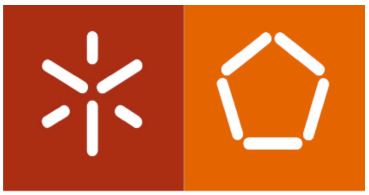

Universidade do Minho
Escola de Engenharia

João António da Cruz Dias

Indoor Positioning System with Android

Master Thesis
Informática Industrial / Mestrado Integrado em
Engenharia Electrónica Industrial e Computadores

Supervisors:
Professor Dr. Sérgio Lopes
Professor Dr. Sérgio Monteiro

January 2016

Acknowledgments

Throughout my academic journey, I made very good friends and met fantastic people that helped my development as a person and as student.

To my supervisors, Professor Dr. Sérgio Lopes and Professor Dr. Sérgio Monteiro, I want to thank for the continuous support, belief and commitment throughout this project. Their incredible devotion and knowledge was determinant to this thesis outcome.

To My parents, that always supported and believed in me, I want to thank for everything that they did, all the love and support throughout my entire life and the endless energy spent to shape me as a person.

To my girlfriend Ana Clara Queirós, that always motivated me, supported me and believed in me, I want to thank for all the patience and love that she devoted to me throughout this entire thesis and the strength that she gave me to beat all the obstacles.

To my friends, I wanna thank all the strength, all the kind words and all the motivation that they gave me unconditionally. A special thanks to my friend João Neto that helped during the dataset collection.

Last but not the least, I want to thank all the teachers and all the people that contributed to my education throughout this amazing journey.

Huge thanks to all.

This page was intencionally left blank!

Abstract

In this thesis, the indoor positioning problem is approached using an Android platform. This approach is based in tests with a set of algorithms and techniques.

The performance of each algorithm and technique, in an indoor environment, is measured using data from a dataset of a known location mapped with a smartphone in a static and dynamic context, enabling the possibility to perform a wide range of tests and get useful insights.

This thesis approach uses a developed dataset to test the performance of a set of algorithms in different contexts using the Android platform. The dataset has information relative to Wi-Fi signal strength, acceleration and magnetic field strength in different locations inside the room.

Algorithms were tested using the useful information provided by the dataset. After those tests, it was concluded that fingerprinting is more suitable than multilateration indoors since multilateration can become unstable due its need of a distance estimation. The algorithm that returned better results was the Minimum Mean Square Error (MMSE) algorithm. Using this algorithm, a functional Android indoor positioning application was built.

Keywords: Positioning systems, indoor, Android, fingerprinting, multilateration

This page was intencionally left blank!

Resumo

Nesta dissertação é abordado o problema de posicionamento em interiores usando a plataforma Android. Esta abordagem é baseada em testes de um grupo de algoritmos e técnicas.

Em interiores, a performance de cada algoritmo e técnica é medida usando dados provenientes de um *dataset*. Este *dataset* diz respeito a uma área conhecida mapeada com um *smartphone* em contexto estático e dinâmico, de forma a abrir a possibilidade de efetuar uma diferente variedade de testes e obter várias estatisticas de interesse.

Nesta dissertação, é desenvolvido um *dataset* que é usado como base para testar o performance de um grupo de algoritmos em diferentes contextos, usando para tal a plataforma Android. Este *dataset* tem informação relativa à intensidade do sinal Wi-Fi, aceleração e campos magnéticos em diferentes posições da sala.

Os algoritmos foram testados usando o *dataset* desenvolvido. Depois dos testes efetuados, concluíu-se que o fingerprinting é mais adequado que a multilateração para efeitos de posicionamento indoor, isto porque a multilateração pode ficar instável devido á sua dependência de uma estimativa da distância. O algoritmo com melhores resultados foi o MMSE. Usando este algoritmo, uma aplicação Android para posicionamento em interiores foi desenvolvida.

Palavras Chave: Sistemas de Posicionamento em interiores, Android, fingerpinting, multilateração

This page was intencionally left blank!

Abbreviations and Acronyms

AOA	*Angle of Arrival*
API	*Application Programming Interface*
APx	*Access Point x*
BLE	*Bluetooth Low Energy*
dBm	*Decibel miliwatt.*
FSPL	*Free Space Path Loss*
GPS	*Global Positioning System*
IDE	*Integrated Development Environment*
IPS	*Indoor Positioning System*
LED	*Light-Emitting Diode*
MAC	*media access control address*
OS	*Operative System*
RP	*Reference Position*
RSS	*Received Signal Strength*
RSSI	*Received Signal Strength Indicator*

RTI	*Radio Tomographic Imaging*
TP	*Test Position*
VLC	*Visible Light Communication*

Contents

This page was intencionally left blank!

List of Figures

This page was intencionally left blank!

List of Tables

This page was intencionally left blank!

List of Algorithms

This page was intencionally left blank!

Chapter 1

Motivation & Objectives

After Global Positioning System (GPS) came out, there was a need to improve its capabilities to work in indoor environments, because it revealed lack of accuracy when used inside buildings. GPS systems are based in satellite broadcast information. This information is encrypted and continuously broadcast to planet Earth by strategically placed satellites. After the information is collected by the receiver, it is decrypted and used to identify data such as emitter satellite and timestamps. Using these timestamps, the receptor is able to make some calculations in order to get the propagation delay and consequently the distance between the satellite and the receptor.This strategy requires 4 satellites to achieve a thustworthy position estimation. GPS signals reach the Earth with very low power levels, leading to even lower levels, or even no signal, on indoor environments. Therefore, it is necessary to build systems that are able to give good position estimations inside these environments. Indoor Positioning Systems (IPS) appeared to address this problem [1, 2].

IPSs are nowadays a big topic of discussion, because of its usability and potential due to the increasing number of smartphones all over the world. Actual smartphones have built-in sensors that can be used in IPSs. They have accelerometers, gyroscopes, magnetometers and Wi-Fi transceivers. Signal strength information can be

collected from Wi-Fi transceivers and then used to try to estimate a position using techniques such as multilateration or fingerprinting. Accelerometers can be used to estimate position using motion detection, step detection and linear acceleration. Magnetometers and gyroscopes can be used to detect orientation, although they both have advantages and disadvantages related to each other. Magnetometer have long term stable accuracy, however their output can become unpredictable when affected by external disturbances. On the other hand, gyroscopes have a short term accuracy but their output is not affected by external disturbances [3].

The output of these sensors together with a selected set of location techniques are the core of IPSs.

IPS have application scenarios such as transportation hubs, shopping malls, convention centers, university campuses, hospitals, office buildings, among others. These systems can be used not only to present the user with an estimated position but also in context aware applications. As an example, they can be used to present the visitor of a store inside a shopping mall with available offers in real time.

The performance of the two most important techniques is tested with developed Android applications. These techniques are multilateration and fingerprinting and they are used in other applications [4].

This thesis objectives are the following:

1. Study Indoor Positioning techniques;
2. Implement solutions already present in literature;
3. Develop an application that can give precise position with error levels similar to the existing technology and if possible better;

1.1 Contributions

In this thesis, an Android indoor positioning system approach based on Wi-Fi Received Signal Strength (RSS) is presented. This approach is based on the result of a set of tests of existing methodologies. The developed approach is delivered not only as the results present in this document but also as a functional Android application capable of positioning estimation in indoor environments.

A dataset was built. This dataset features multiple readings inside a room with 100 square meters in a static and dynamic context, respectively. Those readings are not only from Wi-Fi hotspots, but also from the magnetic sensor and accelerometer inside the smartphone. The dataset is not exclusive for this work, so it can be used to test techniques and algorithms in future researches.

This thesis also features an analysis to the collected accelerometer readings that may help future researchers to ahchieve a sensor fusion based indoor positioning system.

1.2 Thesis Organization

This thesis is divided into 6 chapters.

Chapter 1 introduces the motivation and presents the pre-defined objectives for this work. Chapter 2 presents an analysis over existing indoor positioning technologies, techniques, algorithms and applications. In chapter 3, a set of developed methods is described. These methods consist of the dataset building and a deeper analysis on the two indoor positioning techniques approached: fingerprinting and multilateration.

Chapter 4 describes the used tools and developed Android applications, including the final indoor positioning application and applications used to build the datasets. In chapter 5, all the results obtained are presented. It includes results related to the dataset, algorithms, and sensors.

Finally in chapter 6, a set of conclusions is exposed alongside with work suggestions for future researchers that want to use this work as a base to their own.

Chapter 2

Background

This chapter presents positioning technologies, techniques and algorithms used in other IPSs. Two of those techniques will be tested: multilateration and fingerprinting. Both of them make use of Wi-Fi technology to estimate the smartphone position. In section 2.3, positioning algorithms are discussed. Those algorithms are divided regarding the technique that uses them, either multilateration or fingerprinting. There are different algorithms for the fingerprinting technique, since it can be approached using two different kinds of algorithms: deterministic and probabilistic algorithms. A set of deterministic and probabilistic algorithms are presented and a brief discussion on all the possibilities is made. On the other hand, multilateration is itself an algorithm, so a multilateration algorithm is discussed.

2.1 Positioning Technologies

A number of technologies can be used to develop positioning systems, because they produce and measure physical phenomena that can be mathematically related to a position. This section describes the ones that are widely used in other applications.

2.1.1 Bluetooth

Bluetooth operates in the band of 2.4 to 2.485 GHz [5].

Consists of a Radio frequency (RF) transceiver, baseband, and protocol stack. The system offers services that enable the connection of devices and exchange of data between them [5].

Apple's iBeacon protocol and Google's Eddystone are based in the latest version of Bluetooth, Bluetooth Low Energy (BLE). These devices continuously broadcast information that is received by a smartphone. Indoor positioning can be achieved using collected signal strength from multiple beacons inside a building and applying positioning algorithms to estimate a position. These devices are widely used in a variety of context aware applications [6]. In figure 2.1 a setup of a Bluetooth based application is represented.

This technology can provide acceptable values in what comes to precision, however there is the need to install extra hardware inside the building (Bluetooth beacons).

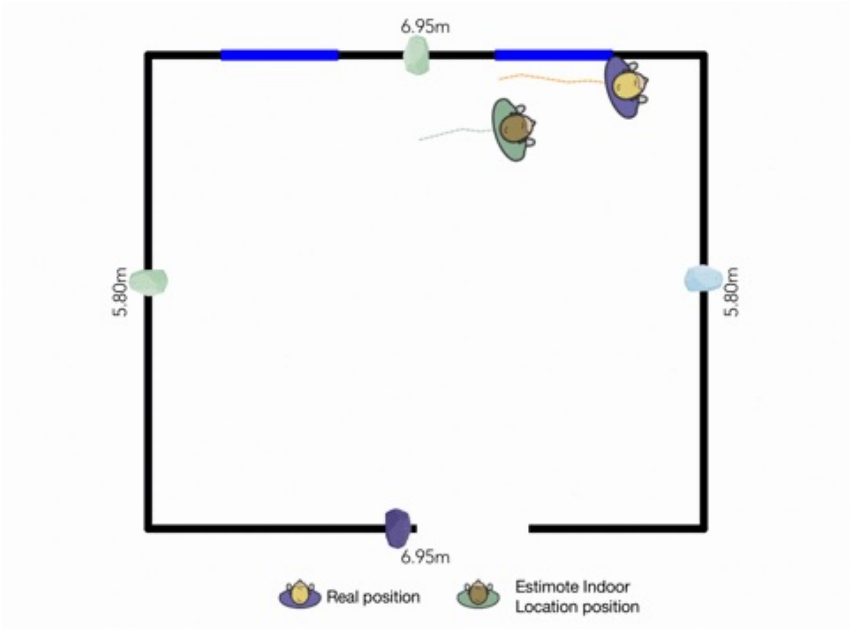

Figure 2.1: Indoor Positioning using BLE beacons. (Source: [7])

2.1.2 Wi-Fi

Wi-Fi works with no physical wires by using radio frequency (RF). This way information can be exchanged between the transmitter and the receiver [8].

This technology is present in most of houses and buildings which makes it interesting for use in IPSs because there is no need to build extra physical infrastructures. Wi-Fi access points (APs) have a unique MAC address that identifies them and alongside with their signal strength it is possible to attempt a position estimation using techniques like multilateration and fingerprinting.

In figure 2.2 is an example of a fingerprint based sytem setup with two fingerprints represented.

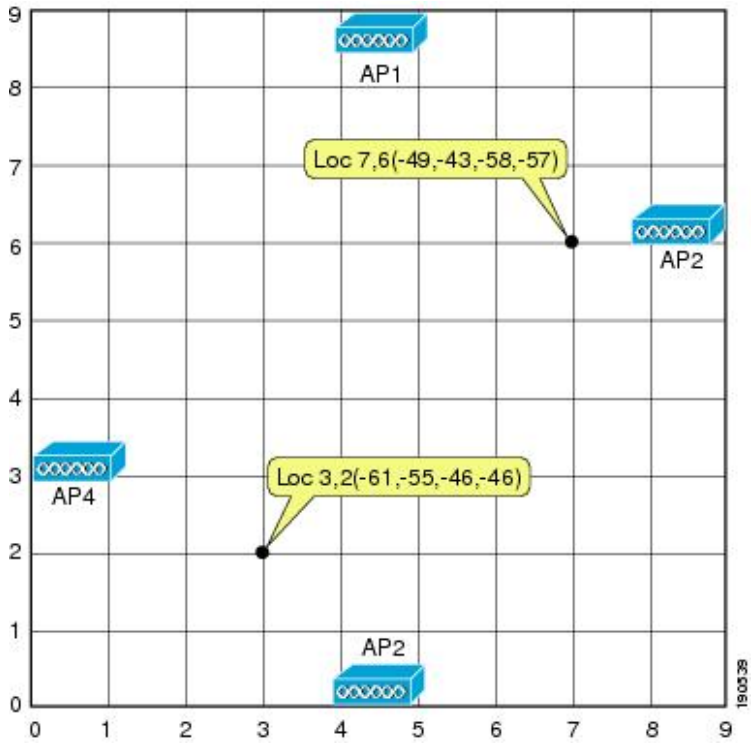

Figure 2.2: Example showing an IPS based on Wi-Fi technology. In this example four APs are placed in a room and two fingerprints are being recorded at two different positions. (Source:[9])

Although, Wi-Fi signals can be used to estimate a position indoors, they still not a complete solution to address the indoor positioning problem due to the nature of those signals. They are affected by a number of spatial factors like shadowing, reflection, wave interference or even by people walking. Wi-Fi signal strength is expected to decrease as the distance increases, however due to these factors it can appear to be stronger or weaker, leading to some errors while measuring the signal strength in a given position and consequently to a wrong position estimation [10].

2.1.3 Ultrasound

Ultrasound frequency band goes from 20kHz up to 40kHz [11].

The characteristics of the ultrasound signal are interesting for use in indoor positioning systems. The accuracy achieved by ultrasound is typically of a few centimeters. The time of flight (TOF) is the preferred method due to the slow propagation of the signal, as shown in the system represented in the figure 2.3 . The time of flight of the signal propagation from the transmitter to the receiver is used to estimate the distance between them taking into account the propagation speed of sound [12].

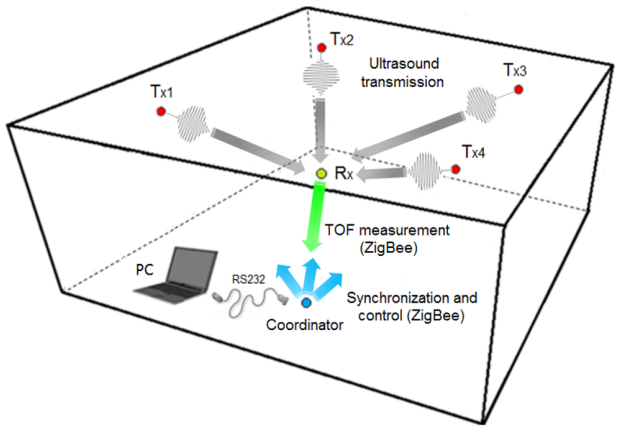

Figure 2.3: An Ultrasound based system. A set of four transmitters is placed in the room and a mobile receiver measures the TOF of the different signals. (Source: [13])

2.1.4 Ultra Wideband

Ultra Wideband (UWB) is a wireless technology that enables the transmission of large amounts of data over a wide frequency spectrum using very low power for a short distance. Unlike Wi-Fi, this technology assures resistance against multipath effects (signals reaching the receiver from two different paths). However, in order to use this technology in an indoor positioning system, receivers and transmitters should be installed in the building [14, 15].

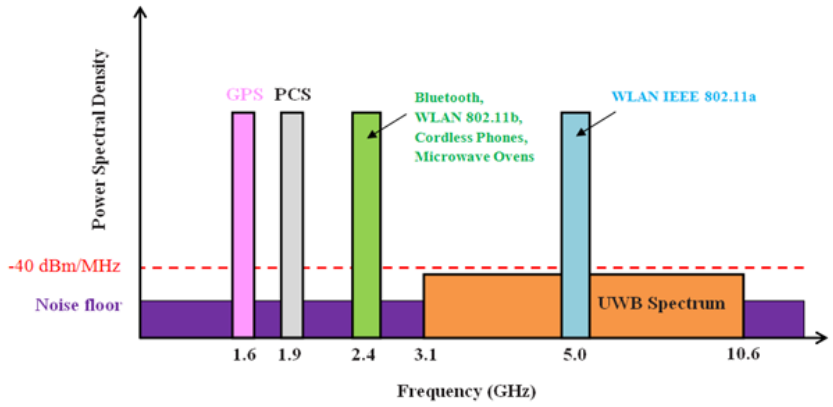

Figure 2.4: UWB Spectrum position compared to other technologies. (Source: [16])

2.1.5 Visible Light Communication (VLC)

Systems based in the VLC technology [17] are emerging and use custom modulated Light-Emitting Diodes (LEDs) lights on the ceiling alongside with a smartphone application. LED lights are installed inside the indoor environment and while the user is navigating through the space with the smartphone, its frontal camera capture and analyze the received light signal (figure 2.5) and deppending on the pattern received from the lights, a position is estimated using techniques such as triangulation. The advantage of such systems is the accurate position estimation. Besides the need to implement a physical infrastructure, this technology has another drawback: there should be no obstruction between the frontal camera

and the LEDs in order for the system to work [18, 19].

Figure 2.5: VLC based system illustration. (Source: [17])

2.1.6 Magnetometer

The magnetometer is commonly found on mobile devices, however it is one of the most difficult sensors to interpret. It is commonly called a compass since it measures the strength of the magnetic field in three dimensions, but it does not necessarily point to the magnetic north. Like a common compass, magnetic interference can cause it to behave unpredictably [20].

Figure 2.6 shows an example of the magnetic strengths inside a building.

Figure 2.6: Magnetic strengths inside a building. Different magnetic strength patterns represent different groups of position thus a position estimation can be performed based on those strengths. (Source: [21])

2.1.7 Accelerometer

The accelerometer is an electromechanical device used to measure acceleration. They are used in wide range of fields, from military to the industry and even on a samartphone to always display the screen upright [22].

Integrating the output once, velocity is obtained. Integrating again, change of position along the axis of the accelerometer is achieved [23]. Using the resultant data, applications like the step detector can be built. Step detector detect steps based on the acceleration values measured at a given time. Figure 2.7 shows an example of a step detector.

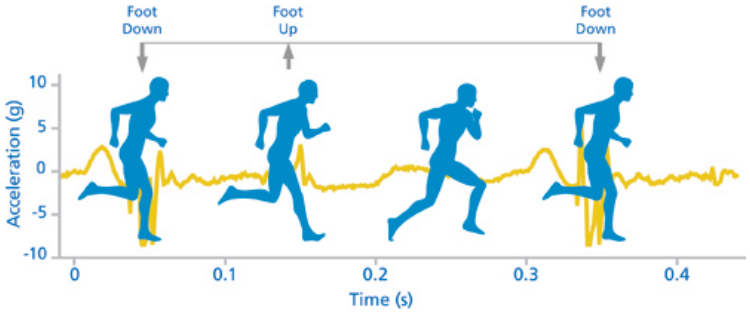

Figure 2.7: Output of an accelerometer measuring inertial movements. (Source: [24])

2.1.8 Computer Vision

Computer vision based systems use 2D or 3D cameras to estimate a target position indoors. They act as a passive system, meaning that there is no need for an exchange of information between the system and the target. As an example, this can be useful for retailers looking to track their costumers inside a store [25, 26].

A number of solutions based in standard surveillance cameras appeared and the use of 3D cameras are being explored in recent years due to their additional depth channel which can improve the performance of computer vision systems [27, 28]. The downside of these systems is the need of a relatively powerful computer or processor to run the detection and tracking software.

Figure 2.8 shows people being tracked by a computer vision based system. Each person is bounded by a rectangle.

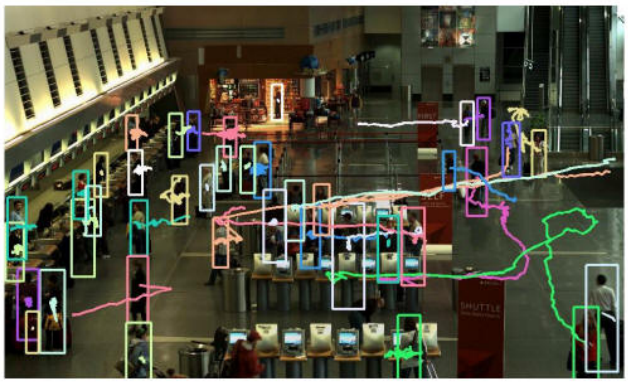

Figure 2.8: People tracking system based on computer vision. (Source: [29])

2.1.9 Radio Tomographic Imaging (RTI)

RTI is a passive indoor positioning technology because there is no need for an exchange of information between the user and the system. Like shown in figure 2.9, RTI is based on a set of wireless transceivers installed on the walls of an indoor environment. Each transceiver broadcast to all the other transceivers and whenever a person or object gets in the middle of a link between two of them, a decrease in the signal strength will happen. Using data from different links between different transceivers enables the system to project the person or object position with acceptable accuracy [30, 31, 32].

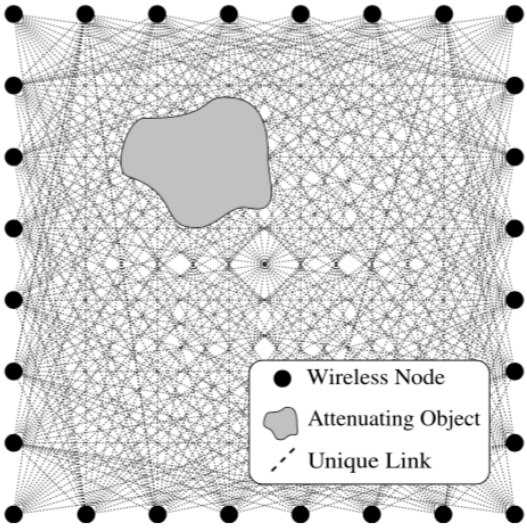

Figure 2.9: Operation example of a RTI based system. (Source: [30])

2.2 Positioning Techniques

The mathematical relationships between physical phenomena and position are described in this section.

2.2.1 Fingerprinting

Fingerprinting technique uses fingerprints to calculate position. Fingerprints are vectors of signal strength measurements recorded from APs near the user [33].

A map with a set of fingerprints related to a building is represented in figure 2.10.

As an example, a Wi-Fi Radio Signal Strength (RSS) fingerprinting system normally consists of two phases: the offline training phase and the online phase. During the first phase (offline phase), RSS fingerprints are collected and in the online phase a database is queried in real time with the RSS at the moment and an estimation of the distance is returned. One should keep in mind that despite this technique is widely used with Wi-Fi RSS, it can be used with other inputs, like magnetic strengths, Bluetooth signals, among others.

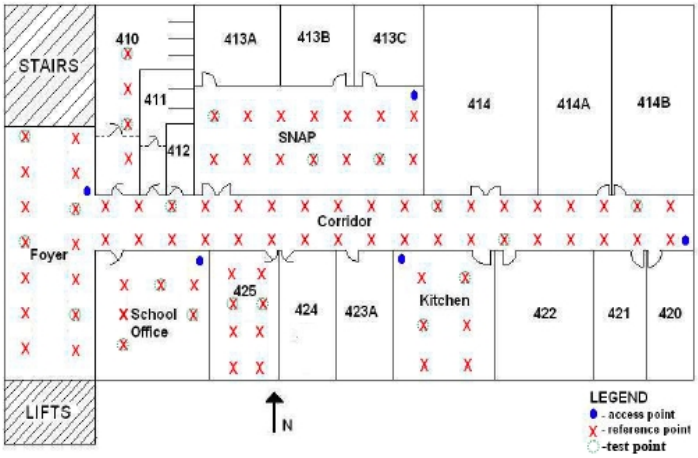

Figure 2.10: Example of a floor plan with fingerprints. This map shows the stored fingerprints in a database and they are marked as reference points (RPs)(red crosses).

2.2.2 CheckPoint

this technique indexes the location of known objects or sensors by using checkpoints. As the object goes through the checkpoint, a tag will be recorded and simultaneously the position associated with this same object will be updated. Systems based in this technique have the advantage of being reliable and accurate, however they have some disadvantages as well: there is a need to have identification tags and a physical infrastructure of transmitters. RFID based location is implemented using this technique [34, 35].

Figure 2.11 shows an example of a system based in the checkpoint technique.

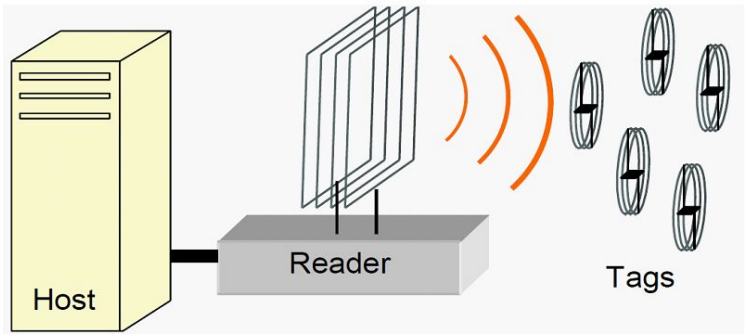

Figure 2.11: Example of a choke checkpoint System using RFID. (Source: [36])

2.2.3 GPS augmentation

Systems that use GPS augmentation [37] consist of a set of devices that collect the external GPS signals and turn it in a usable signal indoors. These systems are accurate [38], however there is the need of an infrastructure to collect the GPS signal and effectively distribute it inside the building and synchronize the signals coming from different satellites (figure 2.12).

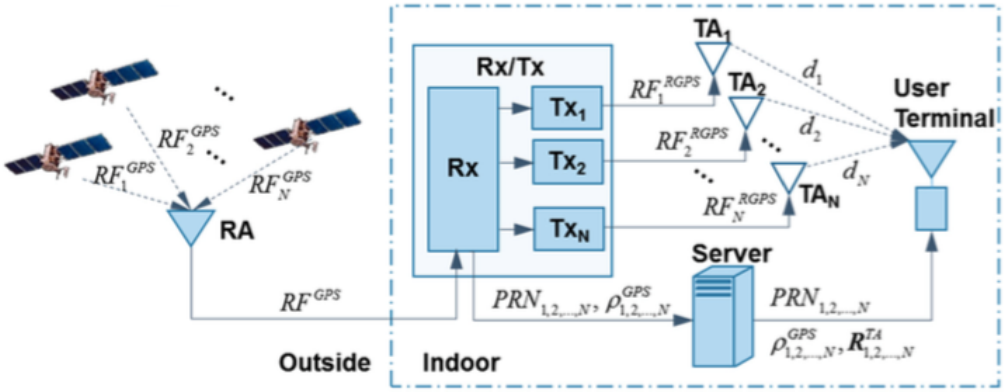

Figure 2.12: Sample architecture of a system based in GPS augmentation. (Source: [38])

2.2.4 Angle of Arrival (AOA)

This technique finds the direction of propagation of a radio-frequency wave at the moment of incision in the receiver (figure 2.13). These directions can be measured either using directional sensors or even measuring arrival timing differences between signals [39].

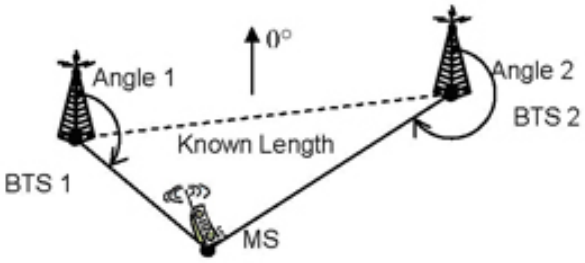

Figure 2.13: Angle of arrival exemplified with signals sent to a mobile device. The phone can use angle information in order to estimate a position. (Source: [40])

2.2.5 Multilateration

Estimates the position of a given device using the RSS from different transmitters (figure 2.14). Considering a relationship between RSS and given distance, an estimation of the distance between the device and each transmitter can be obtained.

This technique is generally designated trilateration when three transmitters are used, however the technique is the same when the number of transmitters is bigger and it is called multilateration.

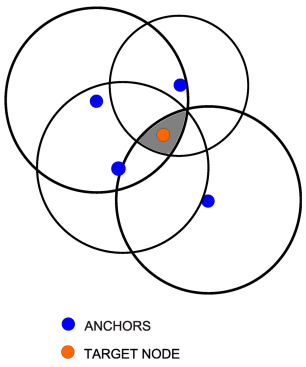

Figure 2.14: Multilateration example. As example, the blue dots are APs and the orange dot is the estimated position of a smartphone. (Source: [41])

2.2.6 Dead Reckoning

This technique estimates a position based in the heading and speed resultant from a previously known position (figure 2.15). When a user is moving from one position to another, a movement vector is created based in the heading and speed. Using this vector, one can estimate the user's current position or a set of possible positions [42].

Figure 2.15: Dead Reckoning demonstration. On the left side there is a representation of the plane's real route (red line) and on the right side the estimated route using the dead reckoning technique. (Source: [43])

2.3 Positioning Algorithms

Algorithms present in this section are divided by the technique to which they are relevant. KNN, WKNN, MMSE and MAP for fingerprinting technique and the Weighted Non-Linear Least Squares Fitting Algorithm for multilateration [44, 45, 46].

2.3.1 Fingerprinting Technique Algorithms

2.3.1.1 K Nearest Neighbors (KNN) - Deterministic Algorithm

This algorithm estimates the unknown user location as the average of K locations which have the shortest Euclidean distances between the currently ob-

served RSS fingerprint and the respective mean value fingerprints in the database (Equation 2.1). The selected K value is selected based on the scenario. That means that for each set of collected data, this value should be different to maximize the performance of the system. However many literature refers to the selection of only the closest neighbor, meaning a $K = 1$. In this thesis case, the K value is set to a different value whenever a new database is being used. The K value will go from 1 to 15 and it deppends on the scenario, meaning that the K parameter will change accordingly to the TPs used to calculate K.

$$\delta_n = \sqrt{\sum_{i=1}^{m}(RSS_{RP} - RSS_{observed})^2} \tag{2.1}$$

Equation 2.1 is the Euclidean distance equation. In the case of indoor position and KNN algorithm, this equation gives Euclidean distances between RSS values. The bigger δ_n, the further it will be from the observed RSS. δ_n corresponds to the Euclidean distance between a given RP's RSS (RSS_{RP}) and the currently observed RSS ($RSS_{observed}$). The sum of values will be made from 1 until m (the number of observed APs that are present in the database). This calculation is performed for all the RPs and a sorted list is created based on this calculation as algorithm 1 suggests. The calculated δ_n should be in ascending order for easier access to the data. Finally, the first K positions will be included into the list Ω and the estimated position will be calculated using the Equation 2.2.

Equation 2.2 finds the estimated position ($P_{estimated}$) using the RPs present in the sorted list Ω. Each RP ($P_{reference}$) included in the list Ω is summed and then a division by the constant K is performed.

$$P_{estimated} = \frac{\sum_{i=1}^{K}(P_{reference})}{K} \tag{2.2}$$

Algorithm 1: Used KNN algorithm

Input : RSS vector

Output : $P_{estimated}$

for *all n RPs* **do**

  Calculate Euclidean distances δ_n using Equation 2.1;

end

Sort RPs in ascending order according to the Euclidean distances;

Include the first K RPs into Ω;

for *all positions in* Ω **do**

  Calculate $P_{estimated}$ position using Equation 2.2;

end

2.3.1.2 Weighted K Nearest Neighbors (WKNN) - Deterministic Algorithm

WKNN is a variant of KNN and estimates the unknown user location as the weighted average of K locations which have the shortest Euclidean distance between the currently observed RSS fingerprint and the respective mean value fingerprints in the database. The weight of each location is set equal to the inverse of the distance ($\frac{1}{\delta_n}$).

Similarly to the KNN algorithm, the WKNN algorithm sorts an Ω list of the first K positions. However, it processes the list in a different way. WKNN algorithm makes a weighted summation shown in Equation 2.3 to find the estimated position ($P_{estimated}$). Instead of summing all the positions and divide them by the constant K, it gives different values of $\frac{1}{\delta_n}$ to the different K positions of the list Ω: for a smaller Euclidean distance, a bigger $\frac{1}{\delta_n}$ and for a bigger distance, a smaller $\frac{1}{\delta_n}$. The Algorithm 2 presents all the process from the input RSS vector until the estimated

position.

$$P_{estimated} = \frac{\sum_{i=1}^{K} \left(\frac{1}{\delta_n} P_{reference} \right)}{\sum_{i=1}^{K} \frac{1}{\delta_n}} \tag{2.3}$$

Algorithm 2: Used WKNN algorithm

Input : RSS vector

Output : $P_{estimated}$

for *all n RPs* **do**

  Calculate Euclidean distances δ_n using Equation 2.1;

end

Sort RPs in ascending order according to the Euclidean distances;

Include the first K RPs into Ω;

for *all positions in* Ω **do**

  Calculate weighted P position using Equation 2.3;

end

2.3.1.3 Minimum Mean Square Error (MMSE) - Probabilistic Algorithm

MMSE is a probabilistic approach that estimates the unknown user location, which is equivalent to the weighted sum of the RPs in the database while the weights are set equal to the conditional probability of each location in the database given the currently observed fingerprint. The algorithm estimates a position based on a conditional probability of a location x given a signal y for a set of n positions [44] (Equation 2.4).

$$P(x_i|y) = \frac{P(y|x_i)P(x_i)}{P(y)} = \frac{P(y|x_i)P(x_i)}{\sum_{i=1}^{n} P(y|x_i)P(x_i)} \tag{2.4}$$

MMSE initially calculates the probability of a user being on a specific position according to the similarity between the observed Wi-Fi RSS and all the RPs RSS. This probability is calculated using the Equation 2.5.

$$Pr = \prod_{i=1}^{m} e^{\frac{-(\bar{x}_i' - \bar{x}_i)^2}{\sigma^2}} \tag{2.5}$$

Pr is the probability of a user being at a selected position, m is the number of observed APs, $\bar{x}'$ is the observed Wi-Fi RSS, $\bar{x}$ is the averaged RP Wi-Fi RSS and σ is a parameter that deppends on the current scenarion [44]. Like K parameter on KNN and WKNN algorithms, this parameter will go from 1 to 15 depending on the test positions used to its calculation.

After the value Pr is calculated for all the n positions in the dataset, $P_{estimated}$ is estimated based in the conditional expectation [47]. This calculation is demonstrated in Equation 2.6 where RP is the set of RPs, and P_i is the location i from the database. An algorithm for the MMSE is shown below in Algorithm 3.

$$P_{estimated} = E(RP|y) = \sum_{i=1}^{n} \frac{P_i \cdot P_r}{\sum_{i=1}^{n} P_r} \tag{2.6}$$

Algorithm 3: Used MMSE algorithm.

Input : RSS vector

Output : $P_{estimated}$

for *all n RPs* **do**

Calculate Pr probability using Equation 2.5;

Add calculated probability into list Ω;

end

Calculate the estimated position $P_{estimated}$ using the Ω elements as input to the Equation 2.6;

2.3.1.4 Maximum A-Posteriori (MAP) - Probabilistic Algorithm

MAP is a probabilistic approach that estimates the unknown user location by maximizing the conditional probability of each location in the database given the currently observed Wi-Fi RSS. This algorithm has almost the same theoretical background as the MMSE algorithm. It uses the conditional probability as base (Equation 2.4) and, like MMSE, initially calculates the probability of an user being on a specific position according to the similarity between the observed Wi-Fi RSS and all the RPs RSS (Equation 2.5). After P_r is calculated, it will select the position where this value is higher. Shortly, it will choose the point in the database with the highest probability of a match between its position and the observed Wi-Fi RSS. The algorithm is described in Algorithm 4.

Algorithm 4: Used MAP algorithm.

Input : RSS vector

Output : $P_{estimated}$

Set $highestPr = 0$;

for *all n RPs* **do**

  Calculate Pr probability using Equation 2.5;

  if *calculated* Pr *is higher than* $highestPr$ **then**

    $highestPr = Pr$;

    $P_{estimated} =$ actual RP;

  end

end

2.3.2 Multilateration Technique

Multilateration can be performed using fitting techniques. These techniques generate an approximated model of the real data, so one can obtain an approximated solution to the position estimation. The fitting technique used is the non linear least squares fitting based in the Levenberg-Marquardt algorithm [48].

The sum of the squares of the errors on the distances is minimized with the least squares technique. r_i denotes the approximate distance between the Wi-Fi APs and the actual estimated position and $\hat{r_i}$ stands for the exact distance:

$$(x - x_i)^2 + (y - y_i)^2 = \hat{r_i}^2 \tag{2.7}$$

Variables x and y are the coordinates of the exact position and x_i, y_i are the coordinates of each Wi-Fi AP. To minimize the sum of the squares of the errors on the distances, one must minimize the function:

$$F(x,y) = \sum_{i=1}^{n} (\hat{r_i} - r_i) = \sum_{i=1}^{n} f_i(x,y)^2 \tag{2.8}$$

with

$$f_i(x,y) = \hat{r_i} - r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} - r_i \tag{2.9}$$

Minimizing the sum of the square errors is a usual problem in applied mathematics. Different approaches can be taken. The selected method, Levenberg-Marquardt, uses the Newton iteration to find the best fitting solution for the estimated position. To find more about the mathematics behind this technique, consult reference [46].

2.4 Related Work & Contributions

In this section related work and contributions to the indoor positioning field are presented. There are solutions and applications that served as base to other works and they deserve to be mentioned also due to their impact on IPS development. These include:

- Airplace [49] [50] - These Android applications make use of the fingerprinting technique. They use the four algorithms used in this thesis approach (KNN, WKNN, MMSE and MAP) to estimate a position. Airplace is divided into two applications. The first application is an offline logger that generates a radiomap containing a set of fingerprints. The fingerprints can be created via crowdsourcing (several users) or by a single user. In the case of crowdsourcing, the server appends and combines all fingerprints created by several users to create the final radiomap that is used for positioning. The users can select how many samples per fingerprint they want to collect and the time between samples can also be adjusted. The users then must upload the collected fingerprints to a main server for further analysis and distribution. The second

application connects to the server to download the radiomap and enables position estimation using a set of algorithms.

- RADAR [51] - Developed by Microsoft Research in 2000, this project was the first research project that put efforts into using Wi-Fi signal strengths for indoor localization. After this study, other projects started to emerge using its methodologies. Two different algorithms were implemented for the indoor location estimate. One of them is the fingerprint technique using the already refered KNN algorithm and the other one is designated as propagation method. The proposed system has an accuracy of 3 meters for 50% of the time when using the Fingerprint approach and at 4.3 meters when using the Propagation method. The results of this approach should are used as reference for this thesis results.

- Trilateration Study [46] - This work describes a system based in trilateration using the nonlinear least squares technique. The system was implemented in a company for a variety of mining applications, including positioning of bulldozers, drills and other instruments. It is a larger scale application when compared to indoor positioning systems, however the fundaments used in the study can be implemented and tested in IPSs.

2.4.1 Other Applications Approach

There is no defined IPS standard yet, though there are already some commercial systems available.

A part of commercial applications use Wi-Fi to provide the user with a position estimation, offering acceptable results. Although, these results can be improved using a combination of technologies, sensors and technologies, resulting in an hybrid system which can improve the performance and accuracy of the system [52].

One of the tested Android applications based on Wi-Fi technology is called

RedPin. This application's approach is different from other commercial applications in the market beacuse it performs room level position estimation. Instead of a spacial position estimation it presents the position related to the area where the user is in: if the user is in the living room it will present a red dot in the living room, if the user is in the kitchen it will display a red dot in the kitchen. The application is open source and it does what it claims to do, however it can't be used for indoor positioning in larger spaces such as big halls or factory stores (for tracking or navigation purposes).

The other tested application uses a different technology: a magnetometer. The application is named IndoorAtlas [21] and it starts by prompting the user to upload the building's map on-line; after this step a calibration is necessary. This calibration is not easy because the users need to characterize all the pathways on which building they want to navigate, going all around with the phone in order to record magnetic values and upload them to the application's on-line database (network connection is required in this phase). Afterwards, the navigation mode is made available and the application's behavior is acceptable, however there are the following problems:

1. The application reacts unpredictably to certain sensor variations;
2. Internet connection is necessary.

2.4.2 Processes Used to Build Similar Datasets

Datasets are a collection of data. They are the base to a group of indoor positioning application and a bad dataset can lead to poor results and vice versa. There are datasets built to serve as base to other applications and theoretical demonstrations. Two of them [10, 53] are used as reference for the comparison with the dataset resulting from the process adopted in this thesis.

- **Approach 1**

  This dataset was created on an entire floor. The floor plan spans an area of 48x$20\ m^2$. They deployed 10 Wi-Fi APs. To build the database, they collected 100 RSS measurements at an interval of 2 m (a total of 72 RPs). The RPs are denoted by the red-colored dots in Figure 2.16 (a). For unseen APs, the mean and measured RSS values are set to $\gamma = -100\ dBm$. The experiments were conducted by walking according to the positions (at 1 m intervals) as seen in Figure 2.16 (b). The results were measured over 10 laps.

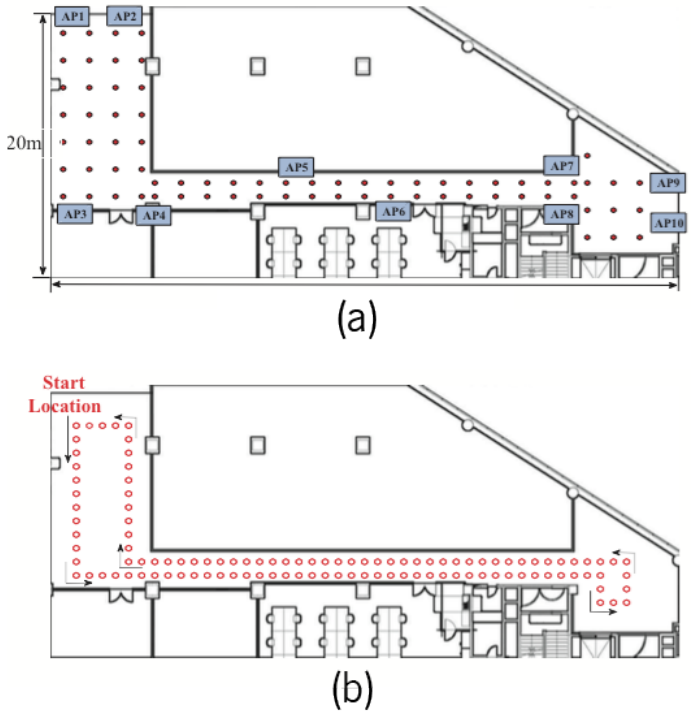

Figure 2.16: Floor plan of used to build the dataset on approach 1. (Source: [53])

- **Approach 2**

  First, the system records a number of fingerprints (signal strength and the associated AP's MAC address) at a given position and then it stores the means and standard deviations of those measurements. Fingerprints are

given by $F = (\mu_i, \sigma_i)$, where μ_i is the mean of the RSS measures from the ith AP and σ_i is the standard deviation. Each fingerprint is related to a position where three measurements with different durations are made. A study is performed to compare different calibration times (15, 30 and 60 seconds) [10].

A calibration file stores the fingerprints and the respective locations. This file is used in the online phase of the fingerprinting technique in order to determine the estimated position. The floor plan used to build the dataset on this approach is in Figure 2.17.

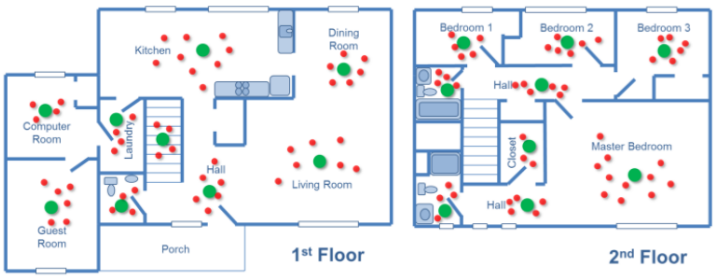

Figure 2.17: Floor plan used to build the dataset on approach 2. Green dots are the RPs and red dots are test positions (TPs). (Source: [10])

Chapter 3

Methods

In this chapter a set of used methods used in this thesis is presented.

A dataset description with two different sets of data is presented alongside with a discussion on its building process.

An important insight over the the used techniques, fingerprinting and multilateration, is presented with the help of a comparison between both of them.

3.1 Datasets

A dataset was built to avoid repetitive on-field tests and to provide a tool to future research. It is used as a solid data structure in all the performed tests. The dataset was built in a classroom inside University of Minho and this classroom has tables and windows in one of the walls and it has only one door. The data was collected using an Android smartphone inside a 10m x 10m room with one Wi-Fi AP in every corner (four APs in total) as shown in figure 3.1. To check how the room looks like, refer to the videos present in the dataset.

All the collected data with the exception of the videos was stored in a SQLite database in the application. These data consists of:

- Wi-Fi RSS at various points;

- Magnetic strengths at various points, standing still and in motion;
- Acceleration values;
- Camera images and video.

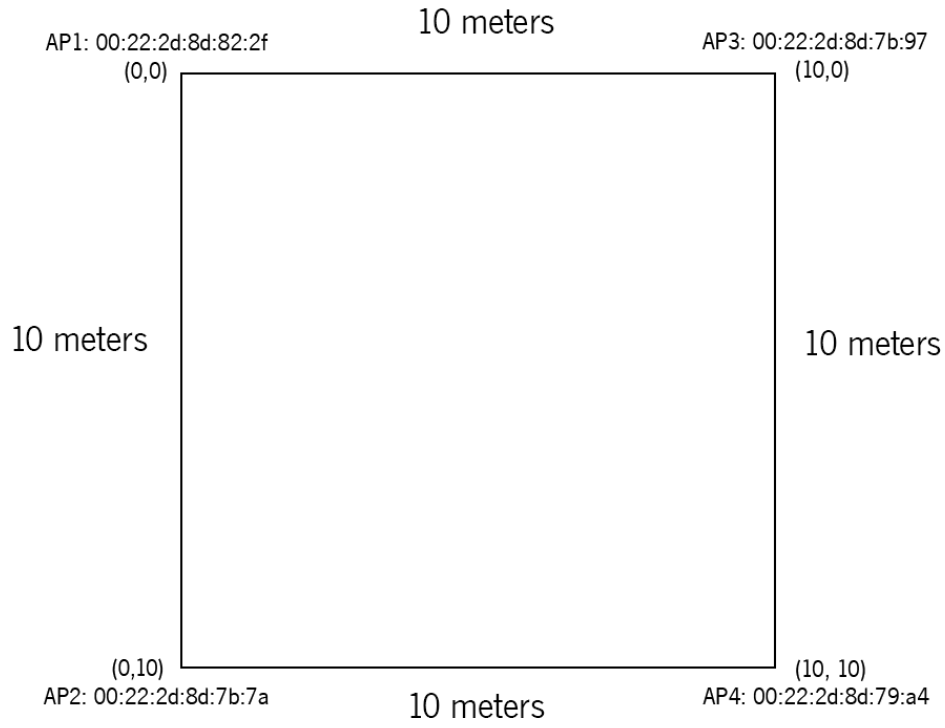

Figure 3.1: Room layout and APs location.

The database was split in two different sets of data:

- Standing still Dataset - Set of data collected at a predetermined set of points without moving. It includes readings from magnetometer and Wi-Fi transceiver in 121 RPs and 7 Test Positions (TPs).
- In motion Dataset - Data collected while moving on the room. The data collected in motion includes magnetometer and Wi-Fi readings plus accelerometer and video data. The video was recorded to serve as a visual validation

of the accelerometer and magnetometer readings. As example, if the user stops at a time x in the video, the accelerometer readings at the same time x should confirm that the user stopped a certain amount of time. This conclusion can be extracted through the absence of steps detected.

In this set of data, three different experiments were performed and later analyzed in this chapter. Those three experiments were collected with different smartphone screen orientations, to collect data that may be useful to future researchers that may want to embed computer vision in indoor positioning and, finally, to serve as base to test the step and movement detection developed in this thesis.

Two different Android applications were used to build the datasets due to their different characteristics. Standing still dataset needs input from the user since it records readings from a given position, on the other hand, in motion dataset is based on a route simulation, meaning that it simulates data collected in real time.

All the variables used in the database layout are defined in table 3.1.

3.1.1 Standing Still

In this set of data, 121 RPs and 7 TPs were collected with the smartphone identified in section 4.1. All points were scanned with a dedicated Android application discussed in 4.2.1. The application asks for the current position and then it starts to gather all the readings on that position. Figure 3.2 shows the database layout for the standing still dataset. For each selected position coordinate, all these parameters are written in the database.

Variable	Description
snumber	Sample number
position	Position (x, y)
absolute	Absolute magnetic strength
magnetic_x	Magnetic strength around the x axis
magnetic_y	Magnetic strength around the y axis
magnetic_z	Magnetic strength around the z axis
timestamp	Timestamp of a sample
mac_address	MAC address of the AP
rssi	WiFi signal strength indicator
absolute_strength	Averaged absolute magnetic strength
deviation	Standard deviation
magnetic_absolute	Absolute magnetic strength
acc_x	Acceleration along the x axis
acc_y	Acceleration along the y axis
acc_z	Acceleration along the z axis

Table 3.1: Variables used in the layout of the databases.

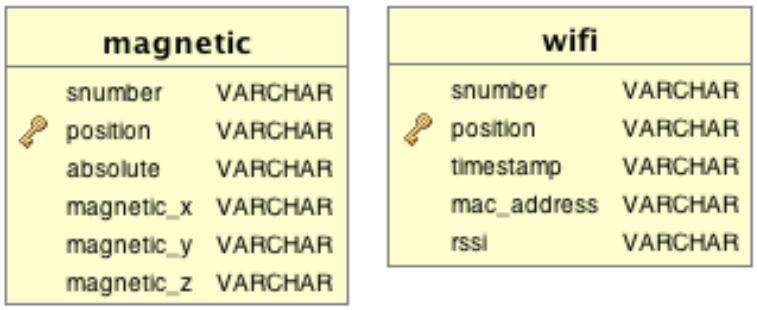

Figure 3.2: Standing still dataset's database layout.

All the parameters were collected in the same fashion. The application receives as input from the user the position and then it collects a sample of Wi-Fi and magnetic field parameters each second until it reaches 10 samples. After 10 seconds, these 10 samples are saved onto the database and the application is ready to collect data in another position. Figure 3.3 shows all the collected points for the standing still dataset. The black crosses are RPs and the blue circles are the TPs. After all the reference points were recorded, there was a need to have a set of TPs. TPs are used to test the database and simulate points received in real time tracking. This way the algorithms can be tested without the need to constantly collect data inside the dataset room. A set of seven TPs were chosen and parameters relative to each chosen TP were stored in the same way as all the RPs.

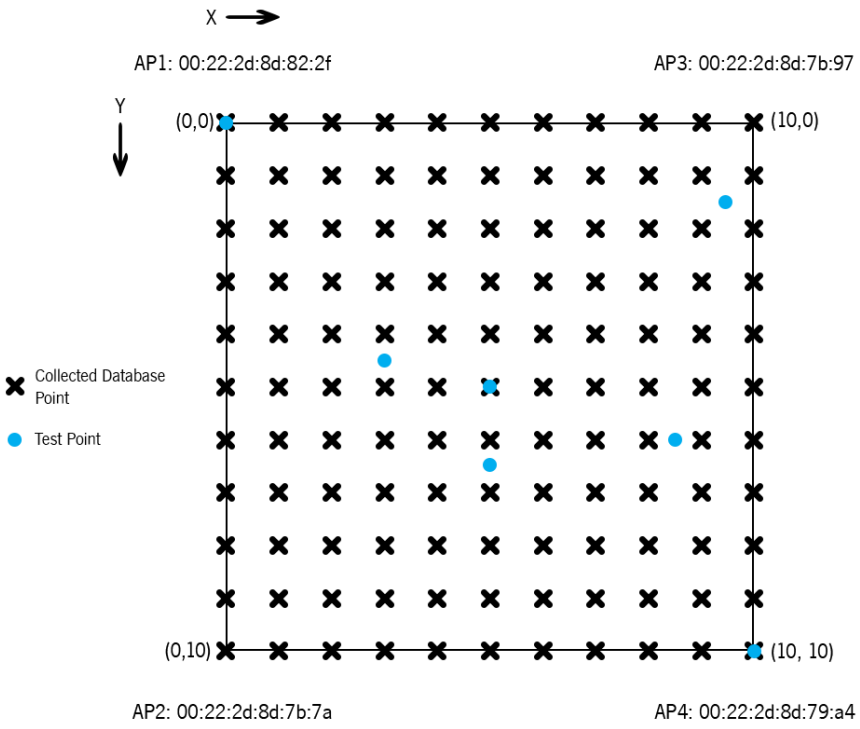

Figure 3.3: Dataset Description.

Averaging all the samples information is necessary and new tables including these values are essential. Averaged values are used by the positioning algorithms and for that reason they should be in the database as well. All the average and standard deviation values are generated in real time by the application. When the data collection for a given position is finished, the Android application processes the 10 readings and it builds the new table in the database. Despite the fact that averaged and standard deviation values were included in the database, the set of samples is still available. This enables to apply other filtering techniques in the future.

The final database structure is shown in the figure 3.4. Database data is now ready to be used by any fingerprinting based application in order to estimate positions inside the dataset's room.

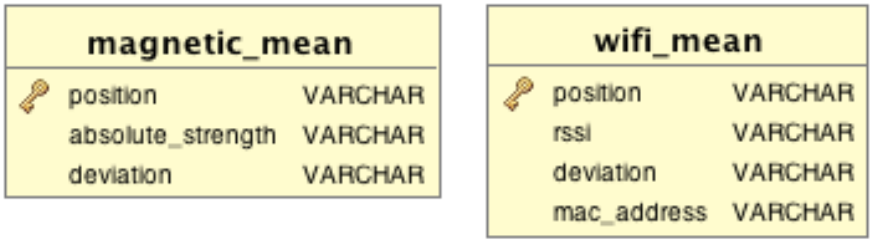

Figure 3.4: Resulting tables after processing collected data.

3.1.2 In Motion

This dataset was built based on three different experiments where the smartphone orientation and walking patterns are different. However, the collected parameters are the ones presented in figure 3.5 for all off them. The configurations were:

1. In the first experiment, the smartphone is in the user's right hand and all the actions are being filmed by another person (Figure 3.6 (a)).

2. In the second experiment, the user carries the filming device in his chest to collect video data while moving. The smartphone remains in his right hand, periodically recording data (Figure 3.6 (b)).

3. Finally, in the third experiment, the user is filmed by another person and he is moving with the phone in his chest (Figure 3.6 (c)).

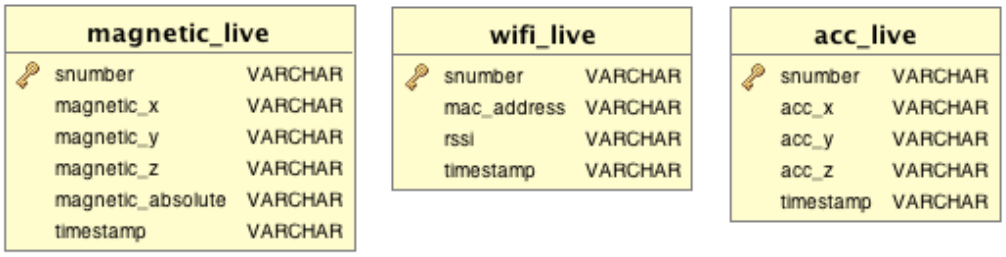

Figure 3.5: In motion dataset's parameters.

The process used to collect data differs from the one used in the standing still dataset. Samples of Wi-Fi RSS, acceleration values and magnetic strength were taken along the user's trajectory every 200 milliseconds. Samples from all sensors are taken in sequence, one per sensor, and as fast as possible to minimize the gaps (to be at almost the same time). A timestamp was recorded alongside with the recorded data, allowing a match between the database values and the recorded video.

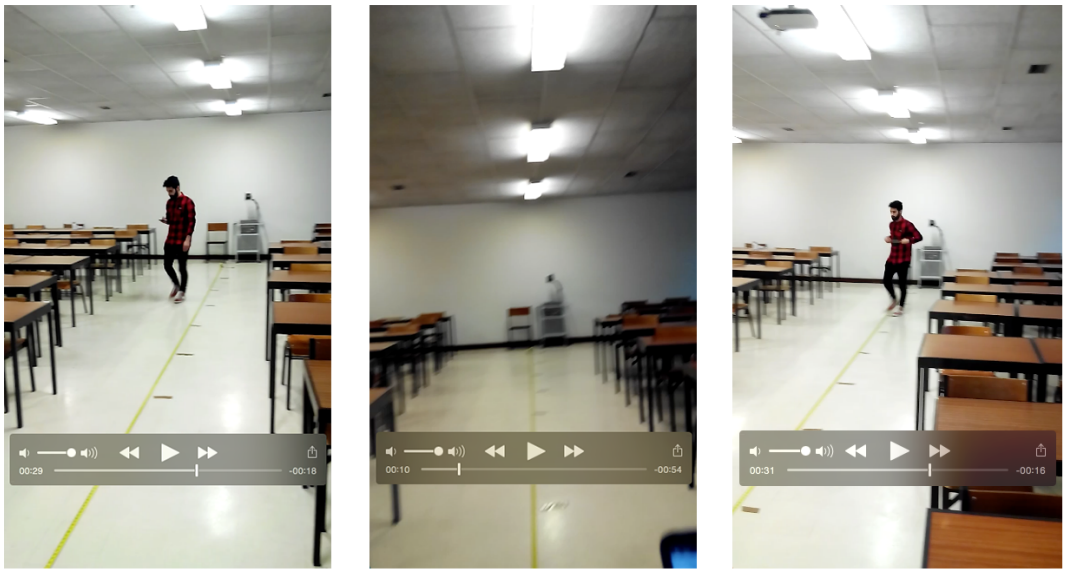

Figure 3.6: In motion dataset experiments. (a) Experiment one; (b) Experiment two; (c) Experiment three.

3.2 Applied Positioning Techniques

This section showcases the tested approaches. The two presented approaches were already mentioned in chapter 2 and they are going to be introduced in order to get a better insight of the advantages and disadvantages of each approach.

3.2.1 Fingerprinting vs. Multilateration

- Data Collection

  To perform an estimation with a fingerprinting based system there is the need to have an offline phase and an online phase. In the offline phase, the database is populated with values that will be used in the estimation during the online phase. The offline phase can be time consuming and it can take a big amount of time. Multilateration technique, on the other hand, needs less data to work properly. Precise APs coordinates alongside with the

media access control address (MAC address) from each AP are needed. The more precise the coordinates of each AP, the better the distance estimation between each AP and the smartphone.

- Capacity to Adapt

  Multilateration is more adaptable than fingerprinting. This happens because the multilateration based systems need less data to be collected in the case of infrastructure modifications, like an AP replacement. If an AP is replaced multilateration need to add the new MAC address and new coordinates, if necessary. Fingerprinting is harder to reconfigure in this case, since a new set of data needs to be collected for all surrounding APs.

- Accuracy

  Fingerprinting technique offers better accuracy than multilateration technique. Multilateration uses a distance estimation as base to estimate a position. In this thesis case the used propagation model is the Free Space Path Loss (FSPL). This means that multilateration do not calculate the interference caused by possible objects between the smartphone and the APs. On the other hand, fingerprinting technique handles this situation, since the values recorded are attenuated already by possible objects, and that value will be used as reference on the online phase. Nevertheless, an important consideration should be made: although multilateration technique does not handle possible obstructions, it still can be better than fingerprinting reacting to environmental changes. Fingerprinting technique is a static approach and subtle environmental changes can lead to a loss of accuracy.

3.2.2 Fingerprinting

Fingerprinting approach is the most widely used approach nowadays. However the amount of data needed to implement it is much higher than in multilateration, since a complete dataset is needed.

As refered in chapter 2, fingerprinting technique uses Wi-Fi RSS fingerprints to calculate position. A Wi-Fi RSS fingerprinting system normally consists of two phases: the offline training phase and the online determination phase. In the training phase, the goal is to build an empirical training database for each RP by sampling the Wi-Fi signal strength from several wireless APs. Hence, each location in the database has a RSS record, and these RSS records are used as a location fingerprint [54, 55, 56]. As shown in Figure 3.7, in the next phase, the online phase, the received Wi-Fi RSS is crossed with all fingerprints stored in the database, and the similarity, or the distance in signal space is calculated on the basis of a positioning algorithm. The otput of this algorithm is then reported as the estimated location [57, 58].

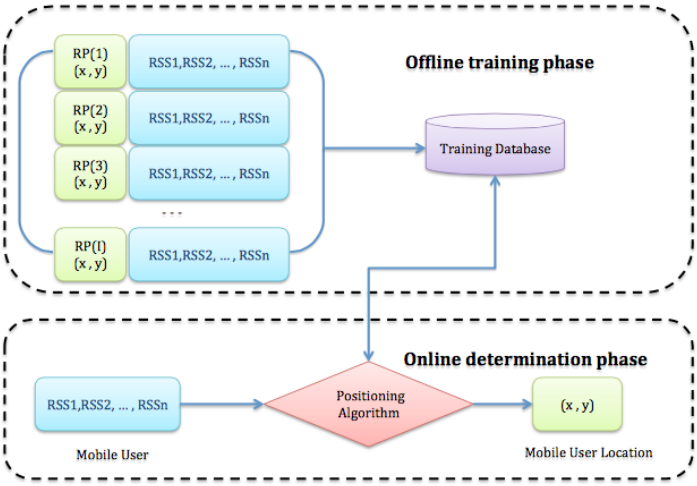

Figure 3.7: A standard Wi-fi Fingerprinting system. (Source: [58])

The database used in this thesis approach consists of the dataset described and since Android system is being used, it is stored in a SQLite database. All the information of the dataset can be obtained with a query and it can be used by other applications in the future.

The algorithms used in this approach were previously described in this chapter and their performance is compared in chapter 5. Two of the algorithms are deterministic and the other two are probabilistic. The deterministic are:

- K Nearest Neighbors (KNN);
- Weighted K Nearest Neighbors (WKNN);

The probabilistic are:

- Minimum Mean Square Error (MMSE);
- Maximum a Posteriori (MAP).

The used approach follows the guidelines shown in figure 3.7.

3.2.2.1 K and σ Parameters Calculation

As refered in chapter 2, K and σ parameters are calculated whenever using the application in a different scenario. Usually, a dataset contains a set of previously recorded fingerprints and they are enough to get a position estimation. However one should be aware that the results can benefit if K and σ parameters are optimized for the current scenario. To achieve that, a set of TPs should be collected. In this approach, a set of seven TPs is used and they are referenced in the figure below:

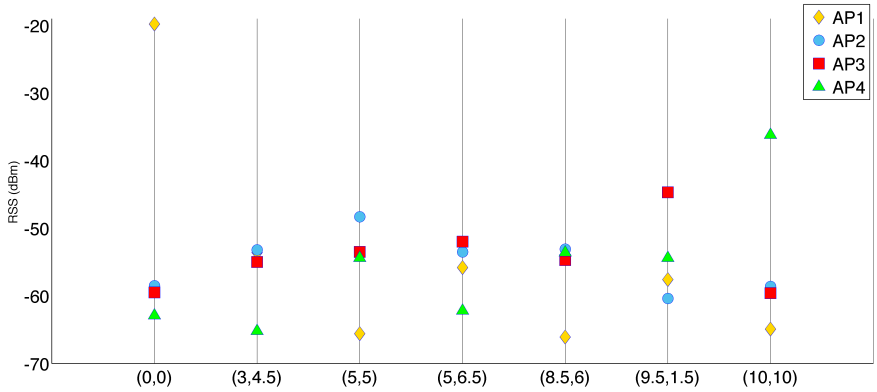

Figure 3.8: TPs from dataset used in the calculation of the parameter K and σ. All the values are in dBm and they correspond to the average of the 10 samples taken during the offline phase.

Algorithm 5: Best fitting parameter calculation.

Input : Set of n Test Positions P

Output : $BestDet$ and $BestProb$

Vector $ParametersDet$ to store all the deterministic calculated parameters ;

Vector $ParametersProb$ to store all the probabilistic calculated parameters ;

for *all n Test Positions of P* **do**

  for *j from 1 to 15* **do**

    Input the RSS values for the position n of P into the algorithm 1;

    Add the output value to the vector $ParametersDet$;

    Input the RSS values for the position n of P into the algorithm 3;

    Add the output value to the vector $ParametersProb$;

  end

end

$BestDet =$ Average of all the elements in vector $ParametersDet$;

$BestProb =$ Average of all the elements in vector $ParametersProb$;

In case that there are no TPs available in the dataset, the parameters K and σ are set to the default value 1. If there are TPs available, the parameters approximation are going to be evaluated. The estimation of the best fitting parameter is explained in detail in algorithm 5.

3.2.3 Multilateration

As referred in Chapter 2, multilateration is a positioning technique to estimate a position using the signal strengths received from several non-collocated, non-collinear transmitters. Multilateration technique uses parameters of known Wi-Fi networks like frequency of Wi-Fi signal, signal strength, network MAC address and real coordinates of Wi-Fi APs in the evaluated location. The observed signal strength together with the AP frequency can be used to estimate the distance between the AP and the mobile device. When using this method, one considers three or more APs allocated in the evaluated location. The signal strengths of these points are decreasing exponentially depending on the distance between transmitter and receiver and random noise factor. Thus this dependency can be considered as function of distance. The distance estimated by signal strength is presented as a circle with a radius around the AP, if the radius have the exact value, the intersection of three access point radius results in a point. This model can be shown as such equation system:

$$\begin{aligned} r_1 &= (x - x_1)^2 + (y - y_1)^2 \\ r_2 &= (x - x_2)^2 + (y - y_2)^2 \\ r_3 &= (x - x_3)^2 + (y - y_3)^2 \\ r_n &= (x - x_n)^2 + (y - y_n)^2 \end{aligned} \tag{3.1}$$

where r_n is the exact distance between the AP_n with center coordinates (x_n, y_n). x and y are the coordinates of the user position.

In order to get a solution to this system, radius distances r_n should consist of exact measurements due to the unique final solution of the equation system. Figure 3.9 shows this situation. The image is made using data extracted from the dataset and is related to the TP $(3, 4.5)$. Distance between this TP and all the APs position was calculated using equation 3.2 and the resulting distance measurements were used as radius for each circle around the center of each AP.

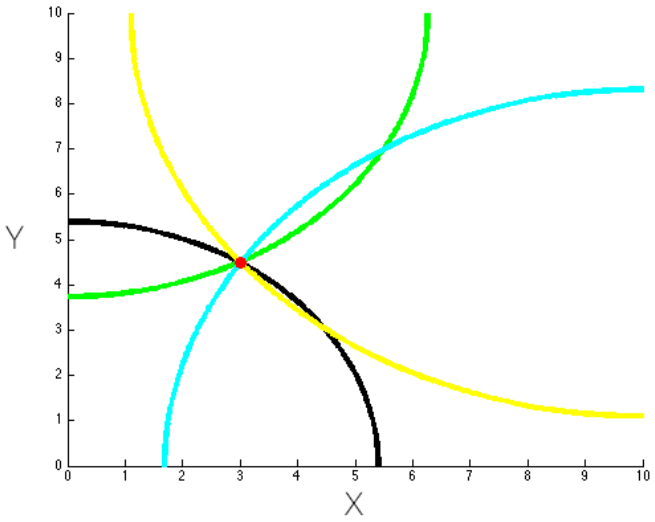

Figure 3.9: Exact application of the multilateration technique using the dataset data for the TP (3, 4.5).

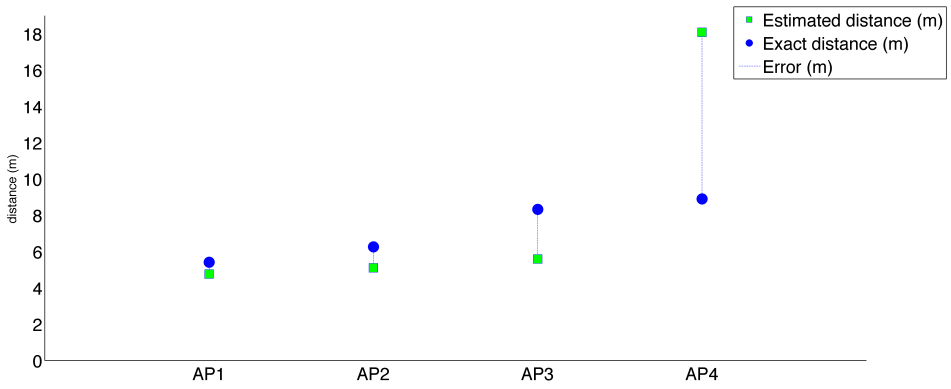

Figure 3.10: Estimated and exact distances from APs for TP(3, 4.5)

The red point on figure 3.9 is the exact position. It's not called estimated position because exact distances were used.

$$dist((AP_x, AP_y)(x, y)) = \sqrt{(AP_x - x)^2 + (AP_y - y)^2} \tag{3.2}$$

In real scenarios exact distances are not achievable, meaning that estimated distances should be used. To find the estimated radius distance for each circle, one should remember that in real situations the objective is to convert the received RSS level into a distance. To achieve this, it's necessary to use a model that is able to perform such conversion. The used model is the FSPL ([59]) and it's represented by:

$$FSPL(dB) = 20 \log_{10} d + 20 \log_{10} f - 27.55 \tag{3.3}$$

where, d is the transmitter-receiver separation distance in meters, f is the signal frequency in MHz, $FSPL$ is the received signal strength path loss in dBm. Constant -27.55 is the recommended constant to use when working with frequencies in MHz and distances in m. The frequency f is set to a default value of 2400 MHz since Android devices cannot detect the frequency band of Wi-Fi access points. 2400 MHz is used because that's the most widely used frequency in Wi-Fi networks.

When using the estimated distance values, it's expected that the equation system 3.1 do not return only a point as solution. There are 3 possible outcomes:

- Circles intersect and the result is an intersection area;
- Circles do not intersect;
- A combination of the above situations.

To estimate a position based in estimated distances, a different approach should be used. The problem can be solved using the weighted non-linear least

Table 3.2: Example of input variables for the MATLAB's multilateration function. All values in meters.

APs coordinates		Estimated distance from TP (3, 4.5) to the APs
0	0	4.7565
0	10	5.0967
10	0	5.5884
10	10	18.0838

squares fitting ([46]). This solution is based in a non-linear minimization algorithm and fits a non-linear model to a set of data.

The algorithm was first tested in the dataset using MATLAB and then implemented in the Android platform. First an estimation of the distance for every TP of the dataset was made using the equation 3.3. The result of this estimation for the TP (3, 4.5) is shown in table 3.2. After those estimations were complete, the previously calculated data was used as input to a MATLAB function that implements the multilateration algorithm.

The output of the multilateration function for this data is shown in figure 3.11. The red dot represents the real position and the blue dot represents the estimated position. Colored curves represent part of the circles around the APs. The effects of the uncertainty included by the estimated distance are visible when comparing figure 3.11 with figure 3.9. The circles do not intersect in a single point, and in this case, the circle around the AP4 (10, 10) is outside the dataset boundaries (10 x 10 meters) since its estimated distance is 18.038 (table 3.2.) This operation was performed for all TPs and the obtained results are in chapter 5. They are presented alongside with a comparison between this approach and the fingerprinting approach.

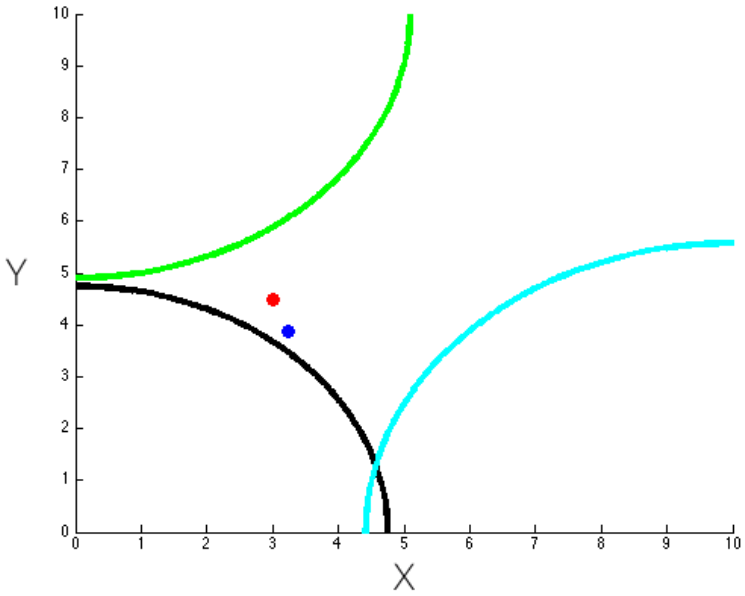

Figure 3.11: Output of the MATLAB's multilateration function for the TP $(3, 4.5)$.

This page was intencionally left blank!

Chapter 4

Android Applications

This chapter showcases the used tools and developed applications. There are three developed applications to be used in different scenarios. Two of the developed applications were developed to collect values to populate the dataset, one for the standing still dataset and the other for the in motion dataset. The final application uses the algorithm with the best performance to estimate positions in indoor environments. The final application is based on the standing still dataset application because the procedures to populate the dataset are the same to populate a new dataset in a different scenario. All the applications work with the fingerprinting technique. The Standing Still application has implemented the KNN, WKNN, MMSE and MAP algorithm. However the final application, only has the MMSE algorithm since it proved to have better results.

4.1 Tools

A set of tools were used while developing this thesis. These tools consist of both hardware and software and they are relevant to the outcome. Hardware tools should be analyzed by future researchers, since their specifications can be of interest.

4.1.1 Hardware

- Alcatel One Touch Pop C7 - The used smartphone. It has built-in accelerometer, magnetometer, gyroscope and Wi-Fi. Part of the hardware specifications are presented below:

Operating System (OS)	Alcatel One Touch Pop C7 Dual 7040D
Operating System	Android 4.2.2 Jelly Bean
OS Upgradeable	Information not registered
Chipset and Hardware	**Alcatel One Touch Pop C7 Dual 7040D**
CPU / Processor	MediaTek MT6582M / ARM Cortex-A7
Processor Clock / Cores	1300Mhz 32bits Quad-Core
RAM Memory	512MB LPDDR2
GPU Graphical Controller	ARM Mali-400 MP2
Internal Storage	4GB (2GB user available)
External Storage	Until 32GB microSD, microSDHC

Figure 4.1: Smartphone's hardware specifications.

- Wi-Fi APs - The used APs were 4 SMC7901WBRA2. The wireless specifications are below.

Maximum Wireless Speed:	54 Mbps
WiFi standards supported:	802.11b (11 Mbps) 802.11g (54 Mbps)
Wifi security/authentication:	WEP WPA (TKIP) WPA2 (AES) 802.1X (EAP-MD5/TLS/TTLS/PEAP) Wireless MAC Address filtering SSID Broadcast disable
WiFi modes:	Access point
external antenna(s):	1
Antenna gain:	2 dBi
WDS compatible:	✓
WMM (QoS):	✓

Figure 4.2: SMC7901WBRA2 wireless APs specifications.

4.1.2 Software

- Android Studio - An integrated development environment (IDE) to develop Android applications [60].
- MATLAB (matrix laboratory) - A numerical computing environment and programming language developed by MathWorks [61].
- Android Software Development Kit (SDK) - which includes UI components, databases, and Application Programming Interface (APIs) to access important devices of the smartphone, as is the case of the Wi-Fi and all sensors.
- Airplace [49] - Fingerprint based indoor positioning application for Android. This application was tested and it is described in chapter 3. All the algorithms used in this application were adapted to the data model developed in this thesis.

4.2 Dataset Applications

4.2.1 Standing Still

The Android application developed to collect the standing still dataset consists of one Android Activity called `Logger`. The most relevant Android APIs used in this Android Activity are:

- `android.net.wifi` - Wi-Fi APIs provide means by which applications can communicate with the lower-level wireless stack that offers Wi-Fi network access. Almost all information from the WiFi device is available, including the connected network's link speed, IP address, negotiation state, plus information about other networks in range. Other API features include the ability to scan, add, save, terminate and initiate Wi-Fi connections [62].

- `android.hardware` - Provides support for hardware features, such as the camera and other sensors [63]. It is used to get readings from the magnetic sensor. This is done using the class `SensorManager`, to select the magnetic sensor and it is registered to sample magnetic field values (For x, y and z axis) each second.

- `android.database.sqlite` - Contains the SQLite database management classes that an application uses to manage its own private database. This API is used to create the dataset's database [64]. In `Logger` it is used to store the magnetic sensor and Wi-Fi readings into the database.

Static dataset application's work flow is shown in Figure 4.3 where the variable $count$ represents the number of samples taken for a certain position (varies from 0 to 9, a total of 10 samples). $snumber$ represents the number of positions scanned in total (one increment in $snumber$ variable means that 10 samples were taken in a certain position). Android's `EditText` widgets were used to collect the position coordinates inputed by the user.

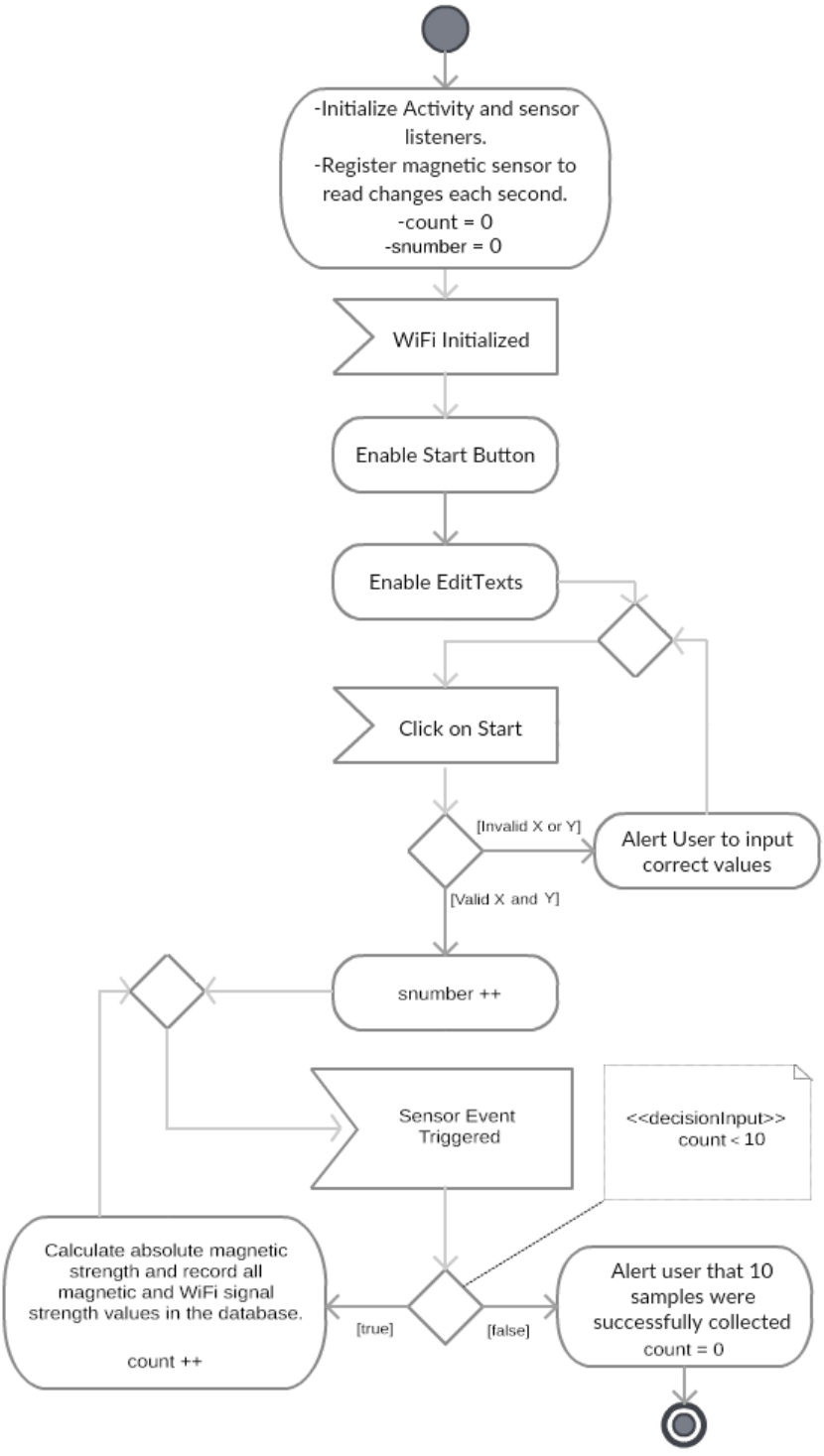

Figure 4.3: Flow diagram to the application used to build the standing still dataset.

4.2.2 In Motion

The Android application used to build this dataset periodically stores readings from Wi-Fi APs signal strength, accelerometer and magnetometer. Like the application used to build the standing still dataset, this application has one similar activity called `LiveLogger`. The most important used APIs are the same used to build the standing still dataset (`android.net.wifi`, `android.hardware` and `android.database.sqlite`), however, in this case, there was a need to get reading from the accelerometer (`android.hardware`) so changes were made to the sensor event listeners. First, an adjust in the reading times was made. The in motion dataset requires faster sampling times when compared to the standing still dataset because motion readings are being extracted as well. Both sensors listeners were registered with the Android default value `SENSOR_DELAY_NORMAL`. This value corresponds to a sampling period of 200 milliseconds, which is enough to get good readings and at the same time perform all the necessary calculations (represented in 4.4) in between the sample readings.
The flow diagram 4.4 shows the activity `LiveLogger` flow in detail.

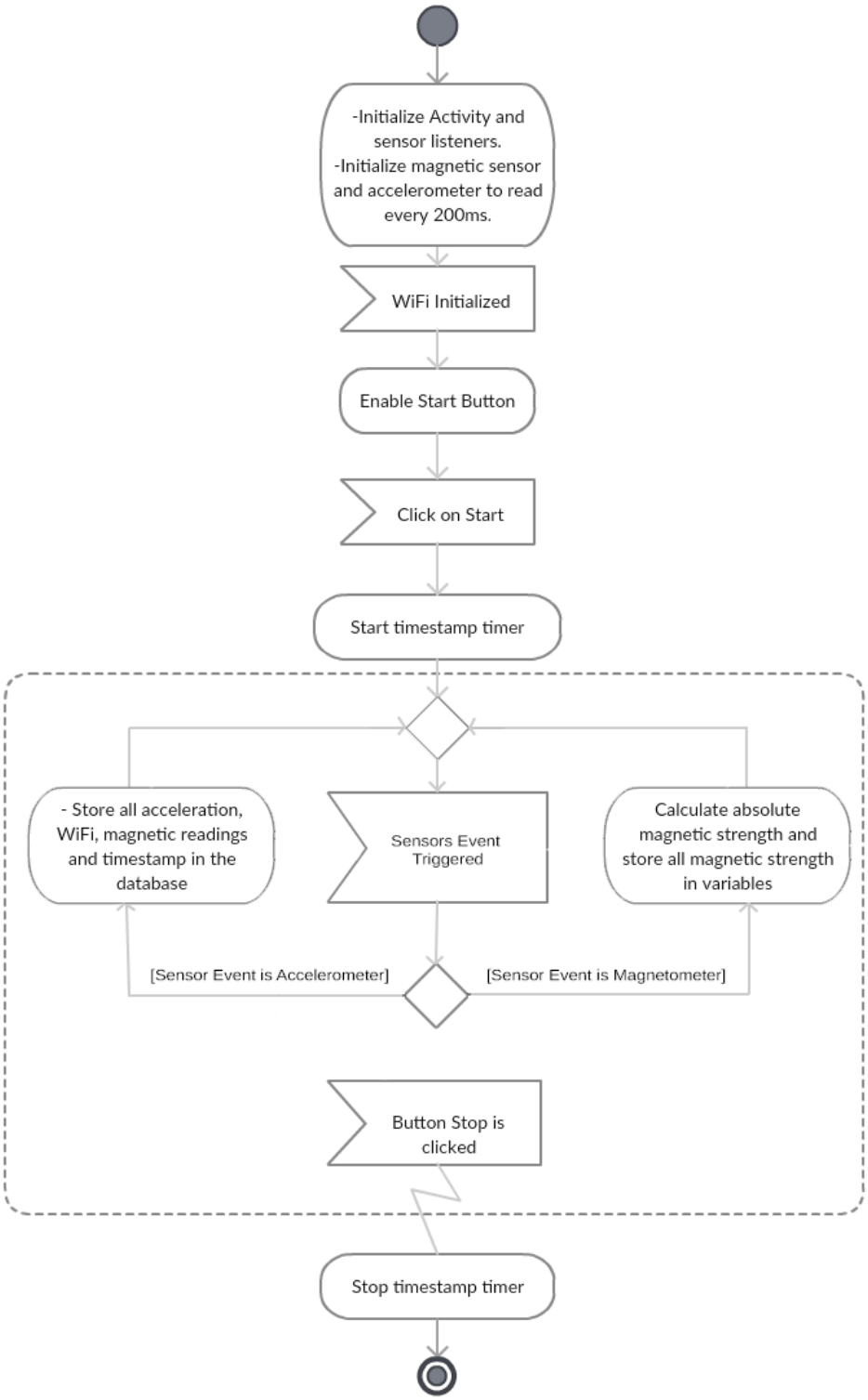

Figure 4.4: Flow diagram to the application used to build the in motion dataset.

4.3 Indoor Positioning Application

An Android application to estimate positions in indoor environments is described in this section. As shown in figure 4.5, the indoor positioning application has 4 Android activities. `MainActivity` is the entry class where the application starts. This class has two buttons and depending on the user action, different activities are created. If the user wants to create a new dataset, `Logger` activity starts. On the other hand, if the user wants to receive position estimations using an already existing dataset, `SelectDB` activity starts and prompt the user with a list of the available datasets. The user then selects the dataset and `Tracker` activity is started. This activity will provide position estimation based on the smartphone's WiFi RSS in real time alongside with the current smartphone's heading orientation.

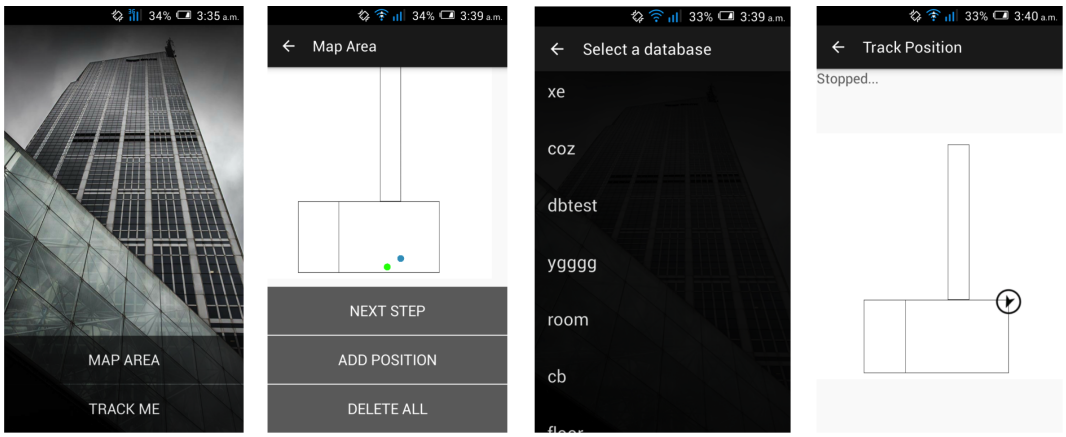

Figure 4.5: Screenshots of the developed Android application.

4.3.1 Architecture

The indoor positioning application runs on a smartphone with the Android operative system (OS). In figure 4.6 the high level architecture of the overall system is presented. The diagram is divided into three layers:

- Smartphone - This layer contains the used smartphone hardware components. Magnetometer, accelerometer, WiFi transceiver and the touchscreen are relevant to the system.

- Android - Contains the used APIs to interface with the relevant hardware blocks.

- Indoor Positioning Application - Includes all the developed software. The software was developed for the Android platform so it will communicate with the hardware blocks using the Android's operating system layer.

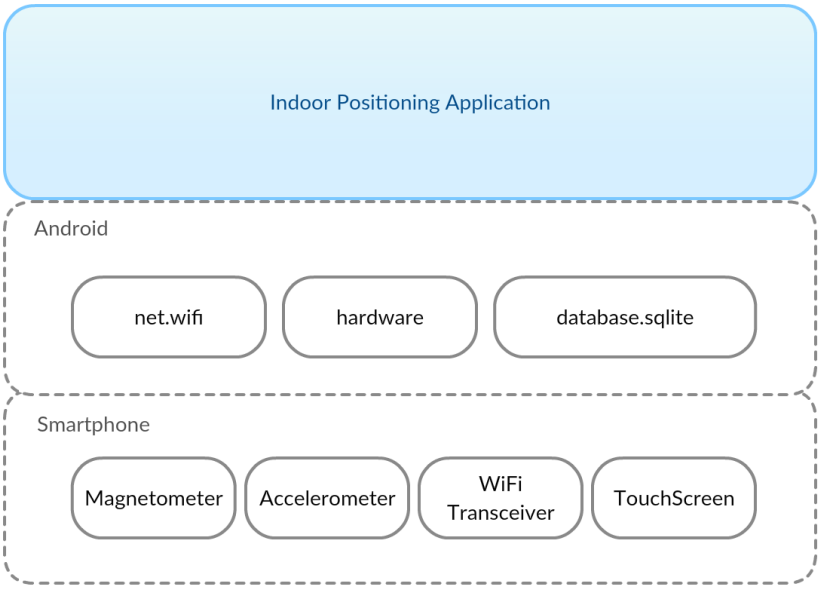

Figure 4.6: Overall system architecture.

The architecture of the indoor positioning application layer is presented using three class diagrams: the first presents the used Android Activities and their relations; the second describes in a deeper detail the `Logger` activity that is the same activity used to collect the data related to the standing still dataset; the third refers to the `Tracker` activity. `Logger` and `Tracker` activities are presented with

more detail because of their importance in the application. The class diagrams are presented using the Unified Modeling Language (UML) [65].

Figure 4.7 shows the relations between the activities and a detailed overview of `MainActivity` and `SelectDB`. All activities are compatible with different version of Android platform. `MainActivity` is launched whenever the application starts and it's the entry point to the application. This activity launches `SelectDB` and `Logger`. `SelectDB` is used to list the existent databases and it launches the `Tracker` activity. All the Android activities presented in 4.7 should inherit from the Android OS's `Activity` class. However in this case it is inheriting from `AppCompatActivity` (this Android OS class inherits from the Android `Activity` class) in order to use the support library action bar features.

The application is structured in 5 different packages:

- Package Adapters - Includes all Android Adapters [66] to the used lists.
- Package Structures - Data structures and data handling classes.
- Package Utils - Wraps utility classes such as the algorithms and the databases class which has all the operations related to the databases.
- Package Views - Contains custom views.
- Package Wifi - Has wifi related classes.

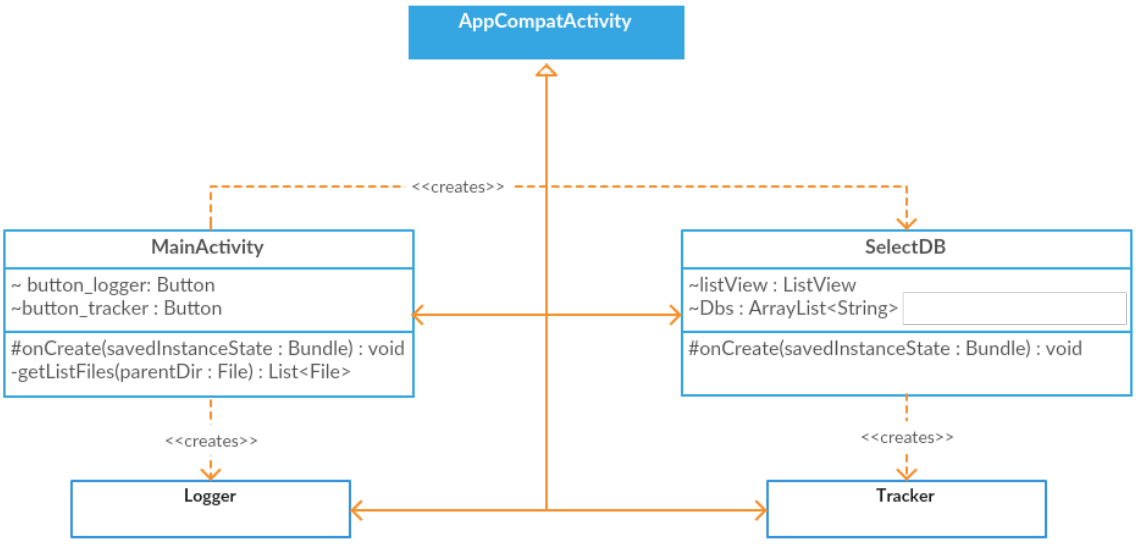

Figure 4.7: Android application's class diagram.

Figure 4.8 describes the `Logger` activity and its relations with other classes. `Logger` activity uses sensor readings, so it has a realization relation with the class `SensorEventListener` class. The WiFi Package is also used by the Logger activity. `SimpleWifiReceiver` has a composition relation with Logger, so the life cycle of this class is dependent of the Logger activity life cycle.

`SimpleWifiManager` has a multiplicity relation with `Logger` activity, because an instance of this activity can have one or more `SimpleWifiManager` objects. In order to store data in the database, it has a connection with the `Databases` class inside the package Utils. This relationship also has a multiplicity, because `Logger` activity can have multiple `Databases` object instances.

Every time the user creates a fingerprint in the database, an averaged version of this fingerprint needs to be created. To temporarily store this data, the data structure `WifiMean` is created and has an unique relation (aggregation relation) with the activity `Logger`, because it is the only class that uses it.

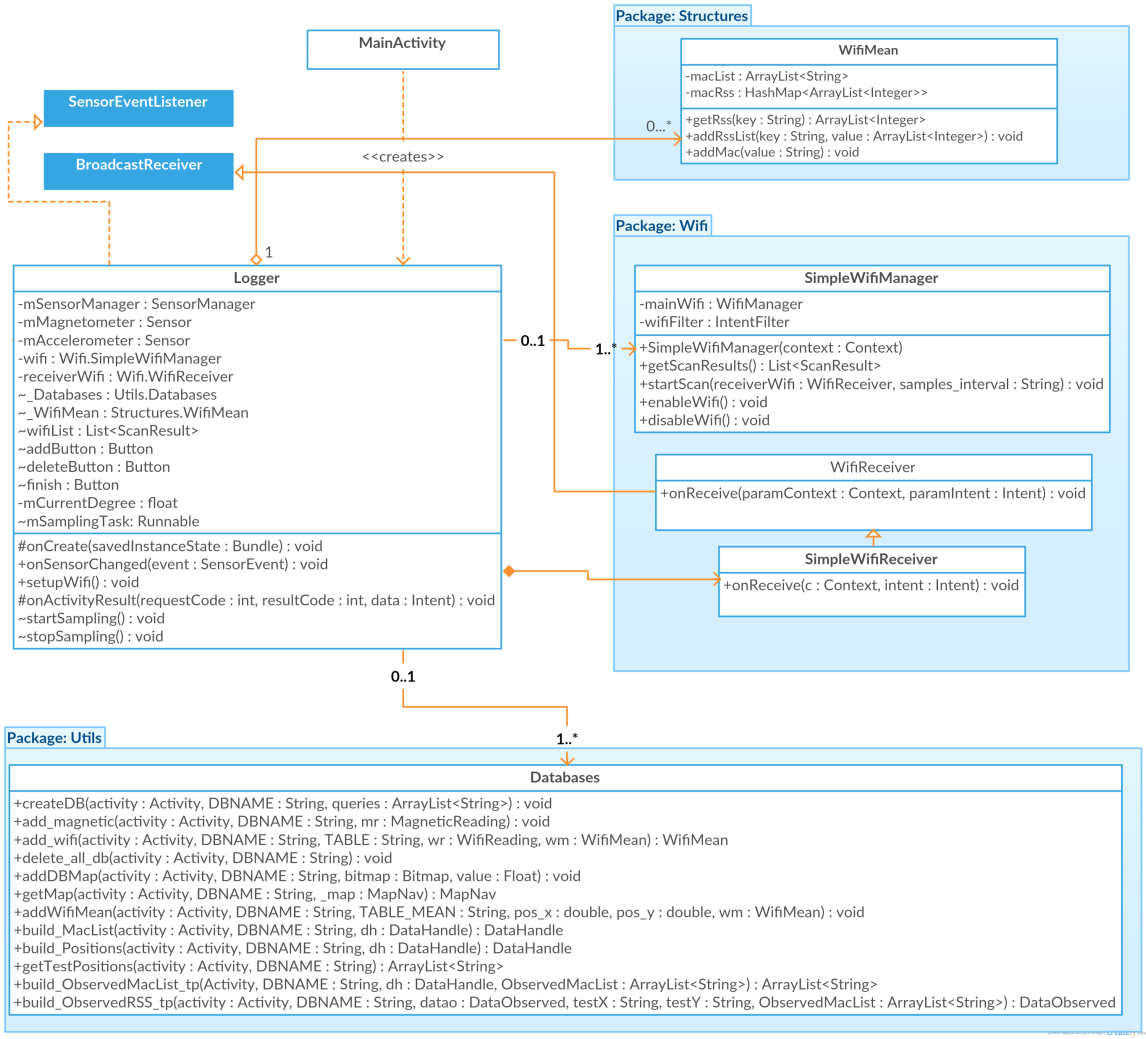

Figure 4.8: Logger Activity description and relations.

Figure 4.9 represents the `Tracker` activity and its relations with other classes. Like the `Logger` activity, this activity uses sensor readings, so it has the same relation with the Android's `SensorEventListener` and with the application's WiFi package classes. This activity reads from the database so, like `Logger` activity, a multiplicity relation with the `Databases` class is necessary. In the package Structures, activity `Tracker` is related with two different data structures with an aggregation relation: `DataHandle` that is responsible for the temporary storage of all dataset's related information; `DataObserved` that is responsible for the temporary storage of every observed WiFi readings. This data structure is updated

whenever a new WiFi reading is available (periodic event). These data structures enable the activity `Tracker` to perform calculations over their data and estimate a position using the indoor positioning algorithm.

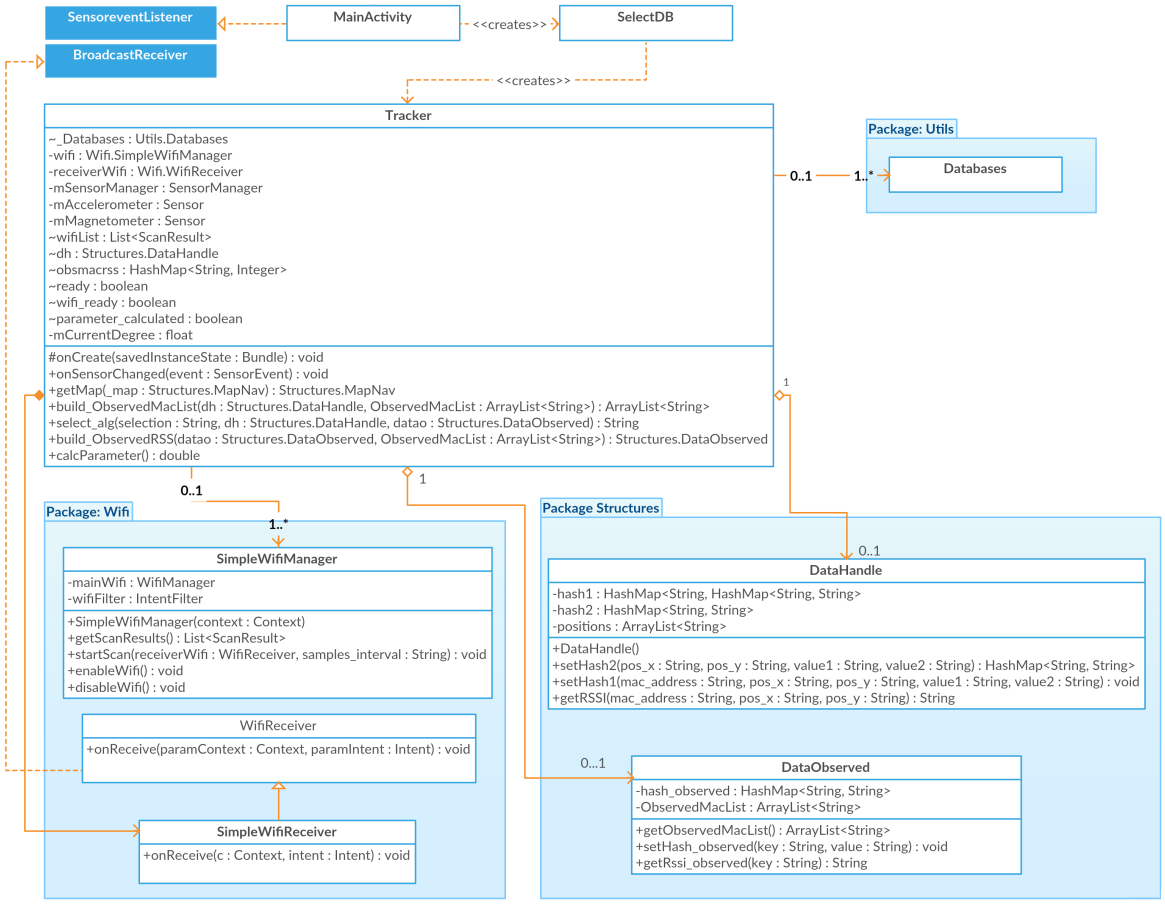

Figure 4.9: Tracker Activity and relations.

This page was intencionally left blank!

Chapter 5

Results

This chapter presents results obtained in all tests performed. First a dataset analysis is presented. In this analysis the data inside standing still dataset and in motion dataset is interpreted and relevant conclusions are extracted. The second set of results is related to the tested techniques. A set of results for fingerprinting and multilateration is discussed. Finally, theoretical optimizations over the dataset's data are applied to verify their impact in the position estimation error.

5.1 Dataset Analysis

In this section, the collected data in both datasets is analyzed. This analysis will be presented in two subsections: standing still dataset and in motion dataset. In the first part, Wi-Fi RSS and magnetic strengths will be discussed; In the second part, results will be discussed based on the observation of the recorded videos and analysis performed on the in motion database. To better understand the following analysis, it's necessary to understand how the axis are aligned into the smartphone. Figure 5.1 shows how the axis are distributed. It is important to note that magnetic fields are measured around the three axis while accelerations are measured along them.

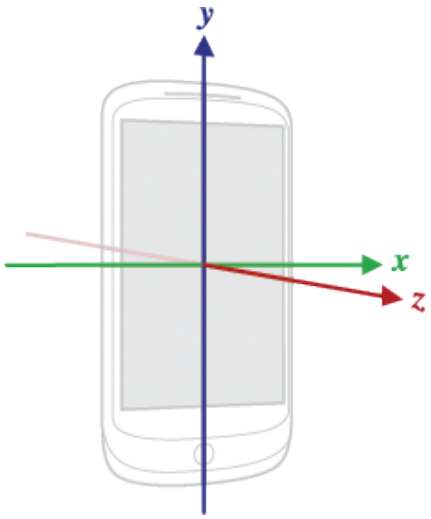

Figure 5.1: Smartphone axis description.

In both datasets the data was extracted from the databases and then analyzed using MATLAB. In the case of the in motion dataset, data recorded by sensors was compared and synchronized with the recorded video.

5.1.1 Standing Still

In the standing still dataset, Wi-Fi RSS is the main concern, however magnetic strengths are also considered. Data collected from all the 4 Wi-Fi APs was used to build 4 tables. Those tables contain the RSS for each AP and they were transformed into heat maps, using MATLAB, for an easier visualization (Figure 5.2). It's notorious that the Wi-Fi RSS gets smaller as the position gets more distant from the AP location.

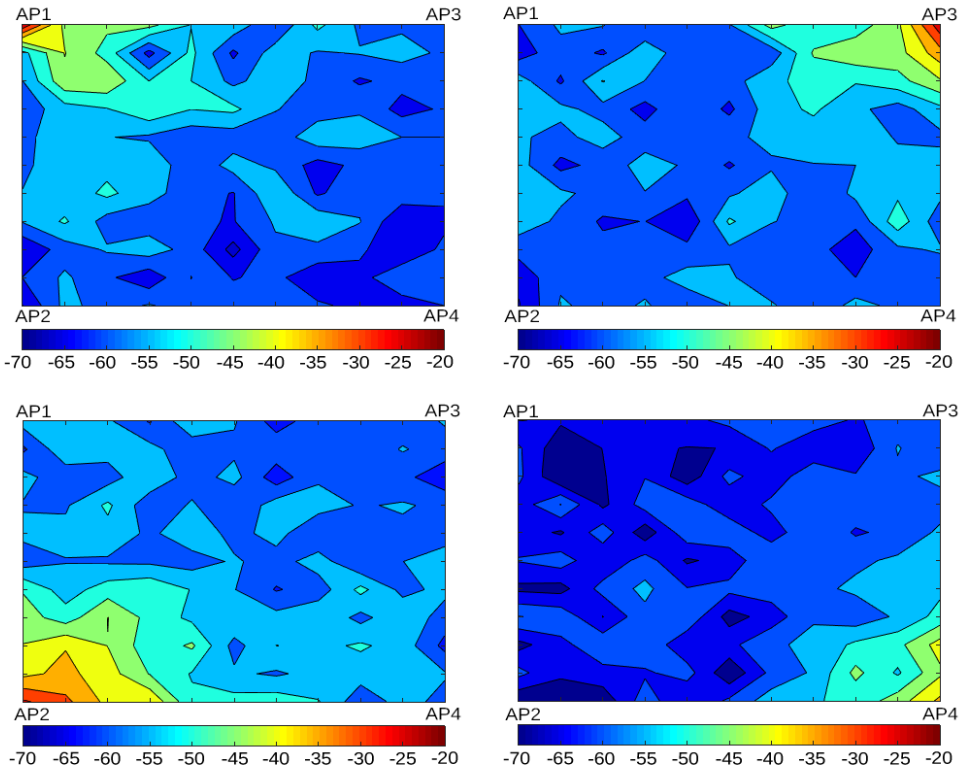

Figure 5.2: Wi-Fi signal strength heat maps from all APs. In each heat map is just represented the respective heat map. Top left: AP1; Bottom left: AP2; Top right: AP3; Bottom Right: AP4

The heat maps confirm the logarithmic like evolution of the Wi-Fi signal strength and they look as nearly as expected. It's visible that there are locations where the Wi-Fi RSS breaks the logarithmic like evolution and its AP Wi-Fi RSS gets stronger. This fact comes from signal variation and interference. However this instabilities can be minimized if the user is moving inside the room and its movement pattern is considered. Using probabilistic algorithms and movement patterns, this problem can be partially solved.

Magnetic field readings from the standing still dataset will not have impact on this thesis approach, however they are present in the database in case they are valuable to future researchers.

Each position has a magnetic field value. The magnetic field pattern inside the

room is presented in Figure 5.3 and it was generated using the absolute magnetic strength in each position. Absolute magnetic strength is calculated through the equation 5.1 where $AbsStrength$ is the absolute magnetic strength for each position and $strengthx$, $strengthy$ and $strengthz$ are the magnetic strength around each axis for a given position. Magnetic field data is useless when performing steady positioning, but when performing tracking it can be used to complement Wi-Fi RSS information, improving the accuracy of the system.

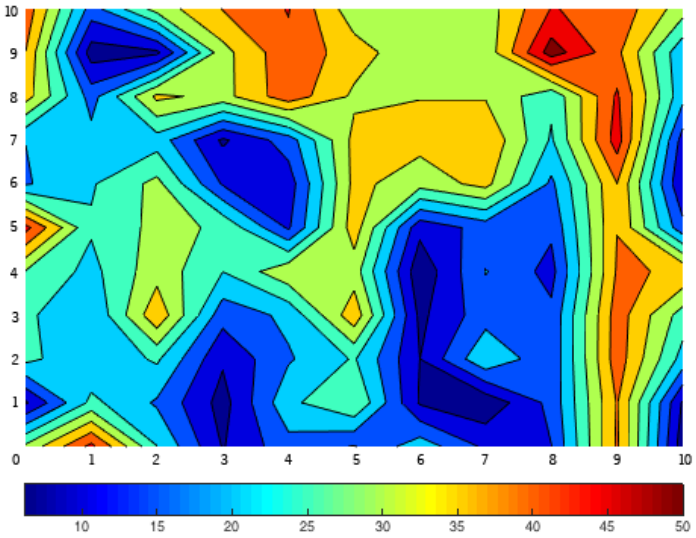

Figure 5.3: In motion dataset's magnetic strength pattern (values in μT).

$$AbsStrength = \sqrt{strengthx^2 + strengthy^2 + strengthz^2} \tag{5.1}$$

5.1.2 In Motion

In this subsection all the data is considered, meaning that Wi-Fi RSS, magnetic fields and acceleration values are analyzed taking in consideration the recorded videos in each experiment (refer to figure 3.6).

When moving, there are important aspects to consider: position and orientation.

Both of these aspects can be estimated using the information recorded in this dataset. Using the Wi-Fi RSS, position can be estimated and using the accelerometer and magnetometer readings, a good estimation to orientation can be achieved. As a complement to those important aspects, a step and movement detection can possibly help in the achievement of a better position estimation.

First, let's consider the Wi-Fi readings. All the readings were passed through the positioning algorithms to achieve a position estimation. To illustrate the position estimation achieved using the Wi-Fi RSS readings in the in motion dataset, the output of one of the used algorithms is presented.

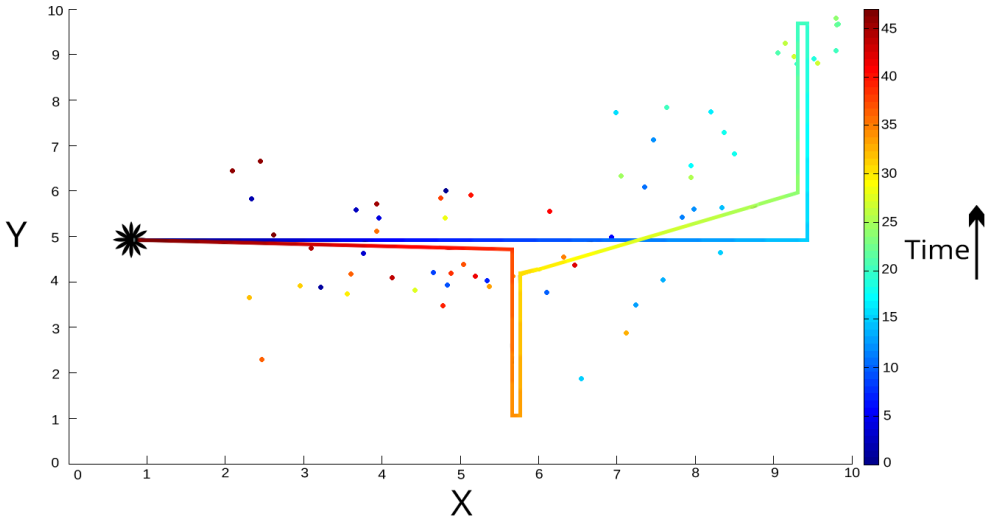

Figure 5.4: Position estimation using MMSE algorithm and readings from the in motion dataset. The points represent a reading thaken during the route simulation. The color shows a time scale in order to represent the time dimension alongside with the position.

Figure 5.4 shows the position estimation (circles) and the real route. It is visible that the estimated positions are near the real route, meaning that a position estimation is not only possible, but at the same time acceptable.

Looking at the orientation subject, a combination between magnetic read-

ings and accelerometer readings can give a good estimation of global orientation [67]. Android's `getOrientation()` public method can give a good estimation based on another Android's public method named `getRotationMatrix()`. The `getRotationMatrix()` public method uses accelerometer and magnetometer readings to build a matrix that will be used by `getOrientation()` public method.

Global orientation results can be interesting to use as a magnetic compass, however in indoor positioning applications, the achievement of a relative position that relates to the current map is the point of interest. A method to get the desired relative orientation was developed. This method consists in the addition of a new flag to the dataset. This flag has a reference value that is obtained by pointing the smartphone to the top of the map at the end of the fingerprint recording process. By doing this, the recorded orientation value is stored and it is used as offset when moving around the map. Instead of the magnetic north, now the reference is the top of the map, giving the desired relative position. Considering the experiment one, a video was created to show the effectiveness of this method. Screenshots of the experience are shown in Figure 5.5 and Figure 5.6. It is important to notice that when the user starts moving (Figure 5.5), he is facing the right side of the room and then, after he makes a turn to the right he is facing now the bottom side of the room (Figure 5.6).

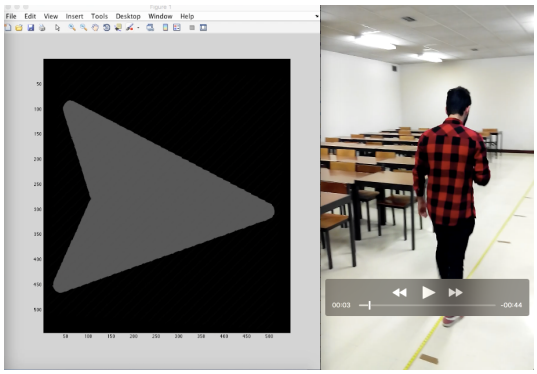

Figure 5.5: Orientation test when user is going towards the right side of the room.

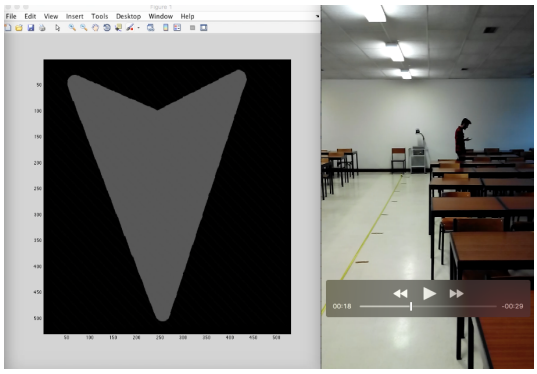

Figure 5.6: Orientation test when user is going towards the bottom side of the room.

Now that positioning and orientation have been analyzed, step and movement detection should be referenced. First, let's consider the step detection since it's used as base to the movement detection (in this thesis approach).

The values from accelerometers recorded during the in motion dataset experiments, were collected from the database and then analyzed in MATLAB. When analyzing the data, the goal was to get a good estimation of the number of steps given by the user. To get a good estimation of step number, the acceleration values from all the three axis should be transformed in an gravity free acceleration scalar using the formula in equation 5.2. This transformation is needed since it enables the detection of steps regardless of the smartphone's screen orientation.

$$AccelerationScalar = (\sqrt{acc_x^2 + acc_y^2 + acc_z^2}) - 9.8 \tag{5.2}$$

The next step is to define a threshold for the step detection. This threshold defines when a step is detected or not. If the acceleration scalar value crosses the threshold when increasing, a step is detected. Figure 5.7 shows an example of this detection for the experiment 1 and with a threshold of 0. The blue line represents the evolution of the acceleration scalar values and the red marker represent the detection of a step.

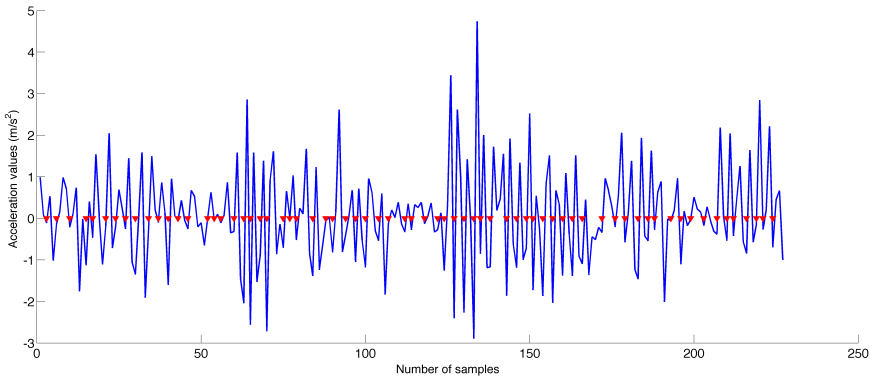

Figure 5.7: Steps detected for experiment 1 with a threshold value of 0.

As expected, the bigger the threshold value is, the least the sensibility will be. In order to estimate the best threshold value for the system, a series of tests were performed for the three experiments. First, the steps were counted for all the videos. This gives the real number of steps given by the user. Then an analysis with different threshold values gives an estimation of the optimum value. The closer the number of steps detected by the step detection algorithm, the better the threshold value is. Figure 5.8 shows the previously described test for all three experiments.

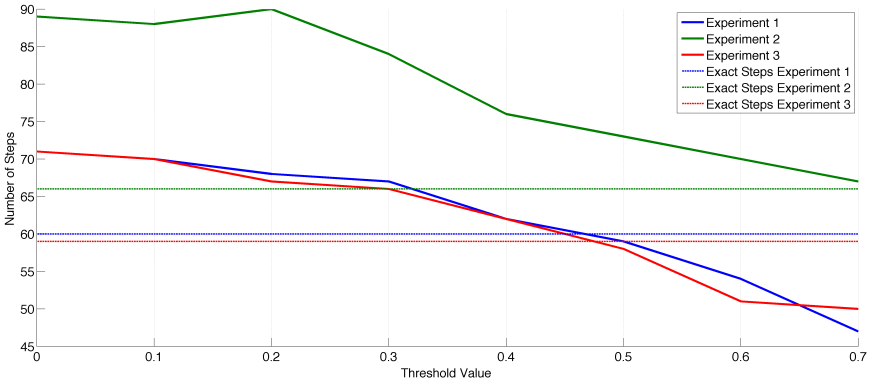

Figure 5.8: Steps detected in all experiments with different threshold values.

The solid lines represent the evolution of the number of estimated steps depending on the parameter variation. On the other hand, the dashed lines represent the exact number of steps counted in the video. After this analysis the

chosen threshold value was 0.5. With this threshold value the estimation gives a trustworthy number of steps for all the three experiments.

Looking at figure 5.7 it is not possible to detect a moment when the user is stopped (absence of steps) since the threshold is so small that any movement triggers a fake step detection. On the other hand, figure 5.9 shows some sample intervals where there are no steps detected, meaning no movement. The most evident interval of samples with no steps occurs between sample 110 and sample 130. If one look at the video for experiment 1, between the second 21 and 25 the user is turning at the same position, not giving any step. This time frame is exactly recorded in the samples between 110 and 130, meaning that a certain level of context awareness can be obtained with this step counting algorithm.

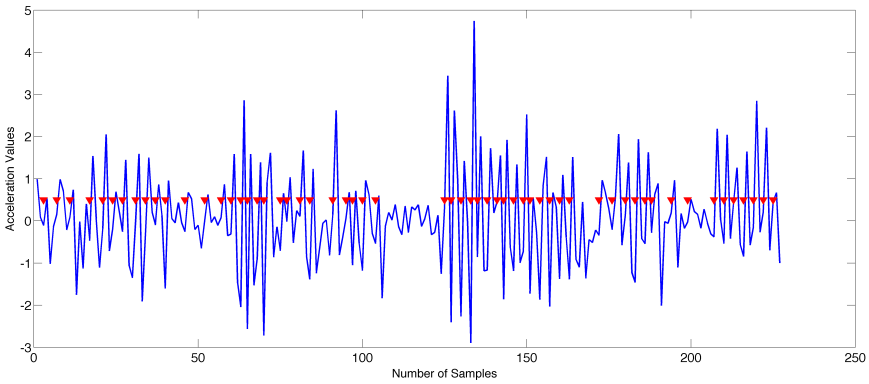

Figure 5.9: Steps detected for experiment 1 with a threshold value of 0.5.

Counting steps is important in certain applications, however, some of the times, the state of movement can become a more valuable insight. If the application is aware of the movement state (moving or stopped), possibly some improvements can be introduced to positioning systems in future researches. Thinking about that, a movement detection algorithm was developed. This algorithm is an attempt to detect the state of movement using the step counting algorithm as a base. Figure 5.11 shows the flow diagram for the movement detection algorithm. Basically, the algorithm detects the step (using the $acceleration_scalar$

and the $thresold$ variables) and counts the time between steps using timestamps ($timestamp$ and $last_timestamp$). If the time between steps is smaller than a threshold ($time_threshold$), the user is considered to be walking. On the other hand, in the case that steps are not detected after a certain $time_threshold$ since the last one, the user is considered to be stopped. The algorithm is simple, but it proves to work. In figure 5.10, the movement state during experiment 1 is shown. Two gaps are visible: one from around the sample 15 to around the sample 20 and the other from around the sample 110 to around the sample 125. Looking again at the video, it's noticeable that these gaps match moments when the user is stopped. The problem with this algorithm is that intervals of time when the user is stopped for less than two seconds will not be detected.

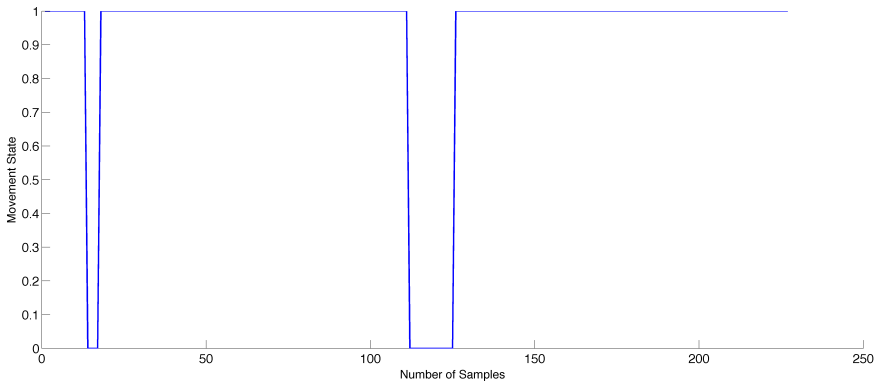

Figure 5.10: Movement state for experiment 1. State 1: Walking; State 0: Stopped.

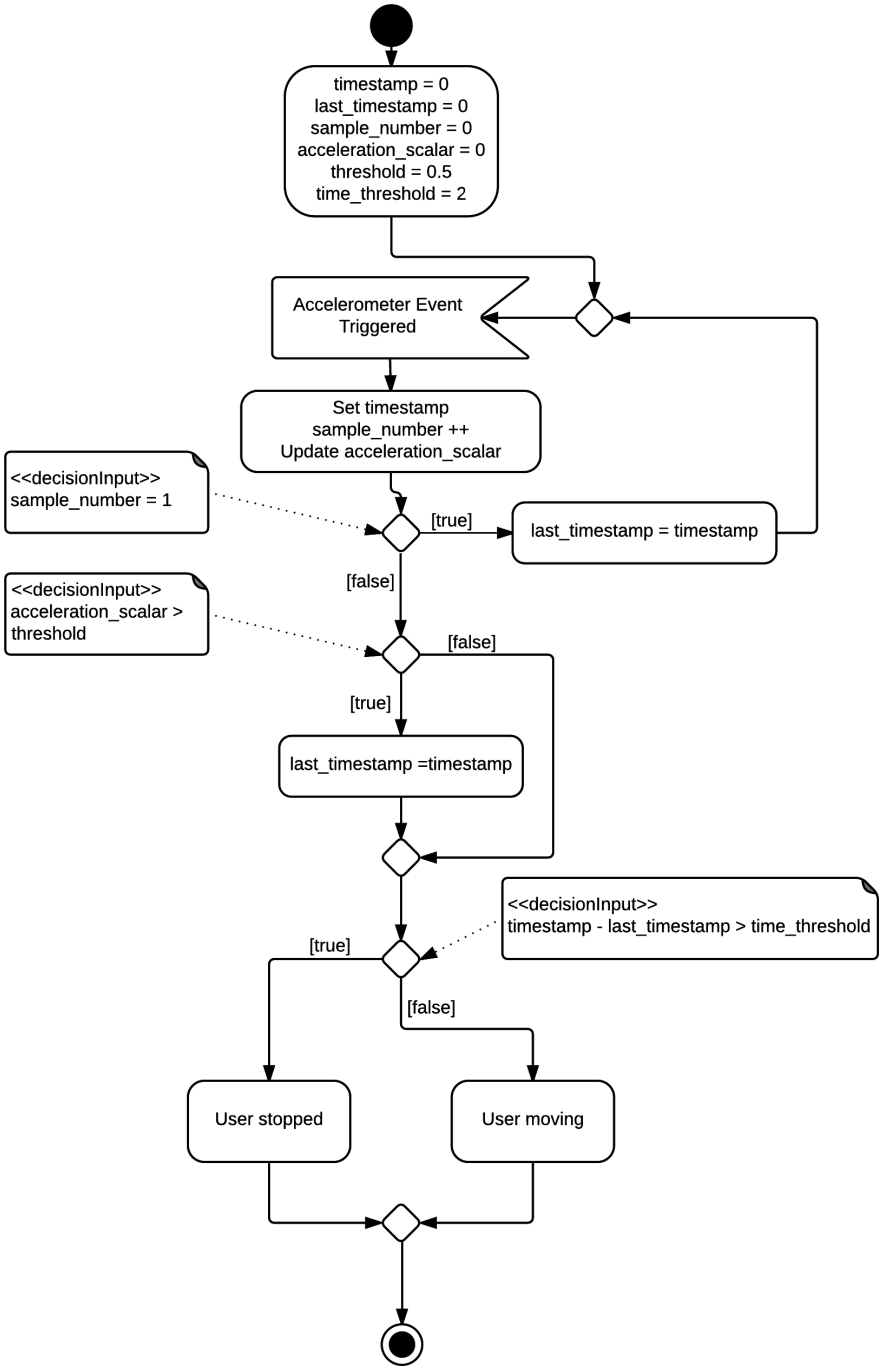

Figure 5.11: Flow diagram to estimate the movement state.

5.2 Fingerprinting Standing Still Tests

The performance of the fingerprinting approach was evaluated based on the relation between the estimated position and the real position, using different K and σ parameters deppending on the algorithm. A total of three experiences for each dataset test position was performed. The first experience was made with the parameters set to 1, that is the default parameter to use when there are no TPs. Figure 5.12 shows the results of the experience for all the four algorithms. In this first experience, the average error is acceptable when comparing to the approaches in chapter 2 . However, there are results in figure 5.12 that go too far from the exact position, being the case of position (5, 5) and (5, 6.5). These big deviations can result in an unacceptable system due to its low accuracy. MMSE algorithm proven to output the best results compared to the other three algorithms, however the difference is not that significant, being in average 29 centimeters.

The second experience sets the parameters K and σ to 15, which is the biggest value they can take. As expected, the averaged results are similar to the first experience. That happens because whenever the parameter value goes beyond the ideal, the error increases and that's why an approximation to the ideal value is a good attempt to get a better solution.

Figure 5.13 shows the results for the second experience. The best algorithm appears to be the WKNN algorithm followed closely by the KNN algorithm. The deterministic algorithms seem to perform better than the probabilistic algorithms, when using higher parameter values. Probabilistic algorithms seem to lose more accuracy than deterministic ones, when the parameter value goes beyond the ideal.

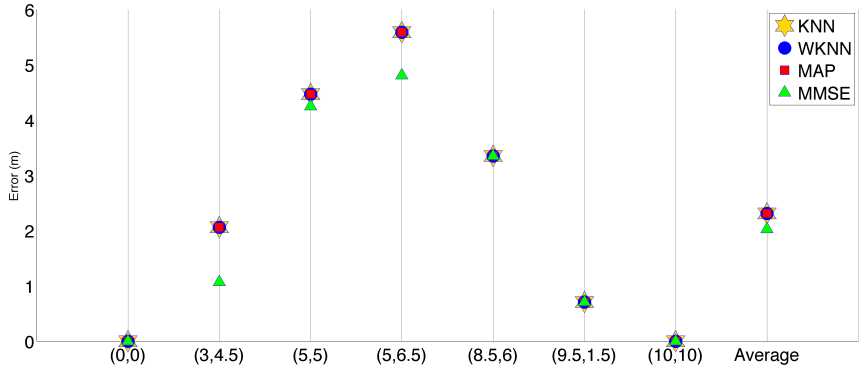

Figure 5.12: Results with the $K = 1$ and $\sigma = 1$.

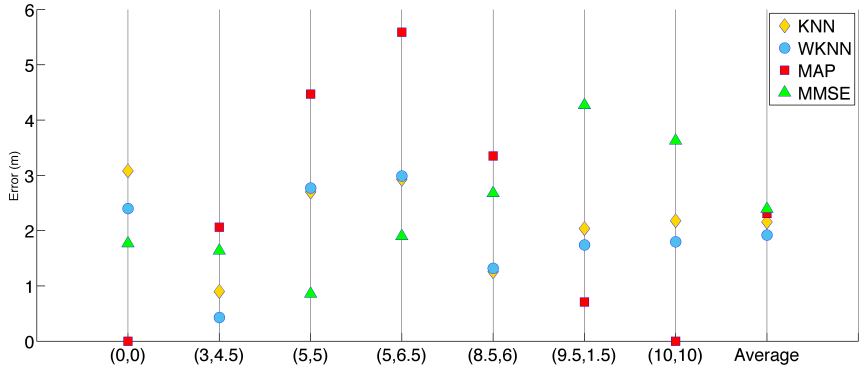

Figure 5.13: Results with the $K = 15$ and $\sigma = 15$.

Experience three (Figure 5.14) implements a smarter approach to the problem. It sets the parameters K and σ to the average of the sum of each TP's ideal value. This time, K and σ have different values, depending on the result returned after the deterministic and probabilistic parameter's estimation (refer to algorithm

5). Against the first two experiences, in the third one, all the algorithms, except the MAP algorithm, return better results, with more accuracy in average and less deviation from the exact position. The improvements are significant and in MMSE algorithm they are, in average, more than one meter more accurate. MMSE returned the best results of all the tested algorithms, however KNN and WKNN also returned good results and both can be used to estimate position, since their estimation has an acceptable error.

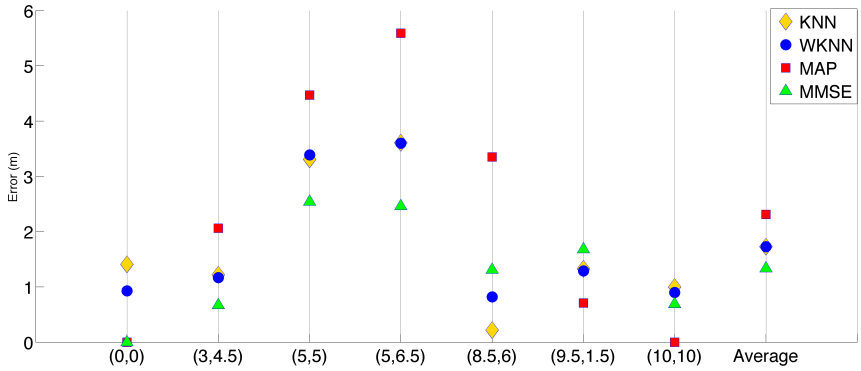

Figure 5.14: Results with the best fitting K and σ parameters.

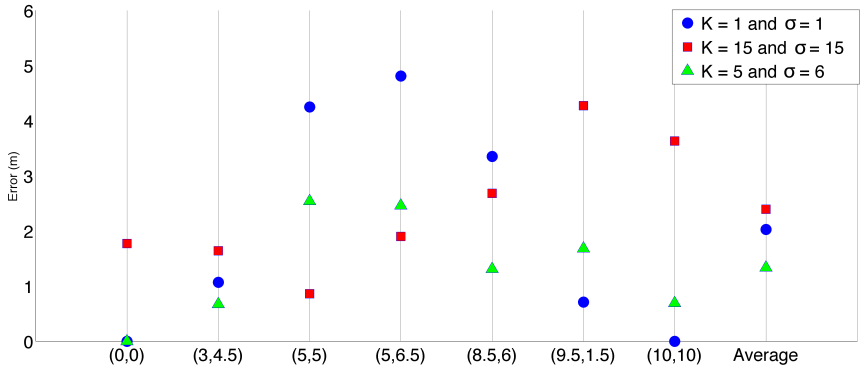

Figure 5.15: Influence of the parameters K and σ in the final position estimation with the MMSE algorithm.

Finally, looking at figure 5.15, it's visible the impact that K and σ have on the final results. The figure represents only the result for the algorithm with the best performance (MMSE) and in average, the final averaged result improved almost one meter when the parameter correction is performed.

Fingerprint based positioning algorithms can give a good estimation of the real position, however there are drawbacks that must be considered before the selection of this technique. The first drawback is that the estimation obtained without TPs (Experience 1 and 2) is plausible but the deviations from the exact positions can induce in wrong estimations (tests shown that the error between estimated position and real position went up to 5.59 meters). The best way to use this technique is to make sure that a set of TPs are build after recording all the RPs. However there is no need to record a big number of TPs. Figure 5.16 shows the position estimation error for the 7 TPs and for 121 TPs. These 121 TPs were generated by selecting a random sample from each RP. The difference between the results is small, being actually bigger when there are more TPs. Another drawback is that fingerprint based positioning systems consume a noticeable amount of time and labor during the collection of Wi-Fi RSS data. This amount of time and labor is even higher if there is a set of TPs in the database.

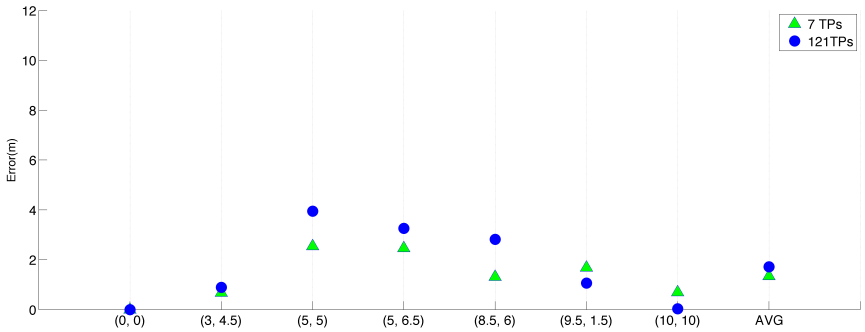

Figure 5.16: Position Estimation errors based on different number of TPs (MMSE algorithm).

Fingerprint positioning systems are dependent on the density of points collected. However there is no need to record a huge amount of RPs. Figure 5.17 shows the position estimation error for three different situations: one is a test performed with all the RPs, the other a test with 50% of the RPs and the other with 25% of the RPs. The results are not that different. When the number of RPs were decreased to 25% there is an increase in the average error of almost 1 meter, but the estimation still an acceptable estimation. In the case of 50% of the RPs, when comparing with the case of 100% of the RPs it's visible that they are almost equal, meaning that the reduction of the number of RPs by 50% didn't affect the position estimation. The RPs used in the tests for 25% and 50% were randomly selected.

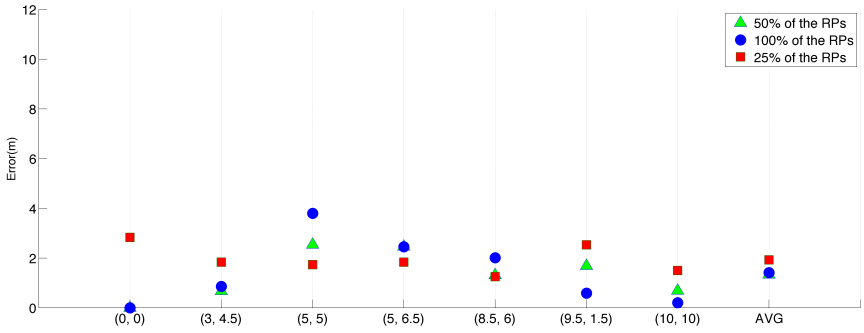

Figure 5.17: Different position estimation erros based on the dataset's density (MMSE algorithm).

5.3 Multilateration Standing Still Tests

Multilateration based systems are expected to have less accurate estimations, since they estimate the position based in distance estimations. Although, the experience based on this thesis dataset shows that multilateration gives acceptable results (few meters). The errors induced by this calculation are presented in figure 5.18.

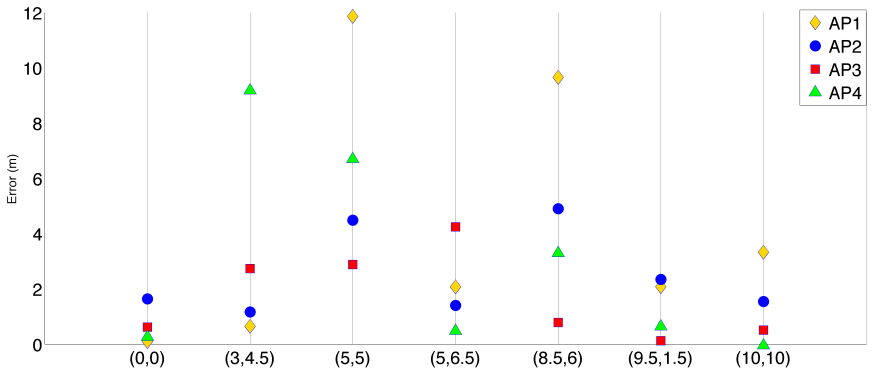

Figure 5.18: Multilateration's technique distance estimation erros. These errors are related to the distance estimated between each TP and the APs.

Figure 5.18 result from a test performed for each dataset TP. For each TP, the distance was estimated using the previously mentioned FSPL (refer to chapter 3). The biggest problem with the FSPL is that, as its name suggest (Free Space Path Loss), it was developed to estimate distances in free spaces where there are no objects between the point and the AP, there are no shadowing and a set of another undesired factors. It happens that in indoor environments, a significant part of these undisired factors are present. For that reason, the distance estimation with this method was expected to return poor results. The results present in figure 5.18 show that most part of the errors are between 0 and 6 meters, however when the previously discussed undesired factores come into scene, the error increases up to 12 meters, which is unacceptable.

FSPL establishes a logarithmic relation between Wi-Fi RSS and distance. To verify the real evolution of the RSS in function of the distance, another test was performed. The results of this test are represented in figure 5.19. The test was performed using all the RPs in the dataset and it envolved all the samples recorded in each of these RPs. For each RP, the maximum, minimum and average RSS value was extracted and ploted in a graph, so one could see the signal strength variation for each point. After all the points were plotted to the graph, a cubic fitting was applied to the maximum and minimum values in each RP, giving a range of the

signal strength variation in function of the distance.

There are some points where the variation is just to big and since the FSPL is not capable to adapt to this changes, multilateration algorithm becomes unpredictable for indoor positioning estimation. A brief observation of figure 5.19 can lead to the previous conclusion: as an example, looking at distance 5 meters, the signal strength expected by the FSPL algorithm would be around -50 dBm. However, looking at the minimum value that a signal can have in that same position, it is visible that for the same distance, the signal strength can be around -60dBm. If that happens, FSPL would return an estimation of around 12 meters, because that is the distance that its curve matches the signal strength of -60dBm. These RSS variation will increase their impact on the result as the user moves away from the AP because of the slower FSPL logarithmic evolution.

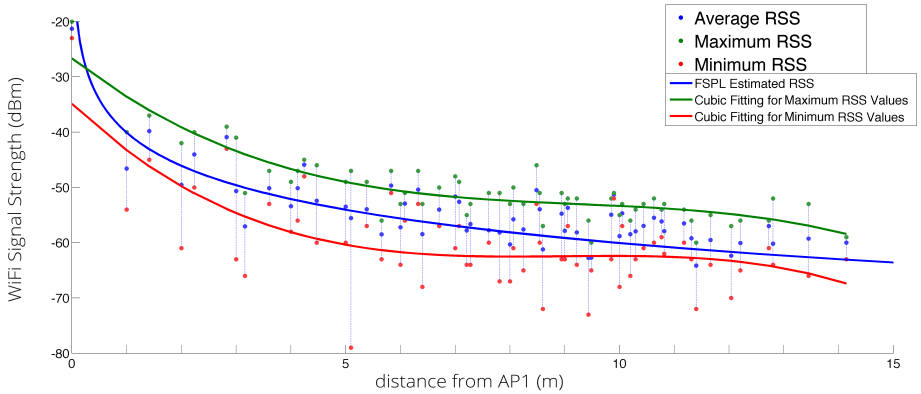

Figure 5.19: Comparison between the FSPL and the real distance estimations considering the variations between the maximum and minimum value of RSS in each dataset RP. This test was performed on the values of AP1.

Finally, figure 5.20 presents the final position estimation error using the TPs as input to the algorithm. The error varies from 0 to 4 meters and presents a nice averaged error of 1.44 meters. On the firs look it looks a good result, but taking in account what was analyzed before, one should understand that multilateration can, sometimes, return completely wrong estimations due to its distance estimation

problem. This situation will be confirmed later in this chapter when analyzing the results obtained with multilateration for a route performed by the user.

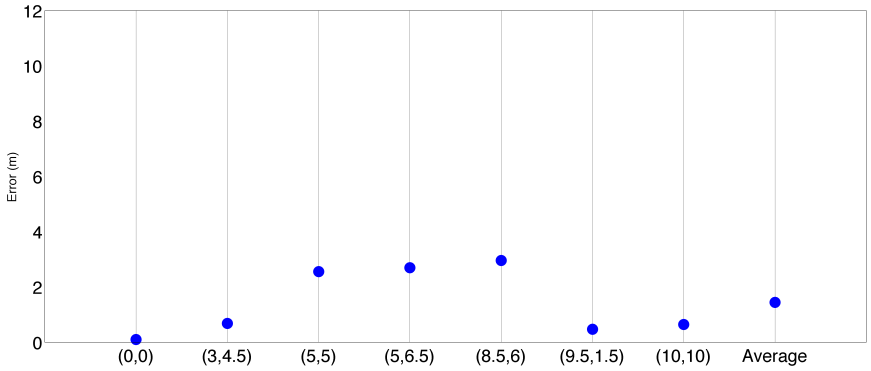

Figure 5.20: Errors in the position estimation using multilateration method.

5.4 In Motion Tests

The route tests are tests performed using the data from the in motion dataset experiment 1. This data was collected by the user while moving around the room where the dataset was recorded. Previously in this chapter, a static analysis was performed for each algorithm and averaged error for each one of them was obtained and compared. In this section the algorithms will be compared based on a set of images generated in MATLAB. These images were built using data processed by the algorithms in the smartphone. The data collected by the in motion dataset experiment 1 was used as input to the algorithms and the 7 TPs were used to optimize K and σ parameters. The output data was stored in a database and analyzed in MATLAB.

The following figures, 5.22, 5.23, 5.24, 5.21 and 5.25 show the results obtained with each algorithm for the route. The black lines represent the actual route with the big black point as a start and end point while the circles represent estimated positions. Points have a color scale that represent time. The total route's length

was around 46 seconds and the color scale goes gradually from blue to red along time.

The best way to observe the figures is to look at the point dispersion and color evolution. The more disperse the points the least accurate is the position estimation. The figure with the least point dispersion and that shows the best estimation is the figure 5.21 that represents the results obtained with the MMSE algorithm.

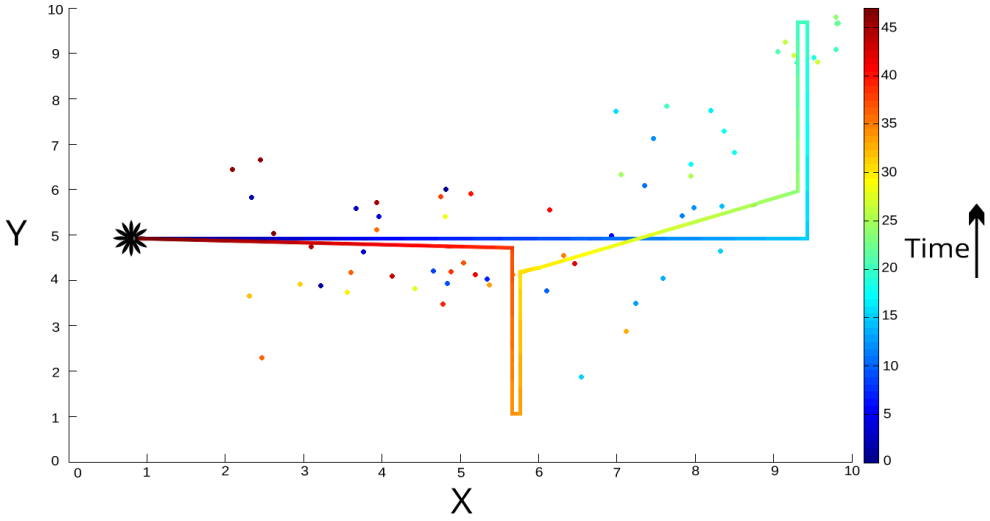

Figure 5.21: Route results for the algorithm MMSE.

Deterministic algorithms performed a little worse than the MMSE algorithm, however the point dispersion is not too big, allowing to have a perception of the performed route. MAP algorithm, in figure 5.24, performed poorly, since it is hard to identify a pattern that relates to the performed route.

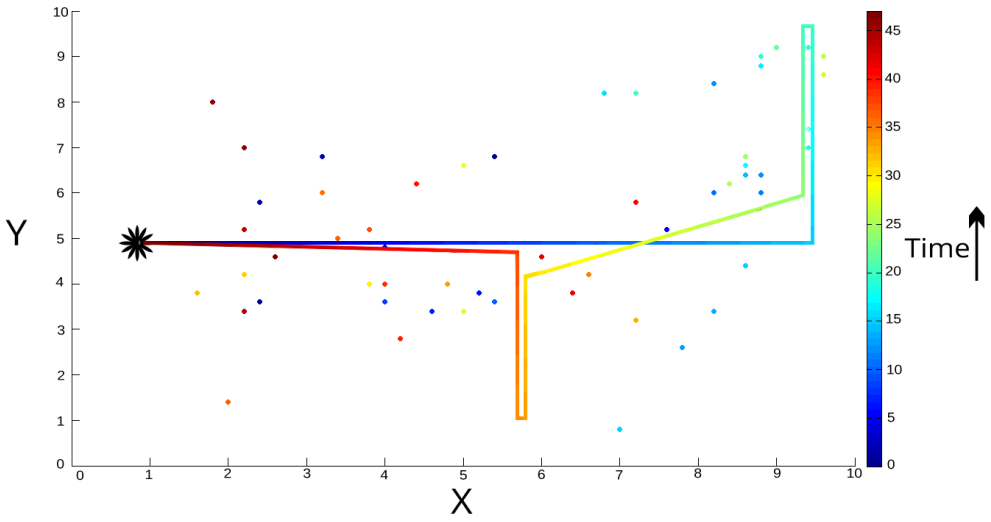

Figure 5.22: Route results for the algorithm KNN.

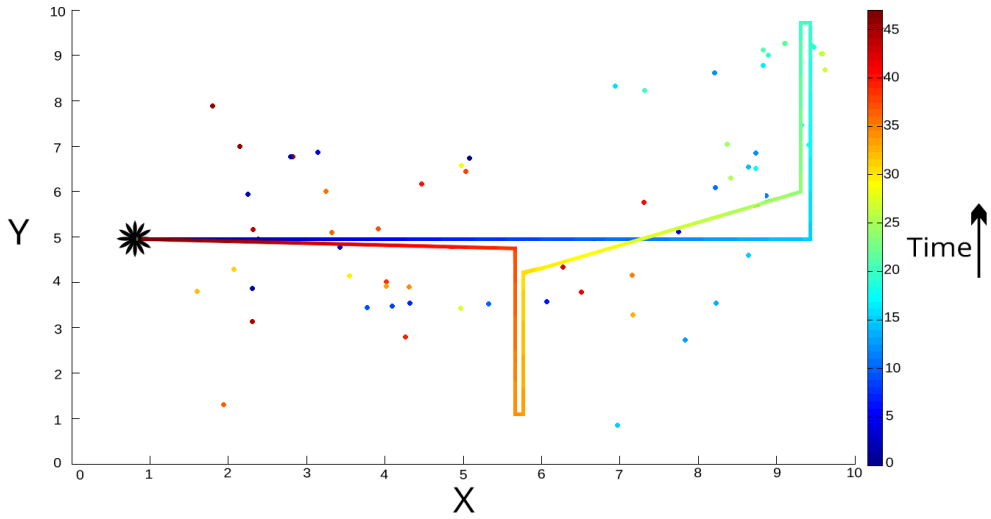

Figure 5.23: Route results for the algorithm WKNN.

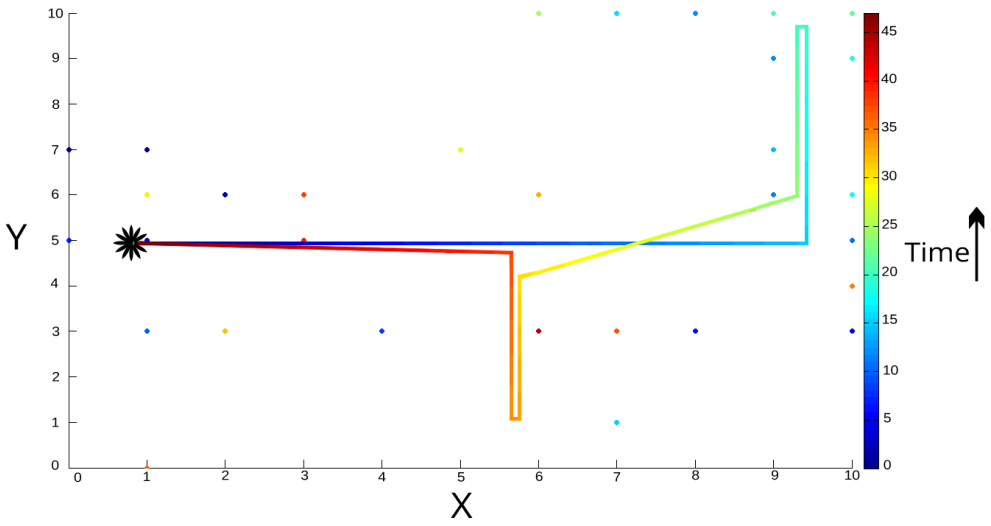

Figure 5.24: Route results for the algorithm MAP.

Multilateration, as expected, and due to its, sometimes, uncertain distance estimations, shows bad results, since the point dispersion is big and covers almost all the room, making almost impossible to relate the points to the route.

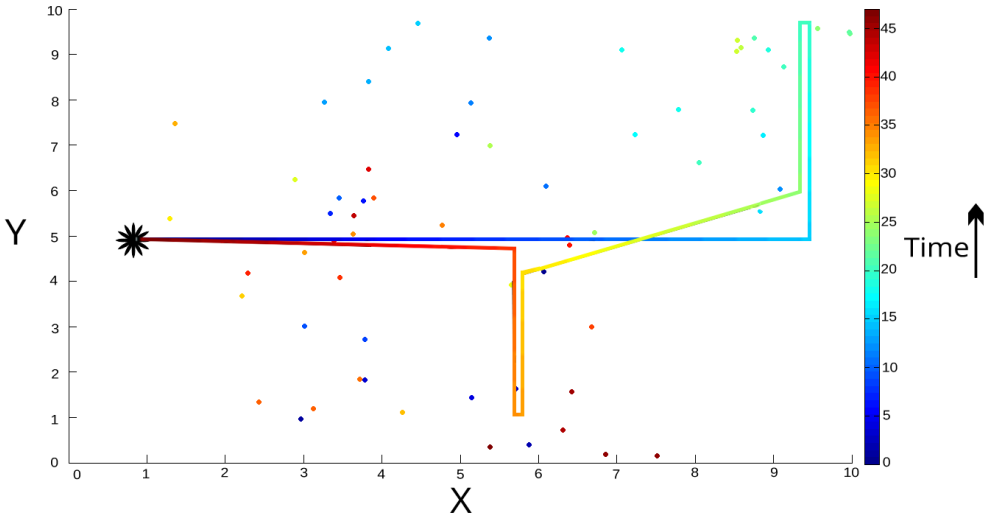

Figure 5.25: Route results for the multilateration technique.

5.5 Dataset Optimization Tests

When the dataset was completely collected, all its data was observed and a search for things that can improve the dataset's performance was made. There were some standard deviation values in the dataset that could be used to remove some RPs readings that, instead of helping the position estimation, were just introducing error in the calculations. Following that, two optimization tests were performed. It is important to notice that all tests performed in this section were made after the previous tests in this chapter and they use only the algorithm that shown best performance (MMSE algorithm).

5.5.1 Optimization Test 1

This first optimization test limited the readings in the dataset depending on their standard deviation. Values with high standard deviation most likely will introduce errors in the calculations. A set of tests were performed, limiting the dataset values to 7 different standard deviation values. The results obtained are shown in figure 5.26 and in table 5.1. The number of readings row represents the total number of averaged readings in the dataset. For each position there are 4 averaged readings, one for each TP. Since there are 121 positions in the dataset, a total of 484 averaged readings exist.

Table 5.1: Evolution of σ and number of readings with the optimization test 1 with MMSE algorithm.

	<1	<2	<3	<4	<5	<6	<7	Normal
σ	7.29	7.57	7	7	7.143	7.57	6	6
Number of readings	63	250	361	442	475	481	483	484

Table 5.1 shows the variation of the σ parameter and the number of readings in function of the standard deviation limitations. As expected, the smaller the

standard deviation limitation, the smaller will be the amount of readings, since some readings are removed due to their higher standard deviation value.

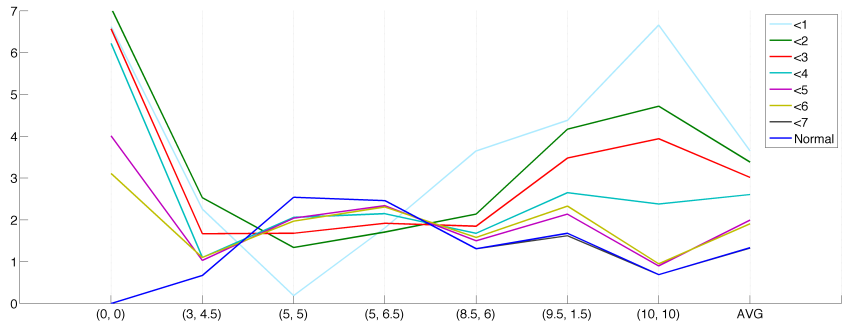

Figure 5.26: Position estimation with the optimization test 1 using the MMSE algorithm.

Looking at the figure above, one can see that the best dataset performance was obtained with the normal dataset values with no standard deviation limitations at all. The bigger the limitation applied, the worse the averaged error becomes.

5.5.2 Optimization Test 2

Optimization test 2 follows a different approach. Although, it still uses the standard deviation as a decisive parameter, this time, it goes deeper than the previous optimization test. It uses a range to define the acceptable samples based on the average Wi-Fi RSS and its standard deviation.

Remembering that in the original dataset, each RP consists of 10 Wi-Fi RSS samples, with this optimization test, in order to stay in the dataset, each one of those samples need respect the following condition:

average Wi-Fi RSS - stdev $<$ sample Wi-Fi RSS $<$ average Wi-Fi RSS $+$ stdev

Only the samples that respect this condition, will be considered in the average Wi-Fi RSS of each RP.

The optimized dataset was tested in the Android application and the results are in figure 5.27.

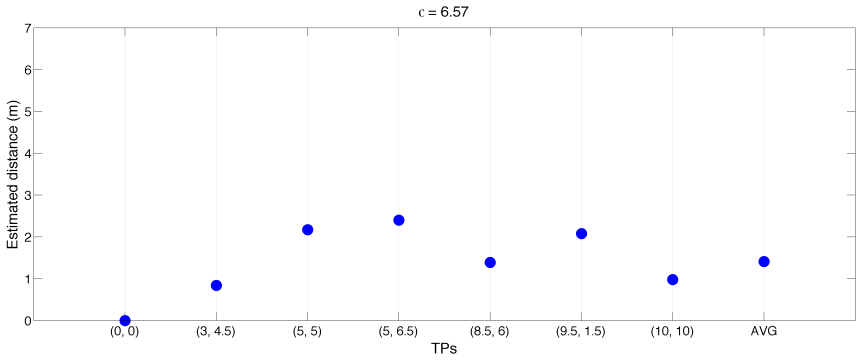

Figure 5.27: Position estimation with the optimization test 2 using the MMSE algorithm.

The results are not too different from the results obtained with the not optimized dataset, actually they are a little worse. In the normal dataset, the averaged error was around 1.33 meters and with this theoretical optimization the result was around 1.4 meters which means that the expected optimizations didn't actualy optimize the dataset performance and even made it worse.

This page was intencionally left blank!

Chapter 6

Conclusions & Future Work

In this thesis, a set of indoor positioning algorithms for two different techniques were tested. On one hand, the fingerprinting technique and on the other hand the multilateration technique. Both techniques work in different ways and have their andvantages and disadvantages as well as different algorithms to estimate the user location. For fingerprinting, the best algorithm, based on this thesis approach, is the MMSE algorithm, which returned an averaged error of 1.33 meters in a static context. Multilateration, on the other hand, proven to have a worse performance when compared with the fingerprinting technique and shows to have undesired results when the distance estimation has a bigger error.

An Android application was developed alongside with this thesis. The application works with the fingerprinting technique and it enables its users to add new maps of fingerprints. Whenever there are available maps, users can navigate. MMSE algorithm is used, since it shows to perform better than the others.

An important part of this thesis is the dataset. The dataset contains relevant information and can help future researches. The dataset has two parts: A static part where a set of points were recorded to the database and an in motion part where points where recorded to the database while simulating a route. This set of points include Wi-Fi readings and magnetic strength readings; the other part

of the dataset have live routes around the room including video footages of those routes. In this part, data from Wi-Fi RSS, magnetic strength and accelerometers are available.

There is a good foundation for future work following this thesis. A deep insight over a Wi-Fi RSS based indoor positioning plus a good ground base of sensory data and movement analysis is available with this document. Following it, one can integrate the movement awareness with the Wi-Fi RSS estimation and create a more accurate system. Another approach would be to explore the possible strength of the magnetic field records present in this document and integrate them as well with the movement awareness and Wi-Fi RSS readings, creating a smarter system.

Bibliographic References

[1] S. Nirjon, J. Liu, G. DeJean, B. Priyantha, Y. Jin, and T. Hart, "Coin-gps: indoor localization from direct gps receiving," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pp. 301–314, ACM, 2014.

[2] M. M. Atia, M. Korenberg, and A. Noureldin, "A consistent zero-configuration gps-like indoor positioning system based on signal strength in ieee 802.11 networks," in *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pp. 1068–1073, IEEE, 2012.

[3] U. Shala and A. Rodriguez, "Indoor positioning using sensor-fusion in android devices," 2011.

[4] T. Wei and S. Bell, "Indoor localization method comparison: Fingerprinting and trilateration algorithm," *University of Saskatchewan. Accessed March*, vol. 24, p. 2015, 2011.

[5] B. S. I. G. (SIG), "Bluetooth core specification." https://goo.gl/RPaJku. Accessed January 7, 2015.

[6] A. Corbacho Salas *et al.*, "Indoor positioning system based on bluetooth low energy," 2014.

[7] E. Inc., "Estimote, real-world context for your apps." http://goo.gl/h4e7rV. Accessed Jan 5, 2016.

[8] V. Beal, "Wi-fi." http://goo.gl/WXSzYQ. Accessed February 4, 2015.

[9] I. Cisco Systems, "Wi-fi location-based services 4.1 design guide." http://goo.gl/ddBwi4. Accessed May 7, 2015.

[10] N. Pritt, "Indoor location with wi-fi fingerprinting," in *Applied Imagery Pattern Recognition Workshop: Sensing for Control and Augmentation, 2013 IEEE (AIPR)*, pp. 1–8, IEEE, 2013.

[11] D. Helgesson and E. Nilsson, "Comparison and implementation of ips," 2014.

[12] C. Medina, J. C. Segura, and A. De la Torre, "Ultrasound indoor positioning system based on a low-power wireless sensor network providing sub-centimeter accuracy," *Sensors*, vol. 13, no. 3, pp. 3501–3526, 2013.

[13] C. Medina, J. C. Segura, and A. De la Torre, "Ultrasound indoor positioning system based on a low-power wireless sensor network providing sub-centimeter accuracy," *Sensors*, vol. 13, no. 3, pp. 3501–3526, 2013.

[14] M. Rouse, "ultra wideband." http://goo.gl/cUO3Es. Accessed January 9, 2015.

[15] L. Zwirello, T. Schipper, M. Harter, and T. Zwick, "Uwb localization system for indoor applications: concept, realization and analysis," *Journal of Electrical and Computer Engineering*, vol. 2012, p. 4, 2012.

[16] O. Haraz, "Why do we need ultra-wideband?," 2012. Accessed November 24, 2014.

[17] A. Brands, "Bytelight services: Indoor positioning." http://goo.gl/mOJaZ6. Accessed July 10, 2015.

[18] C. Swedberg, "Retailers test bytelight's light-based indoor positioning technology." http://goo.gl/c2DaEN. Accessed July 10, 2015.

[19] L. Li, P. Hu, C. Peng, G. Shen, and F. Zhao, "Epsilon: A visible light based positioning system," in *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pp. 331–343, USENIX Association, 2014.

[20] G. Hasan, K. Hasan, R. Ahsan, T. Sultana, and R. Bhowmik, "Evaluation of a low-cost mems imu for indoor positioning system," *Inertnaional Journal of Emerging Science and Engineering*, vol. 1, 2013.

[21] IndoorAtlas, "Indoor positioning made ubiquitous. the positioning service powering location-based apps worldwide.." https://www.indooratlas.com/. Accessed December 10, 2014.

[22] R. Goodrich, "Accelerometers: What they are & how they work." http://goo.gl/zwFBUr. Accessed January 4, 2015.

[23] A. King, "Inertial navigation-forty years of evolution," *GEC review*, vol. 13, no. 3, pp. 140–149, 1998.

[24] Fitlinnx, "Running chart." http://goo.gl/v2a3cn. Accessed January 20, 2015.

[25] Y.-k. Wu, H.-C. Wang, L.-C. Chang, and S.-C. Chou, "Customer's flow analysis in physical retail store," *Procedia Manufacturing*, vol. 3, pp. 3506–3513, 2015.

[26] M. C. Popa, L. J. Rothkrantz, C. Shan, T. Gritti, and P. Wiggers, "Semantic assessment of shopping behavior using trajectories, shopping related actions, and context information," *Pattern Recognition Letters*, vol. 34, no. 7, pp. 809–819, 2013.

[27] M. J. Gómez, F. García, D. Martín, A. de la Escalera, and J. M. Armingol, "Intelligent surveillance of indoor environments based on computer vision and 3d point cloud fusion," *Expert Systems with Applications*, vol. 42, no. 21, pp. 8156–8171, 2015.

[28] K. Heath and L. Guibas, "Multi-person tracking from sparse 3d trajectories in a camera sensor network," in *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pp. 1–9, IEEE, 2008.

[29] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1815–1821, IEEE, 2012.

[30] J. Wilson and N. Patwari, "Radio tomographic imaging with wireless networks," *Mobile Computing, IEEE Transactions on*, vol. 9, no. 5, pp. 621–632, 2010.

[31] M. Bocca, O. Kaltiokallio, and N. Patwari, "Radio tomographic imaging for ambient assisted living," in *Evaluating AAL Systems Through Competitive Benchmarking*, pp. 108–130, Springer, 2013.

[32] Y. Zhao, N. Patwari, J. M. Phillips, and S. Venkatasubramanian, "Radio tomographic imaging and tracking of stationary and moving people via kernel distance," in *Proceedings of the 12th international conference on Information processing in sensor networks*, pp. 229–240, ACM, 2013.

[33] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 775–784, 2000.

[34] A. W. Reza and T. K. Geok, "Investigation of indoor location sensing via rfid reader network utilizing grid covering algorithm," *Wireless personal communications*, vol. 49, no. 1, pp. 67–80, 2009.

[35] S. P. Subramanian, J. Sommer, S. Schmitt, and W. Rosenstiel, "Inr indoor navigator with rfid locator," in *Next Generation Mobile Applications, Services*

and Technologies, 2009. NGMAST'09. Third International Conference on, pp. 176–181, IEEE, 2009.

[36] T. Chou and J. Liu, "Design and implementation of rfid-based object locator," in *RFID, 2007. IEEE International Conference on*, pp. 86–93, IEEE, 2007.

[37] N. LLC, "Somewhere in this building, or precisely here." http://www.nextnav.com/. Accessed January 24, 2015.

[38] R. Xu, W. Chen, Y. Xu, and S. Ji, "A new indoor positioning system architecture using gps signals," *Sensors*, vol. 15, no. 5, pp. 10074–10087, 2015.

[39] R. Peng and M. L. Sichitiu, "Angle of arrival localization for wireless sensor networks," in *Sensor and Ad Hoc Communications and Networks, 2006. SECON'06. 2006 3rd Annual IEEE Communications Society on*, vol. 1, pp. 374–382, IEEE, 2006.

[40] D.-I. P. Ubolkosold, "Positioning in gsm networks." http://goo.gl/Ds0uT0. Accessed January 15, 2015.

[41] G. Colistra and L. Atzori, "Estimation of physical layer performance in wsns exploiting the method of indirect observations," *Journal of Sensor and Actuator Networks*, vol. 1, no. 3, pp. 272–298, 2012.

[42] C. Randell, C. Djiallis, and H. Muller, "Personal position measurement using dead reckoning," in *null*, p. 166, IEEE, 2003.

[43] J. Aronson, "Dead reckoning: latency hiding for networked games," *Gamasutra magazine*, 1997.

[44] O. Costilla-Reyes and K. Namuduri, "Dynamic wi-fi fingerprinting indoor positioning system," in *International Conference on Indoor Positioning and Indoor Navigation*, vol. 27, p. 30th, 2014.

[45] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen, "A probabilistic approach to wlan user location estimation," *International Journal of Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, 2002.

[46] W. S. Murphy Jr, "Determination of a position using approximate distances and trilateration," *Colorado School of Mines*, 2007.

[47] S. M. Ross, *Introduction to probability models*. Academic press, 2014.

[48] H. Yu and B. M. Wilamowski, "Levenberg-marquardt training," *Industrial Electronics Handbook*, vol. 5, pp. 12–1, 2011.

[49] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "The airplace indoor positioning platform for android smartphones," in *Mobile Data Management (MDM), 2012 IEEE 13th International Conference on*, pp. 312–315, IEEE, 2012.

[50] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "Demo: the airplace indoor positioning platform," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 467–468, ACM, 2012.

[51] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 775–784, 2000.

[52] V. Maximov and O. Tabarovsky, "Survey of accuracy improvement approaches for tightly coupled toa/imu personal indoor navigation system," in *International Conference on Indoor Positioning and Indoor Navigation*, vol. 28, p. 31th, 2013.

[53] M. Jin, B. Koo, S. Lee, C. Park, M. J. Lee, and S. Kim, "Imu-assisted nearest neighbor selection for real-time wifi fingerprinting positioning," in *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference on*, pp. Pages 745–748, IEEE, 2014.

[54] P. Prasithsangaree, P. Krishnamurthy, and P. K. Chrysanthis, "On indoor position location with wireless lans," in *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on*, vol. 2, pp. 720–724, IEEE, 2002.

[55] K. Kaemarungsi and P. Krishnamurthy, "Modeling of indoor positioning systems based on location fingerprinting," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 1012–1022, IEEE, 2004.

[56] K.-K. Chu, J.-Y. Ng, and K. Leung, "A new approach for locating mobile stations under the statistical directional propagation model," in *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, vol. 1, pp. 932–940, IEEE, 2006.

[57] W. M. Yeung and J. K. Ng, "An enhanced wireless lan positioning algorithm based on the fingerprint approach," in *TENCON 2006. 2006 IEEE Region 10 Conference*, pp. 1–4, IEEE, 2006.

[58] L. Jiang, *A WLAN Fingerprinting Based Indoor Localization Technique*. PhD thesis, University of Nebraska, 2012.

[59] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*, vol. 2. prentice hall PTR New Jersey, 1996.

[60] T. C. K. f. A. D. B. Ducrohet, Xavier; Norbye, "Android studio: An ide built for android." http://goo.gl/RYPgQ6. Accessed March 15, 2015.

[61] MathWorks, "The language of technical computing." http://www.mathworks.com/products/matlab/. Accessed March 14, 2015.

[62] A. Developers, "android.net.wifi." https://goo.gl/g44Fvb. Accessed March 16, 2015.

[63] A. Developers, "android.hardware." https://goo.gl/Otkiol. Accessed March 16, 2015.

[64] A. Developers, "android.database.sqlite." https://goo.gl/TyVPq8. Accessed March 16, 2015.

[65] G. Booch, *The unified modeling language user guide*. Pearson Education India, 2005.

[66] A. Developers, "Adapter." https://goo.gl/8NnfbT. Accessed July 20, 2015.

[67] A. Developers, "Sensormanager." https://goo.gl/QFiV8s. Accessed March 16, 2015.