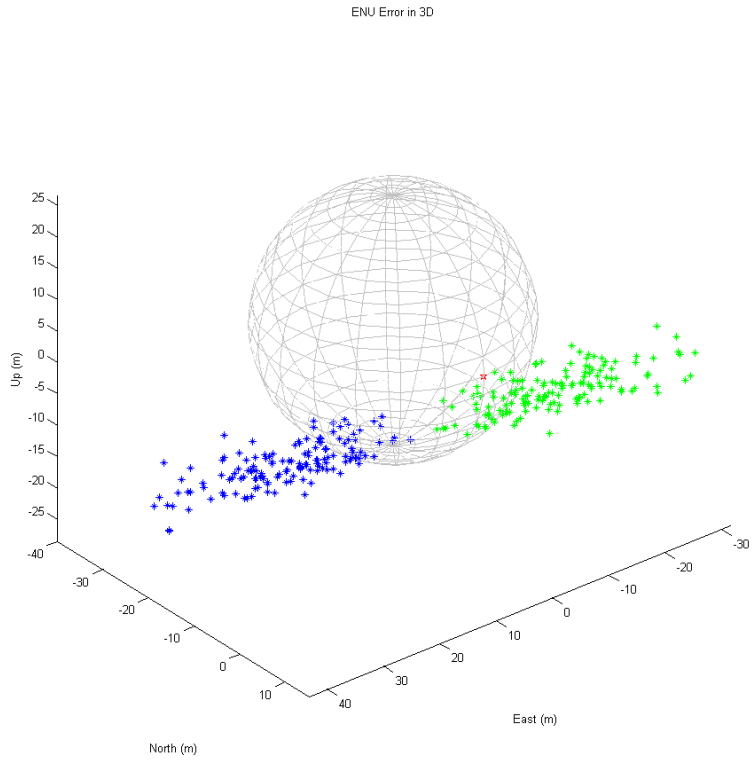

Figure 4.11: Candidate solutions for interval 2 and AP2 with 20m range and large error

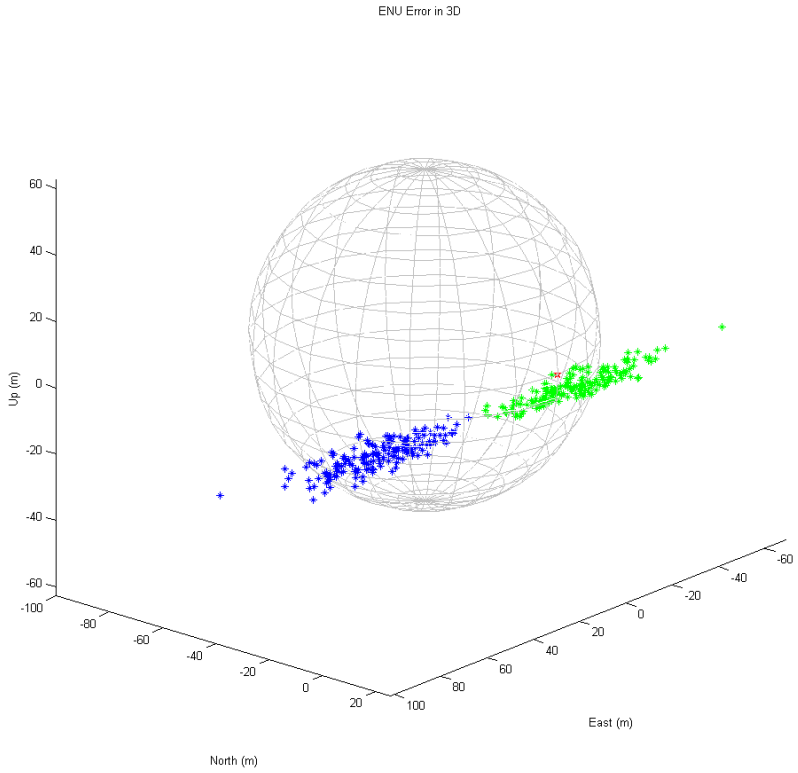

Figure 4.12: Candidate solutions for interval 2 and AP2 with 50m range and large error

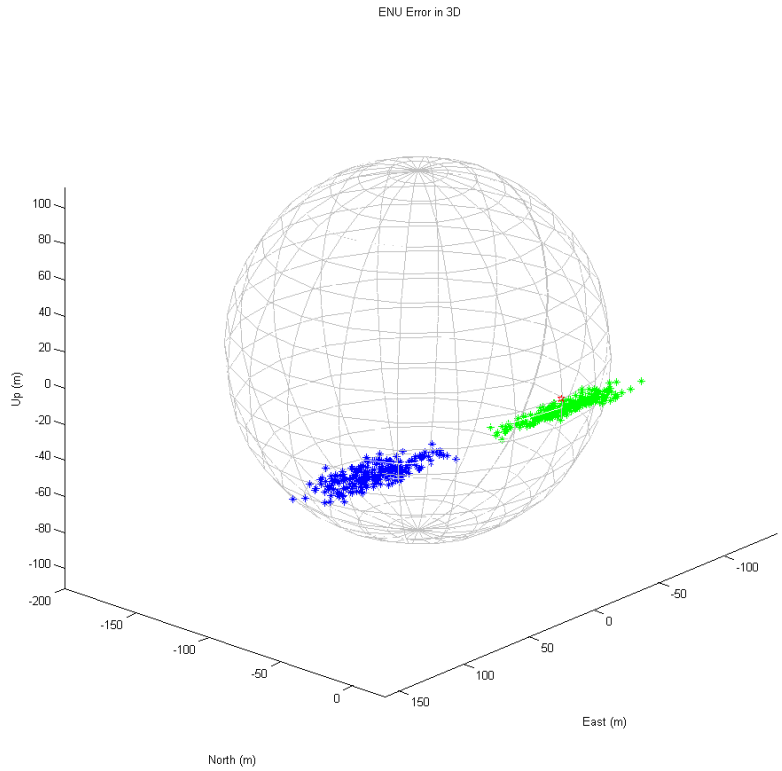

Figure 4.13: Candidate solutions for interval 2 and AP2 with 100m range and large error

4.3.3 GPS SIGNAL WITH ERROR AND TERRESTRIAL SIGNAL WITHOUT ERROR

Epoch	Terrestrial range (m)	AP	Success percentage (Bias for every sat. / Disp = 0.1*Bias)
1700 to 1940	20	1	42 %
		2	63 %
		3	Doesn't work (algorithm doesn't converge in 93% of cases).
		4	52 %
	50	1	39 %
		2	68 %
		3	Doesn't work (algorithm doesn't converge in 69% of cases).
		4	75 %
	100	1	54 %
		2	86 %
		3	34 %
		4	85 %
2400 to 2640	20	1	66 %
		2	58 %
		3	56 %
		4	78 %
	50	1	98 %
		2	87 %

		3	98 %
		4	94 %
	100	1	100 %
		2	99 %
		3	96 %
		4	100 %

Table 4.5: Test of the slope method with only GPS error

It can be appreciated in Table 4.5 that in interval 1 the success percentage does not vary much when it does the range. If AP1 is considered, the percentages obtained are 42% for 20m, 39% for 50m and 54% for 100m. On the other hand, in interval 2 the percentage varies as it does the range. The percentages for AP1 are 66% for 20m, 98% for 50m and 100% for 100m. In Figures 4.14, 4.15 and 4.16, it is seen how the separation of both solutions increases while the range increases. It can be observed that the success of the *slope method* not only depends on the terrestrial range, but on the geometrical arrangement of the reference points (satellites + AP) too.

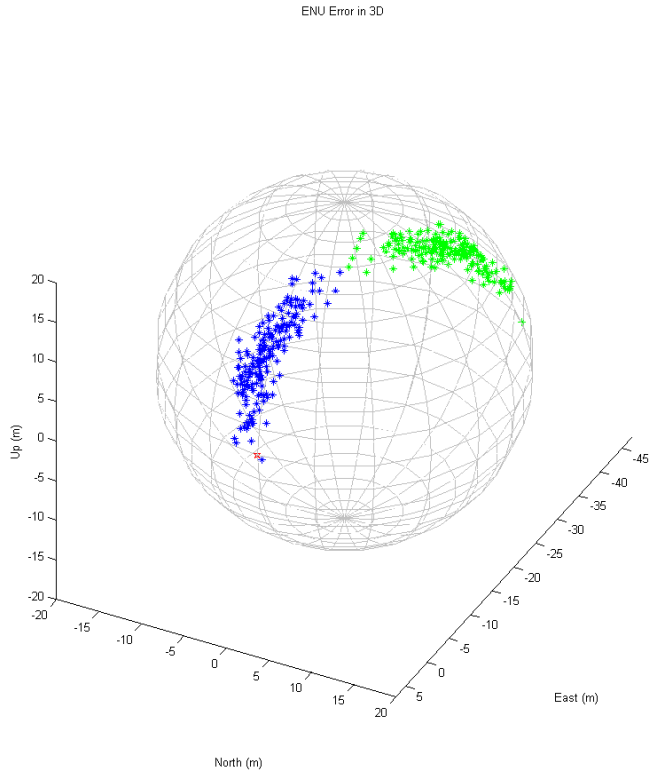

Figure 4.14: Candidate solutions for interval 2 and AP1 with 20m range and GPS error

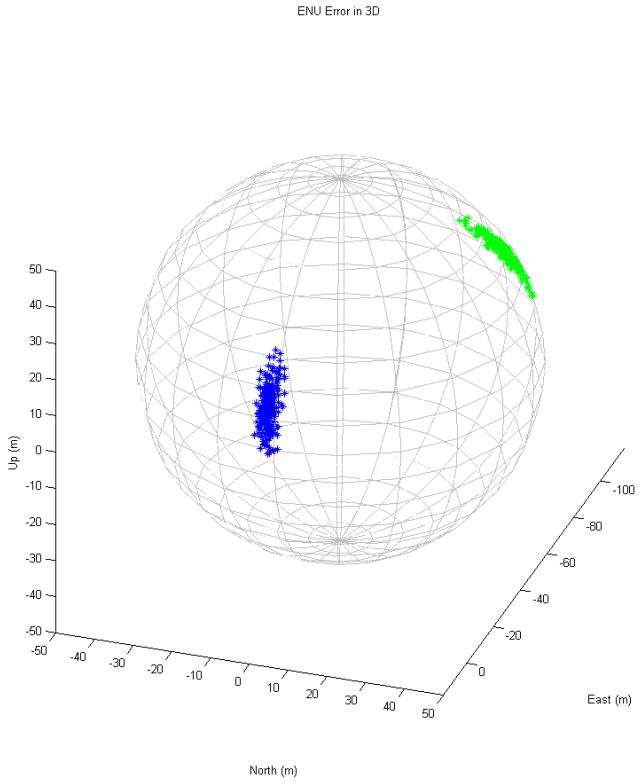

Figure 4.15: Candidate solutions for interval 2 and AP1 with 50m range and GPS error

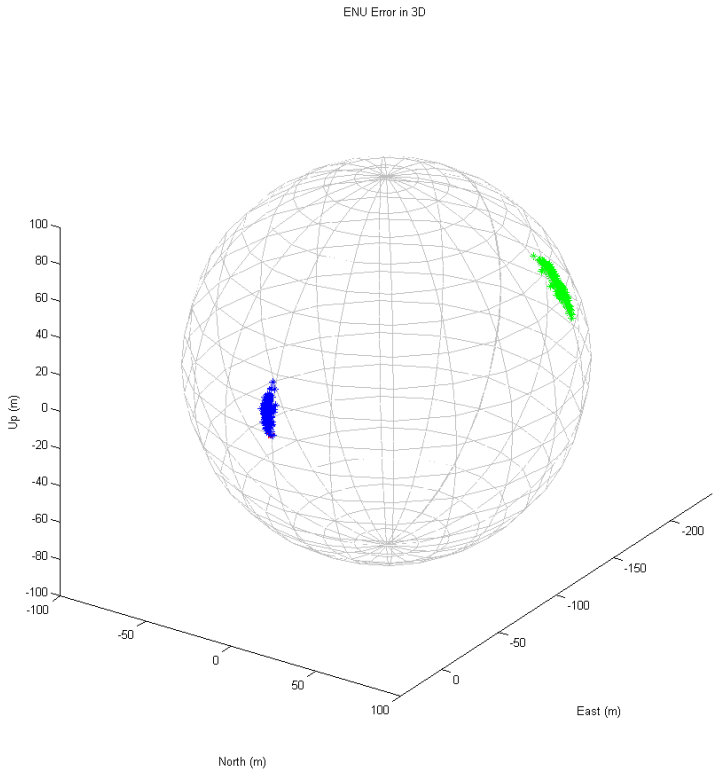

Figure 4.16: Candidate solutions for interval 2 and AP1 with 100m range and GPS error

4.3.4 GPS AND TERRESTRIAL SIGNALS WITH ERROR

Epoch	Terrestrial range (m)	AP	Success percentage (Bias = 0 / Disp = 1)	Success percentage (Bias = 0 / Disp = 10)
1700 to 1940	20	1	42 %	41 %
		2	53 %	56 %
		3	Doesn't work (algorithm doesn't converge in 51% of cases).	33 %
		4	58 %	49 %
	50	1	43 %	35 %
		2	67 %	57 %
		3	42 %	26 %
		4	74 %	65 %
	100	1	50 %	41 %
		2	81 %	66 %
		3	31 %	31 %
		4	85 %	70 %
2400 to 2640	20	1	78 %	56 %
		2	68 %	60 %
		3	53 %	55 %
		4	77 %	63 %
	50	1	93 %	84 %
		2	96 %	65 %

		3	87 %	64 %
		4	90 %	88 %
	100	1	100 %	98 %
		2	97 %	64 %
		3	97 %	84 %
		4	100 %	94 %

Table 4.6: Test of the slope method with GPS and terrestrial errors

In a first analysis, it seems that GPS errors influence more than terrestrial errors, since for the same configuration the error rate does not vary significantly between small and large terrestrial errors. To corroborate that, a table is done comparing all the errors obtained from Section 4.3.2 to 4.3.4 with the AP1:

Epoch	Range (m)	Small terrestrial error	Large terrestrial error	GPS error	GPS + small terrestrial error	GPS + large terrestrial error
1700 to 1940	20	67 %	46 %	42 %	42 %	41 %
	50	82 %	41 %	39 %	43 %	35 %
	100	97 %	38 %	54 %	50 %	41 %
2400 to 2640	20	100 %	68 %	66 %	78 %	56 %
	50	100 %	88 %	98 %	93 %	84 %
	100	100 %	99 %	100 %	100 %	98 %

Table 4.7: Comparison of the slope method tests performed with AP1

It can be seen in Table 4.7 that the percentage of success when considering GPS + terrestrial errors (columns 6 and 7) is much closer to the one obtained with only GPS error (column 5) rather than the one obtained with only terrestrial errors (columns 3 and 4). Therefore, it can be stated that GPS errors influence more than terrestrial errors when using the *slope method*.

4.4 POSITIONING ERROR OF THE PROPOSED METHOD

In order to test the accuracy of the proposed hybrid positioning method, it is assumed that a reliable initial position of the user is available. There are two possibilities of achieving a reliable initial position: calculating the position by GPS alone or executing the *slope method*. To perform the tests, it is decided to obtain the initial position with GPS alone. The selected intervals of epochs are:

- From 1616 to 1856
- From 2360 to 2600

In both intervals, the position corresponding to the first epoch is obtained by 4 GPS satellites, and the 239 epochs remaining by 3 GPS satellites and a terrestrial AP.

As the case of evaluating the *slope method*, tests will be performed with neither GPS nor terrestrial errors, with only terrestrial errors, with only GPS errors and with GPS and terrestrial errors. For each case, it will be represented the success percentage (i.e. the percentage of calculated positions that belong to the correct solution) and the CDF (Cumulative Distribution Function) of the horizontal and vertical positioning error[10]. To correctly represent the CDF, it is considered to show the results with σ (68.26% of the positions) and 2σ (95.44% of the positions).

A test program is created to simulate the errors of the ranges and pseudoranges, compute the position considering the previous errors and show the results with different graphics. As the random variables generated are different for each execution, it is decided to execute the test program 20 times in order to obtain statistically reliable results. The code of the program can be seen in Appendix 2.

4.4.1 GPS AND TERRESTRIAL SIGNALS WITHOUT ERROR

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	58 %	6.7 m	7.7 m	14 m	14.8 m
		2	53 %	2.2 m	2.8 m	6.2 m	9.3 m
		3	65 %	3.8 m	5.5 m	5.3 m	8.5 m
		4	Doesn't work (algorithm	Doesn't work (algorithm	Doesn't work (algorithm	Doesn't work (algorithm	Doesn't work (algorithm

[10] These errors are usually known as HPE and VPE. HPE stands for Horizontal Positioning Error and VPE for Vertical Positioning Error.

			doesn't converge)	doesn't converge)	doesn't converge)	doesn't converge)	doesn't converge)
	50	1	100 %	1.1 m	1.8 m	6.1 m	7 m
		2	28 %	4.5 m	6.3 m	17.2 m	20.6 m
		3	42 %	7.6 m	9.3 m	16 m	19.4 m
		4	58 %	4.3 m	5.8 m	17.6 m	21 m
	100	1	100 %	1.1 m	1.9 m	6.6 m	7.7 m
		2	100 %	2.1 m	3.3 m	4.5 m	9 m
		3	32 %	13.1 m	15.2 m	31.7 m	36.4 m
		4	100 %	1.5 m	2.3 m	8.6 m	10.6 m
2360 to 2600	20	1	100 %	1.8 m	2.5 m	7 m	8.6 m
		2	10 %	18.5 m	21.2 m	11.3 m	12.1 m
		3	76 %	6.4 m	18.2 m	5.4 m	7.5 m
		4	100 %	3.5 m	4.6 m	8.1 m	10 m
	50	1	100 %	1.8 m	2.6 m	7.3 m	8.9 m
		2	100 %	2.6 m	4 m	6.9 m	9.1 m
		3	100 %	3.8 m	4.9 m	6.4 m	8.4 m
		4	100 %	2.3 m	2.8 m	7.7 m	9.4 m
	100	1	100 %	1.9 m	2.6 m	7.4 m	9 m
		2	100 %	2.9 m	4.2 m	6.8 m	8.9 m
		3	100 %	3.5 m	4.6 m	6.5 m	8.6 m
		4	100 %	2.1 m	2.8 m	7.6 m	9.3 m

Table 4.8: Positioning error with neither GPS nor terrestrial errors

It is seen in Table 4.8 that the maximum horizontal error is 21.2m and occurs for the interval of epochs 2360 to 2600, the range of 20m and the AP2. The maximum vertical error is 36.4m and is obtained for the interval of epochs from 1616 to 1856, the range of 100m and the AP3. In both

cases, the percentage of positions corresponding to the correct solution is less than 50%. When the two candidate solutions are more separated, the percentage of correct solutions reflects the accuracy of the system. When the candidate solutions are closer, this percentage does not have so much weight. This is explained in more detail in the next sections.

4.4.2 GPS SIGNAL WITHOUT ERROR AND TERRESTRIAL SIGNAL WITH ERROR

First, the case of small terrestrial range errors (bias=0 and disp.=1) is commented:

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	49 %	6.8 m	8.1 m	13.8 m	16.7 m
		2	51 %	2.6 m	3.8 m	8 m	12.2 m
		3	50 %	4.6 m	6.3 m	7.6 m	11.3 m
		4	51 %	2 m	3.1 m	9.3 m	12.7 m
	50	1	58 %	12.2 m	14.3 m	29.7 m	34.2 m
		2	49 %	4.3 m	6.7 m	16.5 m	22.8 m
		3	48 %	7.8 m	10.3 m	16.3 m	22.3 m
		4	50 %	5.4 m	7.6 m	20.9 m	26.4 m
	100	1	100 %	1.5 m	2.4 m	7.5 m	10.7 m
		2	53 %	8 m	12.5 m	28.5 m	38.7 m
		3	54 %	12.9 m	16.1 m	31.3 m	39.1 m
		4	55 %	13.8 m	17 m	44.4 m	52.2 m
2360 to 2600	20	1	100 %	2.2 m	3.3 m	7 m	8.6 m
		2	51 %	16.4 m	21.3 m	10.2 m	12.3 m
		3	49 %	17.4 m	21.8 m	4.5 m	7.5 m
		4	92 %	4.3 m	25.3 m	8.5 m	12.8 m
	50	1	100 %	2.1 m	3.1 m	7.3 m	9 m

		2	100 %	3.2 m	5.5 m	6.9 m	9.3 m
		3	100 %	4.6 m	7.3 m	6.4 m	8.5 m
		4	100 %	2.7 m	4.2 m	7.7 m	9.6 m
	100	1	100 %	2.2 m	3.3 m	7.4 m	9.2 m
		2	100 %	3.5 m	5.7 m	6.9 m	9.1 m
		3	100 %	4.2 m	6.6 m	6.5 m	8.6 m
		4	100 %	2.5 m	3.8 m	7.6 m	9.4 m

Table 4.9: Positioning error with only small terrestrial errors

It can be appreciated in Table 4.9 that the maximum HPE is 25.3m and it corresponds to the interval of epochs from 2360 to 2600, the range of 20m and the AP4. The maximum VPE is 52.2m and is achieved for the interval of epochs 1616 to 1856, the range of 100m and the AP4. It seems that the positioning error depends more on the geometrical arrangement than the percentage of correct solution. It can be observed that AP2 and AP3 have similar percentages of correct solution for both intervals of epochs. For all the ranges, the percentage is practically the same, but the HPE is always larger for the AP3. It can be observed too that for the interval of epochs from 2360 to 2600 and the range of 100m, the percentages for all the APs are the same (100%). Nevertheless, the HPE and VPE are different for each AP.

Let's see if this can be applied too for large range errors (bias=0 and disp.=10):

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	47 %	9.5 m	14.7 m	22.4 m	35.8 m
		2	51 %	7.5 m	11.4 m	20.4 m	30.5 m
		3	53 %	8.3 m	13.4 m	20.5 m	32.6 m
		4	52 %	7.1 m	10.9 m	19.8 m	30.3 m
	50	1	43 %	16.4 m	22.9 m	41.4 m	59.5 m
		2	50 %	10.6 m	15.2 m	30.2 m	44.9 m
		3	51 %	13.6 m	19.7 m	34 m	50.3 m

		4	54 %	10.6 m	15.8 m	31.4 m	48.9 m
	100	1	50 %	26.2 m	34 m	69.1 m	90.6 m
		2	49 %	14.8 m	21.4 m	43.1 m	64.3 m
		3	47 %	20.3 m	28.7 m	52.7 m	75.2 m
		4	49 %	15.9 m	24.5 m	49.6 m	76.3 m
2360 to 2600	20	1	50 %	32.9 m	48.9 m	7 m	11 m
		2	52 %	23.8 m	38.3 m	11.8 m	17.1 m
		3	49 %	26 m	41.1 m	7 m	11.9 m
		4	52 %	28.6 m	43.9 m	13.4 m	18.9 m
	50	1	58 %	74.3 m	95.5 m	12.6 m	20.2 m
		2	50 %	48.3 m	69.1 m	19.2 m	25.5 m
		3	49 %	51.6 m	71.7 m	9.8 m	15.1 m
		4	52 %	68.9 m	89.6 m	24.5 m	31.8 m
	100	1	100 %	11.4 m	22.4 m	8.1 m	12 m
		2	56 %	88.4 m	118.2 m	29.5 m	38.4 m
		3	50 %	95.7 m	121.4 m	19.1 m	27.9 m
		4	100 %	12.1 m	24.4 m	8.4 m	13.2 m

Table 4.10: Positioning error with only large terrestrial errors

It is seen in Table 4.10 that the maximum HPE is 121.4m and is obtained for the second interval of epochs, the range of 100m and the AP3. The maximum VPE is 90.6m and corresponds to the first interval, the range of 100m and the AP1.

In the case of large errors, AP2 and AP3 have similar percentages of correct solution, and HPE is still always larger for AP3 than for AP2. Concerning to the percentages of correct solution for the second interval of epochs and the range of 100m, it can be observed that are not the same for all the APs. Furthermore, it can be observed that this time the percentage of correct solution is related with the positioning error. For the AP2 and AP3, the percentage is close to 50% and the HPE is close to 120m. For the AP1 and AP4, the percentage is 100% and the HPE is close to 23m.

In order to know which range error has more impact on the positioning error, it is decided to compare for each configuration (i.e. the configurations represented in Tables 4.9 and 4.10) the HPE and VPE with the ones obtained with no errors. In the case of small terrestrial errors, it can be seen in Figure 4.17 that the increase of positioning error is below 4m in almost all the cases. The mean value of the positioning error is 3.4m for the HPE and 5.8m for the VPE. However, in the case of large terrestrial errors, it can be appreciated in Figure 4.18 that the increase of positioning error is larger than before. Almost all of the configurations are below 35m, and the mean value of the positioning error is 36.7m for the HPE and 25.3 for the VPE. As expected, larger terrestrial errors lead to larger positioning errors.

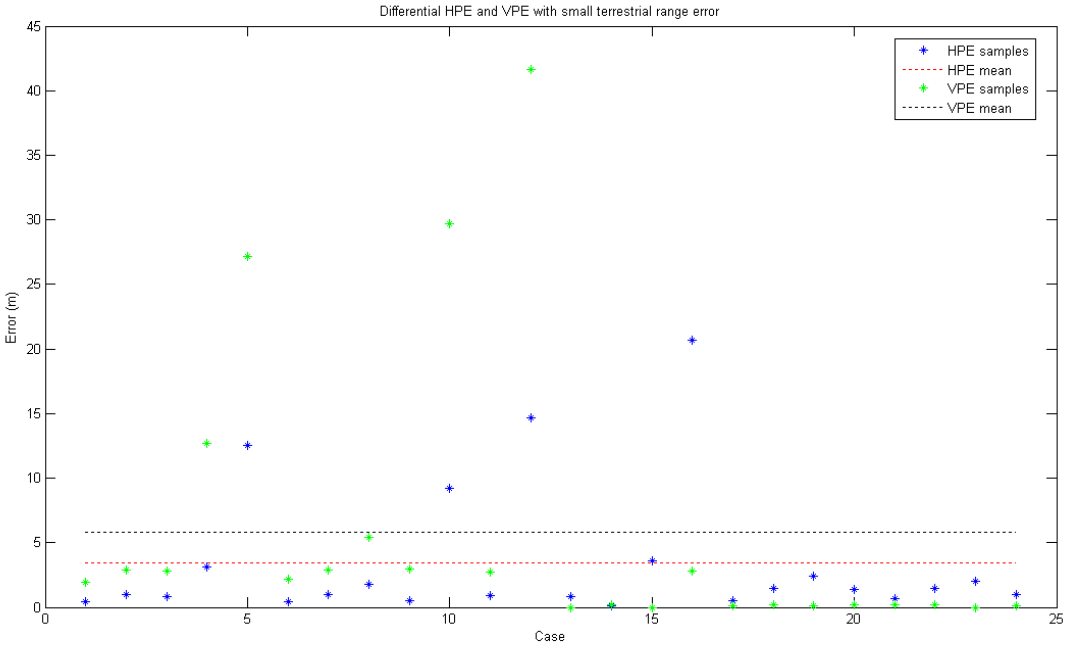

Figure 4.17: Differential positioning error for the case of small terrestrial range error

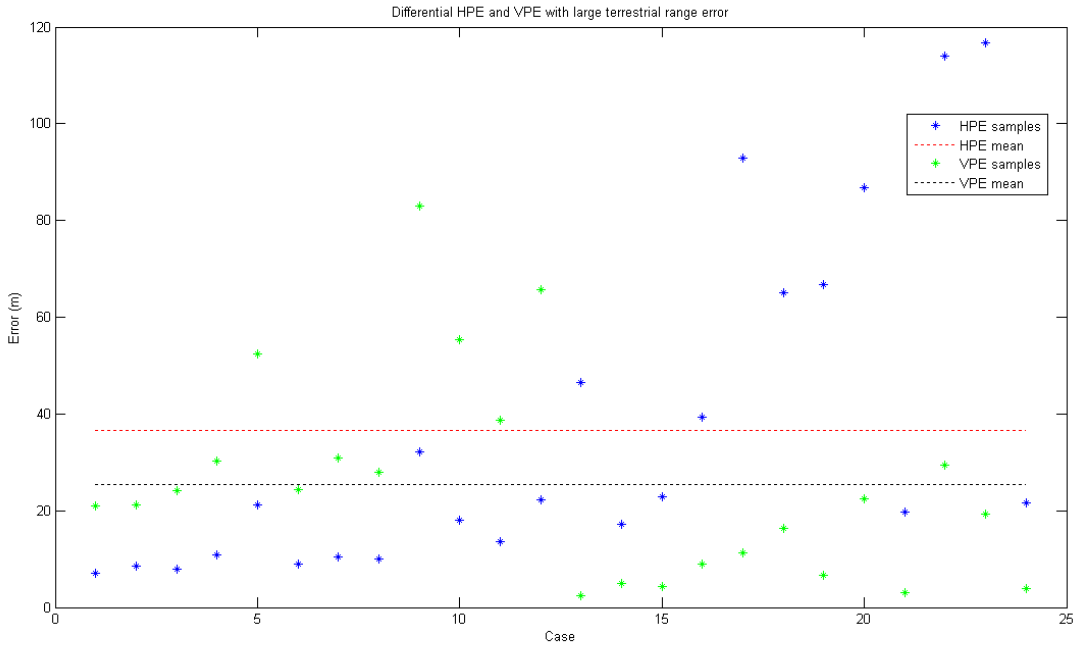

Figure 4.18: Differential positioning error for the case of large terrestrial range error

4.4.3 GPS SIGNAL WITH ERROR AND TERRESTRIAL SIGNAL WITHOUT ERROR

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	57 %	4 m	5.6 m	9.2 m	12.3 m
		2	16 %	10.6 m	13.2 m	14.1 m	15.8 m
		3	86 %	2.9 m	7.7 m	2.7 m	10.1 m
		4	32 %	10.1 m	12.5 m	12.5 m	13.3 m
	50	1	42 %	5.6 m	7.9 m	21.7 m	26.6 m
		2	9 %	15.8 m	18.9 m	28.6 m	31.8 m
		3	29 %	2.9 m	7.6 m	13.8 m	20.3 m
		4	20 %	16.6 m	19.3 m	31.5 m	34 m
	100	1	38 %	12 m	16.1 m	46.4 m	52.6 m
		2	20 %	22.2 m	26 m	47.6 m	52.7 m

		3	32 %	7.4 m	9.4 m	27.4 m	32.2 m
		4	100 %	4.3 m	6.5 m	9.5 m	13.1 m
2360 to 2600	20	1	49 %	23.6 m	30.1 m	15.8 m	18.4 m
		2	52 %	22.3 m	25 m	9.8 m	13.9 m
		3	50 %	13.7 m	18.8 m	7.8 m	10.2 m
		4	52 %	31 m	33.1 m	9 m	13.2 m
	50	1	100 %	3.3 m	5 m	13.3 m	19.6 m
		2	100 %	3.7 m	5.5 m	13.4 m	18.8 m
		3	52 %	34.7 m	46 m	18.7 m	22.3 m
		4	100 %	4.8 m	7.7 m	12.2 m	17 m
	100	1	100 %	3.2 m	4.8 m	13.1 m	18.8 m
		2	100 %	4.3 m	6.4 m	13.9 m	19.5 m
		3	85 %	9.7 m	92.6 m	17.2 m	35.6 m
		4	100 %	4.2 m	6.3 m	12.4 m	17.8 m

Table 4.11: Positioning error with only GPS errors

It can be seen in Table 4.11 that the maximum HPE is 92.6m and occurs for the second interval of epochs, the range of 100m and the AP3. The maximum VPE is 52.7m and is obtained for the first interval, the range of 100m and the AP2.

In the case of having only GPS errors, the positioning error is more influenced by the geometrical arrangement than the percentage of correct solution. For the first interval of epochs, the range of 50m and the AP3, a percentage of success of 29% is obtained. For the same configuration (50m range and AP3), but the second interval, the percentage of success is 52%. However, the HPE of the first interval (7.6m) is much smaller than the one of the second interval (46m). It can be seen in Figures 4.19 and 4.20 that the HPE depends more on the separation of the candidate solutions than the percentage of the correct solution (i.e. one configuration can have a success of 80%, but if the separation is larger than another that have 20% of success, it will end in a worse positioning accuracy). This can be seen for the second interval, the range of 100m and the AP3, where the percentage of success is 85% and the HPE is 92.6m. It is true that 68% of the positions are below 9.7m, but it cannot be assured that the rest is below 9.7m.

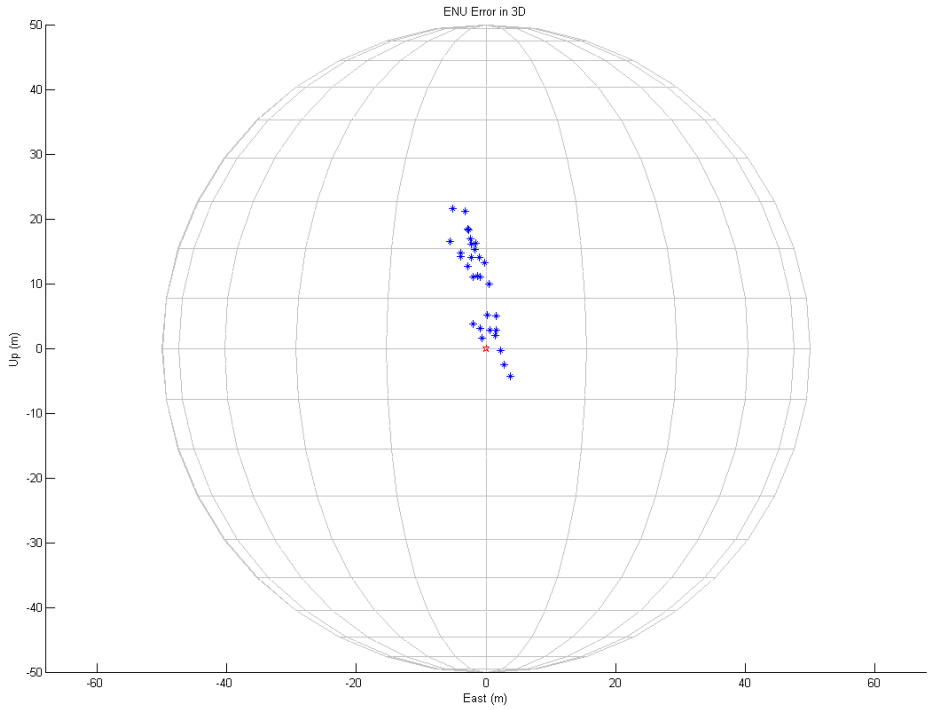

Figure 4.19: Positioning error for the interval of 1616 to 1856, range of 50m and AP3

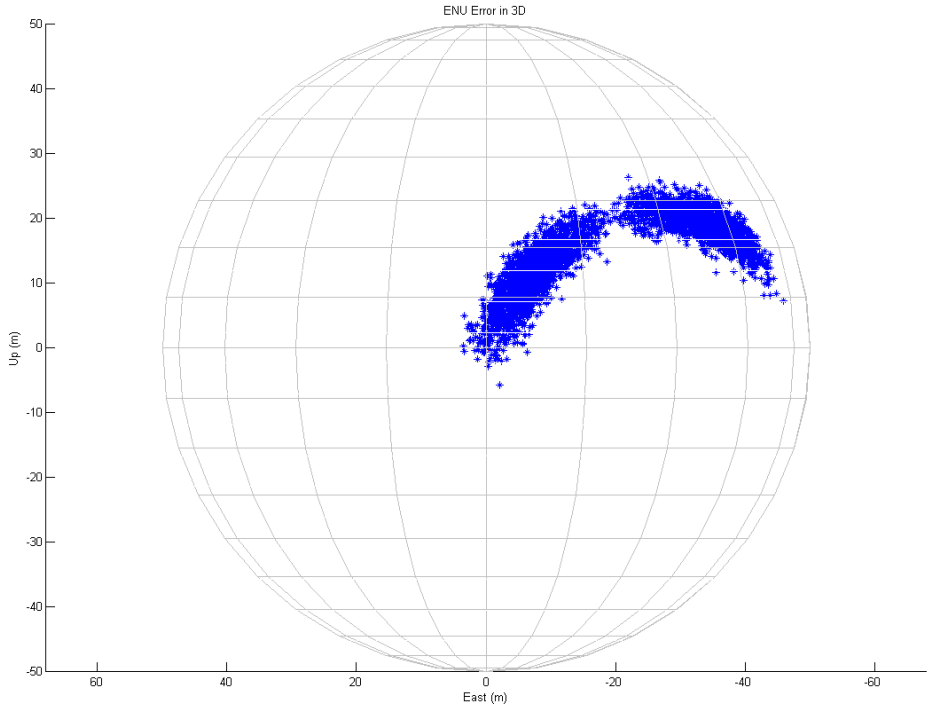

Figure 4.20: Positioning error for the interval of 2360 to 2600, range of 50m and AP3

It can be seen in Figures 4.19 and 4.20 that the algorithm not always converges. In both figures, the estimated positions are represented, but the number of points is much smaller in Figure 4.19 than in Figure 4.20. That is a fact to consider when comparing the results, as the information provided is not the same in statistical terms.

With the objective of quantifying the influence of the GPS pseudorange error in the positioning error, the HPE and VPE for the case of GPS pseudorange error is compared with the case of no errors. It is observed in Figure 4.21 that almost all of the configurations are below 20m. The mean value is 13.1m for the HPE and 11.6m for the VPE. The positioning error is larger than the case of small terrestrial error, but smaller than the large terrestrial error.

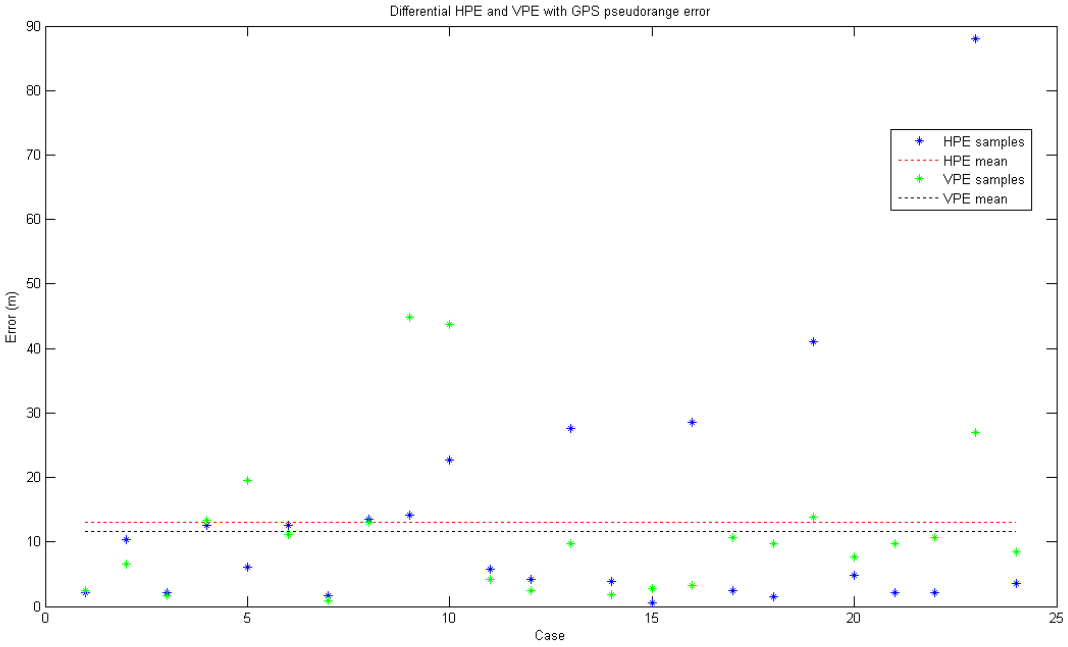

Figure 4.21: Differential positioning error for the case of GPS pseudorange error

4.4.4 GPS AND TERRESTRIAL SIGNALS WITH ERROR

First, the case of having small terrestrial range and pseudorange errors is considered:

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	43 %	4.4 m	6 m	10.7 m	14.4 m
		2	43 %	9.8 m	13 m	13.6 m	16.3 m
		3	40 %	2.9 m	4.3 m	8.4 m	10.9 m

		4	49 %	9.5 m	12.4 m	12.4 m	16.1 m
	50	1	37 %	5.8 m	8.2 m	23.3 m	29.4 m
		2	6 %	15.8 m	18.9 m	28.7 m	32.6 m
		3	50 %	3.5 m	6 m	15 m	20.7 m
		4	42 %	15.7 m	18.9 m	30.6 m	34.8 m
	100	1	38 %	12.6 m	16.6 m	47.5 m	55 m
		2	45 %	20.4 m	25.4 m	44.9 m	52.6 m
		3	35 %	6.2 m	9.3 m	25.5 m	34 m
		4	100 %	4.5 m	6.6 m	10.2 m	14.7 m
2360 to 2600	20	1	51 %	23.4 m	30.3 m	15.7 m	18.8 m
		2	50 %	22.2 m	25.9 m	9.7 m	14 m
		3	56 %	13.1 m	20 m	8 m	11.2 m
		4	53 %	30.6 m	33.8 m	8.9 m	13.2 m
	50	1	100 %	3.5 m	5.3 m	13 m	19.4 m
		2	100 %	4.1 m	6.1 m	13.4 m	18.6 m
		3	51 %	35.8 m	46.7 m	18.7 m	22.9 m
		4	100 %	5.1 m	8.1 m	12.2 m	17.2 m
	100	1	100 %	3.4 m	5.1 m	13.1 m	18.9 m
		2	100 %	4.6 m	7.2 m	13.9 m	19.5 m
		3	73 %	14.1 m	94.8 m	21.5 m	37.6 m
		4	100 %	4.4 m	6.7 m	12.6 m	17.5 m

Table 4.12: Positioning error with GPS and small terrestrial errors

It is seen in Table 4.12 that the maximum HPE is 94.8m and is obtained for the second interval of epochs, the range of 100m and the AP3. The maximum VPE is 55m and occurs for the first interval, the range of 100m and the AP1. These maximum values are very similar to the ones obtained with only GPS errors (92.6m for HPE and 52.7m for VPE). That reflects that GPS errors influence more

than terrestrial errors, because with only small terrestrial errors, the maximum values are 25.3 for HPE and 52.2 for VPE.

As the case of having only GPS errors, the positioning error is more influenced by the geometry than the success percentage. It can be seen in the first interval of epochs and the range of 20m that all the percentages are approximately the same. Nevertheless, the HPE is different for each AP. AP3 has the smaller percentage and, however, the smaller HPE. For the range of 100m, AP3 has the smaller percentage (35%) too, and does not have much more error than AP4, which has a percentage of 100%. That reflects the geometry dependency of the results.

Let's consider the case of large terrestrial range and pseudorange errors:

Epoch	Terrestrial Range (m)	AP	Percentage of correct solution	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	1	46 %	7.9 m	11.7 m	22.4 m	35.1 m
		2	48 %	10.4 m	15.5 m	21.1 m	31.5 m
		3	48 %	8 m	12.9 m	21.5 m	34.2 m
		4	50 %	10.6 m	15.8 m	20.7 m	30.4 m
	50	1	43 %	11.7 m	16.7 m	42.1 m	58.6 m
		2	51 %	13.9 m	21.1 m	32.9 m	46.2 m
		3	54 %	10.5 m	15.1 m	32.4 m	51.6 m
		4	46 %	15.6 m	22.8 m	33.2 m	49.7 m
	100	1	45 %	18.8 m	27.9 m	66.3 m	91.3 m
		2	54 %	18.3 m	28.4 m	46.9 m	67.2 m
		3	52 %	15.6 m	22.3 m	52.3 m	74.5 m
		4	55 %	21.3 m	33.2 m	49 m	79 m
2360 to 2600	20	1	47 %	28.3 m	43.3 m	17.4 m	24.6 m
		2	52 %	26.9 m	42.1 m	11.1 m	18.2 m
		3	51 %	23 m	36.7 m	15 m	22 m
		4	50 %	33.8 m	48.6 m	9.3 m	16.3 m

	50	1	53 %	68.5 m	88.2 m	28.3 m	37.9 m
		2	53 %	53.7 m	75 m	12 m	19.8 m
		3	51 %	44.4 m	64.8 m	21.7 m	30.1 m
		4	64 %	65.5 m	94.1 m	12.9 m	19.5 m
	100	1	99 %	12.7 m	25.2 m	13.7 m	21.4 m
		2	51 %	101.7 m	127.9 m	17.9 m	26.5 m
		3	52 %	84.6 m	114 m	32.8 m	43.5 m
		4	100 %	12.5 m	24.3 m	12.9 m	19.2 m

Table 4.13: Positioning error with GPS and large terrestrial errors

It is seen in Table 4.13 that the maximum HPE is 127.9m and occurs for the second interval of epochs, the range of 100m and the AP2. The maximum VPE is 91.3m and is obtained for the first interval, the range of 100m and the AP1. In this case, the maximum values are more similar to the ones obtained with only large terrestrial errors (121.4m for HPE and 90.6m for VPE) than the ones with only GPS errors (92.6m for HPE and 52.7m for VPE). That reflects that, in the case of GPS errors and large terrestrial errors, the terrestrial range errors influence more than GPS errors.

It can be observed that almost all the configurations have a success percentage of 50% approximately. Nevertheless, the HPE and the VPE vary when it does the range. When the range increases, both candidate solutions are more separated and then the HPE and the VPE increase (except in the cases of AP1 and AP4 for the range of 100m and the second interval, for which the percentage is 99% and 100% respectively).

It has to be taken into account that errors are simulated with random Gaussian variables. Although the experiments are repeated several times, the randomness of the variables makes the results being different each time. When the errors are small, it is not so relevant. However, it is important to consider it when the errors are large.

In order to study the impact of the range and pseudorange errors on the positioning error, all the configurations from Tables 4.12 and 4.13 are compared with the ones without errors. In the case of GPS pseudorange and small terrestrial range errors (Figure 4.22), almost all of the configurations are below 15m. The mean value of the positioning error is 7.9m for HPE and 12.1m for VPE. In the case of GPS pseudorange and large terrestrial range errors (Figure 4.23), almost all the cases are below 50m. The mean value of the positioning error is 37.1m for HPE and 28.1m for VPE. As expected, larger terrestrial errors lead to larger positioning errors.

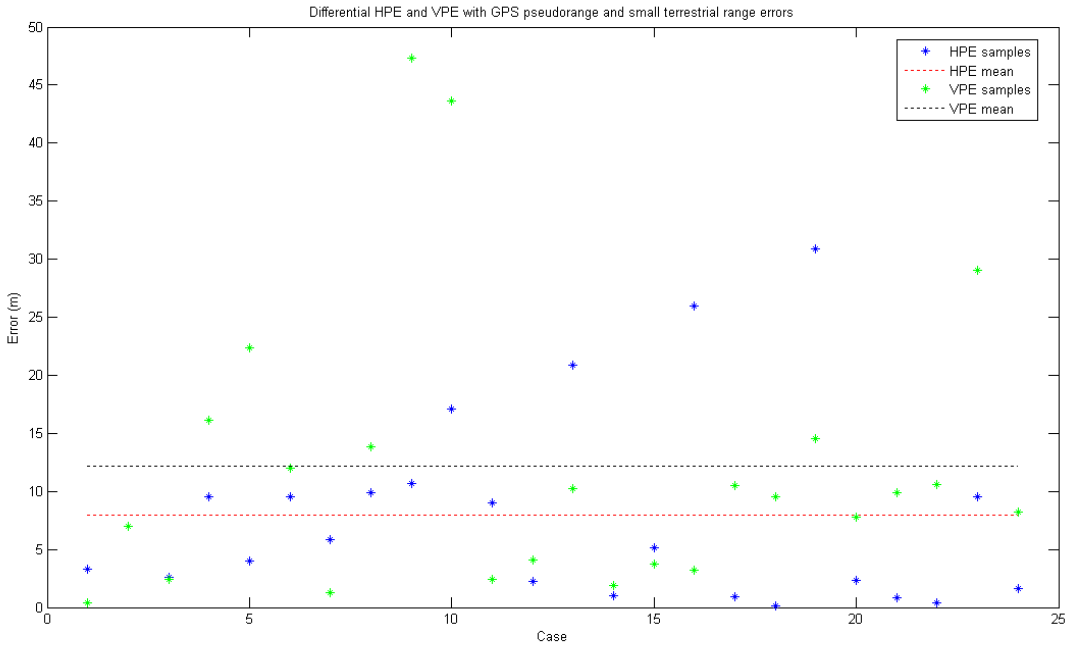

Figure 4.22: Differential positioning error for the case of GPS pseudorange and small terrestrial range errors

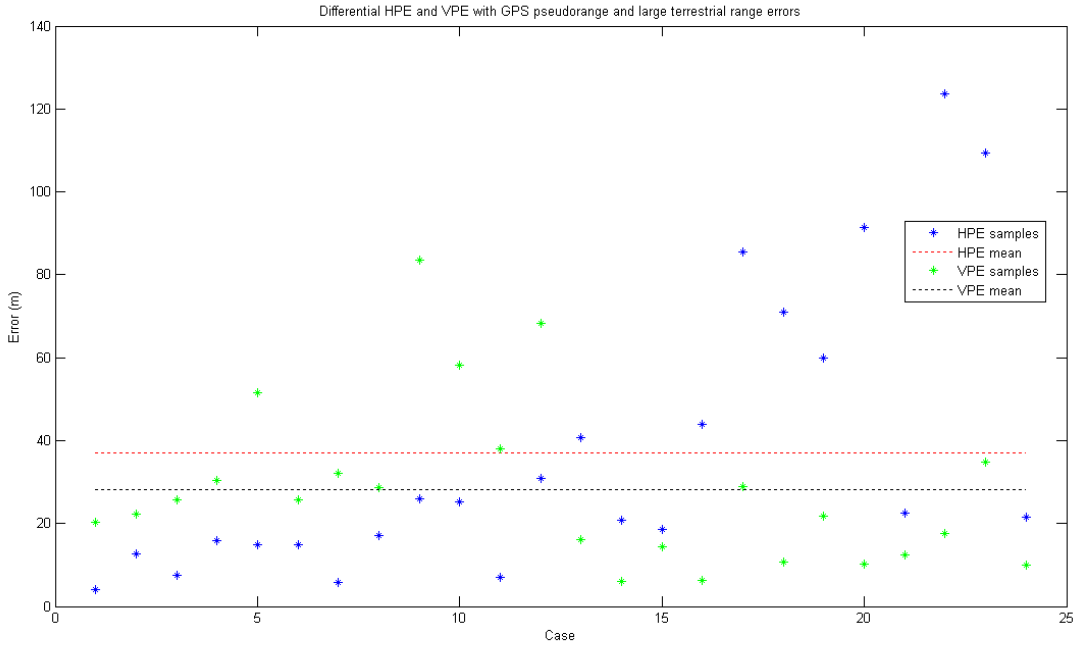

Figure 4.23: Differential positioning error for the case of GPS pseudorange and large terrestrial range errors

To conclude the study of the positioning error, all the means obtained in the previous sections are compared. It can be seen in Figure 4.24 that when the terrestrial range error is small, the GPS

pseudorange error influence more the positioning error, whereas when the terrestrial range error is large, the incorporation of GPS pseudorange error does not modify much the positioning error.

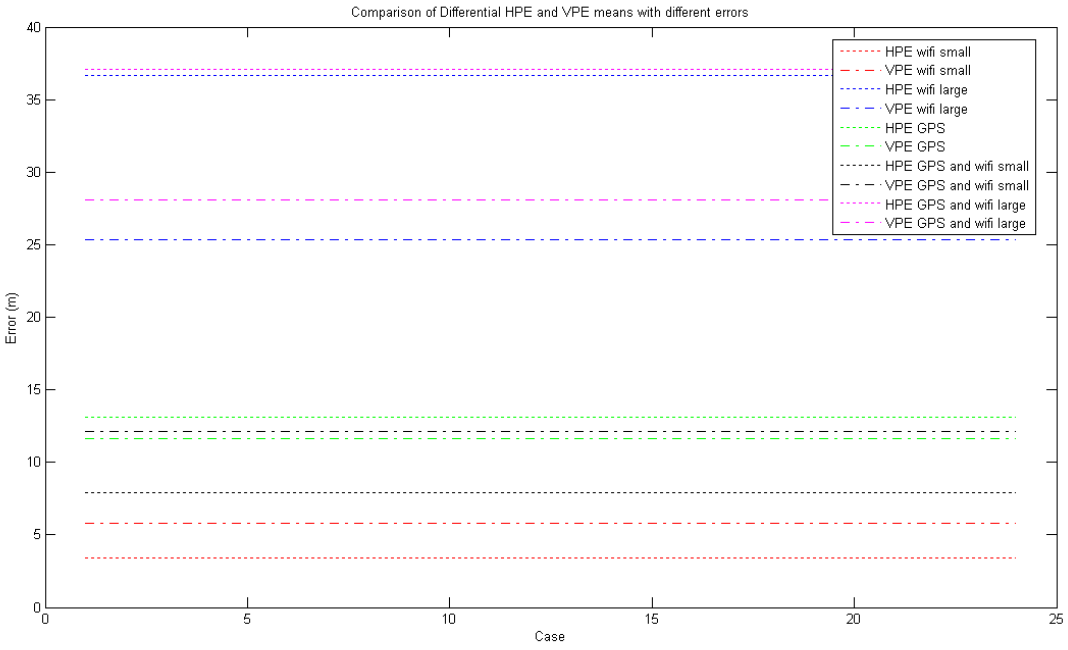

Figure 4.24: Comparison of differential positioning error obtained with different errors

4.5 COMPARISON OF THE POSITIONING ERROR USING THE PROPOSED METHOD AND CONVERGING ALWAYS TO THE CORRECT SOLUTION

In the above tables, a column representing the percentage of times that the proposed method converges to the correct solution has been added. In order to show the difference between using the proposed method and converging always to the correct solution[11], some concrete cases are represented. All of them correspond to the group of positions calculated with GPS and terrestrial errors. In concrete, they correspond to small terrestrial errors (bias=0 and disp.=1) and the AP2. Positioning errors are compared for different intervals of epochs and terrestrial ranges.

[11] To converge to the correct solution, Bancroft's algorithm is applied and then LS is executed. This method is commented in more detail in Section 3.4.

Epoch	Terrestrial Range (m)	Used method	HPE CDF with LS (σ)	HPE CDF with LS (2σ)	VPE CDF with LS (σ)	VPE CDF with LS (2σ)
1616 to 1856	20	Proposed method	9.8 m	13 m	13.6 m	16.3 m
		Correct solution	4 m	5.7 m	11.8 m	14.3 m
	50	Proposed method	15.8 m	18.9 m	28.7 m	32.6 m
		Correct solution	3.9 m	5.6 m	16.8 m	20.8 m
	100	Proposed method	20.4 m	25.4 m	44.9 m	52.6 m
		Correct solution	4.2 m	6.4 m	20.9 m	26.3 m
2360 to 2600	20	Proposed method	22.2 m	25.9 m	9.7 m	14 m
		Correct solution	4.7 m	9.4 m	11.5 m	14.8 m
	50	Proposed method	4.1 m	6.1 m	13.4 m	18.6 m
		Correct solution	4.1 m	6.1 m	13.4 m	18.6 m
	100	Proposed method	4.6 m	7.2 m	13.9 m	19.5 m
		Correct solution	4.5 m	7.2 m	14.1 m	19.6 m

Table 4.14: Positioning error comparison using different methods

It can be appreciated in Table 4.14 how HPE for the interval of epochs 1616 to 1856 reduces to more than a half for all the ranges if the algorithm converges to the correct solution. However, for the interval of epochs 2360 to 2600 an improvement is only appreciated in the case of 20m range.

In the rest of ranges (50m and 100m), the proposed method always converge to the correct solution, thus there is no difference in positioning errors. In the case of VPE, the most significant difference between methods occurs for the interval of epochs 1616 to 1856 and the range of 100m, where the error reduces to half.

It is important to mention that the computational cost is larger in the case of calculating the correct solution, because it implies executing the Bancroft's algorithm and after executing the Least Squares twice (once for each initial position obtained with Bancroft's algorithm)[12]. However, in the case of using the proposed method, one execution of the LS is enough. Therefore, if the computational cost is not a factor to consider, Bancroft's method will be used because a higher accuracy is obtained. If not, a compromise between accuracy and computational cost will have to be reached.

[12] It is supposed that the actual user position is known. Otherwise, the Bancroft's method does not make sense.

5 CONCLUSIONS

A tight hybrid positioning system combining GPS and WLAN signals has been proposed and tested. The objective of the proposed system is to take advantage of the existing WLAN deployments in many cities in order to increase urban positioning coverage of GNSS. When studying different algorithms to hybridize both technologies, the problem of position ambiguity has been found. A method to distinguish both candidate solutions (*slope method*) has been researched. In order to calculate the position employing both kinds of measurements (GPS and WLAN), a method has been proposed considering all the possible conditions. The accuracy of the system has been tested contemplating different types of error. To perform the tests, a static user has been considered. GPS data has been collected from real GPS satellites and WLAN data from real WLAN APs.

Referring to the *slope method*, it has been concluded in Section 4.2 that the method needs 4min of acquiring data to work properly. Therefore, the duration of the *slope method* has been fixed to 4min. The percentage of success in position determination considering different range and pseudorange errors has been tested in Section 4.3. It has been concluded that the method works better when the terrestrial range errors are small rather than large (as expected). After, pseudorange errors have been incorporated, and it has been observed that they influence more than terrestrial errors. Therefore, it can be concluded that pseudorange errors have more impact than the terrestrial errors when considering the success of the *slope method*.

It has been observed too that results depend on the geometrical arrangement, which influences the separation between both candidate solutions. This is an important issue to take into account, because when both candidate solutions are more separated, the *slope method* distinguishes better between both of them. Therefore, it can be concluded that the success of the *slope method* depends on the geometrical arrangement too.

Concerning to the positioning error of the proposed method, the tests performed in Section 4.4 show that positioning errors depend more on the geometrical arrangement than the percentage of correct solution. The separation of both candidate solutions (influenced by the geometrical arrangement) determines the positioning error when the percentage of correct solution is less than 100%. The larger is the separation between both candidate solutions, the larger the positioning error (regardless of the percentage of correct solution). For that reason, the accuracy of the proposed method is better when both candidate solutions are closer.

It has been studied too the positioning error when contemplating different range and pseudorange errors compared to the case of no errors. The mean of the HPE and VPE has been represented for all the errors contemplated, and it has been concluded that GPS pseudorange errors have more impact than small terrestrial range errors but less than large terrestrial errors when calculating the positioning error.

As said before, the *slope method* works better when both candidate solutions are more separated. However, the accuracy of the proposed method is better when candidate solutions are closer. As both of them depend on the geometrical arrangement, when one works better, the other works worse.

In order to conclude, it can be said that tests performed along this project demonstrates that the proposed system enhance the positioning availability compared to the case of GPS alone. The accuracy is not as good as the one obtained by GPS alone, but in harsh environments it is better to have an estimated position (although it is not as accurate as GPS) rather than not having information about the user's position.

6 FUTURE WORK

In this project, a hybrid positioning system has been studied considering a static user. The data used to perform the tests has been collected from real GPS satellites in the case of GPS pseudoranges, and has been simulated using a ranging model in the case of terrestrial ranges.

If the project would have lasted more, the next step would be performing the tests in a real scenario. Measurements would be performed in an urban canyon and the AP would be located at the corner of a street. Knowing the coordinates of a real AP would enable the possibility of studying the behaviour of the proposed system when a user is moving. Taking this into account, the next step would be collecting data considering a dynamic user and see how the system works while performing different trajectories. In that case, different filters (e.g. Kalman filter, particle filter, etc.) could be applied to improve the accuracy of the results.

All the tests are carried out considering 3 GPS satellites and a WiFi AP. The case of 2 GPS satellites and 2 WiFi APs has not been commented in this project, but a first approach has been carried out during it. It might be interesting to finalize the tests performed with 2 GPS satellites and 2 WiFI APs and perform new tests considering the real scenario and the dynamic user.

Finally, it would be interesting to implement the algorithm in a device and see how it works in real time (considering that previous tests were good enough). This would allow studying the behaviour of the proposed system working in real time in all the previous cases. A key point when implementing the algorithm in a device is the measuring time of GPS measurements and terrestrial measurements and how they harmonize.

APPENDIX 1

The code of the *slope method* test program is presented below. In the first part of the code, range and pseudorange errors are created, AP location is selected and the range distance without error is fixed. In the second part, it is set the number of tests, the epoch to start and the number of epochs to calculate. In the third part, the main program is executed. For each epoch, the errors are incorporated to the calculated pseudorange and simulated range. After that, the algorithm calculates both initial positions with Bancroft's algorithm and uses each of them to calculate the candidate solutions with Least Squares. Each solution is filtered and saved in a vector of solutions. Once all the positions are computed, the slope of the vector values of both solutions is calculated and the *slope method* decides which the correct one is. In the last part, different figures are plotted in order to present the results.

```
%---------- Configuration of the range and pseudorange errors ----------%

%CTAE antenna position
usrxyz = [4794682.4427 169625.5396 4188738.7470];
usrllh = xyz2llh(usrxyz);

dist_AP = 50;
% Setting APs parameters
enu_AP1 = [-dist_AP 0 0];
xyz_AP1 = enu2xyz(enu_AP1,usrxyz);
enu_AP2 = [0 -dist_AP 0];
xyz_AP2 = enu2xyz(enu_AP2,usrxyz);
enu_AP3 = [0 dist_AP 0];
xyz_AP3 = enu2xyz(enu_AP3,usrxyz);
enu_AP4 = [dist_AP 0 0];
xyz_AP4 = enu2xyz(enu_AP4,usrxyz);

%selection of the AP to test
enu_AP = enu_AP4;
xyz_AP = enu2xyz(enu_AP,usrxyz)';
range_AP = norm(enu_AP);

%1 bias for every satellite
bias_GPS = zeros(1,32);
bias_GPS(4) = 3;
bias_GPS(11) = 20;
bias_GPS(13) = 9;
bias_GPS(17) = 10;
bias_GPS(20) = 7;
bias_GPS(23) = 5;
bias_GPS(25) = 17;
bias_GPS(31) = 8;
bias_GPS(32) = 6;
```

```
%dispersion of GPS satellites
disp_GPS = 0.1;

%terrestrial error
bias_wifi = 0;
disp_wifi = 1;

%---------- Initial parameters to configure the main test program ----------%

num_test = 100;
sol_1 = 0;
sol_2 = 0;

ii_start = 1700;
ii_num = 240;

c = FLAGS.c;
MINelev = FLAGS.ANGth;

% Catch Alpha & Beta. Needed to apply the Klobuchar model
[ALPHA, BETA] = getAlphaBeta(HEADER_nav);

% Find the index of the C1 parameter in the Observables matrix.
CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C1'));
if isempty(CA), CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C1C')); end
if isempty(CA), CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C5a')); end


% Setting Initial and Ending time range for observations
if isempty(FLAGS.Ini_time)
   FLAGS.Ini_time = GPS_obs.OBSERVATION(1).Epoch.GPSsecond;
end
if isempty(FLAGS.End_time)
   FLAGS.End_time = GPS_obs.OBSERVATION(end).Epoch.GPSsecond;
end

for ii=1:length(GPS_obs.OBSERVATION)
   tempII(ii)=GPS_obs.OBSERVATION(ii).Epoch.GPSsecond;
end

if
((FLAGS.Ini_time>=GPS_obs.OBSERVATION(1).Epoch.GPSsecond)&&(FLAGS.Ini_time<=GPS_obs.OBSERVATIO
N(end).Epoch.GPSsecond))
   near = min(abs(tempII-FLAGS.Ini_time));
   iimin = find ((tempII-FLAGS.Ini_time)==near);
   if isempty (iimin)
      iimin = find ((tempII-FLAGS.Ini_time)==-near);
   end

if((FLAGS.End_time>=GPS_obs.OBSERVATION(1).Epoch.GPSsecond)&&(FLAGS.End_time<=GPS_obs.OBSERVA
TION(end).Epoch.GPSsecond))
```

```
        near = min(abs(tempII-FLAGS.End_time));
        iimax = find ((tempII-FLAGS.End_time)==near);
        if isempty (iimax)
            iimax = find ((tempII-FLAGS.End_time)==-near);
        end
    else
        msgbox('Check End time in ConfigFile.', 'warn' );
    end
else
    msgbox('Check Ini time in ConfigFile.','warn' );
end

%---------------------------------- Starting the main program -----------------------------%

for i=1:num_test

    %white gaussian noise to simulate errors in the GPS pseudoranges
    awgs_sat1 = randn(5000,1);
    awgs_sat2 = randn(5000,1);
    awgs_sat3 = randn(5000,1);

    %white gaussian noise to simulate terrestrial error
    awgs_wifi = randn(5000,1);

    %START_POSITION = [0;0;0];
    jj = 1;
    kk = 1;
    first1 = 1;
    first2 = 1;
    clear enuerr_ref;
    clear enuerr_ref_norm;
    clear enuerr2_ref;
    clear enuerr2_ref_norm;

    for ii = iimin+ii_start:iimin+ii_start+ii_num  % Number of observations

        EPOCH.POSITION = [];
        EPOCH.RHOo = [];
        EPOCH.PREFIT = [];
        EPOCH.SV_accuracy = [];
        EPOCH.Elev = [];
        EPOCH.Az = [];
        C1 = [];
        START_POSITION = [0;0;0];
        clear START_POSITION2;

        OBS = GPS_obs.OBSERVATION(ii);
        Epoch = OBS.Epoch.GPSsecond;
        list(ii,1:OBS.NUM_OF_SATELLITES) = OBS.LIST_OF_PRNs;

        for m=1:2
            clear svxyzmat;
```

```
nn=1;
nn2=1;
for ww=OBS.LIST_OF_PRNs
   Epheme = findEphem(GPS_nav.PRN(ww).TOE, Epoch);
   [totgd] = Epheme.TGD;

   %%%%%%%%%%
   C1 = OBS.OBSERVABLES(nn,CA);
   TOF = C1/c;
   Tsate = Epoch - C1/c;
   [dsatel] = dtSAT(Epheme, Tsate);
   Ttrans = Tsate - dsatel;
   [POSITION, Ek] = svP(Epheme, Ttrans, FLAGS);
   [dsatel] = dtSAT(Epheme, Ttrans);
   %%%%%%%%%%

   [drelati] = dtREL(Epheme.e_Eccentricity, Epheme.sqrtA, Ek);

   if m==2
      [LLHrx, Elev, Az] = RXhorizon (POSITION, START_POSITION);
      dtropo = dtTROPO(LLHrx, Elev, OBS.Epoch.DAYofYEAR,FLAGS.c);
      diono = dtIONO(Epoch, ALPHA, BETA, LLHrx, Elev, Az);
      C1 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati-dtropo-diono)*c;
      if exist('START_POSITION2','var')
         [LLHrx, Elev, Az] = RXhorizon (POSITION, START_POSITION2);
         dtropo = dtTROPO(LLHrx, Elev, OBS.Epoch.DAYofYEAR,FLAGS.c);
         diono = dtIONO(Epoch, ALPHA, BETA, LLHrx, Elev, Az);
         C1_2 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati-dtropo-diono)*c;
      end
   else
      C1 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati)*c;
      C1_2 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati)*c;
   end

   [POSITION] = ECEFrotation(POSITION, TOF, FLAGS.wE);
   if ~isempty(POSITION)
      [LLHrx, Elev, Az] = RXhorizon (POSITION, usrxyz);
      sat_elev(nn) = Elev*180;
      if Elev*180 >= 42
         switch nn2
            case 1
               error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat1(ii);  % error for the first
satellite
            case 2
               error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat2(ii);  % error for the
second satellite
            case 3
               error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat3(ii);  % error for the third
satellite
         end
         prvec(ii,nn2) = C1 + error_GPS;
         prvec2(ii,nn2) = C1_2 + error_GPS;
```

```
            svxyzmat(nn2,1:3) = POSITION;
            sat_vec(ii,nn2) = ww;
            nn2 = nn2 + 1;
          end
          nn = nn + 1;
        end
      end

      if length(svxyzmat) <= 3  % if the number of satellites is less or equal than 3, an AP is incorporated
        error_wifi = bias_wifi + disp_wifi*awgs_wifi(ii);  % terrestial error
        prvec(ii,nn2) = range_AP + error_wifi;
        prvec2(ii,nn2) = range_AP + error_wifi;
        svxyzmat(nn2,1:3) = xyz_AP;

        numAPs = 1;  % this variable indicates if the program has to use a hybrid algorithm or not (1 or more=YES, 0=NO)
      else
        numAPs = 0;
      end

      if numAPs==0
        % calculation of position
        [START_POSITION,num_iter,pseudo,pseudo_err,estpos]= Least_squares(numAPs,prvec(ii,:),svxyzmat,START_POSITION');
        time1 = START_POSITION(4);
        START_POSITION = START_POSITION(1:3)';

        [a,b]=size(START_POSITION);
        if a<b, START_POSITION=START_POSITION';end

      else
        % execution of Bancroft's algorithm to obtain the initial positions
        [ini_pos] = Bancroft(prvec(ii,:),svxyzmat);

        % calculation of position 1
        [START_POSITION,num_iter,maxiter,pseudo,pseudo_err,estpos]= Least_squares(numAPs,prvec(ii,:),svxyzmat,ini_pos(1,1:3));
        time1 = START_POSITION(4);
        START_POSITION = START_POSITION(1:3)';

        [a,b]=size(START_POSITION);
        if a<b, START_POSITION=START_POSITION';end

        % calculation of  position 2
        [START_POSITION2,num_iter2,maxiter2,pseudo2,pseudo_err2,estpos2]= Least_squares(numAPs,prvec2(ii,:),svxyzmat,ini_pos(2,1:3));
        time2 = START_POSITION2(4);
        START_POSITION2 = START_POSITION2(1:3)';

        [a,b]=size(START_POSITION2);
        if a<b, START_POSITION2=START_POSITION2';end
```

```
        end

    end

    estusr=START_POSITION;
    enuerr(ii,1:3) = ( xyz2enu(estusr(1:3),usrxyz) )';    %positioning error
    enuerr(ii,4) = time1;  %error corresponding to the receiver clock drift
    if norm(enuerr(ii,4)-enuerr(ii-1,4))<50 %filtration of big clock errors
       if num_iter ~= maxiter %the algorithm converge
          if first1 == 1
             ref_pos1 = START_POSITION;
             first1 = 0;
          else
             xyz_pos(jj,1:3) = estusr(1:3);  %we save the estimated positions
             enuerr_pos(jj,1:3) = ( xyz2enu(estusr(1:3),usrxyz) )';  %positioning error after filtering
             euclidean_error(jj) = norm(enuerr_pos(jj,1:3));
             enuerr_ref(jj,1:3) = ( xyz2enu(estusr(1:3),ref_pos1) )';  %positioning error referred to estimated
position 1
             enuerr_ref_norm(jj) = norm(enuerr_ref(jj,1:3));  %distance to the ref_pos1
             jj = jj + 1;
          end
       end
    end

    if exist('START_POSITION2','var')
       estusr2=START_POSITION2;
       enuerr2(ii,1:3) = ( xyz2enu(estusr2(1:3),usrxyz) )';    %positioning error
       enuerr2(ii,4) = time2;  %error corresponding to the receiver clock drift
       if norm(enuerr2(ii,4)-enuerr2(ii-1,4))<50 %filtration of big clock errors
          if num_iter2 ~= maxiter2 %the algorithm converge
             if first2 == 1
                ref_pos2 = START_POSITION2;
                first2 = 0;
             else
                xyz_pos2(kk,1:3) = estusr2(1:3); %we save the estimated positions
                enuerr2_pos(kk,1:3) = ( xyz2enu(estusr2(1:3),usrxyz) )'; %positioning error after filtering
                euclidean_error2(kk) = norm(enuerr2_pos(kk,1:3));
                enuerr2_ref(kk,1:3) = ( xyz2enu(estusr2(1:3),ref_pos2) )'; %positioning error referred to
estimated position 2
                enuerr2_ref_norm(kk) = norm(enuerr2_ref(kk,1:3)); %distance to the ref_pos2
                kk = kk + 1;
             end
          end
       end
    end

 end

 if jj==1 || kk==1
    disp('An initial position cannot be calculated');
 else
    y=1:jj-1;
```

```
    z=1:kk-1;

    % we apply polyfit to express the errors like a line
    coeff1 = polyfit(y,enuerr_ref_norm(y),1);
    m1 = coeff1(1);
    b1 = coeff1(2);
    y1 = m1*y + b1;

    coeff2 = polyfit(z,enuerr2_ref_norm(z),1);
    m2 = coeff2(1);
    b2 = coeff2(2);
    y2 = m2*z + b2;

    if abs(m1) < abs(m2)
       disp('The correct position is solution 1 (blue)');
       sol_1 = sol_1 + 1;
    else
       disp('The correct position is solution 2 (green)');
       sol_2 = sol_2 + 1;
    end
  end

end

disp('Solutions 1 (blue):'); disp(sol_1);
disp('Solutions 2 (green):'); disp(sol_2);


%---------------------------------------- PLOTS ----------------------------------------%

scrsz = get(0,'ScreenSize');

figure(1)
plot(enuerr(:,1),enuerr(:,2),'b*',enuerr2(:,1),enuerr2(:,2),'g*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')

figure(2)
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr(:,1),enuerr(:,2),enuerr(:,3),'b*',enuerr2(:,1),enuerr2(:,2),enuerr2(:,3),'g*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
ylabel('North (m)')
xlabel('East (m)')
```

```
figure(3)
set(3,'Position',[1 1 scrsz(3)/2.1 scrsz(4)/2.4]);
plot(enuerr_pos(:,1),enuerr_pos(:,2),'b*',enuerr2_pos(:,1),enuerr2_pos(:,2),'g*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')

figure(4)
set(4,'Position',[1  scrsz(4)/2 scrsz(3)/2.1 scrsz(4)/2.4]);
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr_pos(:,1),enuerr_pos(:,2),enuerr_pos(:,3),'b*',enuerr2_pos(:,1),enuerr2_pos(:,2),enuerr2_pos(:,3),
'g*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
ylabel('North (m)')
xlabel('East (m)')

figure(5)
set(5,'Position',[scrsz(3)/2  scrsz(4)/2 scrsz(3)/2.1 scrsz(4)/2.4]);
plot(y,enuerr_ref_norm(y),'b-',z,enuerr2_ref_norm(z),'g-',y,m1*y+b1,'k-',z,m2*z+b2,'r-');
xlabel('Epoch');
ylabel('Distance');
title('Distance to first positions depending on time');
```

APPENDIX 2

The code of the positioning error test program is presented below. In the first part of the code, range and pseudorange errors are created, AP location is selected and the range distance without error is fixed. In the second part, it is set the number of tests, the epoch to start and the number of epochs to calculate. In the third part, the main program is executed. For each epoch, the errors are incorporated to the calculated pseudorange and simulated range. After that, the program calculates the position with Least Squares. To know if the computed position belongs to the correct solution or not, Bancroft's algorithm is executed and both solutions are used as initial positions of the Least Squares algorithm, thus obtaining both candidate solutions. The position calculated with only Least Squares and both candidate solutions are filtered. If all three positions exist, the position calculated with only Least Squares is compared to each candidate solution and it is decided if it belongs to the correct solution or not. In the last part, different figures are plotted in order to present the results.

```
%---------- Configuration of the range and pseudorange errors ----------%

%CTAE antenna position
usrxyz = [4794682.4427 169625.5396 4188738.7470];
usrllh = xyz2llh(usrxyz);

dist_AP = 50;
% Setting APs parameters
enu_AP1 = [-dist_AP 0 0];
xyz_AP1 = enu2xyz(enu_AP1,usrxyz);
enu_AP2 = [0 -dist_AP 0];
xyz_AP2 = enu2xyz(enu_AP2,usrxyz);
enu_AP3 = [0 dist_AP 0];
xyz_AP3 = enu2xyz(enu_AP3,usrxyz);
enu_AP4 = [dist_AP 0 0];
xyz_AP4 = enu2xyz(enu_AP4,usrxyz);

%selection of the AP to test
enu_AP = enu_AP3;
xyz_AP = enu2xyz(enu_AP,usrxyz)';
range_AP = norm(enu_AP);

%1 bias for every satellite
bias_GPS = zeros(1,32);
bias_GPS(4) = 3;
bias_GPS(11) = 20;
bias_GPS(13) = 9;
bias_GPS(17) = 10;
bias_GPS(20) = 7;
bias_GPS(23) = 5;
bias_GPS(25) = 17;
bias_GPS(31) = 8;
```

```
bias_GPS(32) = 6;

%dispersion of GPS satellites
disp_GPS = 0.1;

%terrestrial error
bias_wifi = 0;
disp_wifi = 10;

%---------- Initial parameters to configure the main test program ----------%

num_test = 20;
sol_1 = 0;
sol_2 = 0;
total_sol = 0;

ii_start = 2360;
ii_num = 240;

c = FLAGS.c;
MINelev = FLAGS.ANGth;

% Catch Alpha & Beta. Needed to apply the Klobuchar model
[ALPHA, BETA] = getAlphaBeta(HEADER_nav);

% Find the index of the C1 parameter in the Observables matrix.
CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C1'));
if isempty(CA), CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C1C')); end
if isempty(CA), CA = find (strcmp(HEADER_obs.TYPES_OF_OBSERVABLES.TYPES , 'C5a')); end


% Setting Initial and Ending time range for observations
if isempty(FLAGS.Ini_time)
   FLAGS.Ini_time = GPS_obs.OBSERVATION(1).Epoch.GPSsecond;
end
if isempty(FLAGS.End_time)
   FLAGS.End_time = GPS_obs.OBSERVATION(end).Epoch.GPSsecond;
end

for ii=1:length(GPS_obs.OBSERVATION)
   tempII(ii)=GPS_obs.OBSERVATION(ii).Epoch.GPSsecond;
end

if
((FLAGS.Ini_time>=GPS_obs.OBSERVATION(1).Epoch.GPSsecond)&&(FLAGS.Ini_time<=GPS_obs.OBSERVATIO
N(end).Epoch.GPSsecond))
   near = min(abs(tempII-FLAGS.Ini_time));
   iimin = find ((tempII-FLAGS.Ini_time)==near);
   if isempty (iimin)
      iimin = find ((tempII-FLAGS.Ini_time)==-near);
   end
```

```
if((FLAGS.End_time>=GPS_obs.OBSERVATION(1).Epoch.GPSsecond)&&(FLAGS.End_time<=GPS_obs.OBSERVA
TION(end).Epoch.GPSsecond))
      near = min(abs(tempII-FLAGS.End_time));
      iimax = find ((tempII-FLAGS.End_time)==near);
      if isempty (iimax)
         iimax = find ((tempII-FLAGS.End_time)==-near);
      end
   else
      msgbox('Check End time in ConfigFile.', 'warn' );
   end
else
   msgbox('Check Ini time in ConfigFile.','warn' );
end

jj = 1;
kk = 1;
ll = 1;
mm = 1;

%---------------- Starting the main program -----------------%


for i=1:num_test

   %white gaussian noise to simulate errors in the GPS pseudoranges
   awgs_sat1 = randn(5000,1);
   awgs_sat2 = randn(5000,1);
   awgs_sat3 = randn(5000,1);

   %white gaussian noise to simulate terrestrial error
   awgs_wifi = randn(5000,1);

   START_POSITION_LS = [0;0;0];
   partial_sol_1 = 0;
   partial_sol_2 = 0;
   partial_solutions = 0;

   for ii = iimin+ii_start:iimin+ii_start+ii_num  % Number of observations

      EPOCH.POSITION = [];
      EPOCH.RHOo = [];
      EPOCH.PREFIT = [];
      EPOCH.SV_accuracy = [];
      EPOCH.Elev = [];
      EPOCH.Az = [];
      C1 = [];

      clear START_POSITION2;
      clear estusr_pos_LS;
      clear estusr_pos;
      clear estusr_pos2;
```

```
OBS = GPS_obs.OBSERVATION(ii);
Epoch = OBS.Epoch.GPSsecond;
list(ii,1:OBS.NUM_OF_SATELLITES) = OBS.LIST_OF_PRNs;

for m=1:2
   clear svxyzmat;
   nn=1;
   nn2=1;
   for ww=OBS.LIST_OF_PRNs
      Epheme = findEphem(GPS_nav.PRN(ww).TOE, Epoch);
      [totgd] = Epheme.TGD;

      %%%%%%%%%%
      C1 = OBS.OBSERVABLES(nn,CA);
      TOF = C1/c;
      Tsate = Epoch - C1/c;
      [dsatel] = dtSAT(Epheme, Tsate);
      Ttrans = Tsate - dsatel;
      [POSITION, Ek] = svP(Epheme, Ttrans, FLAGS);
      [dsatel] = dtSAT(Epheme, Ttrans);
      %%%%%%%%%%

      [drelati] = dtREL(Epheme.e_Eccentricity, Epheme.sqrtA, Ek);

      if m==2
         [LLHrx, Elev, Az] = RXhorizon (POSITION, START_POSITION_LS);
         dtropo = dtTROPO(LLHrx, Elev, OBS.Epoch.DAYofYEAR,FLAGS.c);
         diono = dtIONO(Epoch, ALPHA, BETA, LLHrx, Elev, Az);
         C1_LS = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati-dtropo-diono)*c;
         if exist('START_POSITION','var') && exist('START_POSITION2','var')
            [LLHrx, Elev, Az] = RXhorizon (POSITION, START_POSITION);
            dtropo = dtTROPO(LLHrx, Elev, OBS.Epoch.DAYofYEAR,FLAGS.c);
            diono = dtIONO(Epoch, ALPHA, BETA, LLHrx, Elev, Az);
            C1 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati-dtropo-diono)*c;

            [LLHrx, Elev, Az] = RXhorizon (POSITION, START_POSITION2);
            dtropo = dtTROPO(LLHrx, Elev, OBS.Epoch.DAYofYEAR,FLAGS.c);
            diono = dtIONO(Epoch, ALPHA, BETA, LLHrx, Elev, Az);
            C1_2 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati-dtropo-diono)*c;
         end
      else
         C1_LS = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati)*c;
         C1 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati)*c;
         C1_2 = OBS.OBSERVABLES(nn,CA) + (dsatel-totgd+drelati)*c;
      end

      [POSITION] = ECEFrotation(POSITION, TOF, FLAGS.wE);
      if ~isempty(POSITION)
         [LLHrx, Elev, Az] = RXhorizon (POSITION, usrxyz);
         sat_elev(nn) = Elev*180;
         if Elev*180 >= 42
```

```
                switch nn2
                  case 1
                    error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat1(ii);  % error for the first
satellite
                  case 2
                    error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat2(ii);  % error for the
second satellite
                  case 3
                    error_GPS = bias_GPS(ww) + disp_GPS*bias_GPS(ww)*awgs_sat3(ii);  % error for the third
satellite
                end
                prvec_LS(ii,nn2) = C1_LS + error_GPS;
                prvec(ii,nn2) = C1 + error_GPS;
                prvec2(ii,nn2) = C1_2 + error_GPS;
                svxyzmat(nn2,1:3) = POSITION;
                sat_vec(ii,nn2) = ww;
                nn2 = nn2 + 1;
              end
              nn = nn + 1;
            end
          end

          if length(svxyzmat) <= 3  % if the number of satellites is less or equal than 3, an AP is incorporated
            error_wifi = bias_wifi + disp_wifi*awgs_wifi(ii);  % terrestrial error
            prvec_LS(ii,nn2) = range_AP + error_wifi;
            prvec(ii,nn2) = range_AP + error_wifi;
            prvec2(ii,nn2) = range_AP + error_wifi;
            svxyzmat(nn2,1:3) = xyz_AP;

            numAPs = 1;  % this variable indicates if the program has to use a hybrid algorithm or not (1 or
more=YES, 0=NO)
          else
            numAPs = 0;
          end

          % calculation of position with LS only
          [START_POSITION_LS,num_iter_LS,maxiter_LS,pseudo_LS,pseudo_err_LS,estpos_LS] =
olspos_wifi_weighted(numAPs,prvec_LS(ii,:),svxyzmat,START_POSITION_LS');
          time_LS = START_POSITION_LS(4);
          START_POSITION_LS = START_POSITION_LS(1:3)';

          [a,b]=size(START_POSITION_LS);
          if a<b, START_POSITION_LS=START_POSITION_LS';end

          if numAPs ~= 0
            % execution of Bancroft's algorithm to obtain the initial positions
            [ini_pos] = Bancroft(prvec(ii,:),svxyzmat);

            % calculation of position 1
            [START_POSITION,num_iter,maxiter,pseudo,pseudo_err,estpos] =
olspos_wifi_weighted(numAPs,prvec(ii,:),svxyzmat,ini_pos(1,1:3));
            time1 = START_POSITION(4);
```

```
        START_POSITION = START_POSITION(1:3)';

        [a,b]=size(START_POSITION);
        if a<b, START_POSITION=START_POSITION';end

        % calculation of  position 2
        [START_POSITION2,num_iter2,maxiter2,pseudo2,pseudo_err2,estpos2] =
olspos_wifi_weighted(numAPs,prvec2(ii,:),svxyzmat,ini_pos(2,1:3));
        time2 = START_POSITION2(4);
        START_POSITION2 = START_POSITION2(1:3)';

        [a,b]=size(START_POSITION2);
        if a<b, START_POSITION2=START_POSITION2';end
      end
    end

    estusr_LS = START_POSITION_LS;
    enuerr_LS(ii,1:3) = ( xyz2enu(estusr_LS(1:3),usrxyz) )';    %positioning error
    enuerr_LS(ii,4) = time_LS;  %error corresponding to the receiver clock drift
    if norm(enuerr_LS(ii,4)-enuerr_LS(ii-1,4))<50 %filtration of big clock errors
      if num_iter_LS ~= maxiter_LS %the algorithm converge
        estusr_pos_LS = estusr_LS;
        xyz_pos_LS(ll,1:3) = estusr_LS(1:3); %we save the estimated positions
        enuerr_pos_LS(ll,1:3) = ( xyz2enu(estusr_LS(1:3),usrxyz) )'; %positioning error after filtering
        euclidean_error_LS(ll) = norm(enuerr_pos_LS(ll,1:3));
        HPE_LS(ll) = norm(enuerr_pos_LS(ll,1:2));
        VPE_LS(ll) = abs(enuerr_pos_LS(ll,3));
        ll = ll + 1;
      end
    end

    if numAPs ~= 0
      estusr=START_POSITION;
      enuerr(ii,1:3) = ( xyz2enu(estusr(1:3),usrxyz) )';    %positioning error
      enuerr(ii,4) = time1;  %error corresponding to the receiver clock drift
      if norm(enuerr(ii,4)-enuerr(ii-1,4))<50 %filtration of big clock errors
        if num_iter ~= maxiter %the algorithm converge
          estusr_pos = estusr;
          xyz_pos(jj,1:3) = estusr(1:3); %we save the estimated positions
          enuerr_pos(jj,1:3) = ( xyz2enu(estusr(1:3),usrxyz) )'; %positioning error after filtering
          euclidean_error(jj) = norm(enuerr_pos(jj,1:3));
          HPE(jj) = norm(enuerr_pos(jj,1:2));
          VPE(jj) = abs(enuerr_pos(jj,3));
          jj = jj + 1;
        end
      end


      estusr2=START_POSITION2;
      enuerr2(ii,1:3) = ( xyz2enu(estusr2(1:3),usrxyz) )';    %positioning error
      enuerr2(ii,4) = time2;  %error corresponding to the receiver clock drift
      if norm(enuerr2(ii,4)-enuerr2(ii-1,4))<50 %filtration of big clock errors
```

```
            if num_iter2 ~= maxiter2 %the algorithm converge
               estusr_pos2 = estusr2;
               xyz_pos2(kk,1:3) = estusr2(1:3); %we save the estimated positions
               enuerr2_pos(kk,1:3) = ( xyz2enu(estusr2(1:3),usrxyz) )'; %positioning error after filtering
               euclidean_error2(kk) = norm(enuerr2_pos(kk,1:3));
               HPE2(kk) = norm(enuerr2_pos(kk,1:2));
               VPE2(kk) = abs(enuerr2_pos(kk,3));
               kk = kk + 1;
            end
         end

         if exist('estusr_pos_LS','var') && exist('estusr_pos','var') && exist('estusr_pos2','var')
            err_to_sol1(mm) = norm(estusr_pos_LS-estusr_pos);
            err_to_sol2(mm) = norm(estusr_pos_LS-estusr_pos2);
            if err_to_sol1(mm) < err_to_sol2(mm)
               partial_sol_1 = partial_sol_1 + 1;
               sol_1 = sol_1 + 1;
            else
               partial_sol_2 = partial_sol_2 + 1;
               sol_2 = sol_2 + 1;
            end
            partial_solutions = partial_solutions + 1;
            total_sol = total_sol + 1;
            mm = mm + 1;
         end
      end

   end
   disp('Percentage of solutions 1 (blue):');disp(partial_sol_1/partial_solutions);
   disp('Percentage of solutions 2 (green):');disp(partial_sol_2/partial_solutions);
end

disp('Solutions 1 (blue):'); disp(sol_1/total_sol);
disp('Solutions 2 (green):'); disp(sol_2/total_sol);


%------------------------- PLOTS ----------------------------%

scrsz = get(0,'ScreenSize');

figure(1)
plot(enuerr(:,1),enuerr(:,2),'b*',enuerr2(:,1),enuerr2(:,2),'g*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')
```

```
figure(2)
plot(enuerr_LS(:,1),enuerr_LS(:,2),'b*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')

figure(3)
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr(:,1),enuerr(:,2),enuerr(:,3),'b*',enuerr2(:,1),enuerr2(:,2),enuerr2(:,3),'g*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
ylabel('North (m)')
xlabel('East (m)')

figure(4)
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr_LS(:,1),enuerr_LS(:,2),enuerr_LS(:,3),'b*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
ylabel('North (m)')
xlabel('East (m)')

figure(5)
set(5,'Position',[1 1 scrsz(3)/2.1 scrsz(4)/2.4]);
plot(enuerr_pos(:,1),enuerr_pos(:,2),'b*',enuerr2_pos(:,1),enuerr2_pos(:,2),'g*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')

figure(6)
set(6,'Position',[1  scrsz(4)/2 scrsz(3)/2.1 scrsz(4)/2.4]);
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr_pos(:,1),enuerr_pos(:,2),enuerr_pos(:,3),'b*',enuerr2_pos(:,1),enuerr2_pos(:,2),enuerr2_pos(:,3),
'g*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
```

```
ylabel('North (m)')
xlabel('East (m)')

figure(7)
set(7,'Position',[scrsz(3)/2 1 scrsz(3)/2.1 scrsz(4)/2.4]);
plot(enuerr_pos_LS(:,1),enuerr_pos_LS(:,2),'b*')
hold
plot (0,0,'rp');
axis('equal')
axis('square')
grid
title('GPS ENU Plotting Error (Static User)')
ylabel('North (m)')
xlabel('East (m)')

figure(8)
set(8,'Position',[scrsz(3)/2  scrsz(4)/2 scrsz(3)/2.1 scrsz(4)/2.4]);
hold on
plot_sphere(range_AP,enu_AP(1),enu_AP(2),enu_AP(3),20);
plot3(enuerr_pos_LS(:,1),enuerr_pos_LS(:,2),enuerr_pos_LS(:,3),'b*',0,0,0,'rp');
hold off
title('ENU Error in 3D');
zlabel('Up (m)')
ylabel('North (m)')
xlabel('East (m)')

figure(9)
subplot(3,1,1);
hist(euclidean_error,100);
title('Euclidean Error Distribution (Solution 1)');
ylabel('Percentage')
xlabel('Error (m)')
subplot(3,1,2);
hist(euclidean_error2,100);
title('Euclidean Error Distribution (Solution 2)');
ylabel('Percentage')
xlabel('Error (m)')
subplot(3,1,3);
hist(euclidean_error_LS,100);
title('Euclidean Error Distribution (LS Solution)');
ylabel('Percentage')
xlabel('Error (m)')

figure(10)
subplot(3,1,1);
cdfplot(HPE,'Empirical CDF of HPE (Solution 1)');
subplot(3,1,2);
cdfplot(HPE2,'Empirical CDF of HPE (Solution 2)');
subplot(3,1,3);
cdfplot(HPE_LS,'Empirical CDF of HPE (LS Solution)');
```

```
figure(11)
subplot(3,1,1);
cdfplot(VPE,'Empirical CDF of VPE (Solution 1)');
subplot(3,1,2);
cdfplot(VPE2,'Empirical CDF of VPE (Solution 2)');
subplot(3,1,3);
cdfplot(VPE_LS,'Empirical CDF of VPE (LS Solution)');
```

BIBLIOGRAPHY

[1] J.M. Stone and J.D. Powell, "Precise positioning with GPS near obstructions by augmentation with pseudolites," *Position Location and Navigation Symposium, IEEE 1998*, 1998, págs. 562-569.

[2] F. Barceló Arroyo and I. Martí Escalona, "Coverage of Hybrid Terrestrial-Satellite Location in Mobile Communications," *Proceedings of the European Wireless (EW2004): Mobile and Wireless Systems beyond 3G*, 2004.

[3] J. A. Lopez-Salcedo, Y. Capelle, M. Toledo, G. Seco-Granados, J. Vicario, D. Kubrak, M. Monnerat, and A. Mark, "DINGPOS: A Hybrid Indoor Navigation Platform for GPS and GALILEO," *ION GNSS*, Savannah (USA): 2008.

[4] G. Dona, G. Burden and S. Ingram, "Combining UWB and GNSS Positioning to Extend Location of Emergency Personnel in Forest Fire Environments," *ENC-GNSS (European Navigation Conference - Global Navigation Satellite Systems) 2009*, Naples, Italy: 2009.

[5] G. Macgougan, K. O'Keefe and R. Klukas, "Tightly-coupled GPS/UWB positioning," *Ultra-Wideband, 2009. ICUWB 2009. IEEE International Conference on*, 2009, págs. 381-385.

[6] Ju-Yong Do, M. Rabinowitz and P. Enge, "Performance of Hybrid Positioning System Combining GPS and Television Signals," *Position, Location, And Navigation Symposium, 2006 IEEE/ION*, 2006, págs. 556-564.

[7] D. Bonacci, W. Chauvet, P. Paimblanc and F. Castanié, "Improving vehicles positioning using wireless telecommunication media and GNSS hybridization," *IEEE International Workshop on Intelligent Transportation*, 2008.

[8] E. D. Kaplan and C. Hegarty, *Understanding GPS - Principles and Applications*, Norwood: Artech House, 1996.

[9] S.A. Golden and S.S. Bateman, "Sensor Measurements for Wi-Fi Location with Emphasis on Time-of-Arrival Ranging," *Mobile Computing, IEEE Transactions on*, vol. 6, 2007, págs. 1185-1198.

[10] V. Abhayawardhana, I. Wassell, D. Crosby, M. Sellars and M. Brown, "Comparison of empirical propagation path loss models for fixed wireless access systems," *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, 2005, págs. 73-77 Vol. 1.

[11] L. Liechty, E. Reifsnider and G. Durgin, "Developing the Best 2.4 GHz Propagation Model from Active Network Measurements," *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, 2007, págs. 894-896.

[12] C. Laurendeau and M. Barbeau, "Malicious Node Position Bounding in Mobile WiFi/802.11 Networks," *Technical Report TR-08-04, School of Computer Science*, Carleton University: 2008.

[13] S. Mazuelas, A. Bahillo, R. Lorenzo, P. Fernandez, F. Lago, E. Garcia, J. Blas and E. Abril, "Robust Indoor Positioning Provided by Real-Time RSSI Values in Unmodified WLAN Networks," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 3, 2009, págs. 821-831.

[14] K. Madsen, H. B. Nielsen and O. Tingleff, *Methods for Non-Linear Least Squares Problems*, Technical University of Denmark: Informatics and Mathematical Modelling, 2004.

[15] W.S. Murphy, "Determination of a Position Using Approximate Distances and Trilateration," Colorado School of Mines, 2007.

[16] S. Bancroft, "An Algebraic Solution of the GPS Equations," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. AES-21, 1985, págs. 56-59.

[17] I. J. Fernandez-Corbaton, A. H. Vayanos, P. A. Agashe and S. S. Soliman, "Method and apparatus for determining an algebraic solution to GPS terrestrial hybrid location system

equations," U.S. Patent 6289280, Septiembre 11, 2001.

[18] J. Abel and J. Chaffee, "Existence and uniqueness of GPS solutions," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 27, 1991, págs. 952-956.

[19] N. Kubo and A. Yasuda, "Reduction of Pseudorange Multipath Error in Static," *Proceedings of the 15th International Technical Meeting of the Satellite Division of the Institute of Navigation ION GPS*, Oregon: 2002, págs. 452-459.