**DEPARTAMENTO DE SEÑALES, SISTEMAS Y
RADIOCOMUNICACIONES**

**ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE
TELECOMUNICACIÓN**

Contributions to beacon-based applications in smart spaces

Autor: EDUARDO METOLA MORENO
Máster Ingeniero de Telecomunicación

Director: JOSÉ RAMÓN CASAR CORREDERA
Doctor Ingeniero de Telecomunicación

Madrid, 2017

Department: Señales, Sistemas y Radiocomunicaciones
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid (UPM)

PhD Thesis: Contributions to beacon-based applications in smart spaces
Contribución a las aplicaciones basadas en balizas en espacios inteligentes

Author: **Eduardo Metola Moreno**
Máster Ingeniero de Telecomunicación (UPM)

Advisor: **José Ramón Casar Corredera**
Doctor Ingeniero de Telecomunicación (UPM)

Year: 2017

Committee named by the Rector of Universidad Politécnica de Madrid, on the 201..

Committee: **Nombre Presidente**
Universidad de dónde es
Nombre Vocal 1
Universidad de dónde es
Nombre Vocal 2
Universidad de dónde es
Nombre Vocal 3
Universidad de dónde es
Nombre Secretario
Universidad de dónde es
Nombre Suplente 1
Universidad de dónde es
Nombre Suplente 2
Universidad de dónde es

After the defense of the PhD Thesis on the of 201., at the E.T. de Telecomunicación, the committee agrees to grant the following qualification:

.....

PRESIDENT

MEMBERS

SECRETARY

The research described in this Thesis was carried out within the Grupo de Procesado de Datos y Simulación (GPDS) at the Dept. of Señales, Sistemas y Radiocomunicaciones, E.T.S.I. de Telecomunicación, Universidad Politécnica de Madrid (from 2009 to 2017).

Abstract

The customization of applications and services today is an example of the importance of being aware of the user, her behavior and her context by the service providers. Particularize for each user is a complex task so that machine learning algorithms are used to classify or cluster users in profiles based on parameters that users of the same group share or whose difference is minimal. In this way, there are user groups or clusters that roughly define the behavior of every user in the same cluster from a series of parameters that describe a model.

In this work, since the main line of research carried out previously was focused on indoor localization, modeling the users behavior from indoor users trajectories has been proposed. This modeling is based on both Markov and non-Markov models. Markov models only take into account the current state of the user position to estimate the next position but non-Markov models are memory-aware and take into account previous states as well as the current one.

Proposed models, and more particularly the Markov model and the model with memory 1, which only takes into account the immediately previous and the current states to estimate the following user position, are evaluated in this work to check how models could represent, in an indirect way, the behavior of users by groups.

The evaluation of the models is done using a classification algorithm, which is not evaluated in this work, and classifying trajectories that are generated from the proposed models and evaluating the percentage of correctly classified trajectories according to the following two parameters: number of trajectories needed to generate the proposed models and number of trajectories used to train the classifier.

In addition, it is intended to estimate the number of trajectories necessary to ensure, with a certain confidence degree, the profile a user belongs to if she is identified and classified from her trajectories.

As trajectories are defined as a succession of positions, the symbolic location paradigm has been addressed, which allows us to estimate the user position using wireless technology when indoors. Thus, three algorithms are proposed, which are easy and quick to implement, calibrate and configure, and enable the deployment of a localization system in real environments in a practical way.

Within the possible wireless technologies, we decided to use a Bluetooth Low Energy beacon-based system that have a great autonomy, what enables to have a stable infrastructure and with little maintenance, non invasive and easy to deploy; while the user carries a mobile device that runs a localization application that detects the beacons and estimate its own position.

The proposed algorithms have been designed, developed and evaluated to estimate their usefulness in controlled scenarios and their possible installation in real scenarios. In addition, it has been proposed the design and development of a mobile application of personalization from the user position, where the localization algorithms proposed throughout this work have been implemented.

On one hand, the actual implementation in a controlled environment has been used for its subsequent deployment in a demo environment, where banking or retail businesses could be installed and use the localization system for, for example, the personalization of services as a function of the position of the user from their trajectories; or in a real environment like a real supermarket, in Nice, where an infrastructure has been deployed in part of the enclosure for the evaluation of the complete solution in an uncontrolled environment.

On the other hand, this application offers tools for user tracking and construction of paths or sending personalized notifications depending on the

position of the user, as well as an informative dashboard that allows to know the position data in each zone of the different users.

Resumen

La personalización de aplicaciones y servicios hoy en día es una prueba de la importancia de conocer al usuario y su contexto por parte de los proveedores de servicios. Particularizar para cada usuario es una tarea compleja por lo que algoritmos de aprendizaje automático o *machine learning* se utilizan para clasificar o clusterizar usuarios en perfiles en función de factores que compartan los usuarios del mismo grupo o cuya diferencia sea mínima. De esta manera, se manejan grupos o clusters de usuarios que definen de una manera aproximada el comportamiento de cada uno de los usuarios que pertenezcan a un mismo cluster a partir de una serie de parámetros que describen un modelo.

En este trabajo, puesto que la línea de investigación principal llevada a cabo anteriormente estaba centrada en la localización de usuarios en espacios interiores, se ha propuesto como objetivo modelar el comportamiento de los usuarios a partir de las trayectorias que realizan en un espacio en interiores. Este modelado se basa en modelos tanto markovianos, donde solo se tiene en cuenta el estado actual del usuario, es decir, su posición actual, para estimar la posición en el instante siguiente; como no Markovianos o con memoria, donde sí se tienen en cuenta los instantes previos al actual.

Los modelos propuestos, y más en particular el model markoviano como el modelo con memoria 1, que solo tiene en cuenta el instante inmediatamente anterior y el actual para estimar la posición del usuario en el instante siguiente, se evalúan en este trabajo para ver cómo podrían representar, de manera

indirecta, el comportamiento de los usuarios por grupos.

La evaluación de los modelos se realiza utilizando un algoritmo de clasificación, el cual no se evalúa en este trabajo, y clasificando trayectorias generadas a partir de los modelos propuestos y viendo el porcentaje de trayectorias correctamente clasificadas en función de los dos parámetros siguientes: número de trayectorias necesarias para generar los modelos propuestos y número de trayectorias utilizadas para entrenar el clasificador.

Además, se pretende estimar el número de trayectorias necesario para asegurar, con un cierto grado de confianza, el perfil al que pertenece un usuario si éste es identificado y clasificado a partir de sus trayectorias.

Como las trayectorias se definen como una sucesión de posiciones, se ha abordado el paradigma de la localización simbólica, que permite estimar la posición de un usuario utilizando tecnología inalámbrica, en espacios interiores. Para ello, se proponen tres algoritmos fáciles y rápidos de implementar, calibrar y configurar, que permitan desplegar un sistema de localización en entornos reales de manera práctica.

Dentro de las posibles tecnologías inalámbricas, se ha decidido utilizar un sistema de balizas Bluetooth Low Energy que tienen una gran autonomía, lo que permiten tener una infraestructura estable y con poco mantenimiento, poco invasiva y fácil de desplegar; mientras que el usuario, con un dispositivo móvil una aplicación de localización puede detectar las balizas y estimar su propia posición.

Los algoritmos propuestos se han diseñado, desarrollado y evaluado para valorar su utilidad en escenarios controlados de prueba y su posible instalación en escenarios reales. Además, se ha propuesto el diseño y desarrollo de una aplicación móvil de personalización a partir de la localización del usuario, donde se han implementado los algoritmos de localización propuestos a lo largo de este trabajo.

Por un lado, la implementación real en un entorno controlado ha servido para su posterior despliegue tanto en un entorno demo, donde negocios de tipo banca o retail podrían ser instalados y utilizar el sistema de localización

para, por ejemplo, la personalización de servicios en función de la posición del usuario a partir de sus trayectorias; como un entorno real como un supermercado real, en Niza, donde se ha desplegado una infraestructura en parte del recinto para la evaluación de la solución completa en un entorno no controlado.

Por otro lado, esta aplicación ofrece herramientas de seguimiento de usuarios y construcción de trayectorias o de envío de notificaciones personalizadas en función de la posición del usuario, además de un dashboard informativo que permita conocer los datos de posición en cada zona de los distintos usuarios.

Agradecimientos

Contents

Abstract	i
Contents	xv
1 Context and goals	1
1.1 Introduction	1
1.2 Context-aware services	2
1.3 Goals and challenges.	4
1.4 Summary	7
1.5 Contributions	9
2 State of the art	13
2.1 Techniques and technologies applied to localization applications.	13
2.2 Business Intelligence (BI) applied to context-aware services.	30
3 BI applied to users classification based on their behavior patterns	39
3.1 Motivation and objectives	39
3.2 Description of the problem, a generic case	43

3.3 A case of study	53
3.4 Application to a generic scenario	70
3.5 Conclusions.	72
4 Methods for the localization of mobile devices	75
4.1 Introduction	75
4.2 Design of a localization system based on iBeacons	76
4.3 Localization algorithms.	79
4.4 Evaluation of the localization system.	90
4.5 Conclusions.	103
5 Personalization of applications based on the user position and behavior	107
5.1 Introduction	107
5.2 A user position and behavior-aware application.	108
5.3 Application design	111
5.4 Demonstration.	114
5.5 Conclusions.	127
6 Conclusions and future work	129
6.1 Conclusions.	129
6.2 Future Work	133
A Appendix Trajectories generation tools	135
A.1 Users trajectories and scenarios models description	136
A.2 Design and development of trajectories simulation tool	138
A.3 Trajectories analysis	145
B Appendix Selection of the technology to build the infrastructure of the localization system	151
B.1 iBeacons	152

C	Appendix RSS-based solutions	157
C.1	BLE signal behavior	162

Chapter 1

Context and goals

1.1 Introduction

The increasingly common presence of new technologies in our day to day entails the increment of the information that can be obtained from people and their context. As a part of our work and leisure daily tasks, new technologies are formed by the devices, techniques and tools that we commonly use for communication, interaction, gaming, working, data processing, information storage, etc.

The incorporation of hardware such as smartphones, tablets, e-readers or smart televisions, applications, software and services to our lives, brings the possibility to service providers to obtain information from the users, who are actually the targets of the services they provide. On the other hand, users, who perform as consumers, become data providers, either consciously due to all the information they provide to the current existing services, or unconsciously, through their physical and online activities that can be transparently monitored.

One of the most representative topics in new technologies is the Internet of Things (IoT), which converts objects in sensors and actuators and allow their communication with both users and services. The addition of connection, storage and processing abilities to common objects, convert them in smart objects, easing the possibility to offer ubiquitous and customized services, due to the possibility to detect the properties of the context where the objects are. Therefore, the creation of user context-aware services is possible due to this IoT paradigm.

Ubiquitous computing adds the possibility to offer services whose actual computing tasks are not implemented in the same place where the information is obtained and provided. The internet, the possibility to connect the smart objects to users and services, the cloud computing and the multiple connectable devices used by people allow to provide services with information related to the users and their context, to be computed and represented all around the world.

Nowadays, the amount of information that services and applications can be provided from the multiple sensors and users is permanently increasing. Business Intelligence (BI) is the concept that represents the storage, processing, interpretation and representation of this huge amount of information, in a way, this can be synthesized and analyzed by thirds, so that the information can be parameterized and classified for a better understanding. For instance, companies that offer services according to the type of user and their context - *context-aware services* - can access to a more elaborated information that allow them to reach potential users and customers more efficiently.

1.2 Context-aware services

The real motivation of context-aware services in businesses starts from the need and willingness of companies to offer enhanced and customized services to users. Companies that support this innovation invest large amounts of money in R & D, always trying to reflect this investment into profits. In literature, the motivations and implications of this type of investment in different scenarios, taking into account both external and internal factors, have been studied (Betz, 2011)(Xiao-qiang, 2009). The term *context-based services* refers to the adaptation of a service response to factors related to

the user and her surrounding environment. Users relative information such as their preferences, their recent activities, their current position, the date and time at which the service was required or the type of device that is used to interact with the service are some of the main features that make the provided service to adequate its response to the user. If this information is detected and measured, users profiles can be modeled to study their behavior patterns.

From a practical point of view, banks, malls, groceries or hotels are some examples of public scenarios where people attend for leisure and work activities. The range of products and services offered to customers makes it possible to target customized ads and offer products and services to the users based on their needs (Aida et al., 2011)(Antoniou et al., 2012a)(Antoniou et al., 2012b).

For example:

- In a bank branch, the activities carried out by customers that normally go to take money to the ATM are totally different from the activities performed by a customer that goes to pay the bills at the branch or the businessman that runs credits for running his business. The type of ads and offers a customer receives are totally different according to the profile they belong. The first type of customer, who only goes to the branch to take money, could be tempted to get a card with less commissions for money withdrawal from any ATM, the second one could receive more publicity for household bills in the bank and the third would receive customized offers for the credits required for her business.
- In a mall, some users can go only to do the shopping, others go shopping to the different types of stores (technology, culture, fashion, etc.) and others can go for fun (movies, restaurants, etc.). Stores advertising for users through a smartphone application, discounts and offers can be customized depending on the type of user.
- The hotel business can also be adapted to the types of users and their activities on trips. There are different profiles of users: users with a cultural interest, users with a tourist interest but with a less cultural profile, users with preferences for nature, green spaces and parks or users who want to enjoy the leisure offers of the cities they visit. Offers and discounts on attractions such as museums or restaurants, guided tours

and personalized visits and activities at the hotel are some of the services or products that can be offered to customers in a personalized way.

This work focuses on user-oriented services, so that the information of interest is that related to users, their activities and their context. We might be interested in other user-related parameters such as psychological, cultural or social, but for simplicity we assume that the chosen parameters are reflected in the activities that users perform, which is part of the input data in this case. In other words, the tendency of a person with a high cultural interest in a shopping center is reflected in more frequent and longer lasting visits to bookstores or media stores. Therefore, in this work, users that share behavior patterns are classified in the same profile.

Besides presence-related information about users, internet purchases or browsing (cookies), posts on social networks, words typed in engine searches or the most used mobile applications are also useful information to know customers activities from another point of view. In this work, we have decided to focus on users physical activities, the tracking of user positions in scenarios and their combination to create trajectories to study the users behavior patterns.

1.3 Goals and challenges

The challenges addressed in this work try to approach the problem of context-aware services based on people physical activities related to their movement and the paths that follow in a specific scenario. Two points of view of this approach are discussed. On one hand, the representation of the user behavior patterns in a specific scenario using Markov models, which represent the behavior of users that share the profile type. On the other hand, in order to classify a user in an existing profile and provide her with a customized service, the detection of the trajectories she performs is managed through a mobile-based localization system supported by a Bluetooth Low Energy(BLE)-based beacons infrastructure.

Furthermore, the real implementation of the localization system using the algorithms that are proposed in this work, in order to demonstrate its correct performing and the development of a service based on the user profile is

presented and discussed.

Along this section, the goals and challenges related to every topic in this work are introduced and described.

1.3.1 Representation of user behavior patterns through Markov and non-Markov models

It can be interesting to be able to study and to understand how users behave when moving in a specific scenario. This enables the possibility to know the trends and patterns that the different types of users follow when, for instance, visit a mall, a museum or a bank. In these examples, businesses managers can control a very interesting information that help them to carry out the best strategies to manage their businesses. Therefore, we propose a methodology to create models that could represent the movement of a user through a series of areas that form a scenario.

These models are both Markov and non-Markov based and define the probability of moving between two states. Here, we consider that a scenario is formed by N areas, which represent the states of the models. On one hand, Markov models define the probability to move between the current state and the next one. On the other one, in order to take into account previous states, we will propose a deviation of this kind of Markov-based models.

Users behavior patterns define a user type or a user profile in a scenario. Therefore, a model represents the behavior of a specif user profile, what means that for every profile a new model must be defined. The definition of every model is based on a set of trajectories performed by users that belong to the profile that is going to be defined. In Chapter 3, we try to find the optimum number of trajectories used to define a model that is necessary to ensure a certain quality to the proposed models.

The concept *quality of the model* is defined as the ability of a classifier to correctly detect the profile a trajectory belongs to. Since the classification process can also affect the evaluation of the models, we are going to evaluate how the number of trajectories used to train the classification algorithm affects the quality parameter. Once the methodology to build Markov-based

models is discussed, we can create trajectories based on these models.

On the other hand, if we are able to track a user and build trajectories from the ordered set of estimated positions, and these trajectories are classified by an already-trained algorithm, we propose to find the number of trajectories that are necessary to ensure, with a certain probability degree, that the user belongs to the detected profile.

1.3.2 Creation of trajectories from a Bluetooth Low Energy based localization system

The detection of users in the different zones of a scenario enables the possibility to track the positions of the users along time to build the trajectories they perform. From the different technologies used to detect users presence (infrared, ultrasounds, radio frequency, cameras) we have decided to work with radio frequency based devices for three reasons:

1. They are not very expensive and already deployed elements such as access points can be used. In this work, small BLE beacons are used.
2. Common mobile devices are equipped with radio frequency transceivers, so that most of users can be localized.
3. The use of the users mobile devices allow to know who is the one that is being localized, so that she can be tracked to build the required trajectories.

The main challenge related to the deployment of a localization system is to design and calibrate a BLE beacons-based algorithm, which must be able to detect the position of a user in a scenario that contains N areas. Areas are more or less defined and a primary requirement is the stability of the estimation. In other words, besides to provide a good quality estimated position, this must be stable, so that swaps between areas must be avoided. Let's suppose a service that is enabled when the user is inside an area. If the user is enjoying this service and the localization system wrongly detects the user outside this area, the service would stop. This type of effects must be carefully faced.

Calibration of the algorithm is necessary to be setup according to the configuration of the scenario. Then, its implementation in a mobile-based architecture is necessary to carry out an evaluation of the performance of the algorithms.

1.3.3 Application of a context-aware service to a real scenario

Finally, the design and development of a context-based service using the position of the user is proposed. We rely on a smart space where sensors can be deployed and the localization system can be implemented and tested. Both, a multimedia service and a notifications service are built to demonstrate the concept and how this type of services are easy to be deployed.

On the other hand, we propose and design a user profile based service, which modulates its performance according to the type of user that requires the service.

1.4 Summary

The reader will read in the next chapters:

- **Chapter 2** - A review of the state of the art of the two main fields of study in this work is done. First, BI is a wide concept that integrates, among other things, the processing of a huge amount of data in order to gather the underlying information that enables people, who have to take decisions in base to this knowledge, to obtain the synthesized information instead the raw data. Then, the motivation of why we deep into this paradigm, why companies from different sectors do too and the actual and most used techniques of Big Data (the part of BI aimed at processing lots of data) are revised.

On the other hand, the problem of locating a target has been studied during years, adapting the works to the state of the available technologies. Here, the positioning system problem has been focused from two points of view: the technologies and the methods. The technologies involve those that have been used to detect the targets to estimate their position. E.g. Wi-Fi, Bluetooth or GPS are some of the most used technologies

to estimate the position of people because these technologies are integrated in smartphones. Once there exist a communication between the infrastructure that support the localization and the mobile device that is going to be localized, the techniques to process the exchanged information are based on algorithms aiming at minimizing the error in the position estimation.

- **Chapter 3** - The study of the behavior of users based on the trajectories they do in a generic scenario is proposed. Users trajectories are described as a succession of positions that could define user trends in a specific scenario. Some examples are proposed. For instance, a shopping center. If a user visits the book store every time she goes to the shopping center, a likely-cultural profile could be attached to her.

A generic scenario, where there are N areas or states and M user profiles is proposed. In order to be able to study movements-based user behaviors, a suit of Markov and non-Markov based models are proposed to define the movement of the users (their transitions among the different areas, based on the current position and, optionally, according to the type of model (without or with memory), the l previous positions or states of a user. The generation of the models, its evaluation, the estimation of the number of trajectories that are necessary to create good enough models, with a certain quality, and how to analyze it using a classification algorithm is discussed in Chapter 3 and estimated for a mock-up scenario based on a small shopping center with 5 types of user profiles. The classification process is evaluated based on the number of trajectories that are used to train the algorithm. Finally, if the models are created and the classification of the users based on their trajectories is deployed, we try to find the estimation. The conclusions obtained for this use of case is then applied in the generic case.

- **Chapter 4** - The detection of the user position using wireless technologies such as Bluetooth Low Energy (BLE) is approached in Chapter 4. BLE was released in Bluetooth standard in 2010 and recently has been implemented in commercial devices such as smartphones or tablets. Besides, small objects working as beacons have appeared implementing this new version of Bluetooth, which is both not very expensive and low-power consuming. This last feature make them to be quite autonomous because their batteries have an autonomy of up to two years.

In Chapter 4, the study of the type of signal these BLE-based beacons broadcast and how the spatial conditions affect the RSS, the design of a RSS-based localization algorithm, the beacons-based infrastructure and the evaluation of the proposed algorithm and its variations are approached.

- **Chapter 5** - Finally, the application of these concepts in a real service is carried out. First, we propose the design a user context-aware application. This application can be built over any scenario and is aware of the user context information. This work is based on two measurable types of data related to the user: its behavior and its current position.

The user profile type is considered in this work to be estimated from the classification of users behavior based on the paths they follow when visiting a scenario. On the other hand, a real implementation of a positioning system is proposed and deployed in three different scenarios. One of them is a test scenario, where we describe different business to be supported by the proposed application. Besides, a content-based application that presents a content related to the user position is described and demonstrated. And finally, in Super-U supermarket Nice, an actual deployment was carried out to prove that this could be installed in a real business.

1.5 Contributions

In this work, we propose three contributions, each related to Chapters 3, 4 and 5. The first one, related to Chapter 3, is the proposal of a methodology to define users behavior using Markov and non-Markov models. We have considered to define users behavior from the point of view of their trajectories in a specific scenario, such as a mall, a museum, a bank, etc. Then, if we can model users behavior using proposed models, we can understand most common users behavior patterns when visiting these scenarios and business managers could adapt their services to the different user profiles.

Our proposal covers the models design using simulated trajectories and the evaluation of these models based on two parameters: the number of trajectories used to design them and the number of trajectories used to train

a classifier. The classification of trajectories in each corresponding profile and the correctness of the trajectories classification is the indicator we use to estimate models quality. In other words, the models quality could mean how well models represent, in an indirect way, real life.

Using simulated trajectories instead real trajectories has a practical reason. The collection of real trajectories from a real scenario is a hard work and we have decided to simplify it by simulating models as if they were real and simulate the trajectories to build the proposed models.

Second, we propose a mobile and beacon-based localization system. BLE beacons are rising and used for proximity purposes. Nevertheless, we propose a set of algorithms to find a simple solution to quick and easily deploy a beacon-based localization system, minimizing the calibration that in other proposals is necessary to obtain a good localization quality. Optimal solutions are pretended to offer the position with a high accuracy, however, we have considered to find a solution from the simplicity point of view.

Proposed solution is able to detect users position through their mobile devices, which are able to detect BLE signals and process their information by a suit of algorithms. One of them does not require any calibration, just the definition of the areas and the installed beacons. Other of the proposed algorithms require a minimum calibration and reduces the position estimation errors.

Finally, the combination of the two previous contributions are represented in the design of an application, which extends the mobile localization system with some other features such as tracking, trajectories creation or the customization of messages sent as notifications for user or user type. The main goal of Chapter 5 is to present a solution that propose the customization of a mobile-based application that combines the user positioning and the services personalization. User position is obtained by using the algorithms proposed in Chapter 4 and every time a user exits the localization area a trajectory is created. Though it is not done in this work, these trajectories could be used to create the proposed models in Chapter 3 to study their behavior. A notifications service is proposed and developed, which sends notifications to users according to the profile they belong to. Finally, a real application of this solution has been developed in both a controlled scenario and a grocery in Nice, where the set-up has been done in working days to add

more realism to the actual implementation.

Chapter 2

State of the art

In this chapter we do an approach to the two main topics that are addressed in this work: localization techniques and technologies and Business Intelligence applied to context-aware services. The way we do is not by creating an exhaustive and detailed state of the art, but a summary of some examples that have been carried out related to these topics, in order to contextualize this document and illustrate some orientations.

2.1 Techniques and technologies applied to localization applications

Context-aware services adequate their behavior according to the current and past status of the user, the environment and the user-context interaction. For instance, the position of the user at a specific moment, its trajectory, the current hour of the day and the user sitting on the sofa are some examples that can be a scene that induces a context-aware service to, for instance, start playing on the TV a customized content for that specific user. Here, we focus on the current and previous user position, how the actual user location is estimated and how to optimize the trajectory of the

user to offer a better information for activity estimation to an inference system.

Location estimation is not a novel problem but recently emerging technologies have appeared, such as Bluetooth 4.0, which are more adequate for new mobile devices. They make these not to consume so much energy maintaining their features for the development of location-based (LB) systems. Filtering the position of the user along the time is another task to estimate the correct trajectory of the user, trying to avoid impossible paths or movements, like swaps between two adjacent areas without a direct connection (e.g. a door). Finally, the study of these paths can make context-aware services to improve the way they work if they are aware of how users behave (in time-space terms), even according to their profiles. The user activity estimation using big data tools and the multiuser problem is another issue we try to solve in this work.

The position of the user is one of the main sources of information in context-aware systems: This parameter is estimated by positioning systems and is provided to third applications that offer a dynamic and customized service that can vary according to the user localization (Gu et al., 2009). This kind of systems can be applied to a wide sort of activities, such as fitness centers, to offer a personalized exercise based on the area or the machine the user is; malls, where an offer from a store can be notified when the user approaches; or banks or hospitals to inform clients or patients where or when to be attended if they are waiting. The goal in every case is to offer a more valuable service to the user depending on their position. In addition to the users localization, users tracking from the succession of their positions is an additional source of information for these context-aware services. Adding an upper layer to the localization information, which allows to classify users according to their movements, so that an even more customized service can be provided, what increases the added value of the service.

The estimation of the user position is in the last decade an easier job because of the integration of wireless sensors in commonly used mobile devices. Wi-Fi, Bluetooth, mobile networks or RFID have been originally used for other purposes like internet access, data exchange, phone calls or access control. But recent advances allow developers to use them for positioning purposes using different techniques when the mobile devices interact with the sensor networks. Indeed, big companies dedicated to smartphone manufacturing and operating systems working on them have their own location services. Google Maps, Apple Maps or Nokia Here are some examples of this type of location

systems.

In relation to wireless sensor networks, they are already deployed for communication purposes, but ad-hoc networks can be installed with location purposes. There are other objects in a location-aware infrastructure that can interact with the user without using a mobile device, such as a camera, a pressure mat or an infrared detector. Although they are used as impersonal location systems (except user pattern or body recognition, which can recognize who is the target to locate), these positioning devices can play as a support to other location technologies.

Most common wireless devices used for location purposes, such as smartphones, tablets or laptops, use signal features and sensor identifiers (proximity) to estimate the position. There are different configurations in the creation of a LB system that uses RF signals to be processed to estimate the position of a mobile device. These possibilities vary according to where the raw data are acquired and where the location is computed.

In relation to the data acquisition, the battery consumption is one of the weakest points of LB systems when this is done from a smartphone. It is required an almost permanent connectivity in wireless technologies. Some works try to reduce the battery consumption in mobile devices doing a post processing of an inertial system that indicates when the user is moving. In that case, the data acquisition component is launched (Oshin and Poslad, 2013). It can reduce about a 50% the battery consumption.

On the other hand, the localization computation can be done either in a location server or in the actual mobile device. This decision is conditioned by the required privacy of the LB system, the service purpose and the resources that are consumed in the data processing (Baniukeyevic et al., 2013). Using Bluetooth 4.0, devices discovery is faster than previous versions, which required between 10 and 14 seconds to ensure a complete devices scan. Now, using the new version of Bluetooth technology, 1 second is enough to do a full scan.

In this section a survey of the most used location techniques is summarized while converging to those that in this work are thought to be easy to deploy,

non-intrusive and offer a very suitable result for the purposes of this work. It is intended to be a smartphone oriented solution, where all the smartphone sensors data are gathered by the device itself, combining with proximity and user interaction sensors that are installed in the deployment area. It is notable to remark that the intention of the location system is to be integrated in a real solution and to be multiuser.

2.1.1 Indoors positioning

The emergence of new mobile devices, which are integrated with wireless technologies, and the fact that GPS does not reach indoor spaces have promoted new challenges in indoors localization techniques. Next, a survey of the most widely used techniques and technologies for indoors localization systems are reviewed.

2.1.1.1 Localization techniques

Following an approximation of the classification proposed in (Liu et al., 2007), there are three types of localization techniques used for positioning purposes: triangulation, scenario characterization (where RSS fingerprint is the most widely used) and proximity.

Triangulation: triangulation properties are used to estimate the position using, on the one hand, the distances between the nodes and the target and, in the other hand, the intersection of two lines when the angle of arrival is estimated. Distances-based algorithms estimate the distance from different parameters of the signals such as the time of flight or time of arrival (TOA) or received signal strength (RSS). Next, techniques based on the triangulation method are described:

- **TOA** The distance between an infrastructure node and the target can be estimated using the time of flight. Depending on the architecture and the possibilities of data acquisition of each device, it can be measured in the target or in the node. In relation to the type of system architecture, TOA is obtained. Both cases have their own handicaps. In the first case, both transmitter and receiver clocks must be synchronized to obtain the actual difference of times. In the second one, a delay can appear between the reception of the first signal and the transmission of the second one. In a 2D scenario, once three distances are calculated, ideally, the point the target is located is the intersection of the three circumferences that

can be formed with the center in each node and the radius equals to its respective estimated distance. In a real world, this distance varies due to the errors of the estimated times. Thus, the position could not be an exact point since the three circumferences could not intersect in the same point.

There are two scenario models. The first one is called LOS (Line of Sight) scenario, where both transmitter and receiver nodes can see each other without any obstacle. If there is not a direct path to connect both nodes, this is NLOS (No Line of Sight) scenario. There are works focused on one of the two scenarios like (Dashti et al., 2010) or (Alavi and Pahlavan, 2006) that study and model respectively the error due to the multipath interference, which is the first cause of errors in the estimations. The human body can be one of the most influential elements in both LOS and NLOS scenarios. The effects of the human influence in some parameters such as geographical relationship, bandwidth and power are studied in (Geng et al., 2013).

- **TDOA** Unlike TOA systems, TDOA uses the difference of the time of flight from two nodes to the target to build a hyperbola. Using three nodes, two hyperbolas are built in a 2D map and the position of the target is estimated in the point where they cross. The hyperbola is the curve in the space where the absolute value of difference of both distances to two fixed points (foci) is a constant for every point of the curve. Then, if the position of two nodes are known, once their TDOA is calculated, the hyperbola can be drawn in the 2D space. Some relevant examples found in literature about TDOA-based techniques in location-aware systems are (Tyler, 1999) and (Gao and Zhong, 2010) in mobile location applications or in (James et al., 1996) where vehicles tracking is proposed using TDOA. This seems to be a technique that is merged with some others such as angle of arrival (Liu et al., 2013) or the received signal strength (McGuire et al., 2005) in order to improve the position estimation. The improvement of TDOA accuracy is treated in studied in (Chang et al., 2014), where the integration of cross-correlation and genetic algorithms is used in 3D space-free scenarios.
- **AOA** Mentioned in the paragraph 2.1.1.1, angle of arrival (AOA) was merged to TDOA in (Liu et al., 2013) to improve both single localization techniques. AOA is a technique used in array antennas where the angle of arrival of a signal can be estimated. If the direction of two arrays is known and the AOA of the signal proceeding from a mobile node is

calculated, it is possible to estimate its position in a 2D scenario (Ozeki et al., 1996).

- **RSS - channel model** Received Signal Strength is a feature of the signal that is also related to the distance between two nodes. Once the RSS is known, the distance is calculated. Several channel models describe how the signal is attenuated with distance. Furthermore, other causes such as obstacles and multipath can be integrated into the channel models using 3D ray tracing tools like (El-Kafrawy et al., 2010).

In the following paragraph, we focus on the actual algorithms based on channel model. In section 2.1.1.3, the RSS behavior is studied. Some examples of location systems that use the channel model technique to estimate the position are: (Castano et al., 2004), (Zhou and Pollard, 2006) and (Wang et al., 2013). (Castano et al., 2004) merges a RSS model channel based location system that together a GPS-like triangulation algorithm estimates a mobile target with a 3 meters error but two years later, (Zhou and Pollard, 2006) proves that a model-channel based location system obtains a 1,2 m error. In (Wang et al., 2013) the problem of the no convergence of the n circumferences form by the distances between nodes is to be solved.

Fingerprint: There is a less precise (not accurate) technique that defines an area using a certain parameter. An RSS-fingerprint-based location system starts by the definition of a set of points in the deployment area. Then, an electromagnetic map is created using empirical measurements or theoretical tools to assign each point to an RSS-related information. For instance, this can be the mean of a set of values or an histogram. Then, using an algorithm that is based on the distance between the actual RSS vector and the fingerprint, the LB system approaches the user to the more likely position or area. A fingerprint-based system and some RSS-based algorithms that use NN or KNN techniques are studied and tested to obtain the position (Torteeka and Chundi, 2014) (Shin et al., 2012) (Tran et al., 2006) (Moghtadaiee and Dempster, 2012) (Metola et al., 2009). This technique has been deeply studied in (Honkavirta et al., 2009). It is a survey that runs through different comparing norms (p-norm, which comprises the Euclidean norm and Manhattan norm is the most used) and resolves that both KNN and weighted KNN work better than NN, with $K = 3$ and 4 as optimum parameters. A comparing norm is the mathematical formulation to estimate the non-spatial

distance between the beacon and the target to be located.

Additionally, the creation of the map can be less effort-costly or the map can contain more points if interpolation techniques are used to reduce the calibration work (Koweerawong et al., 2013). This reduction affects the time and effort that is required to calibrate a map. The calibration is comprised by an initial work and successive refinements. These are necessary to update the fingerprint if there are environment changes. In order to solve this last disadvantage, (Koweerawong et al., 2013) tries to minimize the number of new necessary measures (feedbacks) to estimate a new fingerprint using a linear interpolation, just taking a 5% of every point in the map. It ensures a 5 meters accuracy system updating up to 80% of every point just taking the 5% previously mentioned.

2.1.1.2 Proximity

There are further solutions that do not use exactly the already mentioned RSS-based techniques to estimate the position. Maps calibration, model-channel design and other location tools are discarded either because the cost-effort does not make up the entire deployment or because the accuracy is not intended to be so high and a symbolic position is enough in quality terms, e.g. the proximity of the user to a point of interest where any service could be enabled.

There are several technologies that can be used in proximity based location systems. A first type are those that sense the presence of an object or a body in the region they cover. For instance, a pressure mat installed on a sofa can detect when a person sits down or stands up, or an infrared sensor can detect if a user is in its coverage area. The drawback of these two types of sensors used in the examples are their impersonal feature, because they only provide a binary data if any user is inside its area of presence. For a multiuser space, this is a problem because we cannot know who is the user the sensor is detecting. This information needs to be complemented with another source of information, like a camera or a personal device (e.g. a smartphone) to detect the user.

Other technologies, such as Bluetooth or RFID are integrated in recent mobile devices and allow developers to build location systems using smartphones or tablets, which can provide a kind of identifier to know the target to locate. A first example is the possibility of using a device that emits a broadcast signal (e.g. Bluetooth) to play as a beacon, publishing, besides, its location or even matching an existing relation of mac addresses and positioning points - areas (Hallberg et al., 2003). Other solutions try to avoid using RSS creating a network of nodes that estimates the position of a target through search inquiries at two power levels. An average of 2 meters error is obtained (Forno et al., 2005).

Some examples of services based on the location using proximity techniques:

- The TV turns on if the user sits on the sofa (pressure mat).
- Lights turn on if the user enters in a room (infrared sensor).
- Send offer notifications to users if they are next to a shop in a mall (beacons)

Depending on the nature of the application, the distance to ensure there is "a proximity" between the user and the point of interest can change (Liu et al., 2014).

Beacons: In telecommunications, beacons are signal transmitters that send a unique identifier and, optionally, additional information. If the location of the beacon is known, it can be matched when detected by the mobile device to estimate the user position. Normally, it is associated to a point of interest. E.g. in a mall, the location system detects the beacon identifier that is related not to a x and y position, but a store. Then, the user can receive in his smartphone advertisements, discounts or any type of commercial notification. If the beacons are used only as proximity elements, each coverage region and their density of distribution in the deployed area define the accuracy of the location system.

The configuration of a network of beacons for positioning purposes has some features that must be taken into account. The number of beacons, their position (Xu et al., 2011), transmission power and period between two consecutive broadcasts can affect the way a LB service works. Thus in (Bostanipour and Garbinato, 2013) the two last cited parameters, the transmitted signal strength and the period between two consecutive

broadcasts, are studied in order to optimize the discovery of a target in a proximity based location system. Three environments are proposed: indoor with partitions (walls), indoor without partitions and outdoors. The energy consumption is not taken into account but we know that the higher the transmission power and the broadcast frequency are, the higher the energy consumption is, which in devices fed by batteries is a handicap. So that, defining a parameter that describes the detection probability must take into account the time the target is actually in range to the proximity beacon.

2.1.1.3 RSS issues

Two RSS-based location techniques were previously mentioned: channel model and fingerprint. Before operating and locate the target, a previous calibration must be done. Both the model definition channel and the characterization of power map can be performed either empirically or theoretically in a first phase, called calibration phase or offline phase.

(El-Kafrawy et al., 2010) (Gan et al., 2014) are ray tracing tools that support LBS developers to create a fingerprint and a channel model taking into account obstacle or multipaths effects. On the other hand, the calibration phase to create a fingerprint involves making a sort of measures, which initially are gathered from a set of points in the map. The creation of the map can be less effort-costly or this can be enlarged in number of points if interpolation techniques are used to reduce the cost of the calibration work (Koweerawong et al., 2013). This reduction affects the time and effort it implies the map calibration. In (Koweerawong et al., 2013), it is comprised by an initial work and successive refinements. These are necessary to update the fingerprint if there are environment changes. In order to solve this last disadvantage, it is intended to minimize the number of new necessary measures (feedbacks) to estimate a new fingerprint using a linear interpolation, just taking a 5% of every point in the map. It ensures a 5 meters accuracy system updating up to 80% of every point just taking the 5% previously mentioned.

As mentioned before, in a model channel-based positioning system, the previous phase can be either empirical, where the creation of the channel model is highly sensitive to environment changes, what implies a hard work of recalibration must be done eventually or periodically; or, otherwise, consider a loss-path attenuation of free-space for the propagation formula:

$$L_d = L_0 + 10\alpha \log(d) \quad (2.1)$$

where L_d is the signal loss at a distance d , L_0 the signal loss at 1 meter far from the emitter, and α is the coefficient of signal attenuation. The weather, indoors or outdoors propagation, the frequency of the signal and some other parameters can change this value, though this should not be considered static and suitable for certain conditions. For a 2.4 GHz technology such as Wi-Fi or Bluetooth in free space conditions, this value should be considered to be between 2 and 3 dB/meter. Walls and floors influence could be also considerable. Also CAD tools, which can simulate this value, can be used. Although its complex work for localization purposes has not a good accuracy cost-effort return, the calculation of this value in real time can be done. This technique adapts to imprecise changes and estimates the distance-power gradient while locating, using a non linear least squares method of the joint estimation. The estimation of the value offers a better lower average error (RMSE), overall if the prior estimation is lower than the actual one (Li, 2006).

The assumption of the error in the channel model design is shared by (Dulmage et al., 2010). They use the non-logarithmic model to point that α and P_{RX} contain errors respectively in their estimation and measurement. The ideal formula they work with is

$$p_{rx} = \frac{(p_{tx} k_1 m)}{d^\alpha} \quad (2.2)$$

where p_{tx} and p_{rx} are expressed in mW and k is the free-space loss constant at a distance d

$$k = \frac{4\pi d^2}{\lambda} \quad (2.3)$$

Thus, they work with the next formula if d is found, whose estimation can be calculated:

$$\hat{d} = \frac{p_{tx} k^{\frac{1}{\alpha+\epsilon}}}{p_{rx} + n} \quad (2.4)$$

where n and ϵ are the RSS estimate and model errors, respectively. It is easy to move from equation 2.1 to 2.2, where

$$L_d = P_{TX} - P_{RX} \quad (2.5)$$

and

$$L_0 = P_{TX} - P_{1m} \quad (2.6)$$

converting them to mW. Then, we have

$$p_{rx} = \frac{p_{1m}}{d^\alpha} \quad (2.7)$$

and being

$$p_{1m} = p_{tx} k_{1m} \quad (2.8)$$

we have 2.2. The standard deviation of the distance error is calculated using different values of the channel model error variance and two practical RSS estimation techniques: a finite window power estimator and an autocorrelation-based sliding window synchronization algorithm by Schmidl and Cox (Schmidl and Cox, 1997). It is concluded that the chosen RSS estimator varies the distance error according to the number of samples and even the actual distance. The Schmidl and Cox estimator can only be used in a window of distances. It means that if the receiver quite close to the emitter it would increase the distance error.

It can be concluded that RSS is an easy-to-use estimator for positioning purposes but not so robust as it should be. Ideally, it should maintain a fixed value in a static context, but there are works that study how RSS varies depending on the user position and orientation (Kaemarungsi and Krishnamurthy, 2004), the environment temperature, which varies the RSS about 5 dB between a range of 10 °C, the height of the sensor nodes, with an RSS variation of 5 dB between 6 cm and 135 cm over the floor or the antenna type, testing two types such as helix and patch antennas causing an RSS variation about 10 dB (Xu et al., 2011) (Kanaris et al., 2014). The calibration phase can be as costly as precise it is required to be the system. Empirical tests have been done to demonstrate how the mean of measurements from access points vary up to 10 dBs, what makes it difficult to have a robust

LB system (Kaemarungsi and Krishnamurthy, 2004). Even in non line of sight cases, the user presence can hide the visibility of an access point if its behind his back. Deviations of RSS affect to LB systems based on distance calculation. Its value changes and, then, the error in the location is more significant.

In (Wang and Wong, 2014), the study of the RSS deviation goes deeper. The random noise is calculated. Its value results to be less considerable than thought (below 1dB). Deterministic terms are calculated in the RSS formula. They vary even if the user is in the same position, but it depends on the user orientation. The position of the mobile device in relation to the user (carried by the user or placed on a table close to the user) is studied in Chapter 4. These defects are tried to be mitigated by a power-bias mitigation algorithm that is proposed, formulated and evaluated with a decrease of the mean distance error up to 1 meter depending on the used smartphone.

In the second phase of the LB system, actual measurements are compared to the RSS fingerprint created in the previous phase, so that the user localization can be estimated. In a real scenario, the user is holding the mobile device and his presence has been previously proved to affect the RSS. It is supposed that the RSS mean value in a set of measurements, if the user presence exists, significantly decreases. It degenerates the LB system because a mean value to define a point is not a significant parameter of the RSS histogram. Two more features of the RSS behavior proved in (Kaemarungsi and Krishnamurthy, 2004) are that this is left-skewed and that its standard deviation depends on the RSS level. Thus, it can vary with some parameters mentioned previously. So it can be concluded that, in a previous calibration, it must be taken into account not only a mean value but the wide set of values to create a more detailed source of information, for instance, a histogram.

As demonstrated before, a unique value of the RSS is a poor unit of information to estimate the position of a target. A trend in research is to improve the quality of these LB systems using not only a unique RSS value in a precise time instant, but a set of time-spaced samples or optimized RSS estimators (Dulmage et al., 2010). Furthermore, parameters filtering and smoothing algorithms are used to filter the current data and avoid outliers, so that a more representative information of the RSS is obtained (Xin-di et al., 2012) and improve algorithms with a pre-processing that decreases the average error a 18%. Every model of mobile device can contain a different wireless

sensor. It can result in different RSS values for different mobile device models. Thus, normalizing is a task that must be taken into account in LB systems. It is tried to solve in (Kjaergaard, 2007). A linear model mapping and a further processing for user tracking is used. It can improve symbolic location: give the user position not a point (x,y,z) but an actual zone, area or object the user can interact with. The mitigation of the heterogeneity in APs and mobile devices is also studied in (Mahtab Hossain et al., 2013), demonstrating that RSS comparison is more robust than RSS direct, what normalizes the bias differences between two different devices. In this work it is pretended to build a robust LB system that faces the problem of using different mobile devices, which can result in a difference in RSS value acquisition.

Another feature of the RSS signal is the angle of arrival (AoA). It has been studied the improvement of the location using directional antennas (Kanaris et al., 2014), constructing a radio-map using a 3D ray tracing tool and concluding that the positioning accuracy is improved in indoor environments if the antenna directions are right. Otherwise the accuracy could be worse.

2.1.1.4 Localization technologies

Wi-Fi: Wi-Fi is a wireless communications technology widespread and used in various fields such as home and work. The main use of this technology is to provide connectivity to different elements, both personal and at work (TVs, smartphones, computers, tablets). Wi-Fi components are based on one of the 802.11 standards developed by the IEEE and adopted by the Wi-Fi Alliance (WiF). Chronologically: 802.11a, 802.11b, 802.11g, 802.11n and 802.11ac. Wi-Fi uses two frequency bands for operation.

In many environments, the deployment of the necessary infrastructure to install a Wi-Fi based positioning system can be a fairly simple task. This is because the existing network of access points for internet access is used. The main disadvantage of using this already deployed set of devices is that the developer of the location system cannot control certain critical parameters such as the output power adjustment. As previously seen, in some RSS-based localization techniques, which is the case in this work, a variable power of emission causes a RSS vary and, therefore, the prior work of calibration or modeling is ineffective.

Using Wi-Fi or any other technology-based RSS algorithm does not greatly affect the used technique or algorithm. Therefore, any sample location system that uses Wi-Fi could be applied to other technology such as Bluetooth. The main different between both technologies is the range of Wi-Fi, which is larger. Then a series of fairly recent work currently using Wi-Fi and different types of techniques are listed. First, (Huang et al., 2015) proposes a fingerprints creation system from data collected by a large number of users in order to prevent the installer itself have to perform this difficult task. As this solution already has certain flaws beforehand since automatic data updates were performed with non-professional devices, a solution to rectify any fault in the creation of the fingerprint is proposed. This solution is based on creating correction points in areas in particular where the exact position is known, besides detecting attractions or landmarks in places of confluence of many users.

Another type of Wi-Fi-based solution is proposed in (Yang and rong Shao, 2015). In this case, localization methods that measure the time of arrival or the angle of arrival are used. Wi-Fi (or Bluetooth) is a technology with a width of relatively small band (40 MHz in Wi-Fi except the latest standards that increase to double and even at 160 MHz in some cases) to perform measurements of time of arrival accurate. On the contrary, UWB is a technology capable of take accurate measurements of time but this technology is not integrated into commercial smartphones. Therefore, for ToA measurements, a method called "multiple message approach" is used. This uses an antenna array, which also is useful for AoA measurements. The solution, if one AP in infrastructure is used, combines the two techniques, ToA and AoA. If more than one AP, then AoA is the technique chosen.

Another factor to consider in location systems is the current visibility between a source node and a receiving node (LOS vs NLOS). In the case of scenarios where there is no vision, there are works that try to improve the estimation of the location differentiating LOS and NLOS cases with different algorithms. The result is an improvement in accuracy of up to 60% (Xiao et al., 2015).

Bluetooth: Bluetooth is a technology that has been integrated into smartphones and cell phones since years for interaction and communication purposes (e.g. to send a picture) (Liu et al., 2009). The way this technology works, its shorter range and its not so wide use in communication sensor networks are features that allow other possibilities to Bluetooth-based technologies, apart from already commented techniques (fingerprint and channel model techniques used in LB systems), which are also possible.

Basically, this technology is based on RF signal, like Wi-Fi, thus techniques and algorithms documented in the previous section are valid if triangulation and fingerprint techniques are used. There is a wide set of works studying how this technology behaves for positioning, using distance-based algorithms (Castano et al., 2004) (Bluetooth + GPS) (Wang et al., 2013) (Zhou and Pollard, 2006). To enhance the possibilities that can be achieved using just one single technology, Bluetooth can be combined with other technologies such as Wi-Fi (Aparicio et al., 2008) and even GPS (Metola et al., 2010), where it is demonstrated that a combination of both technologies, that increases the number of resources in infrastructure to use (APs), improves the accuracy of the localization system.

Nevertheless, since Wi-Fi signals have a larger coverage, it is common to find Bluetooth as a support technology making a pre-location to isolate some areas and their Wi-Fi access points to enclose the problem (Baniukevic et al., 2011) (Baniukevic et al., 2013).

Bluetooth Low Energy: In relation to the concept of beacons, introduced above in section 2.1.1.2, the new revision of the standard Bluetooth, Bluetooth v4.0 or Bluetooth Low Energy (BLE), published in 2010, completes the previous reviews gathering the following features to the already revisions existing classic Bluetooth: high speed (based on Wi-Fi) and low consumption (Siekkinen et al., 2012). The low power feature is applied as a subset of the review v4.0 for fast communication through single connections. This is intended for integration into small devices with a button cell type battery. Therefore, two types of implementation are allowed:

- Only mode - only integrates BLE protocol stack.
- Dual mode - BLE functionality is integrated within a Bluetooth classic controller.

In this work, we will focus on the application of this new technology for locating systems. To do this, the term iBeacon, by Apple Inc., is briefly introduced. In Appendix B.1 this term is extended, presenting some market products and their actual state of development. They are small devices that implement the BLE standard. iBeacons were launched with the aim of being used as proximity elements. The way to operate with iBeacons is to deploy them strategically in an environment where the localization system is going to be installed, for example, a retail type store. The store interested in this localization system will provide the user with a mobile application capable of detecting iBeacons around and, roughly, estimate the distance between the iBeacon and the mobile device. The APIs provided by the manufacturers of iBeacons provide a poor estimate of the distance dividing the space into three distance ranges: immediate, near and far.

RFID: Radio Frequency IDentification (RFID) is a technology that uses tags or cards with a limited storage for information exchange. A simple RFID system is composed by a reader and one or a set of tags, each with a unique identifier. Additional information can be inserted in some types of tags, depending its nature and capacity. But the simplest systems only use the identifier to know where the user is (Yin et al., 2011) (Yang et al., 2013) (Yang, 2014).

The main advantage of using RFID for positioning purposes in LB systems is its easy deployment. For instance, a smart guide system is deployed in Auto Shanghai 2011 (Yin et al., 2011) where users wear RFID readers that receive notifications from active tags working at 2,45 GHz with a range of working distances between 0,5 m and 50 m. In this particular case, the readers that the users wear provide a sound service of guidance. The problem of using this type of technology for positioning purposes is that if only the id of the tag is used to know where the user is, the position of the tags and its transmission signal strength must be meticulously set up to avoid collisions among two tag signals. Besides, using active tags makes it necessary to be fed.

In (Yang et al., 2013), this problem is faced. Firstly, passive tags are used, which make the RFID system less powerful because the distance between the reader and the tags must be shorter. But here the focus is placed on how to distribute a set of tags over a grid to maximize the accuracy and precision, which are calculated only in function of the density of tags, instead of using

an algorithm to obtain a better precision. It is pretended to minimize the use of tags but maintaining a good precision. Nevertheless, reducing the number of tags implies to decrease the system accuracy. A high density in the tags deployment can increase the accuracy of the location system, but a tag collision problem can appear. But the same author tries to improve the precision using a location algorithm thought it is not so decisive as the configuration of the tags infrastructure (Yang, 2014).

NFC (Near Field Communication): is a subset of RFID that allows the deployment of services based on a real proximity. NFC could be known as an enabling technology (Siira et al., 2009), because unlike common RFID location systems, reader and tag must be intentionally quite close. Approaching the smartphone to a tag can offer the user to obtain information of a product, increasing its basic information. E.g. to open in the web browser the Wikipedia website of the product where the tag is placed.

Control access is a mechanism that can be carried out by NFC-based location systems (Köbler et al., 2010). Here, a well known communication tool is updated with automatic instant messages according to the actual status of the user. Check-ins and check-outs in an office environment are taken into account to provide, in this particular case, the actual status of the user. The main problem of this technology is the necessity of the voluntariness of the user. Other cases of use are proposed in (Nandwani et al., 2012), like personalized offers and advertisements or business information modeling.

2.1.1.5 Others

Acoustic - Ultrasound signals are used. The time of flight is measured and using the sound waves velocity the distance can be calculated. Then, using triangulation techniques the position is estimated (Priyantha et al., 2000) (Priyantha et al., 2001) (Prieto et al., 2009).

Infrared - Simple angle of arrival techniques to estimate the position using infrared technologies (Lee et al., 2004), though Active Badge is a well known and 'old' location system using infrared from personal transmitters to networked sensors (Want et al., 1992).

Visual - Video recordings are used for positioning purposes where outdoors LB systems cannot be used and as an alternative to RF-based techniques, having as a drawback its necessity to have line of sight (LOS) with the object to locate. There are many works about this type of technology that intend to locate in indoors environments (Schroth et al., 2011), object motion surveillance (Hu et al., 2004) and human activity detection (Ni et al., 2013).

Indoors GPS - A not extended technology for indoors positioning is the simulation of GPS in indoors (van Diggelen, 2002) (Soloviev and Dickman, 2011). Pseudolites are devices simulating satellites in indoors (Madhani et al., 2003) (Kim et al., 2008) (Bartone and van Graas, 1998).

Pedestrian dead reckoning (PDR) - method combines the accelerometer, which is used as a pedometer, and magnetometer in order to orientate the localization system. This technique can be used indoors with an error below 1m (Liang and Krause, 2016) or for movement classification (?).

2.2 Business Intelligence (BI) applied to context-aware services

2.2.1 Review of BI techniques

Is it really interesting to be aware of users behavior patterns for business improvement? (Capgemini, 2014) realizes it is - *'Over 60% of financial institutions in North America, for instance, believe that big data analytics offers a significant competitive advantage'*. They are aware of how little BI is used by business managers, while they own a great amount of customers information. The maturity level of BI and its tools in banks is below 50%.

All types of sectors such as hospitals, software engineering, hotels or energy market (Hans and Mnkandla, 2013) (Daryaei et al., 2013) (Argotte et al., 2009) tend to establish a relationship with BI. They realize that it is

impossible to do human analysis in the enormous number of data that are generated, so that a survey that can be applied to other disciplines is proposed.

BI or Business Analytics is the name given to the solution that faces the problem of how to manage in contemporary business organizations the huge amount of information (Chen et al., 2012) they handle. Since the emergence of new technologies, all the information that companies store and process for their working, both internal and external, is exponentially growing up. Data prediction and processing, pattern discovery, classification or clustering of elements are some of the fields where BI has entered to help data to be analyzed (Duan and Xu, 2012).

Among all the concepts that BI includes, we focus in this work in the Data Processing. Then, the objective in third chapter is to obtain the underlying information that allows companies to understand the users behavior. This allows business managers to get the necessary knowledge from a large list of customers profiles and their historic activity data, which are based on the user-context information. This information can be acquired using sensors or simulated. Before the availability of this powerful suit of tools that BI provides, elements registries, such as clients in a shop, were manually stored, their transactions and the creation of profiles to understand users behavior was not possible. Now, the availability of storing large amounts of data, the high capacities of processing and the possibility of using BI tools to simulate *real* profiles of users, cluster, classify and detect the type of profile to study their trends and provide a most suitable service is possible.

Works in literature divide the Big Data problem into different classifications. According to the type of work they do, (Argotte et al., 2009) classification divides it in: prediction, pattern recognition, modeling and others. Meanwhile, (Duan and Xu, 2012) classifies works:

- Supervised algorithms
 1. Decision Tree
 2. Bayesian Statistics
 3. Neural Network
 4. Support Vector Machine

5. Regression

6. Nearest Neighbor

• Unsupervised algorithms

1. Item-set mining

2. Clustering

Following this last classification by (Duan and Xu, 2012), some examples of the enumerated algorithms are described below.

Supervised algorithms: are used by machines to learn patterns from a set of input data, also called *training data*, whose output is also known. Once the algorithm is trained, a set of data, called *test data*, can be used either to obtain the output information starting from an input data or to evaluate the algorithm using an input data and comparing the actual output data and the estimated output data by the machine. Training and test data set should provide from the same distribution $p(x, y)$. Otherwise, it would be almost impossible too obtain y from a x that is not close to the data set that was used to train the algorithm. But there are some cases in which an arbitrary selection of the input data must be chosen, because any case of x can be very improbable and the algorithm could not be trained with that case. For instance, if the data set contains two successes for x : $x = a$ and $x = b$, and $p(x = a) = 0,01$ and $p(x = b) = 0,99$, many samples are needed if the distribution $p(x, y)$ is used to train the algorithm to have at least one case when $p(x = a)$. Nevertheless, if some samples for each case are used to train the algorithm, the distribution changes but the trainer is correctly trained to detect every case. Next, a classification of supervised algorithms is listed:

- *Decision Trees* are used in many diverse areas such as radar signal classification, character recognition, remote sensing, medical diagnosis, expert systems, and speech recognition (Safavian and Landgrebe, 1991). Next, the two most used algorithms for decision trees are described: *C4.5* and *CART* (Kohavi and Quinlan, 1999). *C4.5* is an algorithm used to generate decisions trees, based on ID3 algorithm from the same author, Ross Quinlan (Quinlan, 1993), and is commonly referred as a classification algorithm. The division of the data in each node is based on the use of the difference of entropy when choosing an attribute. The open

source implementation in Java of this algorithm is used in the classifier in Chapter 3 in *weka* software. This is named J4.8. This algorithm is improved with C5.0 with more velocity and a lower memory consume. CART algorithm is a binary decision tree that is constructed by splitting a node into two child nodes repeatedly, beginning with the root node that contains the whole learning sample (Breiman et al., 1984).

- *Bayesian Statistics* are used to classify elements as a supervised algorithm. Training data set could contain the relation of every combination of y , the clusters in which elements are classified, and x , the observed values of an element to infer its cluster. The combination that maximizes $p(y/x)$ provides the cluster for an observation x of an element. But, if the trainer does not contain every combination of $p(x, y)$, this value must be estimated. (Learned-Miller, 2011) presents one of the most used methods to estimate the value $p(y/x)$. From the training data, the class likelihoods $p(x/y)$ and the priors $p(y)$ are obtained:

$$p(y/x) = \frac{p(x/y)p(y)}{\sum_{z \in \varphi} p(x/z)p(z)} \quad (2.9)$$

being φ the combination of every cluster y . If priors cannot be obtained from the training data set, $p(y) \simeq 1/k$ where k is the number of clusters.

- *Neural Networks* are structures formed by nodes based on the neural structure of the brain. These nodes are connected each other or bidirectionally. The basic topology of directional networks consist of an input layer where each node receives one of the n inputs of the system, a hidden layer formed by m nodes that can combine until n inputs and outputs one signal to the output node, which receives the m outputs from the hidden layer. Its output is the system output of the neural network. On the other hand, the transfer function is also used to represent a neural network. This representation indicates that inputs are first weighted and summed, what combined form the propagation function. Second, an optional activation function and finally a transfer function to delimit the output value between a pair of values, such as 0 and 1 (sigmoid function) or -1 and 1 (hyperbolic function) are added to form the transfer function.

The weight values of the input connections are updated along a series of iterations in the learning phase. The supervised version of this algorithm compares the output of the neural network with the output related to

the corresponding input. The error is used to adjust the weight values so that the estimated output values fit to the actual output values (S.B. Hiregoudar, 2014).

- *Support Vector Machine* is based on the idea of classifying a set of points in one of two groups of states. A new point, whose group is not known, is inferred to belong to one of the two existing points. The construction of a hyperplane that optimally separates the space in two parts is intended. This hyperplane must have the maximum distance with the closest points of every group. Kernel functions, such as homogeneous polynomial or perceptron, are used to solve the problem of the computational constraints of lineal learning machines in no academic and ideal cases (e.g. more than 2 variables, no lineal curves to separate groups, etc.). (Cortes and Vapnik, 1995).
- *Regression*, unlike classification methods, which try to find the class a vector of data belongs, predict a value for a variable. For instance, the future price of a product, if the trend is known. Different fields such as medicine, computer science or traffic use regression algorithms to predict diseases, machine failures or traffic jams.
- *Nearest Neighbor* is a non-parametric algorithm used to predict (regression) and classify or pattern recognition classes from a set of data. The classification of an input vector in a cluster is done based on the calculation of the distance of the input vector $x_i = (x_{1i}, x_{2i}, \dots, x_{pi})$ with all the elements used to train the algorithm. The most common method to estimate the distance between two vectors is the euclidean distance.

Unsupervised algorithms: try to find a function to fit the input and the output of a set of elements using a *training data* whose output is not known. Here, the input data used after the training is used to obtain their output but not to evaluate the algorithm.

- *Neural Networks* Neural networks are introduced above and are also used in unsupervised learning. Some examples are Hopfield Networks (?) (Rojas, 1996) and Boltzmann machines (Hinton and Sejnowski, 1986) (Hinton, 2002) (Hinton et al., 2006).
- *Item-set mining* or association is a method to discover rules in series of data that are not previously matched. Unlike supervised learning methods such as decision trees, where a series of data were related to

an item so that the training is supervised-type, here the series of data are formed by a sort of items whose relationship is tried to be inferred. (Agrawal and Srikant, 1995) and (Srikant and Agrawal, 1996) refer to the Apriori algorithm and its improved version, GSP, whose objective is to find attributes commonly occurring together in a training set.

- *Clustering* techniques group a series of data according a criteria, based normally in distance. Clustering is not an algorithm itself, it is a methodology that can be applied using a specific algorithm. Some examples are k-means, k-means++, online k-means, k-medoids, which belong to the k-means family, whose main feature is to classify clusters in the *closest* cluster, being *closest* that with a lower distance. Other family of algorithms are hierarchical methods, which build a grouping tree according to their similarity.

2.2.2 Use of BI tools for behavior study

In this work, we have focused on using and modeling real scenarios where the application of BI and Big Data techniques to study the users behavior have been contemplated in the literature. Shopping is a very common activity for leisure. Malls and stores are willing to obtain information of their interest from customers like, for instance, how much time they spend in stores or which are the those they most visit. One of the most passive ways to obtain customers information is using their smartphones. Obviously, they must give their consent and, sometimes, install an application to be tracked and monitored or carry a support device so that the experiment is less intrusive. Privacy constraints are not contemplated in this work but we are aware that the consent of users is necessary to obtain contextual data using their mobile devices. (You et al., 2011) proposes a mobile-based solution to know how much time customers stay in stores, grouping trajectories in shopping paths and non-shopping with a hit over 88% of right classifications. The position of the customers is estimated using a match between the Wi-Fi access point they are closer and the region of interest it covers (Wilk and Karciarz, 2014). They also help themselves using inertial sensors integrated in almost every smartphone (e.g. accelerator and gyroscope) for movement detection and then classify them in complex activities (Nascimento et al., 2010).

(Yoshimura et al., 2014) uses also Bluetooth proximity techniques for positioning purposes. In order to show another scenario, this is focused in a mall to know common user trajectories. Entry and exit detection of regions

of interest are useful to know the presence of a user in a zone and the time she spends inside that area. In (Yoshimura et al., 2014) the relation between time of stay and behavior patterns are studied using Bluetooth technologies. They take into account the *wayfinding* concept: the ability of a customer to find the right path to get the goal and they also take into account negative tendencies, for instance, the user turns off the smartphone or the Bluetooth connection is off. Then, a large set of data such as total time of visit, number of visited nodes using repetitions and without repetitions, the number of unique visited nodes without repetition, etc., are some of the data that bring the useful information about users behavior.

On the other hand, another type of parameters can also be taken into account in order to understand the users behavior in a mall: household income, age, gender or education are demographic and economic indicators (Khare, 2012). Nevertheless, in this work we are focusing in observable parameters that do not enter in detail in personal, economic and social features of the customers. Entering in detail in the path customers do when shopping a clustering algorithm is proposed in (Lee, 2012) in order to discover hot-spots in trajectories carried out when shopping and to identify the customers' main pattern. It uses RFID for users positioning - RFID reader and active tags. Based on the Longest Common Sub-sequence technique, which is used to obtain the longest sub-sequence common to all sequences from a set of them, they compare two sequences using this parameter. Adding the factor that if this parameter is more valuable if compared sequences are shorter, they recognize pattern sequences using LCS-based parameter to group trajectories in different clusters.

K-means is one of the most common algorithms and most of new proposals are based on it or use it for clustering purposes (Lee, 2012) (Gil et al., 2009). This technique tries to minimize the distance between the elements and the representative mean in each cluster. K-means based algorithms are also used in real proposals for clustering of user profiles through their trajectories. An RFID-based system is proposed in (Larson et al., 2005) where a k-medoids clustering algorithm is implemented to make k-means tougher to outliers. The user tracking is based on observations done each 5 seconds. The application is also a grocery and the goal is to study the customers trajectories dividing them into three categories according to the time they spend inside the grocery.

From the point of view of marketing, not just the users paths are the most reliable source of information to know their tendencies, but, for instance, browsing through websites can be another way to follow a user in a not physical way. This changes the philosophy of the scenario studied in this work from the point of view of the user and the environment. The user, in the case referred to this work, is tracked through a set of already defined areas, so a trajectory is generated. In other cases different agents or objects to follow have been studied. For instance, the behavior of users when they are in a group or scenarios that are not discrete as an area whose position is given by x and y coordinates (Hui et al., 2009).

Users behavior and modeling is more deeply studied in (Rhee and Bell, 2002)(Zancanaro et al., 2007). For instance, when shopping customers normally have a main store that captures most of the user purchases, are they willing to change their preferences? Offers and low prices do not affect users preferences about main stores. After modeling users main stores preferences and studying why they would change it, next conclusions are obtained (Rhee and Bell, 2002): users do not tend to short-term preferred store changes; if a user has not changed her main store in a short-term, it will be more difficult to happen in a long-term; and location-related changes are more influential to main store changes in users.

Classifying users in profiles eases the understanding of how they behave in a specific scenario. In other words, it is not necessary or useful to define the behavior of every single user. For instance, as previously told, a bank may have different customer profiles such as a person who visits the branch specifically to obtain money from the ATM or an business manager to run credit issues. The trajectory and the behavior of each type of user vary depending on the profile.

The application to other scenarios has been conducted in other works like (Zancanaro et al., 2007), which first proposed to classify users from their trajectories rather than by historical activities or interests of the users like most of user behaviors studies do. (Zancanaro et al., 2007) characterizes users in different profiles and relate each one to an animal according to their similarity in behavior: ant, fish, butterfly and grasshopper. The chosen scenario is a museum.

- *Ant*: this type of user takes a specific way, and spend time seeing everything in the museum.
- *Fish*: type of user that mainly moves in the middle of the room without stopping in detail in the pictures.
- *Butterfly*: type of user that does not follow a specific path but is guided by the orientation of the exhibition of the museum and goes into detail on existing pictures.
- *Grasshoppers*: user with specific preferences that spends much time watching what matters.

The parameters used in the experiment with 143 people are mean time of visit, mean time in front of a picture and its standard deviation, the percentage of visited pictures and an indicator between 0 and 1 that represents in how many pictures the user has stopped. Then, a list of vectors is generated, as much as actual trajectories are. The previous mentioned parameters are used in order to obtain the 4 types of groups of user profiles using next two clustering algorithms: k-means and a recurrent artificial neural network.

In this work, the effort is done in both the BI techniques we use to look for the user behavior patterns and the tool to monitor the users trajectories in a real scenario.

Trajectories analysis: Entering in detail in clustering algorithms to obtain sequential patterns, unlike the use of parameters of the sequences like in (Zancanaro et al., 2007), (Agrawal and Srikant, 1995) and (Srikant and Agrawal, 1996) had studied the field of mining sequential patterns proposing a set of algorithms called AprioriSome, AprioriAll and GSP which are introduced above. The two first algorithms try to find similar patterns in a set of users transactions, while GSP introduces some constraints to give some coherence to the information: for instance, there is not too much time between two consecutive transactions because they should be considered independent.

In this work the method chosen for sequence mining is the conversion of trajectories to parameters indicating then how many times an event occurs, its duration and the number and type of transactions or transitions between two different states.

BI applied to users classification based on their behavior patterns

3.1 Motivation and objectives

3.1.1 Motivation

The detection of the kind of users that visit a specific environment can be a powerful source of information. Companies, for instance, can use this knowledge to offer adequate and customized services to their potential customers. Let's imagine a common business such as a mall. By tracking user movements, we could detect those users that are interested in clothes and those that are more interested in technology. Even there can be users that are interested in both types of stores. In this case, we could study user trends if the profile the user belongs is detected by tracking their trajectories. We could model different user profiles behavior, according to the profile they belong to. These models would help us to generate trajectories using a simulator and study with more information their behavior, without the necessity of taking so many trajectories.

Then, in this chapter we first propose the representation of users movement through Markov and non-Markov models. Markov models define the transition probability between two states and assume that the future state is only a function of the current one. Therefore, if this is applied to the movement of users, these models can describe how users move in a scenario, which is defined by a set of states that represent in turn a set of areas. Hence, Markov models define the transition probability of a user between two areas.

For example, Markov models define the users transitions probabilities as follows:

$$P_h(s_n/s_m), \forall m, n \quad (3.1)$$

where s_m is the starting area of the transition and s_n is the end area. This is the definition of the probability that a user that belongs to a user with a given profile h moves from s_m to s_n .

The motivation to model the movement of users is to know their behavior drawn from the users trajectories over a specific scenario. Many more features could be taken into account to infer the behavior of a user in a specific scenario, like her interaction with the environment and the kind of activities she performs. Nevertheless, movement patterns such as the number of times a place is visited or the places that are never visited are parameters that can define user behavior patterns.

The transition probability between two zones is a parameter that implicitly includes the number of times a user is in every area, besides the direction of the transition. Furthermore, this parameter can be defined using Markov models, which indicate the probability of moving between two states or zones, or a deviation of these models, which are not Markov type and that define the probability of moving to a specific zone, based on the current and the previous one. Markov models do not take into account previous state, what means they do not have memory. On the other hand, non-Markov models are proposed trying to take into account previous states and include one more element in the equation and adding the memory concept to the model.

3.1.2 Objectives

The main objective we face in this chapter is the definition of the Markov and non-Markov models that represent the behavior patterns of the different users, which are classified in profiles. In order to help the reader to understand the entire process that is carried out to fulfill the objectives described in this section, the schema in Figure 3.1 is presented to support their explanation.

Figure 3.1: General diagram to define the entire methodology

First of all, in order to create the models that describe the users behavior, a set of real trajectories must be used to estimate the transition probabilities between every pair of states (or areas) and create the models. As seen in Figure 3.1, from the real world we can get real trajectories. Nevertheless, if we consider both to use an example scenario, which actually does not exist, and that gathering many trajectories is not feasible in this work, we have created a mock-up real world to simulate it. Then, if in the real world we get real trajectories, with the mock-up world we generate simulated trajectories.

These are used, as explained later in this chapter, to generate both types of models, Markov and non-Markov.

The optimum number of simulated trajectories to create valid models is going to be estimated. If few trajectories are used, the models could not be accurate enough. But needing many trajectories to create the model could convert this task unpractical. Then, the first issue to be addressed is the estimation of the minimum trajectories that are necessary to create good enough and valid models that represent with a certain quality and confidence the movement of the different user profiles in a scenario divided in areas.

According to Figure 3.1, we have developed an application that, from both, Markov and non-Markov models, generates trajectories, which from now on are going to be defined as model-generated trajectories. Since models are supposed to represent user profiles, a good classification of the users in their corresponding profile, using the model-generated trajectories, is a good indicator to evaluate how models represent, in an indirect way, the real world. Therefore, the percentage of these model-generated trajectories that are correctly classified for every profile in a particular scenario is the indicator we have chosen to evaluate the quality of the models in this work.

The classification of the users in profiles through the model-generated trajectories is done once the algorithm is trained. Training phase requires a set of trajectories, each related to the profile of the user that performed it. In this work, we have used simulated trajectories to train the algorithm. Besides, we discuss the number of simulated trajectories that are necessary to do a correct training of the algorithm. The evaluation is also done taking into account the percentage of model-generated trajectories correctly classified in the corresponding profile.

The last objective in this chapter is to estimate the minimum number of trajectories that are necessary to ensure, with a certain confidence degree, the profile she belongs. To do that, we first identify a single user and estimate its profile by using the classifier, which has been previously trained. For every profile, we can ensure with a certain confidence degree the probability that the user belongs to every profile.

Finally, we want explain that our proposal in this chapter does not include a formal and strict statistical study. Particularly, statistic strictness about samples size, fit adjustment, etc., has not been considered in this chapter.

3.2 Description of the problem, a generic case

In this section, we are going to introduce the proposed methodology, which is applied for a generic case. This methodology describes the whole process. First, the simulation of trajectories using mock-up models, which represent the real life. Second, the creation of the Markov and non-Markov models, which represent the users behavior based on transitions probabilities. Finally, the estimation of the minimum number of required trajectories for both models creation and classifier training. The classifier is used to estimate the models quality and this quality describes how models represent the real world from an indirect way. Besides, once we are able to classify users from their trajectories with a certain probability of success, we can estimate how many trajectories are necessary to ensure the profile a user belongs to with a certain confidence degree.

This methodology will be described in detail in Section 3.3, using an example and, in Section, 3.4 using a generic scenario.

There is a background about the discovery of knowledge related to movement behavior. Trajectory pattern is considered as a concept in which frequent behaviors are detected, in terms of both space (areas frequently visited) and time (movements duration). There are works in literature that approach this problem by defining these frequent behaviors and finding areas of interest (Giannotti et al., 2007) (Aung et al., 2013). In our case, we propose a methodology to define Markov and non-Markov models that model user behavior, which are used to create mode-generated trajectories and to study users behavior.

3.2.1 Description of a scenario

Let's consider the scenario S . This consists on a set of N areas that represent the states in the Markov model:

$$S = [s_1, s_2, \dots, s_N] \quad (3.2)$$

A trajectory is the succession of visited areas. For instance,

$$tr = s_1, s_2, s_1 \quad (3.3)$$

represents a trajectory that starts in s_1 , continues in s_2 and ends in s_1 . Additionally, the time is a parameter that can be also added to enrich the information related to a trajectory. Then,

$$tr = s_n(1), s_n(2), \dots, s_n(K) \quad (3.4)$$

would become

$$tr_t = [s_n(1), t_{In}(1), t_{En}(1)], [s_n(2), t_{In}(2), t_{En}(2)], \dots, [s_n(K), t_{In}(K), t_{En}(K)] \quad (3.5)$$

where $t_{In}(k)$ and $t_{En}(k)$ are, respectively, the entry and exit times in s_n at step k .

3.2.2 Profile definition

The classification of users in profiles eases the study of users behavior if we now the profile they belong to. A scenario contains a set H of M profiles:

$$H = [h_1, h_2, \dots, h_M] \quad (3.6)$$

One of the objectives in this chapter is to model users behavior using Markov models, which indicate the probability of moving from the area m to n :

$$p(s_n/s_m), \forall m, n \quad (3.7)$$

Markov models define the probability of being in s_n at $k + 1$ step based only on the current state at k . In a real scenario, for instance, in a shopping center,

a person who is in the bookshop has a probability of going to the supermarket after visiting the bookshop: $p(\text{supermarket}/\text{bookshop})$. Nevertheless, if that person had already been in the supermarket before visiting the book shop, she will not likely return to the supermarket. Thus, the condition to estimate the probability of being in s_n at $k+1$ would depend on both the current state and the previous one. We add the memory concept to the equation. Then:

$$p(s_n(k+1)) = f(s_m(k), s_{m'}(k-1)) \quad (3.8)$$

where $n \neq m$ and $m' \neq m$. This is the second type of proposed model, a non-Markov model, which takes into account both the previous and the current states.

Then, in this work, we are going to define two types of models: 1st order models, Markov models or *M0*, which only take into account the current state k to infer the probability of being in s_n in the $k+1$ state; and 2nd order models or *M1* models, which take into account the current and the immediately previous state of the trajectory to infer the probability of being in s_n in the next state. These and a generic model that takes into account l previous states are defined as:

- *M0 model*: $p(s_n(k+1)/s_{m_0}(k))$
- *M1 model*: $p(s_n(k+1)/s_{m_0}(k), s_{m_1}(k-1))$
- *Ml model*: $p(s_n(k+1)/s_{m_0}(k), s_{m_1}(k-1), \dots, s_{m_l}(k-l))$

where $s_n(k+1) \neq s_m(k)$. In other words, two consecutive states cannot be the same.

In the case of the *M0* model, the estimation of the probability of going to s_n if the current state is s_m is calculated taking into account $N-1$ possible states, where the scenario S contains N different states. In case of the *M1* model, that same value depends on $(N-1)^2$ states. And in case of *Ml* model, the probability of going to s_n will depend on $(n-1) \times (l+1)$ states.

3.2.3 Estimation of the models

Proposed models, M0 and M1, are created from simulated trajectories that are generated in a scenario S that contains N areas. Let's remember that simulated trajectories are obtained from mock-up models that represent the real world and that are used instead of real trajectories to simplify the acquisition of real data (see Figure 3.1). Every trajectory, either real or simulated, is related to a user profile. Since there are M user profiles, M clusters of trajectories are created. Each cluster contains only those trajectories that have been acquired from the corresponding profile. Figure 3.2 represents the transitions diagram in S , where $s_n(1) = \text{Entry}$ and $s_n(K) = \text{Exit}$. *Entry* and *Exit* are two symbolic areas that represent the beginning and the end of the trajectory, respectively.

Figure 3.2: State diagram in a generic scenario

For each profile, the transition probabilities are independently estimated. Once we are provided with a set of trajectories, which each belongs to a unique profile, the definition of the parameter $p(s_n/s_m)$ of a M0 model is described as follows:

$$p(s_n/s_m) = u_{nm}/u_m \quad (3.9)$$

where u_{nm} is the number of transitions from s_m to s_n and u_m the number of transitions that start from s_m to all the states.

If $l = 2$ ($M1$ model or non-Markov model), the probability of being in s_n in the $k + 1$ step depends on the current and the immediately previous state. Besides, $p(s_n(k + 1)/s_m(k), s_{m_1(k-1)})$ and $p(s_n(k + 1)/s_m(k), s_{m_2(k-1)})$ are independent values, where $s_{m_1} \neq s_{m_2} \neq s_m$.

If we return to the generic Ml model, the transition probabilities depend on the current state $s(k)$ and the $l - 1$ previous ones. This must be considered that $l \leq k$. In other words, the number of steps (the previous and the current one) that are taken into account in the model, l , cannot be higher than the existing steps (k). If $l > k$, the model takes into account more elements than previous steps exist in the trajectory. The previous steps that do not exist are taken as null values.

To exemplify this case, let's take the $M1$ model as an example. If a trajectory is being created from the model and we are going to move from the $k = 1$ to the $k + 1$ step, and

$$s(1) = \text{Entry} \quad (3.10)$$

and

$$s(2) = s_n \quad (3.11)$$

then, $p(s(1))$ is only a function of the current state, regardless $l = 1$. So that the probability value is represented as $p(s_n/\text{Entry}, 0)$.

3.2.4 Generation of model-generated trajectories

In order to create model-generated trajectories, a desktop application has been designed and developed along this work, and validated in Appendix A.3.1, where the reliability of this type of trajectories is demonstrated. The comparison of the results of the trajectories with the theoretical values of the models is done to evaluate the models generation tool. The results compared are probability parameters such as $p(s_n/s_m)$. This value can be calculated, on one hand, using the theoretical models and, on the other hand, through the

previously simulated trajectories. Both the difference of these values for a set of parameters and the deviation of these differences are used to evaluate the model generation tool.

This tool enables the definition of scenarios with N areas and M profiles for both types of models: $M0$ model, which only takes into account the current position to estimate the probability of being in s_n in the $k + 1$ step, (see equation 3.7) and $M1$ version, which defines the probability of being in s_n in the $k + 1$ step as a function of both the current and the immediate previous steps.

3.2.5 Evaluation of the quality of the models

Regardless the type of model that is being used when the proposed methodology is applied ($M0$ or $M1$), every trajectory, either simulated or model-generated, is related to a unique profile. The correct classification of the trajectories in its corresponding profile is used to evaluate the models quality. Let's remember that these models are defined from a set of real trajectories r or simulated, in order to simplify the acquisition task. Besides, we described the models quality as how they can represent the real world in an indirect way. Then, the number of trajectories per profile is equally divided among the number of profiles. If there are M user profiles and r trajectories in total, there are $r' = r/M$ trajectories per profile that are used to define the Markov models.

In order to classify the model-generated trajectories in the corresponding profile, there are many algorithms as seen in Chapter 2. They are supervised algorithms that need, on one hand, a training data set to learn the patterns of the information that is studied and, on the other hand, a *test* data set that is going to be classified. Every unit in the training data set T is formed by the actual data and the profile it belongs. Meanwhile, the test data set t only contains the actual input data, so that the profile is inferred using the classification algorithm. In this case, the profile of the test data set is also managed to be aware of the quality of the classification.

In this section, the quality of the models is studied based on two parameters:

- the number of real or simulated trajectories that are used to create the models and
- the number of real or simulated trajectories that are used to train the classifier

We have decided to quantify the quality of the models according to the number of correctly classified trajectories, considering that we are not going to evaluate the classifier and that we use it just as a tool. These are model-generated trajectories. Before the classification, we know the profile every trajectory belongs to. Therefore, once the trajectories are created from the models and classified using the classifier, this quality is given by:

$$p = \alpha/t \quad (3.12)$$

where α is the number of correctly classified trajectories and t the number of simulated trajectories.

3.2.5.1 As a function of the trajectories used to create the models

First of all, the algorithm is trained using real or simulated trajectories. We manage three parameters:

- r : the number of real or simulated trajectories that are used to create the proposed models (either $M0$ or $M1$) or $r' = r/M$, which is the number of trajectories per profile, where M is the number of profiles.
- T : the number of real or simulated trajectories used to train the classifier.
- t : the number of model-generated trajectories that are used to evaluate the models.

The quality of the models is firstly evaluated as a function of r . Since the influence of T and t is intended to be minimized, a large amount of data to train the classifier and to test the models is used, so that these values do not significantly affect the evaluation process.

If, on one hand, t_m is the number of model-generated trajectories that belong to the profile m and are classified after the training phase and, on the other hand, α_m is the number of correctly classified trajectories, then:

$$p_m = \alpha_m / t_m \quad (3.13)$$

is the ratio of correctly classified trajectories related to the profile m . The quality result is given as P , which is the mean of all the ratios from every profile:

$$P = \frac{1}{M} \sum_{\forall m} p_m \quad (3.14)$$

For one profile, the association between r' and p_m , that is to say, the minimum number of trajectories r' for every profile that are necessary to create models with a probability p_m are related by:

$$\min \{r'\} \rightarrow \sigma p_m < th, \quad (3.15)$$

where σp_m is the deviation if i iterations are performed to estimate a more precise solution and th a threshold, which is defined according to the constraints and requirements of the models design.

On the other hand, for the total number of profiles:

$$\min \{r\} \rightarrow \sigma P < th \quad (3.16)$$

where P is the set of ratios of correctly classified trajectories from i iterations, individually represented as $P(i)$:

$$P = P(1), P(2), \dots, P(I), \quad (3.17)$$

σP is the deviation of P and th a threshold that defines the dispersion of $P(i)$ values

3.2.5.2 As a function of the trajectories used to train the classifier

The number of correctly classified trajectories, which defines the quality of the proposed models, depends on one hand on the number of trajectories that were used to create the models. Besides, the classifier could also affect the classification process of the model-generated trajectories. Hence, we wonder how many trajectories are necessary to correctly train the classifier, so that it does not affect the estimation of the models quality.

In this case, r and t are supposed to be large enough, so that they do not affect the classification process. A range of values to define T , the number of trajectories used to train the classifier, is chosen to detect the minimum of necessary trajectories to isolate the classification process from the training phase.

The evaluation process starts with the creation of models based on the previously inferred $\min\{r\}$ value. T is modulated using a wide range of values, in order to look for the lower bound that fulfills, for the complete case of all the profiles:

$$\min\{T\} \rightarrow \sigma P < th \quad (3.18)$$

where T is the minimum value, for the combination of every profile, which ensures that P is over a threshold th , following the same reasoning than in equation 3.15.

3.2.6 Classification of a single user

During this work, profile and trajectory concepts have been treated. Trajectories are created based on a model that defines user profiles behavior. On one hand, in order to evaluate the quality of the models, the number of required trajectories, real or simulated, to create the models has been discussed. On the other hand, it has also been evaluated how the number of trajectories, either real or simulated, that are used to train a classifier affects when the models are evaluated, since the quality of the models was decided to be quantified from the ratio of model-generated trajectories that are correctly classified in its corresponding profile.

However, if we particularize in a single user, whose identity and profile are also known, we know the a priori probability of classifying a user in her corresponding profile: h_m . In this section, we will look for the number of necessary trajectories n to ensure, with a certain probability degree, that a user belongs to her profile m .

This parameter depends on two variables: the user profile and a criteria value. This criteria c indicates, from a set of n trajectories, which is the parameter we are searching, the minimum subset to ensure the user belongs to the profile m , with a given certainty:

$$c \leq n_{min}/n \quad (3.19)$$

The binomial function is used to indicate the successful results from n trajectories using a criteria c :

$$f(n_{min})_m = \binom{n}{n_{min}} (h_m)^{n_{min}} (1 - h_m)^{n - n_{min}} \quad (3.20)$$

where h_m is the probability of detecting the correct profile from one trajectory related to the profile m .

For a range of values that n takes so that $f(n_{min})$ is evaluated, the minimum value is searched. For every different value of criteria, the minimum value of n where $f(n_{min})$ starts to converge is the required number that determines how many trajectories are necessary to ensure the profile of a user.

In Section 3.3, a real case is proposed and discussed whose results and conclusions are intended to be applied then to the generic case.

3.3 A case of study

In this section, the generic case is particularized using a mock-up scenario and five user profiles, so that proposed models are discussed using real values. In order to be more specific in this section, only *M0* and *M1* models are studied.

3.3.1 Scenario and user profiles definition

First of all, we have define a mock-up scenario formed by 5 areas, besides the Entry and Exit areas. This scenario is based on a simple shopping center that contains a supermarket, a technology store, a book store, a clothes store and a restaurant. The five areas are named by their initial to simplify their representation: S, T, B, C and R.

Next, five user profiles have been defined ad-hoc. These profiles are used to generate the trajectories along the mock-up scenario. It is important to note that these models are not based on *M0* or *M1* models. These are models based on a mock-up of the real world (see Figure 3.1. This first version of the models is based on a supposed behavior of real users. In other words, since we do not have real users for this mock-up models, we have decided to create them choosing a representation of five types of users, according to the zones that represent the mock-up scenario. They are completely invented and arbitrary, with the purpose of giving number and we only intend to represent with numbers the generic case discussed above

The models represent five types of common visitors that adequate to a set of ordinary behavior patterns. Trying to give a sense to the different user profiles behavior, users start visiting the supermarket, then they go to the stores and end their journey to the mall visiting the restaurant. Visits to the supermarket, the different stores and the restaurant are not *mandatory*. Obviously, at least one of them is actually visited. There is not such a trajectory that starts in the Entry area and suddenly is finished in the Exit area.

The five customer profiles are respectively based on persons with a certain cultural interest (*cultural*), persons with fashion preferences (*fashion*), persons who basically goes to the mall to do the shopping (*practical*), young who likes technology and clothing stores and finally goes to the restaurant (*young*) and

finally *shoppers*. These are represented by the models in figures 3.3, 3.4, 3.5, 3.6 and 3.7. Model trajectories can finish either in *exit* area or in another type of area. For instance, if a trajectory finishes in Restaurant (*R*), last state will be represented as *R*, assuming that the next one would be *exit*.

Figure 3.3: Model of the user profile type *culture*

Figure 3.4: Model of the user profile type *fashion*

Figure 3.5: Model of the user profile type *practical*

Figure 3.6: Model of the user profile type *young*

Figure 3.7: Model of the user profile type *shopper*

3.3.2 Application of the methodology to the case of study

Once we have defined a specific scenario and a set of models to particularize the generic case discussed in Section 3.2, we want to estimate the following figures of merit. First, the number of trajectories that are necessary to build these models and this number affects the way these models indirectly represent the real world. Second, the number of trajectories to correctly train a classifier, which is the tool we use to quantify the quality of the models. And finally, provided the trajectories of a single user, the number of required trajectories to ensure, with a certain confidence value, the profile she belongs.

First of all, the acquisition of real data from users is simulated through the ad-hoc profiles that were described in Figures 3.3, 3.4, 3.5, 3.6 and 3.7. Let's remember that at this moment, models have not been defined yet. These models represent the actual behavior of users if their patterns would have been observed in a real scenario. Nevertheless, we have decided to simplify this task by defining these models according to the profiles that we described previously, representing a mock-up of the real life. Therefore, we have a set of trajectories, which have been simulated, where each one belongs to one of the 5 existing profiles.

The collection of trajectories that are generated per profile approaches to T_r/M , being T_r the set of *real* trajectories that are simulated and M the number of profiles. In this case, $M = 5$. A uniform distribution is used, whose domain is divided into 5 equal sectors, so that the probability of generating a trajectory relative to a specific profile p_m is the same, regardless the value of m . Every time a new trajectory is going to be generated, the profile is chosen using a random variable over the previous uniform distribution and the proposed tool generates a new trajectory based on the model that correspond with the actual profile.

The estimation of the models based on the simulated trajectories is described below. If there are T_r trajectories in total, T_{rm} belong to the profile m . For instance, if we want to obtain the description of the *M0* model for every profile, the operation is performed individually. First, one of the profiles is chosen along with all the trajectories simulated from this profile. Every profile is going to be defined using simple transitions. The possible transitions are: *Entry - S*, *Entry - T*, *Entry - B*, *Entry - C*, *Entry - R*, *S - T*, *S - B*, *S - C*, *S - R*, *T - C*, *T - B*, *T - R*, *T - exit*, *B - T*, *B - C*, *B - R*, *B - exit*, *C - T*, *C - B*, *C - R*, *C - exit* and *R - exit*. Then, the number of times every transition appears in a trajectory, a counter related to the actual trajectory is incremented. This counter takes into account the total number of transitions of the combination of every trajectory of a single profile. The probabilities that define the model are calculated separating the transitions taking into account the origin state. For instance, if there are Tr_S transitions that start from the Supermarket, the probability of being in C if the current position is S is Tr_{SC}/Tr_S , being Tr_{SC} the number of transitions from S to C.

Example Let's use 10 trajectories using the *Shopper* profile to particularize it. The ten trajectories randomly created for the *Shopper* profile are:

- Entry, B, C, T, exit
- Entry, T, exit
- Entry, T, C, B, exit
- Entry, T, C, exit
- Entry, C, T, B, exit
- Entry, T, B, C, exit
- Entry, C, B, exit
- Entry, B, T, R, exit
- Entry, B, C, T, exit
- Entry, T, exit

In this example, there are 10 transitions that start from *Entry*. Therefore:

- $p(S/Entry) = 0$
- $p(T/Entry) = 5/10 = 0,5$
- $p(B/Entry) = 3/10 = 0,3$
- $p(C/Entry) = 2/10 = 0,2$
- $p(R/Entry) = 0$

and $\sum_n (s_n/s_m) = 1$ is proved. Hence, if the current state is *Entry*, every possibility for a next movement is contemplated and taken into account. And so on for every position to create the *M0* model.

The *M1* model is created using both the current and the previous positions of the user. Since Markov models only take into account the current state in a transitions model, this situation is solved converting the combination of *current state* / *previous state* as a current state. So that, going back to the

previous example of ten trajectories, one current state would be T/C . That is to say, the current position is the technology store and the previous one the clothes store. In this example there are only three transitions that start in T coming from C . These are:

- $p(B/T, C) = 1/3$
- $p(exit/T, C) = 2/3$

and

$$\sum_{\forall n} (s_n(k+1)/s_m(k), s_{m_1}(k-1) = 1 \quad (3.21)$$

This example takes into account only ten real trajectories to estimate the models of the user profiles, so that the representation can be evidently skewed due to the low number of trajectories to estimate the parameters to define the model. So, at this moment the first challenge is to determine how many trajectories are necessary to ensure a certain quality for both proposed models: $M0$ and $M1$ models in the proposed scenario, using the five user profiles described above.

3.3.2.1 As a function of the simulated trajectories used to create the models

Since the challenge is to determine the number of trajectories that are necessary, in this case, to have an accurate quality value that define the two versions of the proposed models ($M0$ and $M1$), a range between 10 and 50000 trajectories is used. 10, 20, 30, 40, 50, 100, 500, 1000, 5000, 10000 and 50000 are the number of trajectories that have been used to build the Markov models in the experiment described below. This gives a set of eleven sets of trajectories, each containing a different number of them. Using the method described above, the models are calculated for every profile.

The evaluation of the models is done using a classifier that is trained with simulated trajectories, but tested with model-generated ones. For example, if we start with 100 trajectories based on the mock-up models that are represented in Figures 3.3, 3.4, 3.5, 3.6 and 3.7, each profile would be represented by approximately 100/5 trajectories, and $M0_{100}$ and $M1_{100}$ are

the proposed models. On the other side, 100000 trajectories, which have been simulated from the mock-up models, have been used to train the classifier and two sets of 100000 model-generated trajectories, each of them using $M0_{100}$ and $M1_{100}$ models. Each trajectory that is going to be classified is associated to a specific profile, which helps the classifier to know if the classification of the trajectory was correctly done. Then, the accuracy for each type of profile is obtained, for models created using 100 real trajectories:

$$p_m^{100} = \alpha_m / t_m \quad (3.22)$$

being α_m the number of trajectories that belong to the user profile m and have been correctly classified and t_m the total number of trajectories that belong to the user profile m . This classification has been done using Weka application.

Weka is a free software platform created to carry out machine learning and data mining tasks. In this work, it has been used to implement the C4.5 algorithm, a supervised learning-based decision tree used to classify elements. Its open source implementation in Java is J48 and this is used in Weka to carry out the classification tasks of this work (Witten et al., 1999) (Quinlan, 1993).

Therefore, we estimate one value of accuracy for every user profile, and a combined value of accuracy from the five proposed profiles, P^k , being k the total number of trajectories used to estimate how many are needed to create models with a certain confidence degree. Since some times, using a low number of trajectories k_1 , the accuracy value can be higher than using a higher one k_2 being $k_2 > k_1$, the intention is not to find a good value of accuracy for the studied model, but a stable value that can be defined with a certain degree of confidence.

Figure 3.8 represents the average, for all the user profiles and separated by the number of simulated trajectories used to create the proposed models, of the quality of the two type of discussed models, M0 and M1. Let's remember that we have decided to define *model quality* to that parameter that defines the number of simulated trajectories, based on the proposed models, that are correctly classified to that user profile that defines the model. Besides, the evaluation process has been done 10 times, each with 10000 different simulated trajectories used to train the classifier and 100000 different model-generated

trajectories to be classified. A separate process has also been done to test every type of model, M0 and M1. The presented results are averaged for the type profiles. A high number of simulated trajectories and model-generated trajectories used, respectively, to train and to test the classification algorithm was chosen to reduce the effect of the number of trajectories used in this task as much as possible.

Figure 3.8: Ratio of correctly classified trajectories, for *M0* and *M1* models, as a function of the number of trajectories used to create the models

As mentioned below, 10 iterations were done for each set of trajectories. Thus, the maximum and minimum values of correctly classified trajectories are represented for each set of simulated trajectories used to generate the proposed models. Besides, the mean value is represented too. Figure 3.8, for every domain unit, is less dispersed when the domain value is bigger. Roughly, from 100 trajectories on, the image values tend to converge, which means a higher certainty in the number of trajectories a classifier correctly classifies, if these trajectories are simulated using models that are, in turn, created using the number of trajectories represented by the actual domain value.

The selection of the optimum number of trajectories to create valid models should be done, based on the standard deviation of the 10 experiments in each case, by the models designer according to the requirements of the quality

values (number of correctly classified trajectories) and their dispersion. As seen in Figure 3.8 and Tables 3.1 and 3.2, the quality values are not directly proportional to the number of trajectories, but its dispersion.

For instance, if the requirement of the design is to offer a model whose quality standard deviation is under 1%, the creation of the M0 model would require at least 30 trajectories. This value is not a very fine approach, because the domain values that have been chosen is quite disperse. On the other hand, in order to create a M1 model, at least 500 trajectories should be used to create a M1 model with a quality standard deviation above 1%.

	<i>number of trajectories</i>					
	<i>10</i>	<i>20</i>	<i>30</i>	<i>40</i>	<i>50</i>	<i>100</i>
<i>M0 average</i>	87,69	85,9	87,4	85,45	85,95	88,5
<i>M0 SD</i>	2,61	1,73	0,58	0,57	0,6	0,71
<i>M0 average</i>	93	91,62	94,73	90,05	91,98	94,71
<i>M0 SD</i>	1,57	1,97	1,16	0,81	0,77	1,09

Table 3.1: Mean and standard deviation of the quality values as a function of the trajectories used to build the models (I)

	<i>number of trajectories</i>				
	<i>500</i>	<i>1000</i>	<i>5000</i>	<i>10000</i>	<i>50000</i>
<i>M0 average</i>	86,68	86,1	86,11	86,05	86,07
<i>M0 SD</i>	0,34	0,21	0,22	0,12	0,15
<i>M0 average</i>	92,64	92,44	92,51	92,37	92,3
<i>M0 SD</i>	0,32	0,27	0,2	0,12	0,09

Table 3.2: Mean and standard deviation of the quality values as a function of the trajectories used to build the models (II)

3.3.2.2 As a function of the trajectories used to train the classifier

The previous experiment does not take into account the number of trajectories that are used to train the classifier. Next, the search of the minimum acceptable value of training trajectories is discussed. Using the same classification algorithm, a set of simulated trajectories has been created in the same range of values than above: 10, 20, 30, 40, 50, 100, 500, 1000, 5000, 10000 and 50000; which is used to train the classifier. For each case,

10 different samples have been created to avoid not precise values. In this case, the classifier is trained using from 10 trajectories to 50000, while this is tested using 100000 simulated trajectories that correspond with the obtained in the experiment explained above. Figure 3.9 represents the accuracy of the classification, averaging the 10 cases and the 5 different profiles, and shows how the value converges to the actual accuracy values that correspond to each model.

Figure 3.9: Accuracy of the classification for $M0$ and $M1$ models based on the number of trajectories used to train the classifier

The quality of the classification and the dispersion of this quality has been demonstrated that also depend on the number of simulated trajectories used to train the classification algorithm (see Table ??). The selection of the number of trajectories that are going to be used to train the algorithm is done under criteria of the user that is going to analyze the trajectories. In this case, unlike the previous chart, the quality increases as the number of trajectories does too until it converges to the qualities obtained above for $M0$ and $M1$ models. The standard deviation is minimized if the number of trajectories increases, then the user that analyzes the trajectories can choose the optimum value of trajectories to train the classifier according to the quality of this classification. Tables 3.3 and 3.4 contains the values for the measurements

done for this experiment.

	<i>number of trajectories</i>					
	<i>10</i>	<i>20</i>	<i>30</i>	<i>40</i>	<i>50</i>	<i>100</i>
<i>M0 average</i>	47,86	69,05	76,51	74,69	79,22	79,14
<i>M0 SD</i>	11,73	10,1	5,96	5,45	1,74	2,48
<i>M0 average</i>	52,44	75,88	84,99	81,84	87,63	87,28
<i>M0 SD</i>	12,88	11,1	6,03	4,9	1,51	2,58

Table 3.3: Mean and standard deviation of the quality values as a function of the trajectories used to train the classification algorithm (I)

	<i>number of trajectories</i>				
	<i>500</i>	<i>1000</i>	<i>5000</i>	<i>10000</i>	<i>50000</i>
<i>M0 average</i>	84,49	83,53	84,96	84,66	86,07
<i>M0 SD</i>	2,85	1,54	0,11	0,80	0,15
<i>M0 average</i>	90,62	91,58	92,1	92,13	92,3
<i>M0 SD</i>	1,44	1,24	0,11	0,12	0,09

Table 3.4: Mean and standard deviation of the quality values as a function of the trajectories used to train the classification algorithm (II)

3.3.2.3 Classification of a single user

Proposed models represent the behavior of groups of users, who are separated or clustered in user profiles, according to their behavior. Thus, if we particularize to one single user, her behavior is described by the model that represents the profile she belongs. Nevertheless, in real life, users do not behave always in the same manner. In other word, in the case of the proposed scenario in this chapter, a user is not 100% *Buyer*, let us say sometimes she might behave as a *Buyer*, others as *Fashion* and some visits to the mall will have a combination of patterns of two or more profiles.

The case of mixture profiles could have perfectly been considered, nevertheless, we have considered to assume users only belong to one profile type. Then, the user can be either correctly classified in the actual profile she belongs or one of the other four profiles, what would be a bad classification. If the probability of correctly classifying a user is known and one trajectory is observed (see Table 3.5), we are going to look for the number of trajectories that are necessary to

ensure, with a certain confidence degree, that the user actually belongs to the corresponding profile. The probability of correctly detecting a user from one trajectory provided she actually belongs to the m profile is h_m :

$$h_m = P(\hat{E}_m/R_m) \quad (3.23)$$

what means the probability of estimating a user in m , $\hat{E}_m$, if she really belongs to m , R_m . If n trajectories are observed from the same user, the probability of correctly classifying the user as m is:

$$h_m^n = \sum_{i=x}^n \binom{n}{i} (h_m)^i (1 - h_m)^{n-i}, \quad (3.24)$$

being i an integer that starts from x , and x the next integer that fulfills $c \leq x/n$.

100000 *real* trajectories have been created using the models describe in Figures 3.3, 3.4, 3.5, 3.6 and 3.7. Weka offers a cross-validation tool, which divides the set of elements into f equally blocks. By default, a good value for f is 10. Then, 10 blocks of 10000 trajectories each are used to estimate the quality of the classifier. Ten iterations are done, in which 9 of the blocks are used to train the classifier and one to test it. In every iteration the test block changes so that every block acts just once as a test block. Ten results are obtained and their average represents the conditional probability of Bayes' theorem: $P(\hat{E}_m/R_{m'})$ where m and m' are profiles that do not have to be the same.

<i>Actual profile</i>	<i>Estimated profile</i>				
	<i>Fashion</i>	<i>Culture</i>	<i>Shopper</i>	<i>Young</i>	<i>Practical</i>
<i>Fashion</i>	1	0	0	0	0
<i>Culture</i>	0	0,98	0	0,02	0
<i>Shopper</i>	0,03	0,08	0,86	0,03	0
<i>Young</i>	0,09	0	0	0,91	0
<i>Practical</i>	0	0	0	0	1

Table 3.5: Confusion matrix of real trajectories classification

Then, if a user belongs to the *Fashion* profile, she will be certainly classified in the *Fashion* cluster. Nevertheless, if the user belongs to the *Culture* profile,

the probability of being correctly classified will be $h_C = 0,98$, what means the relation between the correctly classified trajectories and the actual number of trajectories of *Culture* profile. Therefore, the conditional probabilities for each profile to correctly classify a user in the profile she belongs are:

- *Fashion*: $h_F = 1$
- *Culture*: $h_C = 0,98$
- *Shopper*: $h_B = 0,86$
- *Young*: $h_Y = 0,91$
- *Practical*: $h_P = 1$

Once we are provided with a set of trajectories from the same user, if we pretend to classify this user in one specific profile, we have to decide the percentage of trajectories that must be classified in that profile. For example, if a user that belongs to the *Culture* profile is observed 10 times, how many trajectories must be classified as *Culture* to claim that the user belongs to the actual *Culture* profile? This is solved using the parameter *criteria*, which is a percentage where: $c \leq x/n$, being c the criteria, $0 \leq c \leq 1$, n the total number of trajectories and x the number of correctly classified trajectories. For instance, for $n = 10$, if the criteria is the 50% of the trajectories: $c = 0,5$ and $x = 5$. Thus, to ensure a user belongs to the *Culture* profile using a criteria of 50%, if 10 trajectories have been observed, at least 5 must be classified as *Culture*.

Focusing on the five profiles whose probabilities of detection have been estimated above, the probability of correctly classifying a user profile type *Fashion* or *Practical* is 100%. That is to say, every time a user whose profile is *Fashion* or *Practical*, every trajectory she performs is going to be classified correctly. Nevertheless, if the probability of correctly detecting a single trajectory that belongs a profile m is $h_m < 1$, the number of trajectories that are necessary, according to the criteria, to ensure the correct classification of the user is presented in Equation 3.24.

Setting a range for n that goes from $n = 1$ to $n = 1000$ trajectories and 5 criteria: 50%, 60%, 70%, 80% and 90% for the three profiles whose $h_m < 1$, *Culture*, *Young* and *Shopper*, the charts that reflect h_m^n are respectively

presented in Figures 3.10, 3.11 and 3.12.

Figure 3.10: Probability of correctly classifying a *Culture* type user based on the number of trajectories

In Figure 3.10, whose $h_C = 0,98$ the probability of correctly classifying a user converges quickly to 100% before having 100 trajectories. The most restrictive case is $c = 0,9$, this is, a 90% of the trajectories must be correctly classified. If the criteria is 80% or lower, 20 trajectories are enough to confirm a user whose profile is *Culture*.

Sawtooth effects happen because the number of trajectories is an integer value. For instance, if $n = 20$ trajectories are observed and the criteria is $c = 90\%$, the number of required trajectories to correctly classify to ensure the user belongs

Figure 3.11: Probability of correctly classifying a *Young* type user based on the number of trajectories

to the *Culture* profile are $x = 18$, $x = 19$ or $x = 20$. The binomial values in the three cases are:

- $h_C^{20}(x = 18) = 0,0528$
- $h_C^{20}(x = 19) = 0,2724$
- $h_C^{20}(x = 20) = 0,6676$

what makes $h_C^{20}(c = 90\%) = 0,9929$. In other words, if we observe 20 trajectories from a user that belongs to the *Culture* profile, the probability of ensuring that the user belongs to the *Culture* profile is 99,29%, if we set a criteria $c = 90\%$ (at least 18 times must be correctly classified from 20 observations). If n increases, next closest values for $c = 90\%$ require to ensure the user belongs to the corresponding profile in three cases (e.g. if $n = 25$, $x = 23, 24, 25$ until $n = 30$, what makes x to take four values: $x = 27, 28, 29, 30$ and the summation related to Equation 3.24 increases. From $n = 20$ to $n = 30$ the elements of the summation have a lower value, which is represented by a descendant until $n = 30$ where the jump occurs.

Figure 3.11 represents the probability of correctly classifying a user that belongs to the profile *Shopper* as a function of the number of trajectories. In this case, $h_Y = 0,91$, slightly lower than p_C , what means the number of

Figure 3.12: Probability of correctly classifying a *Shopper* type user based on the number of trajectories

necessary trajectories to confirm the profile of the user is larger. Looking at the curve that represents $c = 0,8$, it converges to 1 if $n > 50$ trajectories. And if the criteria is $c = 0,9$, it will converge to 1 if n is extremely large.

The last example refers to the *Young* profile (see Figure 3.12). In this case $h_S = 0,86$, this is, lower than 0,9, so that if $n \rightarrow \infty$ and the criteria is $c = 0,9$, $p_m^n = 0$. The reason is that if the requirement is to ensure that from 10 trajectories, 9 of them must be correctly detected, and if $h_S^1 = 0,86$ and n trends to ∞ , this assert is not possible to be ensured with the criteria of 90%. If $c = 0,8$, at least 300 trajectories are necessary to correctly classify this type of user and if c is decreased, a lower number of trajectories are required.

The main conclusion that can be obtained after the observation of the three charts that represent the confidence that a user belongs to her corresponding profile, for the three profiles that $h_m \neq 1$, is that if $h_m > c$, if $n \rightarrow \infty$ then

$h_m^\infty = 1$. Otherwise, if $h_m < c$, if $n \rightarrow \infty$ then $h_m^\infty = 0$. Finally, if $h_m = c$, $h_m^\infty \approx h_m$.

3.4 Application to a generic scenario

In this section, we propose a guide to follow our methodology to create models based on either real or simulated trajectories, optimizing the number of necessary trajectories to be created and to train the classifier. Besides, we add the steps to follow to obtain the number of trajectories that are necessary to ensure, with a certain probability, the profile a user belongs to.

1. *Scenario definition*: first of all, the scenario and its transitions matrix must be estimated and created. If the scenario contains N states, $N = s_1, s_2, \dots, s_N$, including both *Entry* and *Exit*, an $N \times N$ matrix must be defined where s_{nm} is the possibility of moving from s_m to s_n , being 1 if possible and 0 if not. To resume, it represents the organization of the space and the possibility to move from an area to another.
2. *Profiles definition*: second, M user profiles are ideally described from real user movements. If it is not possible to acquire real data from users, a mock-up of the real life is proposed to create mock-up models that could represent different types of users. These models are tree type models, so paths are entirely defined (see Figures 3.3, 3.4, 3.5, 3.6 and 3.7 as examples).
3. *Models design*: models creation methodology, based on simulated trajectories from mock-up models, which in this case are $M0$ and $M1$ type models, is designed and described in Section 3.2.3.
4. *Model-generated trajectories*: once the models are designed, these are loaded in the models-generation tool, along with the scenario. The number of trajectories that are required to create models with a certain quality is estimated for the combination of N states and M profiles. A discrete range R of number of trajectories is arbitrarily chosen $T = [T_1, T_2, \dots, T_R]$. For each set T_r of trajectories, T_r trajectories are simulated or acquired in a real scenario to estimate the models. If this is simplified for $M0$ and $M1$ models, R models of each type are obtained: $[M0_1, M0_2, \dots, M0_R]$ and $[M1_1, M1_2, \dots, M1_R]$. For each model, a very large set of trajectories is generated to be classified, while

the classification algorithm is trained using simulated trajectories. The same percentage of trajectories for each profile is used to train the algorithm and to be classified. So, if there are T_r trajectories and m user profiles, there must be about T_r/m trajectories for each profile.

5. *Classification:* for each model, a quality of classification is obtained: $QM0_r$ and $QM1_r$. If this experiment is done 10 times, with different real trajectories that bring each different Markov models $[M0_r^1, \dots, M0_r^I]$ and $[M1_r^1, \dots, M1_r^I]$, 10 qualities are obtained for each model: $[QM0_r^1, \dots, QM0_r^I]$ and $[QM1_r^1, \dots, QM1_r^I]$. The qualities of the models $QM0_r$ and $QM1_r$ are estimated through the average of the 10 qualities:

$$QM0_r = \frac{\sum_{\forall i} QM0_r^i}{I} \quad (3.25)$$

$$QM1_r = \frac{\sum_{\forall i} QM1_r^i}{I} \quad (3.26)$$

being I the number of total iterations. In this case, $I = 10$. The qualities are given in percentage. This value is an indicator of the quality of the model but the selection of optimum number of trajectories is described below.

6. *Estimation of the minimum number of trajectories to create the models:* that set r where the difference between $QM0_{rmax}$ and $QM0_{rmin}$ converges to less than one point is the minimum number of trajectories needed to create models with a convergent accuracy. If the range of sets is not quite precise, the operation can be repeated using an approximation to the values around the first minimum, so that the number of trajectories can be more precise.
7. *Estimation of the minimum number of trajectories to train the classifier:* the number of trajectories that are necessary to use correctly the classifier is estimated in the generic case using those models that were created with the optimum number of trajectories. If a model based on more simulated trajectories is used the result will not get worse. Then, another discrete range of sets is chosen $T = [T_1, T_2, \dots, T_T]$. For each

set, 10 groups of as many *real* trajectories as T_t indicates is created (or acquired if this is possible in the actual scenario): $T_t^1, T_t^2, \dots, T_t^{10}$. Using the optimum models to create a large group of test trajectories to not affect the classification, the classifier is trained using every single group of trajectories and a tested with the test trajectories. Then, a quality for each case is obtained: $QT_t^1, QT_t^2, \dots, QT_t^{10}$. In this case, the qualities for each set of trajectories is averaged to estimate the value and the maximum and minimum are obtained to choose the optimum number of trajectories to train the classifier. That value of T whose difference between QT_{tmax} and QT_{tmin} is lower than 1% is chosen as optimum value. The result can be optimized if an iteration is done using more precise values.

8. *Classification of a single user:* once the models are created and the classifier is trained using at least the optimum number of trajectories, a user can be classified into one of the existing profiles after n trajectories. The estimation of that n value for a generic case is done based on the classification of real trajectories to obtain $p_m \forall m$. A criteria must be chosen to decide how many trajectories are enough to ensure a user belongs to one of the profiles. This value is $c < x/n$ being x and n integer values. For a range N of trajectories, for each value of n in that range: $p_m^n = \sum_{i=x}^n \binom{n}{i} p^i (1-p)^{n-i}$. If p_m^n is represented in a chart, it will tend to 1 or 0 depending on the p_m value and the criteria. If $p_m < c$, $p_m^n \rightarrow 0$, otherwise $p_m^n \rightarrow 1$. The quickness the curve tends to 0 or 1 is reflected in the number of required trajectories to estimate the correct classification of a user. If $p_m = c$, $p_m^n \rightarrow 50\%$.

3.5 Conclusions

The representation of users behavior using Markov (no memory) and non-Markov (with memory) models has been proposed in this chapter. We have focused on users behavior as described by their movement and the activities they perform in a scenario. Scenarios, which have been described as a combination of areas where the transition among them is defined by a transitions matrix, can represent any real environment such as a hotel, a museum or a hospital. On the other hand, in order to distinguish different types of user behaviors, users have been divided in profiles. Thus, every profile can be defined by a model that represents the probability of moving between

two areas.

Markov models, *M0*, only take into account the most recent state in a succession of states. Nevertheless, we have added a second model type, *M1*, which takes into account the two most recent states. In the same conditions, Pictures 3.8 and 3.9 show how the model that takes into account the two last states instead the last one provides a better result.

The quality of the models, which we decided to evaluate as their ability to represent in an indirect way the real world, is obtained using a classifier. On the other hand, the ability of the classifier to correctly classify a set of trajectories, once it has been trained, is evaluated using the ratio between the number of correctly classified trajectories and the total number of trajectories that have been classified.

Again, we must clarify that we call *real* to those trajectories that are generated in the real world, while *simulated* are obtained by simulators that represent a mock-up of the real world. Once models are created from *simulated* trajectories, we can generate *model-generated* trajectories to understand the behavior of the different user profiles.

The number of simulated trajectories used to create the proposed models and the number of simulated trajectories used to train the classifier affect the quality of the models and the quality of the classification, respectively. In the first case, when the number of trajectories used to create the models increases, the quality of the models converges to a certain value, which enables the possibility to ensure how the simulated trajectories represent the different user profiles. The number of trajectories used to train the classifier also affects to the *result* value. In both cases, the challenge is to estimate the optimum number of trajectories for both the creation of the models and the classification of the simulated trajectories. Scenarios with N areas, the definition of the transitions among these areas and M profiles that define the different users behavior are potential situations to apply the contribution proposed in this chapter.

In this work, the classifier is not of our interest. That is to say, we do not evaluate the classifier, but the models. We are aware that the models quality

could vary depending on the classifier, but we do not look at the absolute quality of the models, but their relative quality according to the variation of the number of trajectories used to be created and to train the classifier.

Finally, if we manage a set of user profiles and we observe one single user and we are able to detect the trajectories she performs, we propose to estimate the number of trajectories that are necessary to ensure, with a certain confidence degree, the profile the user belongs. Depending on the user profile the particular user belongs, this result varies. For instance, in the case of study, two of the proposed profiles have a probability of being correctly classified in the correct profile $h_m = 1$, which is an extreme case. In that case, if the user is classified in one of these two profiles and actually belongs to the profile that was estimated, one trajectory is necessary to affirm the profile she belongs.

Chapter 4

Methods for the localization of mobile devices

4.1 Introduction

Tracking people to obtain real trajectories is the next step to create an entire methodology to define models, as seen in Chapter 3. Users localization via their mobile devices and the proposal of a series of methods to track people based on the succession of their estimated positions is the objective in this chapter.

In Chapter 2, a review of the most used location technologies and techniques to estimate the position of objects was made. Since we decided to use mobile devices to detect the positions of the users, whose integrated technologies basically allow the communication with other devices and sensors using RF technologies, we made first a review of the location systems based on RSS, a commonly used parameter in positioning systems.

Recently, the standard Bluetooth 4.0 emerged, also known as Bluetooth Low Energy (BLE), which is currently being integrated into small devices that

work as beacons. This type of BLE beacons periodically emit a signal that normally contains a set of identification parameters. This technology has the advantage of consuming very little battery power, so that its autonomy is considerably lengthened to two years in some cases. In addition, this feature allows to place this type of devices in strategic places for the development of the location system.

BLE-based beacons are also known as proximity beacons, since its main use is the detection of these beacons by mobile devices, which enables any type of service, for instance, the mobile device displays notifications or advertising related to a product or service close to the position of the beacon. However, this use has the limitation that we can only know the proximity of a user to a particular point of interest when the user is near the coverage area of the beacon.

Therefore, our goal in this chapter is to design a series of localization algorithms based on the study of the behavior of signals such as beacons in a test scenario. Proposed algorithms enable the easy deployment of a tracking system.

As seen in Chapter 2, solutions to find the most accurate localization of a target is widely considered in literature. In Section ??, an already studied optimal solution based in least squares is presented. Nevertheless, our objective is to simplify the deployment task, minimizing the necessary calibration to carry out the installation of a tracking system in a real environment. Therefore, we propose a methodology to estimate the user position when indoors, using BLE beacons, and optimizing the number of beacons and the necessary calibration in order to deploy this solution in any type of scenario as quick and simple as possible.

4.2 Design of a localization system based on iBeacons

There are many possible configurations to deploy a localization system. The decision to choose a specific configuration depends on the target we are trying to localize, the supporting infrastructure and where the positioning algorithm is being implemented. In this case, we try to find the position of a user through her mobile device. Most of people usually carry a smartphone, which

is equipped with common RF transceivers such as Wi-Fi and Bluetooth.

As mentioned before, BLE-integrated beacons transmit signals that contain a set of identifiers that enable any BLE detector to identify the transmitter. The localization process we propose starts from a BLE beacon based infrastructure. These are strategically placed and configured to periodically transmit a signal. At the same time, a mobile device is searching BLE signals. The mobile device that is going to be localize runs an application that periodically detects BLE signals. When a detection period finishes, the RSS value from each beacon is measured and all the information is sent to the localization algorithm so that the mobile device position can be estimated. The localization algorithm can be implemented either in the mobile device or in a services server that runs this algorithm.

If the localization system is completely mobile based, all the necessary information to estimate the position must be inside the mobile device and it would estimate its own localization. On the other hand, there are centralized architectures, which compute the localization of all the mobile devices. These send the RSS information from every beacon to the centralized computer, which estimates the position of each device.

At this moment, we do not enter in the detail of the implementation of the localization system, so we just define a BLE beacons-based architecture, which are periodically sending a signal with a constant signal strength. We will describe the behavior of the BLE beacons signals from the point of view of the localization and discuss about the actual position of the beacons in the deployment area.

In Appendix C we have done a survey of the RSS behavior using beacons in different conditions, such as RSS vs distance between beacon and receiver, the smartphone position or the time.

4.2.1 Beacons-based infrastructure

One of the tasks when deploying the localization system is the installation of the beacons in the scenario. Depending on the type of scenario the choice of its placement can vary. For instance, we present in Figures 4.1 and 4.2 two types of configurations. In the first case, areas are adjacent each other. We pretend to limit every area with another. We must understand that in some scenarios, these limits do not actually exist (e.g. walls). That is to say, there is not such a separation between two areas, so it is not easy to physically separate them, what could make the localization process harder than in physically separated areas.

Figure 4.1: Zone configuration with adjacent areas

In the second case, we introduce an extra area that is named *out-of-zone*.

Figure 4.2: Zone configuration with separated areas

Out of Zone is an area in the scenario that combines every zone that is not considered an actual area. Normally, corridors and common areas are defined as *out-of-zones*. For example, corridors in a mall, an office or a museum could not be considered as areas of interest to localize people. Nevertheless, if we want to continuously track a person and she enters in an area which is not of our interest, we can use the area *out-of-zone* to create the complete trajectory.

In the following sections, the proposed algorithms are defined and evaluated. Before, the performance of BLE beacons in a controlled scenario has been deeply tested as presented in Appendix C.1. BLE beacon signals behavior has been studied since several points of view. Some of the most relevant examples that are interesting to design the algorithms are the RSS variation as a function of the distance between transmitter or beacon and receiver or mobile device.

Besides, the spatial variation in a controlled scenario has been also done to evaluate how walls, user position between the beacon and mobile position and distance affects the RSS. The way the user carries the mobile device is also studied. Carrying the mobile in the hand or putting it on a table also affects the RSS value that is detected.

4.3 Localization algorithms

In this section we propose some RSS-based algorithms that, supported by a beacons-based infrastructure, estimate the position of a user through her mobile device. The user is localized in one of the N areas that form a scenario, which can be either adjacent or not (see Figures 4.1 and 4.2). The algorithms we propose aim at exploiting the information users mobile devices periodically detect from the deployed beacons (RSS and beacon-ID) so that their position can be estimated.

As mentioned above, the intention of this work is not to find the optimal solution like other works (e.g. (Tarrío, 2011)), but to find a simple solution that is low cost and effort (we aim at minimizing the previous calibration phase) and fast to deploy.

In this section we propose some of algorithms based on RSS:

1. First, a basic algorithm is proposed, where the estimated position is the area whose beacons are providing the highest RSS value.
2. Second, the values of RSS from beacons of the area where maximum power is being received, $A_n(max)$, are compared with the RSS values from the beacons of other areas. Based on a previous calibration to measure the RSS difference between areas, the estimated position could be $A_n(max)$ or *out-of-zone*.
3. Finally, we propose an algorithm that estimates the probability of being in every area according to the measured RSS values, the transition probabilities between areas and the probability of being in the other zones in the previous stage. For this algorithm we propose an enhancement that modulates the probability of transition between areas depending on the user's movement, which is detected by the mobile device movement sensor. Then, the probability of the user to stay in the current area is higher if the user is not moving.

Besides, we propose to add a spatial filtering after the localization algorithm to provide the localization system with a certain stability. This filter takes as input the estimated position and outputs actual positions changes. It considers a position change when a specific position is estimated a number of times in a row.

4.3.1 Simplest Direct RSS algorithm

This solution estimates as the user position that area whose detected beacons RSS values are the highest. So, let us assume that the scenario where the localization algorithm is to be used contains N areas:

$$A = [A_1, A_2, \dots, A_N] \quad (4.1)$$

and that each area contains M_n beacons. The user mobile device is periodically detecting beacons. At the same time, the algorithm is running and using the most recent RSS value from each beacon to estimate the user position. Let's call $b_{m,n}$ the RSS value from the beacon m in the area n . The RSS value that is used to estimate $A_{n(max)}$, the area whose beacons provide the highest averaged RSS, is calculated as follows:

$$A_{n(max)} = \max(\frac{1}{M_n} \sum_{m=1}^{M_n} b_{m,n}) \forall n \quad (4.2)$$

This is the simplest algorithm we propose in this work. There is no necessity to calibrate the scenario. If all beacons are emitting their signals with the same power, we can start using the algorithm once the entire localization system is deployed.

The two main disadvantages of this algorithm are:

- The solution is always one of the A_n existing areas in the scenario. If the user is not in one of the zones of interest, one of these will be given as estimated positions, what could be considered as a false alarm.
- The instability in the RSS can be reflected in changes in the estimated position, when the user has not actually changed her position.

4.3.2 Thresholds-based algorithm

Thresholds-based algorithm is based on the difference of RSS from the beacons deployed in each area. We start from a scenario that contains N areas and M_n beacons per area. To simplify, let's suppose that every area has the same number of beacon associated, so M is not n dependent. Therefore, there are NM beacons in the scenario.

First, zones are calibrated to estimate the RSS difference between all the beacons in each zone. That is to say, if zone A contains two beacons, A_1 and A_2 , and zone B contains beacons B_1 and B_2 , the difference between the average of beacons in A and the average of beacons in B is measured from both A and B . These values are used, as explained in subsection 4.3.2.2, to compare RSS values when localizing and estimate if the user is actually in one of the specific zones.

4.3.2.1 Out-of-zone

Let's remember that an *out-of-zone* is an additional area that contains that region in the scenario where no specific zone is actually defined. Normally, corridors and common areas can be defined as *out-of-zones*. For instance, corridors in a mall, an office or a museum could not be considered as areas of interest to localize people. When the algorithm is able to provide as estimated position one of the existing areas A_n or *out-of-zone*, then:

- If the localization algorithm estimates that the user position is A_n :
 - If the user is actually in A_n , the localization algorithm has done a right estimation. This is a *success*.
 - If the user is not in A_n (another zone or *out-of-zone*), the localization algorithm has done a wrong estimation. This is a *detection error*.
- Otherwise, if the localization algorithm estimates the user is in *out-of-zone*:
 - If the user is not in any area, then the estimated position is right. In this case, the goal of the localization algorithm is not to give any area as an estimation. This is a *success*.
 - If the user is actually inside a particular area, it is produced what will be called from now on as a *false alarm*, but also considered as an *error*.

Then, if the user is inside a particular area and the localization algorithm estimates the position correctly or she is not in any area and the localization algorithm does not give any area as a solution (*out-of-zone*), the localization is right. However, there are two cases in which the system is not correct: when a detection error or a false alarm occur. Right estimations, detection errors and false alarms are the indicators of the quality of this algorithm.

4.3.2.2 Algorithm description

The average value of the RSS from every beacon in each area is used to define b_n , which is the information used to be compared between areas to estimate the position of the users. The RSS from the beacon m in the area n is $b_{m,n}$, so:

$$b_n = \frac{1}{M} \sum_{m=1}^M b_{m,n} \quad (4.3)$$

The first step is to find that area $A_n(max)$ whose RSS b_n is higher than the others ($b_n(max)$):

$$b_n > b_{n'} \quad (4.4)$$

where $n \neq n'$.

The algorithm will only provide a defined solution if the difference of RSS respect to the rest of areas is over a certain threshold.

The objective of the algorithm is to confirm that the user is in a certain area only if she is clearly in the area. If the user is close to the division between two areas or is out of any area, one of the areas will still provide the highest RSS value $b_n(max)$. However, the difference of RSS between the area that is providing the highest value $b_n(max)$ and the rest of areas is the key to confirm whether the user is actually in $A_n(max)$.

Let us call D to the vector of dimension $n - 1$ that represents the difference between $b_n(max)$ and the rest of RSS of each other area $b_{n'}$:

$$D = [d_1, d_2, \dots, d_{n-1}] \quad (4.5)$$

Each element in $D = [d_1, d_2, \dots, d_{n'}]$ is the actual difference of RSS between $b_n(max)$ and $b_{n'}$, for every area $A_{n'}$, where $n \neq n'$. If the Simplest Direct RSS algorithm is used, then $A_n(max)$ is the solution. In this case, every element in D is compared with a set of already predefined thresholds. Then, the actual position of the user is:

$$S = \begin{cases} A_n(max) & \text{if } d_{n'} > th_{n,n'}, \forall n' \neq n \\ out-of-zone & \text{otherwise} \end{cases} \quad (4.6)$$

where $th_{n,n'}$ is a threshold value between the ordered pair of areas A_n and $A_{n'}$. For instance, in a scenario where $N = 2$, the threshold of the area A_1 respect area A_2 , $th_{1,2}$ is the minimum required difference of RSS to ensure the user is inside A_1 if $b_1 > b_2 + th_{1,2}$. The threshold is an asymmetric value. That is to say, th_{21} is not necessary equal to th_{12} .

Other variation that relax the requirement that confirms that the user is in $A_n(max)$ is

$$S = \begin{cases} A_n(max) & \text{if } d_{n'} > th_{n,n'}, (\forall n' \neq n) - i \\ out-of-zone & \text{otherwise} \end{cases} \quad (4.7)$$

where $i = 0, \dots, N - 1$. For instance, if $i = 1$, all distances in vector D must be higher than every corresponding threshold except in one case. The higher is i the less restrictive is the algorithm. By default, $i = 0$.

4.3.2.3 Thresholds calculation

Thresholds are the values used by the Thresholds-based algorithm to determine if the user is in one of the existing areas A_n or *out-of-zone*. Let's recall that *out-of-zone* is that area we use to define the combination of all the spaces in the deployment scenario that do not belong to any defined zone.

As described above in Section 4.3.2.2, a threshold value th_{AB} determines the minimum difference $b_A - b_B$ that is required to ensure that the user is in A in the scenario $S = A, B$.

If there are n areas, for each area n there are $n - 1$ thresholds that relate the minimum difference between b_n and $b_{n'}$, where $n \neq n'$, that must be fulfilled to estimate if the user is inside A_n .

The calculation of the threshold th_{AB} is carried out taking a set of measurements of RSS from the beacons in A and B. The observer is in A because this threshold value is used to detect if the user is actually in A. To calculate th_{BA} , a set of measurements from the beacons in A and B must be acquired but the observer must be placed in B. Depending on the size of the area, we have decided the observer to be placed in one or more points that are significant in the area. If the observer takes the measurements at the end of the area, these values will not represent that the user is in the corresponding area.

R_A and R_B are an ordered series of measurements taken from A and B respectively:

$$R_A = r_{A_1}, r_{A_2}, \dots, r_{A_Q} \quad (4.8)$$

$$R_B = r_{B_1}, r_{B_2}, \dots, r_{B_Q}, \quad (4.9)$$

and the measurements are taken at the same time:

$$time(r_{A1}) = time(r_{B1}). \quad (4.10)$$

So we build a vector that contains the difference of the measurements $R_{AB} = R_A - R_B$. If R_{AB} is represented in a histogram, we obtain the minimum and maximum difference of RSS from beacons in A and beacons in B, measured from A. The histogram represents in the x axis every value between $R_{AB_{min}} - 1, h_0$ and $R_{AB_{max}}, h_{max}$; and in the y axis, the frequency. If Q measurements are taken from every beacon in A and B, what means that R_{AB} length is Q , then 100% out of every value in R_{AB} is higher than h_0 and lower than $h_{max} + 1$.

If we want to relax the conditions and find, for instance, the element in R_{AB} where 90% out of all the measurements in R_{AB} are higher, we have to choose that element h_p between h_0 and h_{max} where, if q'_p are the number of elements over h_p , then:

- $(q'_p)/Q > 90\%$ and

- $(q'_{p-1})/Q \leq 90\%$.

This percentage is named *confidence degree* and is selected when configuring the algorithm to be more or less strict and determine the threshold for every pair of areas. The higher the confidence degree is, the lower is the threshold and stricter the algorithm.

4.3.3 Bayes-based algorithm

Thresholds-based algorithm tried to solve the limitation of the Simplest Direct RSS algorithm, whose solutions were only one of the existing areas. The solution proposed with Thresholds-based algorithm adds the *out-of-zone* areas using the thresholds concept described above through the comparison of RSS values.

The purpose of this algorithm is not to provide a specific area as a solution, but the percentage to be in each area at every step. To do that, we still consider the highest probability to be in a specific area provided RSS values, using the Simplest Direct RSS algorithm, but in every step we are going to consider the previous state to $P(A_n(t), P(A_{n'}(t-1)))$, which is calculated for every area in every step, and the probability of moving between two areas. Then:

$$P(A_n(t)) \propto \sum_{\forall n} P(A_n(t)/A_{n'}(t-1), O(t)) P(A_{n'}(t-1)) \quad (4.11)$$

where:

- $P(A_n(t))$ is the probability to be in the state A_n at time t ,
- $P(A_n(t)/A_{n'}(t-1), O = (t))$ is the probability to be in A_n at t as a function of both the previous state, $A_{n'}(t-1)$ and the RSS observation, $O = (t)$.
- $P(A_{n'}(t-1))$ the probability to be in the state $A_{n'}$ at $t-1$, which is estimated in the previous step for every state and normalized.

Since $P(A_{n'}(t-1))$ and the observation $O(t)$ are independent, $P(A_n(t))$ can be determined as follows

$$P(A_n(t)) \propto \sum_{n'=1}^N [P(A_n(t)/A_{n'}(t-1))P(A_{n'}(t-1))]P(A_n(t)/O(t)). \quad (4.12)$$

The Equation 4.12 is disaggregated below. First, the probability to be in A_n at instant t given that the position of the user at the instant $t-1$ is $A_{n'}$ is represented by the term

$$P(A_n(t)/A_{n'}(t-1)) = \begin{cases} \neq 0 & \text{if the transition between } n' \text{ and } n \text{ is possible} \\ 0 & \text{otherwise} \end{cases} \quad (4.13)$$

where $n = 1, \dots, N$ and $n \neq n'$.

Second, $P(A_{n'}(t-1))$, which is the probability that the user position is in $A_{n'}$ in the previous instant and the normalized output of the previous algorithm iteration. In the first iteration these values are 1 because the algorithm does not have any previous estimation and $O(t)$ is the unique element to be taken into account.

Therefore,

$$\sum_{n'=1}^N [P(A_n(t)/A_{n'}(t-1))P(A_{n'}(t-1))] \quad (4.14)$$

represents the summation of the probability to be in $A_{n'}$ in the previous instant multiplied by the probability to move from $A_{n'}(t-1)$ to $A_n(t)$.

Finally,

$$P(A_n(t)/O(t)) \quad (4.15)$$

indicates the probability to be in A_n as a function of the observation $O(t)$, which is a vector of pairs formed by beacon identifiers and their RSS values. Obviously:

$$\sum_{n=1}^N P(A_n(t)/O(t)) = 1. \quad (4.16)$$

This is, given an observation $O(t)$, the user must be in one of the n existing areas with a probability $P(A_n(t)/O(t))$. A previous calibration is necessary in order to obtain this term.

Let's suppose a number p of positions where q measurements were taken in each one. In general, there are pq measurements in A_n . Using the same procedure proposed in 4.3.2.3, the RSS of an area A_n refers to the mean of every valid RSS from all the beacons installed in that area. Thus, $O(t)$ represents the combination of the areas where $b_n = b_n(max)$. Actually:

$$P(A_n(t)/O_1(t), O_2(t), \dots, O_{N'}(t)) = \frac{1}{N'} \sum_{l=1}^{N'} [P(A_n(t)/O_l(t))] \quad (4.17)$$

where N' is the number of areas in which $b_n = b_n(max)$.

To summarize, there are three sources to obtain $P(A_n(t))$: the transitions table that indicates the possibility to move from one area to another, $0, 1$; the previous probability to be in each existing area $(0, 1)$; and the table that relates the reference area and the probability to be in each area, $(0, 1)$. Once the probability for each area $P(A_n(t))$ is calculated for $n = 1, \dots, N$ being N the number of areas, the results are normalized to obtain:

$$\sum_{n=1}^N P(A_n(t)) = 1 \quad (4.18)$$

Motion-aware component When a user is in a particular area and stands, the only factor that could alter the estimated position is the fluctuation of the RSS values from the detected beacons. This could cause a change in the estimated position by the algorithm and cause positioning system errors, when the user has not actually changed her position. Therefore a module that infers if the user is moving is added to the previous improvement to modify the transition probability depending on the actual status of the user: moving

or still.

The transitions probability among the different zones that make up the deployment scenario is obtained from the next formula:

$$P(A_n(t)/A_{n'}(t-1)) \begin{cases} \neq 0 & \text{if } n = n' \text{ or the transition between both areas} \\ & \text{is physically possible} \\ = 0 & \text{otherwise} \end{cases} \quad (4.19)$$

When the Bayes-based localization algorithm is configured, we have to take into account that:

$$\sum_{\forall n} P(A_n(t)/A_{n'}(t-1)) = 1 \quad (4.20)$$

The transitions probabilities can be configured to be the same for all users or customized for every user or every type of user, since she can be detected.

For instance, in the implementation of this algorithm, we have considered to make $P(A_n(t)/A_{n'}(t-1)) = 1/L$, where L is the number of areas where $P(A_n(t)/A_{n'}(t-1)) \neq 0$, including $n = n'$. In other words, those areas where the user can go if she is in $A_{n'}$, including the current area.

When the motion-aware detection component infers the user is moving, these values do not vary. Nevertheless, if the movement component detects that the user is still, then:

$$P(A_n(t)/A_{n'}(t-1)) = \begin{cases} 0,8 & \text{if } n = n' \\ \frac{0,2}{L} & \text{otherwise} \end{cases} \quad (4.21)$$

where L is again the number of areas that originally $P(A_n(t)/A_{n'}(t-1)) = 1$. We have considered to set a higher weight, 0.8, if the user is still, to the probability of not changing her position, because if the movement detector does not estimate that the user is moving, she will more probably stay in the

same zone.

4.3.4 Spatial filtering improvement

Due to the instability of the RSS and regardless of the algorithm, sometimes it is inevitable that the estimated position has variations that would be considered as errors. For instance, if a user is still in A_n and, due to the variations of RSS that provide isolated outliers of RSS values, the localization algorithm estimates that the user position has changed. Filtering the position estimation provides a stability to the output of the entire localization system.

Therefore, we propose to add, after the localization algorithm, a module that implements the proposed spatial filtering with two configuration parameters called *detection-in* and *detection-off*. These parameters respectively indicate the number of consecutive times the localization algorithm must estimate the user position in a determinate area or "out-of-zone", so that the localization system provides either the corresponding area or "out-of-zone".

For example, if a user is "out-of-zone" or in an area called A and the parameter *detections-in* = 2, the localization system indicates that the user is in B if the algorithm estimates twice consecutively that the user position is B . On the other hand, if the user is in A and the parameter *detections-off* = 3, the localization system will indicate that the user is "out-of-zone", if the case of Thresholds-based algorithm is used, if it indicates that the user is "out-of-zone" three times in a row.

4.4 Evaluation of the localization system

The evaluation of the proposed algorithms will be carried out in the scenario represented in 4.3, where eight areas are defined according to the deployment scenario. In this scenario, some areas are separated by shelves or pieces of furniture while others do not have any separation. This means that a localization system may have limitations because there are not well-defined areas. These have been formed by regions in the space that the system developer has decided to create as areas, according to the configuration of the

elements in the space.

Two beacons per area have been installed and a prior calibration has been carried out so that the values of the thresholds used in Thresholds-based algorithm and the probability values $p(A_n/O(t))$, used in Bayes-based algorithm, are calculated.

First, the three algorithms discussed above are evaluated, along with the modification of the Bayes-based algorithm, in which the movement of the user is taken into account and detected using the inertial sensors that are integrated in mobile devices. So a space-time evaluation is carried out.

Second, an evaluation of the spatial filtering over one of the algorithms evaluated in Section 4.4.2 is done. In this case, we have decided to implement the spatial filtering over the Thresholds-based algorithm.

4.4.1 Algorithms configuration

The Simplest direct RSS algorithm does not require a previous calibration like Thresholds-based and Bayes-based algorithms do. This offline phase is necessary, in the first case, to estimate the thresholds that ensure that user is in A_n or *out-of-zone* (see Equation 4.7). The estimation of the thresholds is explained in Section 4.3.2.3.

On the other hand and according to Equation 4.12, Bayes-based algorithm requires the previous acquisition of data to calculate $p(A_n/O(t))$. As discussed in Section 4.3.3, given an observation, the summation of the probability to be in each area is 1 (see Equation 4.16). It is possible to receive the same RSS from the beacon in two areas: $b_n = b_{n'}$, where $n \neq n'$, due to its quantified measurement. In this case, we solve this case in different manners for each algorithm:

- *Simplest Direct RSS algorithm* - if beacons from A_n and $A_{n'}$ are providing the highest RSS value, the solution of the algorithm is randomly chosen.
- *Threshold-RSS algorithm* - in this case, *out-of-zone* is the solution since none of the pair of beacons is providing the highest RSS.

- *Bayes-based algorithm* - $P(A_n/O(t))$ is estimated for every area in every step. Let's remember that $O(t)$ indicates those area whose beacons are providing the highest RSS value at t . Then, as most of cases happen, $P(A_n/O(t))$ is directly obtained. If $O(t)$ indicates two areas, we weight $P(A_n/O(t)) = P(A_n/O(t))/N_O$, where N_O is the number of areas whose beacons are providing the highest RSS values, according to the observation $O(t)$.

According to the scenario represented in Figure 4.3, the trajectory followed in the dynamic experiment, in which the user is moving, consists of the succession of the areas D, F, E and C, which some times is traveled in loops. As discussed in Section 4.3.3, $P(A_n)$ is estimated as a function of three elements. First, the current RSS observation. Second, the probability to be in every other area in the previous stage. And finally, the probability to move from every other area, including A_n , to A_n . Since the former parameter is obtained from a previous calibration phase and the second one is the result of the previous state, the algorithm designer must decide:

$$P(A_n(t)/A_{n'}(t-1))\forall n, n'. \quad (4.22)$$

B	D	F	H
A	C	E	G

Figure 4.3: Testing scenario

$P(A_n(t)/A_{n'}(t-1))$ is a parameter that actually defines the user behavior. We can relate it with the Chapter 3, in which the user behavior was modeled. Nevertheless, for the purpose of this section, in which the algorithm is evaluated, we assume that we have some knowledge about trajectories and users behavior based on their movement and then, we have chosen a set of values that favors the trajectory that is carried out for the algorithm evaluation.

So that, under our criteria we have chosen the next values:

- $P(A_{n'}(t)/A_{n'}(t-1)) = 0,5$, which is the probability of staying in the same zone
- $P(A_n(t)/A_{n'}(t-1)) = 0,25$, which is the probability of moving to the next zone in the configured trajectory
- $P(A_m(t)/A_{n'}(t-1)) = 0,25$, which is the probability of returning to the previous zone in the configured trajectory;

where n and m are respectively the next and the previous states of n' . For instance, if the user is in D at instant t , and:

- $P(D(t+1)/D(t)) = 0,5$ (stay in D)
- $P(F(t+1)/D(t)) = 0,25$ (continue the trajectory)
- $P(C(t+1)/D(t)) = 0,25$ (back to the previous position)

and the Bayes-based algorithm wrongly detects a forward movement to F but the user is going to stay at D for a longer interval, the probability to go back to the previous position is

$$P(D(t+1)/F(t)) = 0,25 \quad (4.23)$$

what offers to the algorithm the possibility to correct this error.

4.4.2 Algorithms evaluation

The three algorithms that we propose in this work, along with the modification of the algorithm based on Bayes, which takes into account the motion sensor, have been evaluated in areas B and D from the scenario presented Figure 4.3, under static conditions, besides a route that follows the order of areas: B-D-F-E-C.

This experiment has been carried out by a user that holds the mobile device in their hands. in some selected points in both areas. Once every algorithm is calibrated and the beacons-based infrastructure is configured as discussed above, we have used a mobile application that first is periodically gathering RSS data and, finally, when the application timeout triggers, algorithms do

an off-line processing to estimate the position at each moment.

Along this work, we have done a lot of experiments and we select some of them in this and the following sections as a representation of the evaluation of the algorithms.

For their evaluation under static conditions (in B and D), where the user stands for 5 minutes in each area, the percentage of time the user is correctly localized is calculated for each algorithm (see Table 4.1).

<i>Area name</i>	<i>Simplest Direct</i>	<i>Thresholds</i>	<i>Bayes</i>	<i>Bayes + Motion detection</i>
<i>B</i>	88	88	88	88
<i>D</i>	76	76	76	76

Table 4.1: % of time the user is detected in the correct area

The results are the same regardless the algorithm. Other experiments have provided similar results. The result given by the Table 4.1 indicates the percentage of the time that the user is correctly localized, respect to the total time of the experiment. The experiment is done entirely in the same area, because we intended to evaluate every algorithm in static conditions.

The spatial filtering is not used in this experiment, so that every change of position in each algorithm means a change in the estimated position by the localization system. And the Motion module added to the Bayes-based algorithm does not provide a stability to the still condition of the observer, that increases the probability value to stay in the current area.

As a conclusions, in static conditions, if the user is inside the area, there are oscillations in the RSS values that the algorithms are not able to correct. Below, the spatial filtering will be evaluated to check if this is able to reduce the RSS fluctuations effects.

The evaluation of the algorithms in dynamic conditions, through a path formed by five areas, is done evaluating two parameters. Motion module is not taken into account because the entire experiment is done while the user is moving.

First, as above, the percentage of time the user is correctly detected in each area. In this case, the tester indicates the time she enters every area so that a master file indicates the real instant the user entered and went out from each area, so it is possible to compare the estimated entry and exit time values with the real ones. Results are shown in Table 4.2.

<i>Area name</i>	<i>Simplest Direct</i>	<i>Thresholds</i>	<i>Bayes</i>
<i>B</i>	8	31	8
<i>D</i>	74	74	87
<i>F</i>	76	76	76
<i>E</i>	93	93	93
<i>C</i>	74	74	74

Table 4.2: % of time the user is detected in the correct area

Algorithms present a similar performance, except in two cases: Thresholds-based algorithm in *B* and Bayes-based algorithm without the motion module in *D*. According to the geospatial configuration of the scenario, zones *B* and *D* are not separated and, when the user holding the smartphone with her hand is in *B*, most of times is detected in *D*. It is not possible to be in a very central position to be correctly detected in *B*.

Nevertheless, the evaluation of the performance of the algorithms must go beyond the time the user is correctly placed inside every area. When the user enters, she is not instantly localized inside the new area. The spaces are not completely separated so beacons from the previous area keep being detected. The values presented in Table 4.2 are calculated as follows:

$$t_{\%}(A) = \frac{t_e(A)}{t_r(A)} \quad (4.24)$$

where $t_e(A)$ is the time the user is estimated to be in *A* while user is really in *A*, and $t_r(A)$ the real time the user was in *A* (see Figure 4.4).

Figure 4.4: Relation between the intervals of the estimated position and the real position

Chart in Figure 4.4 demonstrates how the loss of accuracy in the detection of the user as seen in Table 4.2 can be due to the delay in the detection of the user in the actual area. Therefore, in Table 4.5 we present a timing diagram that compares the actual change in the position of users with the detection of zone change from every algorithm.

Figure 4.5: Comparison of instants of detection of change of zone

Chart in Figure 4.5 represents some features related to the proposed algorithms.

If both Simplest Direct RSS and thresholds-based algorithm are compared, *out-of-zones* are detected in the second case in zones where the Simplest Direct RSS algorithm fails in the area detection. In other words, Thresholds-based

algorithm solves certain errors by setting as output position *out-of-zone*.

On the other hand, if we compare the Bayes-based algorithm (without the motion sensor improvement) and the Simplest Direct RSS and Thresholds-based algorithms, we can observe in the third stage, when the user is actually in F, that there are two types of mistakes when the RSS information is the same. Simplest Direct RSS and Thresholds-based algorithms detect a change from D to E, while the Transitions algorithm detects a change from D to C. Transitions between D and E are impossible if the user does not pass through C or F before (see Figure 4.3). We are considering in the experiment that the movement of the user is slow enough to give the possibility to every algorithm to detect the user in every area she passes through.

Finally, if the motion sensor is activated, the Bayes-based algorithm corrects this error because $p(A_n(t)/A_n(t-1))$ value increases when the movement-aware component detects the user stops.

Table 4.3 presents the delay of an experiment of one instance of the previously mentioned path. These values are not quite representative but we observe that the delay value has resulted the same regardless the algorithm, so that one value per area is presented. The delay in area B is not taken into account due to its bad performance and the lag until the localization system starts to correctly detect the user position.

<i>area</i>	<i>Delay (s)</i>
<i>D</i>	2,1
<i>F</i>	2,5
<i>E</i>	0,5
<i>C</i>	1,8

Table 4.3: Sample of the delay introduced by the algorithms in each area in the one loop experiment

A trajectory formed by the repetition of ten visits to the areas *D*, *F*, *E* and *C* is carried out to roughly estimate the mean value and the standard deviation of the errors introduced by the proposed algorithms (see Table 4.4). An error is considered when a wrong estimation happens. Therefore, the mean value

of the errors are the mean number of errors per loop. The improvement of the motion detector to the Bayes-based algorithm provides the same result if compared with the motion detector disabled. The reason is that the user is never still so that the transitions probabilities do not change, which is the only difference if the motion detector is enabled.

<i>Area name</i>	<i>Simplest Direct</i>	<i>Thresholds-based</i>	<i>Bayes-based</i>
<i>Mean</i>	2,4	1,7	0,1
<i>Standard deviation</i>	0,71	0,45	0,1

Table 4.4: Mean error (number of errors per loop) and standard deviation introduced by each algorithm

The improvements that have been incorporated in Thresholds-based algorithm and Bayes-based algorithm offer a better result. The error value is calculated based on the relation of the number of bad detections per loop.

The improvement of the Thresholds-based algorithm in relation to Simplest Direct RSS algorithm is due to the *out-of-zone* estimations in the former one while Simplest Direct RSS algorithm estimates a wrong area. Figure 4.6 represents the difference of detection between Simplest Direct RSS and Thresholds-based algorithms. Grey parts of the bars are *out-of-zone* detections. Master bar is the actual position of the user and Simplest Direct RSS and Thresholds-based algorithms introduce a delay respect to the actual position. We do not consider a wrong detection this delay, but the wrong detection of the actual.

Figure 4.6: Comparison of the estimations in one loop of the three proposed algorithms

On the other hand, in the experiment proposed in this section, the configuration of the Bayes transitions was set up according the parameters proposed in Section 4.4.1. These parameters are:

- $P(A_n(t)/A_n(t-1)) = 0, 5$
- $P(A_n(t)/A_{n'}(t-1)) = 0, 25$
- $P(A_n(t)/A_m(t-1)) = 0, 25$

where $A_{n'}$ and A_m were respectively the next and the previous states of A_n in the defined path.

We discussed in Section 4.4.1 that these values did not model any type of user but the actual trajectory carried out in the experiment. Therefore, we are going to present next the same experiment with different values of the three previously mentioned parameters. Ten set of parameters are used, where

$$P(A_n(t)/A_n(t-1)) = [0, 1; \dots; 0, 9] \quad (4.25)$$

in steps of 0, 1, and

$$P(A_n(t)/A_{n'}(t-1)) = P(A_n(t)/A_m(t-1)) = \frac{1 - P(A_n(t)/A_n(t-1))}{2}. \quad (4.26)$$

The results of this experiment for ten loops to the trajectory formed by the sucession of positions $D - F - E - C$ presents also the mean and the standard deviation of the errors per loop (see Table 4.5).

$P(A_{n'}(t)/A_{n'}(t-1))$	Mean	Standard Deviation
0,1	0,4	0,51
0,2	0,3	0,48
0,3	0,3	0,48
0,4	0,1	0,31
0,5	0,1	0,31
0,6	0,1	0,31
0,7	0,1	0,31
0,8	0	0
0,9	0	0

Table 4.5: Mean error (number of errors per loop) and variance introduced by Bayes-based algorithm vs. parameters value

Errors happen when there are swaps in the position estimation. So that, the stability of an algorithm produces a lower number of errors. As $P(A_{n'}(t)/A_{n'}(t-1))$ increases, the probability to move forward or backward decreases, what makes the swaps between zones to be less probable than staying in the area. Changes of positions then happen when the RSS observation is inferred as an actual movement of the user to the new position.

We can conclude after this experiment that errors are minimized as $P(A_{n'}(t)/A_{n'}(t-1))$ increases. If this value is higher, the algorithm is more restrictive to positions changes, so once the user is estimated in a specific position, the term $P(A_n/O(t))$ is not so relevant and RSS outliers that imply estimation errors are avoided. Nevertheless, a user position change is detected when $P(A_n/O(t))$ is considerably high, what implies a higher delay.

On the other hand, if $P(A_{n'}(t)/A_{n'}(t-1)) = 1$, the estimated position would never change, what would be an error if the user actually moves to a different area.

4.4.3 Evaluation of the spatial filter component

Once the algorithms have been evaluated in static and dynamic conditions, we intend to incorporate the spatial filtering improvement already introduced in section 4.3.4. In Figure 4.6, there are wrong position estimations that are quickly corrected. The stability component introduced in section 4.3.4 as a spatial filtering is evaluated to prove if this actually correct these errors.

We will take into account the Thresholds-based Algorithm to evaluate the *detections-in* and *detections-off*. The rest of algorithms are not affected by the *detections-off* values, because the *out-of-zone* concept is not taken into account.

Figure 4.7 presents the comparison of the detection along the time of the user position in the trajectory type case of use described above using the Thresholds-based algorithm and a subsequent layer formed by the spatial filtering.

Three configurations are used to compare with the actual user trajectory. First, "Thesholds-based algorithm 1-1" represents the detection using the algorithm without the spatial filtering, also presented in Figure 4.5. Increasing the number of required *detections-in* and *detections-off* at the same time causes the elimination of jumps between areas, what makes it a more stable system. *Out-of-zones* are also removed due to the increase of the *detections-off* value. In a scenario where the distance between areas is bigger, *out-of-zones* should not be removed so easily, because our proposal would be not to localize the user anywhere.

On the other hand, the delay introduced by the system is understandable due to the restriction of the spatial filtering to detect n consecutive times the same area (or *out-of-zone*) to ensure the user position.

Figure 4.7: Comparison of times of areas detection for Thresholds-based algorithm and a spatial filtering

From Figure 4.7, we can also observe how including *detections-on* and *detections-off* increases the time detection when detecting a new position. Below, Figure 4.8 represents how this spatial filtering affects the estimation using Bayes-based algorithm. Since this algorithm used to work fine in dynamic conditions, this filtering only adds a small delay (1 second) to the position estimation.

Since in the 10 loops test the mean error introduced by the Bayes-based algorithm is quite low, we are going to introduce the spatial filter to the Threshold-RSS algorithm to compare how the error is improved (see Table 4.6).

Area name	Threshold 1-1	Threshold 2-2
Mean	1,7	0,1
Standard deviation	0,45	0,1

Table 4.6: Comparison of errors (number of errors) as a function of *detections-on* and *detections-off* parameters

In Thresholds-based algorithm the addition of the spatial filtering is more useful than, for instance, using the Bayes-based algorithm, because this algorithm does not introduces errors in the experiment we performed:

Figure 4.8: Comparison of times of areas detection for Bayes-based algorithm and a spatial filtering

The spatial filtering using 2 *detections-in* and 2 *detections-off* is enough to reduce the errors in the detection but a delay of one second is introduced. A higher value in the filtering configuration is not useful in this case.

4.5 Conclusions

In this Chapter we have proposed the design and development of a beacon-based localization system and some algorithms to estimate the user position using a mobile device. This device, detects Bluetooth Low Energy beacon signals and run one of the proposed algorithms to estimate its own position. Though the proposed approach tries to be a simple instead an optimal solution, we need to calibrate some of the algorithms.

Before designing, calibrating and developing the algorithms, we have done some experiments to understand BLE beacon signals by studying their RSS behavior in different conditions (see Appendix C). Once we understood how beacon RSS behave in a controlled scenario, we have deployed the installation of some beacons to cover all the zones in the scenario.

First, we have designed the Simplest Direct RSS algorithm, which estimates the user position from those area whose beacons are providing the highest RSS value. Second, we have compared RSS from beacons in different areas to infer if the user is actually in the area that whose beacons are providing the highest RSS. This solution adds the zone *out-of-zone*, which is that area that

is not defined as a zone of interest. Finally, a Bayes-based algorithm has been proposed, which takes into account three parameters:

- The probability to be in each area as a function of the RSS observation
- The transitions probability, which is the probability of moving between two areas
- The probability of being in every area in the previous step

Since we have observed that RSS fluctuations make the algorithms to make isolated wrong position estimations, we have designed a spatial filtering, which is implemented after the localization algorithms and takes as input the estimated position and outputs that the user position changes after n identical consecutive estimations. For instance, if user is in A and the requirement to infer the user position changes are three consecutive estimations, a detection in B three consecutive times is necessary to detect the change of position by the spatial filtering.

Once the algorithms have been developed, we have evaluated them in static and dynamic conditions. In static conditions, we have obtained the same results for the three algorithms. Other experiments have provided similar results. Nevertheless, in dynamic conditions, we have detected how Bayes-based algorithm reduces the errors in the experiment we have done. These errors also are affected by the transitions probabilities configuration. On the other hand, Thresholds-based algorithm also reduces the errors compared to the Simplest Direct RSS algorithm.

Furthermore, we have proved that, in our experiment, adding a spatial filtering to the transitions algorithm eliminates the errors. If parameters *detections-in* and *detections-off* (let's remember these parameters indicate the number of consecutive times a position or an *out-of-zone*, respectively, must be estimated by the localization algorithm, so that the entire localization system infers a position change) are set to 2, some of the errors are eliminated but the detection that the user position has changed is delayed. If parameters are set to 3, all the errors are eliminated in this example but the detection delay increases (see Figure 4.7).

The solution we propose is recommended as a simple deployment to estimate the position of a user indoors. The Simplest Direct RSS algorithm does not require calibration but makes more errors when estimating the user position. This algorithm contemplates that all the area must be divided into zones. If the area contains some undefined zones, Thresholds-based algorithm includes the *out-of-zone* area. This algorithm requires an off-line calibration phase but makes less errors than Simplest Direct RSS algorithm. Finally, Bayes-based algorithm takes into account the previous states and the transitions probabilities. Our experiments obtained better results using Bayes-based algorithm. According to our experiments, if the user requires to minimize the errors in the position estimations, the integration of a spatial filtering after any of the localization algorithms decreases the errors though adds a small delay.

Finally, we must recall that our proposed solutions are not optimal solution to find a very accurate position, but a simple, quick to deploy and easy to implement beacon-based solution to create a localization algorithm when indoors.

Chapter 5

Personalization of applications based on the user position and behavior

5.1 Introduction

The goal of this chapter is to represent, using some examples and their subsequent analysis, the concepts we have discussed in Chapters 3 and 4. As explained in Chapter 3, we considered that, from a sequence of positions, we are able to build trajectories, which can be used to estimate the user behavior. We defined users behavior as a set of user profiles, which are modeled using Markov and non-Markov based models that define, in its most basic version, the probability to move between every two areas. A more elaborated version of our proposed models define the probability to move between two areas as a function of the previous states.

Therefore, we define a position and behavior-aware application as one whose performance and response to requests may change according to the user position (computed using location algorithms in Chapter 4) and the profile

where the user is classified (Chapter 3).

First, the application and its components are described. The main one is a mobile localization system that estimates the position from the smartphone of the user using RSS from BLE beacons, applying the algorithms described in Chapter 4. Besides, other services are implemented as additional modules. The core component and the additional modules share their output information using two interfaces. One of them is synchronous, so services answer requests providing their output information. The other one is asynchronous and is based on subscriptions and notifications. One module can subscribe to outputs from another, therefore whenever there is a new data in the output component, this is shared to its subscribers.

Furthermore, three implementations or instances of the application are presented. First, the localization system has been installed in a retail scenario (see Figure 5.1), which can be used to deploy multiple business. Second, the localization application has been installed in our laboratory (see Figure 4.3), where a contents service is added to present the localization-aware application. Finally, we deployed our localization system in a supermarket, running at business time, in order to demonstrate how an implementation in a real operating setting is feasible.

5.2 A user position and behavior-aware application

Applications or services have moved from their static condition, whose response to requests did not vary regardless the context where it was required, to a dynamic one, in which their response is a function of multiple parameters related to the user context, for instance, the user position, the time, the weather, the user likes or her activities history.

Beyond the dynamic behavior of context-aware applications, the personalization of their performance as a function of the user position and behavior enables to offer different responses depending on both the user position and the user behavior, which in our case, is described by a profile defined by the user behavior over a scenario (Chapter 3). If users are tracked, the probability of moving between two areas can be obtained by the trajectories formed by the succession of user positions estimated by

the localization system and, whenever the user visits the scenario, these values can be updated. It means that the characterization of a user by our behavior models proposal is not static and new iterations in the classification algorithms can be carried out if new information is available.

5.2.1 A user position-aware application

In this section, we will describe our proposal in detail. First, if we take into account only the position of the user, we can define a user position-aware application as that whose response varies depending on the actual position of the user. As an example, whenever the user changes her position, an application or a service that provides, for instance, a piece of information or any content to the user, varies this content or this information depending on the area where the user is localized. For instance, an audio-guide in a museum that detects the user position and adapts its explanations to the actual estimated position.

The position of the user is given by a localization system, which implements a localization algorithm. A user position-aware application is either subscribed to changes in the user position or demands this information when necessary. In this chapter we propose, design and build a localization application based on contents that are displayed on the screen where the user is localized. The logic of the changes of the display to present the content should be subscribed to the changes in the estimation of the positions

There is a correlation between the accuracy of the localization system and the type of service that can be provided by the user position-aware application. A very accurate application that needs a position information with an error below, for instance, 1 meter, could not use our localization system, proposed in Chapter 4, whose output is a symbolic localization giving the estimation area where the user is supposed to be. Nevertheless, other services like that described above about contents displayed in different areas would work correctly with our proposed localization system.

So far, we have not still talked about personalization. Only the user localization has been taken into account for services deployment. Using a localization system, the personalization is characterized by the user

identification. This is, the personalization can go beyond the position and give a different response to the user not only as a function of her position but any information related to herself.

In Section 5.4 a user position-aware application is described, varying the device place of presentation of a video depending on the current position of the user.

5.2.2 Personalization of an application based on the user behavior

In Chapter 3 we have tried to infer the profile a user belongs to from a set of trajectories. To do that, we must detect the user through an identifier, such as her mobile device, which is also used to track the user trajectories.

This is an improvement in the personalization of applications, which not only depend on the user position and identification, but on the user behavior, given by the profile she is classified in.

On top of the localization system, which provides the current position of the user, it is possible to build a consuming service that builds the trajectories and, then, these can be used to study the user behavior.

In Section 5.4 we will describe three examples of user behavior-aware applications that could consume the information of the user profile type to offer a different content or information to the user depending on her profile. For instance, in the bank branch example, a user that is only detected outside getting money from the ATM is classified in a profile of people different to the business man that frequently visits the branch. The information of a mobile application when the user is detected around the branch could be different depending on the type of user. The user can be detected, identified and her profile is obtained if she has previously been tracked and classified. Besides, every new visit to the branch, the classification of the user is improved.

To simplify, users are commonly classified in one of the existing profiles. Nevertheless, one user can be partially classified in different profiles. This is, a person who goes to the bank branch to retire money from the ATM can

also visit this place to ask for credits. Therefore, the user is classified in two different profiles with different weights.

As mentioned before, the classification of users in different profiles is a dynamic task. When new users trajectories are provided, new iterations in the classification algorithms are performed and the allocation of users to the exiting profiles could vary.

5.3 Application design

The proposed application is a final service that consumes information from users, both their position and profile, to offer different responses depending on the user-related information.

Any user position and user behavior-aware application must be designed to be working over a framework formed by a localization component, which implements the algorithms presented in Chapter 4, and a set of modules that implement multiple services, some of them described below. Among these services, we propose to build a classification service that using the trajectories built from the user positions, offers an API to be used by user-behavior aware applications. This is based on a IoT platform developed in (Corredor et al., 2014).

The actual implementation of the localization component is mobile oriented but supported by an infrastructure formed by the deployed beacons, a database with the space-related information and a localization service that gathers every new position from all the users and publishes them to services that are subscribed to this type of information.

Mobile devices download, from the server where the localization service and the database are installed, the information related to the space where the localization system is going to be working. This information configures the algorithms, which run in the mobile device, and contains useful data from the beacons and the configuration of the space. Periodically, the mobile application, which implements the localization system, receives new values from the beacons RSS and computes its own position through one of the algorithms. If the estimated position changes, this is published to the

localization service, which is running in a support server, and this publishes this information to those services that are subscribed to this type of events.

Figure ?? represents a block diagram to understand the configuration of this solution from the server side point of view. The position estimation is done in the user mobile device and when this is updated in the *Localization component*, which is integrated in a cloud server, proposed services can consume this data to update the information they provide.

5.3.1 Sessions service

This service is one of the proposed services that are subscribed to change of position events, published by the localization service. This component builds entire trajectories that represent the succession of places the user visits in a scenario along with the entry and exit times from every position.

In this section we are not going to describe the technical details of the implementation. Nevertheless, for a better explanation, we advance that the user position is estimated in the mobile device and an applications server consumes the position of every user and creates trajectories for all of them. We could also propose a mobile application that creates its own trajectories and provides them to other services, nevertheless we decided to decouple the *localization component* and the consuming services.

Therefore, when a user is not detected any more by the *localization component* or decides to turn off the application, the session is closed and a trajectory is built. This trajectory can be offered either to an external context-aware application or to other services. In this case, we recall the Chapter 3, which proposed a tool that classifies a user according to her trajectory.

5.3.2 *Classification service*

Using the trajectories provided by the *sessions service*, we propose a classification service that, using a set of models previously defined as discussed in Chapter 3, infers the user profile using her trajectories. The more trajectories this component has, the more accurate the classification will be. Users do not have to belong to one type of profile, they could be partially classified in different profiles. Therefore, a consuming application that adapts its behavior to the type of user disposes a more refined information than the most probably profile.

Previously, we talked about a context-aware application that presents a content according to the user position. Besides, if the application knows the identity of the user, the content can be adapted to its preferences. In this case, we go further and instead of providing the user identification, we are able to estimate the percentage of affinity to every profile from a set, therefore the content presented can be adapted to the main profile. Advertising, offers and notification can be customized if we dispose a set of types of targets and the user is classified in one or some of them.

The classification service is not integrated in the solution that is presented in this chapter. However, in Chapter 3, we proposed a methodology to classify users by using any type of classifier (though we used C4.5 algorithm). In future works, we propose to integrate the on-line classification in modules using this solution.

5.3.3 *Notifications service*

Another service that we propose and implement in our entire application is a notifications service, which, using both the Google and Apple messaging services, subscribe to events such as changes in the position or the finalization of a trajectory to send the user a notification. This notification is handled by the mobile device and this is not of our interest, as well as the content of the message.

As previously mentioned, our open framework enables the subscription to any event, which provides the identification of the user that either has changed her position or has ended a trajectory, therefore, if possible, a message can be

sent to this user.

5.4 Demonstration

Finally, the demonstration of the actual implementation in three different scenarios is presented in this section. In the three cases, the architecture of the application is the same. The user carries a mobile device that runs an application that receives the information from the beacons and, periodically, estimates its own position.

5.4.1 Test scenario

First of all, the scenario presented in Figure 5.1 was used to test the proposed localization algorithms described in Chapter 4. This scenario contains six different areas and a large and shapeless *out-of-zone*. In every area two beacons were placed, quite inside and close each other. Here, we tested our application with a reasonable performance and the installation of actual businesses in this retail-type space is possible as seen in the three use cases presented below. These use cases are stated but not actually deployed. We have supposed some use cases that could be deployed using our proposed solution.

Figure 5.1: Retail space

- **A bank brunch** is a first example to build a position-aware service based on our localization application. According to the scenario configuration, there is an entry zone that would be the first one where users are detected if they enter or even they walk close to the deployment scenario. Once they are detected, they can enter a kind of small room where, for instance, an ATM could be installed and the presence of the user is used to enable the machine. A proximity beacon could be installed in the ATM to ensure that the user is close enough and then enable the ATM to be used by the user. Nevertheless, there is a proximity zone if the user enters the main area where, for instance, a personal computer could be installed and customized according to the user that is detected in the proximity area.

The center zone of the scenario could be a waiting area where users are waiting to be assisted. There is a video wall installed in the curve

structure that is in the middle of the room, where customized contents could be presented according to the types of users that are waiting and, besides other context issues such as the date and the types of products they offer. Two more areas at the bottom of the plan are open to the main space so that non-private transactions or the attention of bank personal can be done. Again, the customization of the attention is possible if the client is detected and her profile is known, so that the worker can handle this type of information. Finally, at the bottom right of the plan there is a private room that could simulate the director office. The reception of the director to a special client is possible if she is notified whenever this type of client enters the branch.

To summarize, customizing the assistance of a client, showing the advertisements of the products that clients inside the branch would acquire, detecting users arrivals for their special treatment or enable the possibility to do some operations are a clear set of examples that would require a positioning system and, in some cases, a further processing of the set of positions of the user, forming a trajectory, is necessary to detect the type of client a user is. For instance, clients that only go to the bank branch to get money from the ATM would receive a different attendance than business managers that make trades with the actual bank.

- **An exhibition hall** could be also installed in this development space. At entry, a big area in the middle with a video wall and two more areas at the bottom of the plan, which could be complemented with more presence-type zones can be used to detect users or visitors and aim the exhibition to the type of user.

First of all, the entry of a visitor into the hall could enable to provide her with information related to the exhibition. For instance, a guide. The language, the amount and precision of information are parameters that can be adapted to the type of user. This is, a user that moves fast along the exhibition is not interested in detailed information, so that the amount of data and details that should bring a guide dedicated to this type of user should be lower if compared to a visitor that pauses in most of collection samples and requires a deeper behavior.

An audio-guide is a very used device in museums. The content that is presented to the visitor varies depending on her position. This variation could be automated if the positioning system detects the area the user is or her proximity to any part of the hall and then the audio-guide could adapt the content that is playing to the actual state of the visitor.

Videos that are presented in a loop can become boring and useless if they are shown while there is nobody watching it. Nevertheless, lots of visitors coincide at the same time in the reproduction of a video, for instance, in the video wall area or one of the two private areas that are in the retail scenario. In this case, a smart system that is able to detect how many people are waiting to be played using a timeout, so that they are not waiting for a long time, can optimize the viewing of a audiovisual content.

- **A retail store** is the third and last example to be described in order to understand how this real scenario can use the detection of the different types of user profiles and the positioning system to offer a better service. For instance, a clothes store can divide its areas in brands or types of clothes. Using presence type location or proximity beacons to detect what type of paths clients do, offers can be directed to the type of client. Private areas could become cash registers so that the detection of how many people are waiting to be attended is interesting to give an optimum service to the client when paying.

In general, regardless the type of use of the retail space, it is possible to give a specific service to a user according to her actual position. Besides the adaptation of this service to the type of user profile and the context, clusters of users can be considered and individual users can be classified into the different clusters, so that the customization of services is easier.

There are context and personal factors that cannot be acquired using the services proposed in chapters 3 and 4 such as date and time, user gender or preferences that can provide the context-aware service with a more detailed and accurate response to the actual user.

Thus, once we are able to imagine a set of examples that could be installed in the already existing retail space described in this chapter, next step is

the proposal of a real context-aware service that, using some of the features described along this work, achieves the real recreation of the context-aware service.

5.4.2 Multimedia service based on the user position

In the scenario presented in 4.3, some of the areas are provided with a display. A multimedia service is developed, in which a content is presented in the display corresponding to the area where the user is localized. Besides testing the performance of the algorithms, a service that consumes changes of positions and notifies a content manager that, using DLNA, sends a video to the corresponding display.

In our particular experiment, areas A, C, D and F are provided with two beacons, installed over the ceiling and a display, centered in the area. Our proposal starts from the detection of the user in one of the existing positions. This user, previously registered, is recognized by an identifier of her mobile device. In our lab, we implement other applications that adapt their behavior, for instance, the video presented on a screen, the color of ambiance lights and the music played whenever the user enters the scenario, to the user, which is recognized by many types of sensors: face recognition, hand signature recognition, mobile movement patterns, etc.

In this application, the user chooses the video she wants to be reproduced in the screen of the area where she is. The video selection, as presented in the video linked below, is carried out by NFC technology, through a set of tags that are attached to some elements, in this case, a magazine. In this case we merge the augmentation of information, giving the objects the possibility to become smart objects.

Link to video: <https://www.youtube.com/watch?v=WrbHFu65Bn0>

In the video, we can observe how the user approaches his mobile device to the NFC sensor attached in one of the articles in the magazine, then a video related to the article is presented in the screen placed in the area where he is. Whenever the user moves to another area, the localization algorithm, which runs in the smartphone and is continuously detecting new beacons information (id - RSS), estimates a change in the user position, so that the content that

is being displayed in the television where the user was previously, changes to the new television where the user is currently.

5.4.3 Actual deployment in a supermarket

Finally, in "Super U La Madeleine" super market (Nice), part of the store was used to deploy an actual localization system. In this case, the session service was also implemented along with an interaction service that detects the proximity of users to points of interests that are provided with a unique beacon transmitting a low signal. If the user is more than a specified time close to the beacon, which means that the RSS is over a threshold during a configured time, we consider there has been an user-point of interest interaction.

In this scenario (see Figure 5.2), a prototype of our user context-aware application was installed and tested. As presented in Figure ??, the localization component is the main element in the entire application. We provide the tester with a mobile device in its development version to test the correctness of the application performance. The display of the smartphone shows the current estimated position by the thresholds algorithm, as shown in video.

Figure 5.2: Actual scenario in Super-U La Madeleine (Nice)

We divided the scenario surrounded in red in 10 zones:

- Parfumerie
- Hygiène 1, 2
- Entretien 1, 2
- Animaux 1, 2, 3
- Alcools
- Caisses (at cash registers)

The placement of the beacons is carried out according to the possibilities the scenario provides, as seen in Figures 5.3 and 5.4.

Figure 5.3: Example 1 of beacon placement in the supermarket

Figure 5.4: Example 2 of beacon placement in the supermarket

Two beacons per area are placed and at four points each zone, a set of measurements of RSS from every beacon were taken during 5 minutes. The information gathered in these measurements are useful to understand the propagation in this scenario and to calibrate both the Thresholds-based algorithm, whose thresholds values were estimated, and a priori probabilities in Bayes-based algorithm, which inform $P(A_n(t)/O(t))$.

The main functionality of the application is the localization system, formed by iBeacons, which are transmitting periodically their identifier, and a mobile device, carried by a user, which detects this iBeacons signals. The demonstration of the performance of the entire service is presented in the video linked below. The test presented in the video was carried out using the Thresholds-based algorithm, so that *out-of-zones* are also included.

Link to video: <http://www.grpss.ssr.upm.es/index.php/es/videos/video/mobiloc-en-supermercado>

Besides, we performed a set of trajectories, in which our proposed algorithms (see Chapter 4) are evaluated. Every algorithm was further processed with a set of combinations of *detections-in* and *detections-off*, between 1 to 3. Thus, 9 combinations of further processing are carried out. Let's remember that *detections-off* are only valid in the Thresholds-based algorithm, so that, for the rest of algorithms there are only 3 different further processing, based on the *detection-in* value.

<i>Actual path</i>		<i>Simple 1-1</i>		<i>Simple 2-2</i>	
<i>Zone</i>	<i>Entry time (s)</i>	<i>Zone</i>	<i>Entry time (s)</i>	<i>Zone</i>	<i>Entry time (s)</i>
Animaux 3	0	Animaux 3	2	Animaux 3	3,1
Entretien 2	10,2	Caisses	12,6	Entretien 2	14,6
Entretien 1	24,3	Entretien 2	13,6	Alcool	26,3
Alcool	34,9	Alcool	25,2	Entretien 1	29,4
Hygiene 1	53,8	Entretien 1	28,3	Alcool	34,7
Parfumerie	71	Alcool	33,6	Hygiene 1	56,8
Caisses	90,1	Hygiene 1	55,8	Parfumerie	78,9
		Parfumerie	77,9	Caisses	92,6
		Caisses	91,6		

Table 5.1: Detection along time of the algorithms in an actual trajectory

Table 5.1 presents the comparison of entry time, if the actual path starts at instant 0 (in seconds), in every detected area. Errors of the Simple algorithm introduce two more estimated positions than zones in the actual path. Caisses zone is detected as the second position but this is an error corrected one second later. This error, in the Simple 2-2 algorithm is avoided. We can also detect the delay existing between the user actually changes her position to the change of the estimated position. In the Simple 1-1 algorithm, this delay is 3 seconds. In the Simple 2-2 algorithm, this delay is 4 seconds, just one second more due to the spatial filtering improvement presented in Chapter 4.

The sessions and notifications services have been developed for this experiment. The localization application is installed in the smartphone, while the rest of services are in a server. We have created a *hub* module that receives the notifications when there is a change in the position of the user, (push events to the server), and allows the subscription from other services to this information. Then, the sessions component subscribes to changes in every user position, so that, whenever there is a user position change, this new one is added to the current trajectory. When the user is not detected anymore or stops the application, the "off" message is sent to inform the sessions component that the user position has changed.

Figure 5.5 presents a session in the database used in our entire system. This shows a list of positions along a list of date-times, which represent the entry time to every area. Whenever the user is not detected any more or stops

the application the session is closed and, if the user is detected later, a new session is created with that user as target.

```

{
  "_id": ObjectID("579b06f8fdb6993137d4ce3d"),
  "area": "superu",
  "userId": "52B2EA15-6E43-41DD-966A-96B48A58FC80",
  "zones": [
    "hygiene1",
    "no_zone",
    "hygiene1",
    "entretien2",
    "cajas",
    "no_zone",
    "hygiene2",
    "cajas",
    "animaux3",
    "no_zone",
    "animaux3",
    "cajas",
    "animaux3",
    "no_zone",
    "hygiene2",
    "no_zone",
    "hygiene2"
  ],
  "times": [
    ISODate("2016-07-29T07:34:16.930Z"),
    ISODate("2016-07-29T07:34:20.818Z"),
    ISODate("2016-07-29T07:34:41.696Z"),
    ISODate("2016-07-29T09:25:51.902Z"),
    ISODate("2016-07-29T09:25:53.956Z"),
    ISODate("2016-07-29T09:25:58.396Z"),
    ISODate("2016-07-29T09:26:08.452Z"),
    ISODate("2016-07-29T09:26:25.487Z"),
    ISODate("2016-07-29T09:26:29.691Z"),
    ISODate("2016-07-29T09:26:36.446Z"),
    ISODate("2016-07-29T09:27:21.193Z"),
    ISODate("2016-07-29T09:27:23.441Z"),
    ISODate("2016-07-29T09:27:26.423Z"),
    ISODate("2016-07-29T09:27:32.019Z"),
    ISODate("2016-07-29T09:27:43.408Z"),
    ISODate("2016-07-29T09:27:47.817Z"),
    ISODate("2016-07-29T09:28:15.667Z")
  ],
  "session": 1,
  "finished": ISODate("2016-07-29T09:28:15.680Z")
}

```

Figure 5.5: Database sample of session

On the other hand, we have introduced in this deployment some other beacons, also BLE tags, used to simulate the interaction between users and points of interest such as objects or, for instance, vendors. Figure 5.6 shows this type of beacons, RadBeacon Dot (Rad), and when the user approach this beacon during a period of time over a set-up time threshold, a new interaction is registered along with the actual user position (see Figure 5.7).

Figure 5.6: iBeacon used for interactions

```
{
  "_id": ObjectID("579b130b160e2c0011b43606"),
  "id": "a6c5637b2c3a1376",
  "poi": "coffee",
  "area": "superu",
  "location": "alcohol",
  "entryTime": ISODate("2016-07-29T08:25:38.539Z"),
  "exitTime": ISODate("2016-07-29T08:25:47.754Z")
}
```

Figure 5.7: Database sample of interaction

5.5 Conclusions

We have proposed in this chapter the design of an application whose response would change based on the user context. This type of personalization has been focused on the user behavior when visiting a specific scenario. Users trajectories are useful to know the profile or profiles a user belongs (as seen in Chapter 3), thus, the merge of localization systems and the generation of trajectories are combined to detect users behaviors and trends.

We have proposed in our test scenario three different use cases that could be used to build actual business and, if users trajectories are built through the succession of users positions, the user profile is possible to be estimated, according to the type of business.

The actual deployments that have been carried out in this chapter only take into account the current user position (and previous in some algorithms) in order to estimate where the user is and offer him customized service, like the multimedia service. Besides, the storage and management of the successions of positions of every user is carried out in the final application described as 'Sessions service'. This is the start point of the trajectories creation.

Actual deployments require a calibration. In every different scenario (three in this case), the data acquisition is necessary to build good enough algorithms. As a future work, a value commitment is chosen between the time spent taking measures and the accuracy of the electromagnetic information acquired in the calibration phase.

Multimedia services allow the evaluation of the performance of the proposed application in real time. As seen in the related video to this section, whenever the user moves to a new area, there is a slight delay between a change of the area the user is and the transition of the video to a new display. Quick detection of changes of user positions could bring a unstable response of the application. Changes in the display where the video is playing could become annoying, so that it is preferable a small delay if the application is totally stable. This commitment is also present in Table 5.1, where Basic 2-2 algorithm avoids one of the two existing errors in Basic 1-1, but there is one more second delay in the general system.

Finally, modules-based application enables the possibility to get information from any module (the localization as final response or source to the multimedia service), and its use to feed further components.

Conclusions and future work

6.1 Conclusions

6.1.1 *User profiles*

The first contribution of this thesis is to propose a methodology to study the users behavior in any type of scenarios from their trajectories. First of all, we have described our methodology for a generic scenario, which contains N areas, and we have defined M user profiles, based on their behavior. We defined a trajectory as an ordered succession of positions since a user enters the scenario until she exists.

The number of user profiles and their features could be defined by clustering user real trajectories. Nevertheless, since we could not rely on the enough resources to carry out this task, we decided to define them, in the particular case, according to a specific scenario type: a shopping center.

Then, a mock-up of the real world has been built, based on the different user profiles. These user types have been modeled using ad-hoc models, which provide simulated trajectories that are used to create Markov and non-Markov

models. Markov models are models that take into account only the current state of the user, that is to say, the current position, to define the probability of moving to the next state or position. And non-Markov models define the probability of moving to the next position based on the current and the immediately previous ones.

We also developed an application that, using the proposed Markov and non-Markov models, outputs model-generated trajectories from the different user profiles. The proposal once we are able to build trajectories from the proposed models is to find how many simulated trajectories are necessary to both, create the proposed models and train a classification algorithm, with a certain quality. First, we must say that the classification algorithm is used to classify the model-generated trajectories in its corresponding profile. And second, the quality of a model is measured as the number of correctly classified trajectories, what would mean how good are they to indirectly represent the real world.

This methodology has been designed for a generic case and a specific case, a mall with 5 user profile types. Finally, the methodology is explained to be carried out by anyone whose objective is to know the user behavior in a scenario she would manage based on the user trajectories.

6.1.2 Localization

In Chapter 4, our contribution was to find a simple solution to estimate the user position, using her mobile device, in an indoor scenario. In literature there are many works that try to find the optimal solution to estimate the user position with the minimum distance error. Nevertheless, our intention was to minimize the effort in time and resources to build a localization system, using BLE beacons, creating also simple algorithms that estimate the user position. The scenario was divided in zones and the first solution, the simplest direct RSS algorithm estimates the user position as that whose beacons are providing the highest RSS value. This algorithm does not require a previous calibration.

Second, the comparison of the RSS values from the different beacons enables the design of a new zone, called *out-of-zone*, which covers every region that is

not defined as a zone of interest. Depending on the distance of the RSS values, which are compared with a set of thresholds that are previously calibrated for every pair of zones, the area where the user position is estimated is one of the zones of interest or *out-of-zone*.

Finally, we have added the possibility of counting with the previous state or user position. A Bayes-based algorithm has been designed, which takes into account the probability of being in every area as a function of the RSS, the probability of being in every area as a function of the previous area and the probability of being in every area in the previous step. This memory-based algorithm requires the definition of the transition probabilities, which indicate the probability of moving between every pair of areas.

In order to provide a certain stability to the estimated positions, due to the variances of this estimation value provided mostly by RSS fluctuations, we decided to add a spatial filtering after the localization algorithm. Then, the entire localization system was composed by the localization algorithm followed by the spatial filtering. It is configurable by two parameters: *detetions-in* and *detections-off*. *Detections-in* indicates the number of consecutive times the user position must be estimated by the localization algorithm, so that the spatial filtering and, therefore, the localization system outputs that zone as the estimated user position. On the other hand, *Detections-off* indicates the number of consecutive times that the area *out-of-zone* must be estimated in order to provide that area as the estimated user position.

We evaluated the performance of the three algorithms in two different conditions. First, in static conditions, where the user was placed in some positions of two different zones and calculated the ratio of good estimated positions respect to all the estimations. We observed in that case that all the algorithms provided the same ratio. We are aware that the experiment that we present in this work is only a sample from many others we did. If all algorithms provide the same ratio when estimating in static conditions, that is, the user is not going to move, we should choose the simplest direct RSS algorithm, which does not require a previous calibration. Nevertheless, as mentioned above, we only presented the results of one example.

On the other hand, in dynamic conditions, where the user is moving between different zones, we actually detected in the experiment we performed that the

Thresholds-based algorithm improves the simplest direct RSS algorithm, while Bayes-based algorithm improves the Thresholds-based algorithm performance. We also added the spatial filtering to the Thresholds-based algorithm and it was proved that it reduced the incorrect estimations if *detections-in* and *detections-off* were set to 2 and completely eliminate incorrect estimations if were set to 3.

6.1.3 Personalization of applications

Our contribution in Chapter 5 was the design of an example of mobile-based application that applied the concepts from Chapters 3 and 4.

First, we designed and developed an application that estimates the user position from BLE beacons RSS. In this application we installed one of the proposed algorithms in Chapter 4, letting the spatial filtering configuration, that is, *detections-on* and *detections-off* parameters, configurable for every different type of deployment.

We also defined a services layer that contains three services:

- *Session service*: it creates trajectories from users estimated positions. This service has been implemented and allows to create a list of trajectories per user to study their behavior as proposed in Chapter 3.
- *Classification service*: this service has been designed to classify users in different profiles according to their trajectories, as proposed in Chapter 3.
- *Notifications service*: this service provides a notifications web application that enables the user to send default messages to those users that enters in some specific areas or customized message to some users or user profiles. It combines some concepts from both Chapter 3 and 4, through the classification of users to send different types of messages according to the user profile and works while the localization system detects a user enters in a specific zone.

6.2 Future Work

6.2.1 *Classification*

The clustering of objects consists on the detection of different groups from a set of elements. In this case, in Chapter 3, we assumed a set of groups to study our tools and models. Nevertheless, those groups can be detected using clustering tools from source data. That is to say, if a set of trajectories are acquired from different types of users, we could try if our system is possible to detect the different types or profiles of users existing in source data, through their trajectories.

Acquisition of actual data from users would provide a more realistic information to create models. Unlike our tool of generating mock-up trajectories from 'real' users, data from actual people would be harder to obtain but would fit better to real life.

6.2.2 *Localization*

In relation to our work in the localization system, the calibration phase could be optimized. This is, it would be interesting to find a value of commitment between the number of measurements that are necessary to provide a certain quality for every algorithm.

The exploration of new technologies, techniques, algorithms and configurations of infrastructure is an infinite field of study to improve and find better solutions to our proposal.

Auto-calibration or self-calibration is a concept we propose to avoid the necessity of hard tasks of calibration. Furthermore, this task can be carried out by the customer. Now, when the localization system is installed in any scenario, we perform some calibration tests. In the case of the grocery described in Chapter 3, it took us two hours to calibrate the ten areas (four points per area) and two more hours to process and update this information to the corresponding databases. Our proposal is to build a self-calibrated product that taking a few measurements with a mobile calibration application so that the algorithms set-up is automatically performed.

Besides, if the environment changes its configuration, a self-calibrated product could optimize algorithm parameters for the new configurations.

Another handicap we have found along our years of research in this field is the use of multiple mobile devices. Both the calibration and test phases have been done using the same model. Nevertheless, changing the smartphone could change its components, so that different values can be detected. Though we have not addressed this problem or make test results, we actually used migrated our code to an iPhone and the results were correct.

6.2.3 Application

Our personalized application takes into account the user position to change its behavior (e.g. multimedia service). Nevertheless, the combination of users classification (Chapter 3) along with the sessions creation through the trajectories created thanks to the succession of positions (Chapters 4 and 5) is a new step in the development of user context-aware applications. For instance, in the multimedia service described in Chapter 5, the video could be different according to the user (we can detect who the user is now), but the video could also be different according to the type of user if she is classified.

Therefore, we propose the actual classification of users through the sessions information. When the user performs a new trajectory, this could be used by the classification algorithm to improve the classification of the user.

Appendix A

Trajectories generation tools

The simulation of trajectories based on Markov models is done using a desktop application that has been designed and developed in this work. In Chapter 3, the cases of study for *M0* and *M1* models have been discussed and two versions of this tool have been developed, each corresponding to both used Markov models. Both accept a scenario configuration formed by N areas (states) along with the *Entry* and *exit* zones. The versions based on the *M0* model describe the probability to be in a zone s_n based on the current state s_m , meanwhile the version based on the *M1* model requires the definition of the probability to be in s_n , based on the current state s_m and the previous state s_l .

In this appendix, first, the description of trajectories, scenarios and user models are introduced. Second, the design and the description of the tools are presented along with the possibilities it offers a user to simulate the generation of trajectories of user profiles that are designed for the working scenario. Finally, the evaluation of the tool is discussed to prove its correct performance.

A.1 Users trajectories and scenarios models description

A trajectory is a succession of positions $tr = tr^1, tr^2, \dots, tr^K$. Every position corresponds to one of the possible states existing in a scenario $S = s_1, s_2, \dots, s_N$, where $tr^k \neq tr^{k+1}$ and $tr^1 = Entry$ and $tr^K = exit$. A position, tr^k , is represented as an object formed by the next three elements:

- Position - name of the area the user has been.
- Entry Time - Date and time the user entered the position.
- Exit Time - Date and time the user left the position.

Figure A.1 represents a sample of a trajectory defined in a xml file.

```
<?xml version="1.0" encoding="UTF-8"?>
<trajectories>
  <trajectory userProfile="User_B">
    <data>
      <position>A</position>
      <entryTime>0</entryTime>
      <exitTime>1.69118</exitTime>
    </data>
    <data>
      <position>C</position>
      <entryTime>1.69118</entryTime>
      <exitTime>3.60165</exitTime>
    </data>
    <data>
      <position>A</position>
      <entryTime>3.60165</entryTime>
      <exitTime>5.27606</exitTime>
    </data>
    <data>
      <position>C</position>
      <entryTime>5.27606</entryTime>
      <exitTime>6.41501</exitTime>
    </data>
  </trajectory>
</trajectories>
```

Figure A.1: Sample of the xml of a trajectory

The development of a tool of simulation of trajectories is based on an existing scenario formed by 8 areas, besides the *Entry* area and an *exit* area that, in this case, they are the same. The training scenario is represented in Figure 4.3 and the user profiles are described based on it. As told above, this tool

has been done in two different versions. The former one is able to create trajectories based on Markov-type models of user profiles (*M0*), where the probability to go to a state s_n only depends on the current state. It means that previous states are not taken into account to define the probability to be in, for instance, B if the user is currently placed in A ($P(B|A)$).

On the other hand, a second version of the application enables the configuration of a scenario whose profiles are defined by the probability of the users to change their position according to their current and previous state. This is an approach to a more real situation and is based on the *M1* model. In relation to the mall scenario described in Chapter 3, a person who has already been to the supermarket, if she is currently in the book store, she most likely will not return to the supermarket. So, the probability to go to the supermarket if the user had previously been there is not the same than the probability to go to the supermarket if she had not been there before. In other words: $P(\text{Supermarket}|\text{Bookstore}|\text{Supermarket})$ and $P(\text{Supermarket}|\text{Bookstore}|\text{TechnologyStore})$ do not have to be equal.

In order to carry out the definition of scenarios in the simulation tool, they are defined using a binary matrix that indicates the existence of a relation between two areas. Rows and columns represent every area along with the *exit* area, and the value that relates a row and a column indicates next:

- If $[A,B] = 1$; the transition from A to B actually exists,
- if $[A,B] = 0$; the transition from A to B does not exist.

Since *Entry* area is only connected to the A area, this is going to be obviated and the first position will be assumed to be A for every trajectory.

Looking at the scenario represented in Figure 4.3, the transitions matrix is not symmetric because $[A,B]$ and $[B,A]$ do not have to be equals. In other words, it may be possible to move from A to B but it may not be possible to move from B to A . This statement can be controversial because there are not physical laws that forbid the transitions from one place to another if the opposite one is possible in ordinary scenarios. Nevertheless we could think in a museum where there is an entry door and an exit door. If the entry door communicates, for instance, the *outdoors* and the *hall* and the exit door communicates the *exit area* and the *outdoors*, in the transitions matrix next

parameters could be defined (case without memory - Markov-type model), for visitors:

- $P(hall|outdoors) = 1$ and $P(outdoors|hall) = 0$
- $P(exitarea|outdoors) = 0$ and $P(outdoors|exitarea) = 1$

Thus, reviewing the training scenario (see Figure 4.3), areas in the corners are connected only to one another area, while centered areas have a connection to three other areas. Let's remember that the transitions between zones are only possible in vertical and horizontal directions. And both the transition between areas A and B , and the transition between G and H are not possible because the existence of walls. This is a real scenario where this work has been carried out and the deployment and evaluation of the positioning system proposed in chapter 4 has been done. The matrix that describes the possible transitions in the training scenario is represented in Figure A.1.

	A	B	C	D	E	F	G	H	$exit$
A	0	0	1	0	0	0	0	0	1
B	0	0	0	1	0	0	0	0	0
C	1	0	0	1	1	0	0	0	0
D	0	1	1	0	0	1	0	0	0
E	0	0	1	0	0	1	1	0	0
F	0	0	0	1	1	0	0	1	0
G	0	0	0	0	1	0	0	0	0
H	0	0	0	0	0	1	0	0	0
$exit$	1	0	0	0	0	0	0	0	0

Table A.1: Transitions matrix for scenario A

A.2 Design and development of trajectories simulation tool

Once the transitions matrix, which describes the possible transitions among all the zones of a scenario, has been created, user profiles should be then modeled in order to describe their movements in the scenario. Thus, for each user profile a matrix called *user probability matrix* is created. It describes the probability of a user to move from an origin position to the destination position. The values in this matrix vary from 0 to 1, both included. A 0 in the cell of the matrix relative to the row A and the column B indicates that the user will never move from A to B . This happens in both cases: either the

movement is not possible (this is a 0 in the transitions matrix that describes the scenario) well this type of user profile will never do this movement. Meanwhile, a 1 in that cell indicates that if the user is in A , the next position will be sure B .

The tool of simulation of trajectories has been implemented in the form of an application with a GUI that enables the configuration of the scenarios and the user profiles. Next, a list of the features of this tool is presented:

- **Creation and edition of scenarios** - the user of this application can create and edit a scenario, using the GUI. To do this, the user can add and remove areas to the scenario that is being edited. The area is defined by its unique name. Once the scenario is defined as a set of areas, the next step is to configure the possible transitions between every pair of areas in both directions. This is the *transitions matrix*.

Let's remember that the transitions matrix should be symmetric to allow the creation of every type of user profiles (the restrictions in transitions would be implicit in the user profile model). Nevertheless, the flexibility of the tool allows the user to create an asymmetric matrix so that both the transitions from A to B and the transition from B to A must be enabled by the user in the application. One more thing is necessary to finish the configuration of the scenario. The user must indicate at least one area to be connected with the *exit* area.

- **Creation and edition of user profiles** - once the scenario has been defined through a set of areas and the possible transitions among them, the next step is to define the user profiles. A user profile is defined by a unique name. The configuration of the user profile is then done setting a probability value for every possible transition. In this case, there are two versions of the application. Let's name *current area* to the area the user is supposed to be before the transition; *previous area* to the area the user was in the previous step to *current area*; and *next area* to the area the user can move after *current area*. The description for both models is done below:

- **MO model.** The configuration of the user profiles in this version of the tool is done area by area. For each one, the sum of probabilities to transition from the current area to every other area where the movement is possible is equals to 1, for every user profile. This is:

$$\sum_{\forall n} P(s_n|A) = 1 \quad (\text{A.1})$$

where s_n represents every area where the transition from A is possible. In relation to the training scenario, the transitions matrix was already represented in Table A.1. Furthermore, if *current position* is C , we can get the possible *next areas* by searching in the matrix represented in Table A.1. These are A , D and E . Therefore, for every user profile, the transitions that must be configured from position C are represented in Table A.2:

	<i>next areas</i>		
<i>current area</i>	A	D	F
C	P(A/C)	P(D/C)	P(F/C)

Table A.2: Transitions probability area C for a markov01 model

- **M1 model.** In this case, the configuration of a user profile is done area by area too. However, for each one, the probability to move from *current area* to all the possible *next areas* must be configured for each possible *previous area*. Thus, for every peer $[currentarea, previousarea]$ the equation A.1 must be fulfilled. If there are n areas, there are not necessarily $n \times n$ combinations. The combination $[currentarea, previousarea]$ for every pair of areas being $currentarea \neq previousarea$ only exists if this is possible to move from *previous area* to *current area* (see Table A.3).

	<i>next position</i>		
<i>current position / previous position</i>	A	D	F
C/A	P(A/C/A)	P(D/C/A)	P(F/C/A)
1-1 C/D	P(A/C/D)	P(D/C/D)	P(F/C/D)
1-1 C/F	P(A/C/F)	P(D/C/F)	P(F/C/F)

Table A.3: Transitions probability area C for a M1 model

Modeling of training scenario. Training scenario is a test scenario that is based on the laboratory where this work was carried out. This is defined in Figure 4.3 and its transitions matrix is defined in Table A.1. Besides, the entry door is placed in *A*. This is, the first area in every trajectory is *Entry* and suddenly, the second position is *A*. Furthermore, unless a preset limit of transitions is reached, there is a probability to go from *A* to *exit* that enables the trajectory ends when this case is success. So, if the trajectory is not finished due to a limit of transitions, the last area is *exit*, which is preceded by the position *A*.

In this scenario, based on an existing office, people is usually placed in areas *B* and *G*. Thus, two users profiles have been designed, each corresponding to a user that is usually placed in *B* and another that is usually placed in *G*. Let's call these two user profiles *userA*, to refer that one that is working in the area *B*, and *userB* to refer the user profile that is working in the area *G*. Tables ?? and A.5 represent the probabilities matrices for both user profiles in the case of the *M0* model.

<i>Area</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>exit</i>
<i>A</i>	0	0	0,7	0	0	0	0	0	0,3
<i>B</i>	0	0	0	1	0	0	0	0	0
<i>C</i>	0,2	0	0	0,6	0,2	0	0	0	0
<i>D</i>	0	0,4	0,4	0	0	0,2	0	0	0
<i>E</i>	0	0	0,6	0	0	0,1	0,3	0	0
<i>F</i>	0	0	0	0,65	0,15	0	0	0,2	0
<i>G</i>	0	0	0	0	1	0	0	0	0
<i>H</i>	0	0	0	0	0	1	0	0	0
<i>exit</i>	1	0	0	0	0	0	0	0	0

Table A.4: Probability Transitions Matrix for user a

These models are used in the configuration of the application for the *M0* models. The probability to go to a specific area only depends on the *current area*.

In the other hand, the model for the *M1* model is composed by *N* matrices, being *N* the number of existing areas, taking into account the *exit* area. Each matrix corresponds to the *previous position* condition.

<i>A</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>exit</i>
<i>A</i>	0	0	0,6	0	0	0	0	0	0,4
<i>B</i>	0	0	0	1	0	0	0	0	0
<i>C</i>	0,35	0	0	0,05	0,6	0	0	0	0
<i>D</i>	0	0,4	0,4	0	0	0,2	0	0	0
<i>E</i>	0	0	0,4	0	0	0,2	0,4	0	0
<i>F</i>	0	0	0	0,3	0,6	0	0	0,1	0
<i>G</i>	0	0	0	0	1	0	0	0	0
<i>H</i>	0	0	0	0	0	1	0	0	0
<i>exit</i>	1	0	0	0	0	0	0	0	0

Table A.5: Probability Transitions Matrix for user b

This is, if we want to check the probability to go to *C* if *current area* is *A*:

- Matrix corresponding to area *exit* must be used if *previous position* was *exit*.
- Matrix corresponding to area *C* must be used if *previous position* was *C*.

If *current area* is *A*, *previous position* can be only *Entry* or *C*.

- **Times configuration.** Since trajectories are defined using the entry time and exit time per area, the minimum time and the maximum time every user profile could spend in each area can be configured. This parameters are optional because the generation of trajectories can be done without using the time.
- **Trajectories generation.** Once the areas, the transitions among them, the user profiles, the transition probabilities among areas for each user profile and, optionally, the minimum and maximum time a specific user profile could spend in each area are configured, the application is ready to generate aleatory trajectories. The generation of trajectories is done through the scenario that is loaded in the application, using the different user profiles that have been configured for the actual scenario. Every trajectory that the simulation tool creates corresponds to one of the existing user profiles. This choice is aleatory according to a configurable parameters described in the next paragraph.

Before the simulation starts, next parameters must be also configured, which are independent for every new trajectories generation process:

- **Probability trajectory-user profile.** This parameter indicates the probability a used profile is chosen to generate a new trajectory. Obviously, the sum of the probability values of all user profiles should be equal to 1.
- **Maximum number of stages.** The *exit* area is used to end the generation of one trajectory. Once the trajectory generation process is in one of the areas where this is possible to move to the *exit* area, if this event happens the trajectory is over. There is some probability that the simulated trajectory performs a large number of loops by some or all its states and do not get to go to the exit area. In this case, this parameter indicates the maximum number of stages of a path so that it is not infinite.
- **Maximum time.** In minutes, this parameter indicates the maximum time of a trajectory before it is terminated. As the previous parameter, its proposal is to avoid a supposed infinite trajectory.
- **Number of trajectories.** The number of trajectories that are going to be created.

The user of the application, before running the process that starts the simulation of trajectories, chooses if the time is going to be taken into account. The process that creates the trajectories works as follows:

1. The entry area that will be the starting position of every trajectory is chosen.
2. Before starting the creation of a new trajectory, using a uniform random variable between 0 and 1 the user profile is chosen.
3. Once the user profile has been chosen to generate the next trajectory, the model of that user is loaded and using the values in Table ??, a new process to generate a new trajectory starts. In this example based on the training scenario, position *A* is the first one. A new random variable is generated to estimate the next position in the trajectory. The limits in the random variable that affect to the next position in the trajectory are set by the row of the matrix that describes by the user profile. For instance, using the matrix of the

user a (see Table ??): if the first position was the area A and the random variable is between 0 and 0.7, *next position* will be the area C , but if the random variable is between 0.7 and 1, *next position* will be *exit*.

4. If the trajectories creation process is running and the *time* parameter is enabled, the entry time is the exit time of the previous position, or 0 if this is the first position in the trajectory, while the exit time is obtained using a probability function where the domain is between the minimum and maximum time of the *current position*.
5. Finally, when *number of trajectories* trajectories have been simulated, an XML is created. This XML file stores a list of the trajectories, each with an ordered list of the positions that form the trajectory, and each position along with the entry time and the exit time in each position, in minutes. These two last values are 0 if the *time* parameter is not enabled.

A.2.1 Trajectories generation tool evaluation

Before analyzing the simulated trajectories, the generation tool must be evaluated and validated to prove its correct performance. To do this, we checked some basic parameters in the simulated trajectories, which were compared to the same parameters obtained using the user profiles design models.

The parameter that we have used to evaluate and validate the trajectories generation tool is the probability of being in a specific area if the user had already been in another defined area. For instance, if the user enters in the scenario A, the first position in the trajectory is the area A. Let's remember this scenario is based on a real one and the entry is actually there. We can argue this parameter by indicating that we want to obtain the probability that the user goes to her work area whenever she enters in the scenario (she has already been to area a). For instance, in relation to the user a : this parameter could be $P(B_n(t)|A_m(1))$, being $t \geq 1$ and $n \neq m$.

Once a large number of trajectories are simulated, in this case 10.000 trajectories, which all of them belong to the user profile a , a script to calculate this parameter has been developed. The script performance is described in

the next section.

The entire process of evaluation and validation is widely explained in appendix A.3.1.

As a first conclusion, we are able to confirm that this tool is able to generate simulated trajectories based on modeled scenarios and user profiles. User profiles are modeled according to the *M0* model, the probability to be in the *next position* only depends on the *current position*; or according to the *M1* model, in which the probability to be in the *next position* depends on both the *current position* and the *previous position*.

Once this tool is correctly implemented, evaluated and working, this is possible to answer the proposed questions such as: "*What is the number of transitions between two consecutive areas?*" or "*What is the probability to be in B if the user has been in A before?*"

A.3 Trajectories analysis

A.3.1 Relevant information from trajectories

Trajectories have been defined as an ordered list of positions than either end with an exit position or are terminated when they reach a maximum number of stages. Every position in a trajectory can go with an *entry time* and an *exit time* that define the minutes the user entered and left in the area. The trajectory starts in the minute 0. If the simulated trajectory was not created using the time parameter both *entry time* and *exit time* are 0.

From one trajectory, there are works that try to study patterns from a list of events or items (Srikant and Agrawal, 1996) (Agrawal and Srikant, 1995) using algorithms that, from a list of item-sets, search for patterns to find the user trends. Our goal is to demonstrate how BI tools can be useful to characterize user profiles according to trajectories if these are converted to a set of parameters. For instance: one trajectory can be expressed through next parameters:

- Times every area is visited in the trajectory.
- Times every transition is done in the trajectory.
- Mean time (per stay) every area is visited in the trajectory. The time can be expressed in minutes or normalized.
- Total time every area is visited along the trajectory.

The underlying information in the trajectories can be used to detect the user profile related to each trajectory, once some of them have been previously classified. Then, the intention is to be able to use the simulation tool to generate trajectories over simulated scenarios and user profile models for clustering and classification of user profiles. This helps business managers because they could work with simulated models to study users behavior.

The evaluation and validation of the simulation tool is done separately to the two existing models. This is done comparing the parameter *probability to be in B if the user had been before to A* estimated using two different methods: a large set of simulated trajectories and the theoretical model.

This parameter can be written as $P(B_n|A_m)$, where n and m are stages in the trajectory that fulfill $n > m$.

A.3.2 M0 model

The estimation of the parameter $P(B_n|A_m)$ using the simulated trajectories based on the *M0* model of the user profile a starts with the iteration of the list of trajectories. Since the processing is done separately for both proposed profiles, two lists of 10000 trajectories for each profile are created. For the user profile a , we are going to look for how many trajectories fulfill that after being in A (this happens in every trajectory), the user has been in B . For the user profile b , we are going to look for how many trajectories are in G after being in A :

$$P(B_n|A_m) = \frac{\text{number of trajectories the user has been to BorG}}{\text{total number of trajectories}} \quad (\text{A.2})$$

Once this parameter is obtained using the simulated trajectories, our purpose is to compare it with the theoretical model, described in the matrix represented

in figure ??.

To do this, a script in Matlab has been implemented and is represented in Table A.2. In this script:

- *input* is the vector $[1, 0, 0, 0, 0, 0, 0, 0, 0]$ that represents the position A .
- P is the matrix represented in Table ?? that propagates the position of the user along the future stages.
- N is the maximum number of stages.

Every iteration, which represents one more position in the trajectory, the element corresponding to the area B is set to 0 because this is supposed that, if the user gets to the pretended area B in the stage n , that trajectory fulfills the condition $P(B_n|A_m)$ and the rest of the trajectory is not taken into account.

```

result = 0;
a = input;
for i = 1:N
    a = a * P;
    result = result + a(2);
    a(2) = 0;
end
end

```

Figure A.2: Script to theoretically obtain $P(B_n|A_m)$ for user a through the $M0$ model

The theoretical value for each profile is unique and ten sets of 10000 trajectories have been analyzed for each profile. Therefore, for each profile, we have obtained 10 values, which are the difference between $P(B_n|A_m)$ simulated and theoretical. The mean value of error for the profile a is $\mu_a^{M0} = 0,00378$ with a standard deviation $sd_a^{M0} = 0,00366$. For the profile b : $\mu_a^{M0} = 0,03521$ with a standard deviation $sd_a^{M0} = 0,004276$. This error represents the difference between the theoretical and the simulated calculation of $P(B_n|A_m)$ from 10 repetitions of 10000 trajectories.

A.3.3 M1 model

The evaluation of the *M1* version of the simulation tool is done also comparing the parameter $P(B_n|A_m)$ using the trajectories generated by the version *M1* model version and the model that indicates the probability to be in *next position* based on the *current position* and the *previous position*.

The difference with the comparison with the *M0* model is the theoretical comparison because the script uses *M* matrices according to the number of areas, each corresponding to the *previous position*. In this case, *input* is a 9x9 matrix, if this is applied to scenario *A*, where there are 8 areas besides the *exit* area. Each row represents the *previous position* and each column the *current position*. Thus, a value in the row *j* and column *k* indicates the probability to be in *k* if *previous position* was *j*.

```
function [ result ] = processm1( input, T , N )
%UNTITLED2 Summary of this function goes here
% Detailed explanation goes here

column = 2;
result = zeros(size(input));
X = input;
[P,Q] = size(input);
result = 0;
for i = 1:N
    B = zeros(P,Q);
    %% filas x columnas
    for j = 1:P
        for k = 1:Q
            A = zeros(1,P);
            A(k) = X(j,k);
            C = zeros(P,Q);
            C(k,:)=A*T(:,j);
            B = B+C;
        end
    end
    for j = 1:P
        result = result + B(j,column);
        B(j,column) = 0;
    end
    X=B;
end
end
```

Figure A.3: Script to theoretically obtain $P(B_n|A_m)$ for user *a* through the M1 model

T is a set of 9 9x9 matrices that represent the probability models. Each matrix is related to a *previous position*, the row of each matrix is related to the *current position* and the value in the column j is the probability to be in j if *current position* is the area indicated by the row and the *previous position* the one related to the corresponding matrix.

Finally, N is the maximum number of stages that the $P(B_n|A_m)$ value is propagated. The script that evaluates $P(B_n|A_m)$ for a theoretical value is described in Figure A.3 and the entire process is similar to the used in the previous section for the $M0$ model. The mean value of error for the profile a is $\mu_a^{M0} = 0,00343$ with a standard deviation $sd_a^{M0} = 0,00191$. For the profile b : $\mu_a^{M0} = 0,00368$ with a standard deviation $sd_a^{M0} = 0,0032$.

A.3.4 Conclusion

The results obtained in Section A.3.2 and A.3.3 show the error of the trajectories through the parameter $P(B_n|A_m)$ if compared with the theoretical value. This is below 1% and the standard deviation too. This demonstrate the fidelity of the simulated trajectories to the proposed models and the guaranty of the right performance of the designed and developed tool.

Appendix B

Selection of the technology to build the infrastructure of the localization system

The selection of iBeacons as support technology, between the existing RF technologies, is due to its low battery consume and, thus, their great autonomy that makes it possible to place them in strategic positions without the necessity of wires to be powered nor to change their batteries quite often.

Besides, the indoors calibration of these devices makes it easier the comprehension of the propagation of this type of signals. The goal of this calibration study is to design a robust and stable positioning algorithm minimizing the number of used beacons.

B.1 iBeacons

Technologically, iBeacons are small, low-consume and low-cost devices that use Bluetooth Low Energy technology to transmit a unique identifier (UUID), plus the Major and Minor values, its own MAC address and the theoretical RSS at one meter, among other information that is not of interest in this part of the work. Next, different parameters are explained in order to clarify what information can be useful for us:

- *UUID* - unique identifier of 16 bytes that is used to distinguish beacons from others that belong to a different vendor or work in a totally different project (installer choice). Normally, when a set of iBeacons are purchased, they come with the same UUID.
- *Major* - 2 bytes size identifier that is used to group a set of beacons. This value is commonly configured after purchased through the vendor application that is provided. It can also be done programmatically.
- *Minor* - similar to Major but is used to distinguish each individual beacon.
- *MAC address* - network address of the level 2 OSI reference model.
- *Measured power* - vendor estimation of RSS at 1 m far from the iBeacon based on the broadcasting power.

There are two working modes with iBeacons. The former one is *monitoring* mode. It consists of iBeacons detection and allows to perform an action when a specific iBeacon is discovered. On the other hand, iBeacons *ranging* returns an array of the detected iBeacons along with their properties. They are equipped with a kind of positioning system that offers an approximate distance among three options, according to the comparison of the RSS and the :

- *Immediate* - distance $< 0,2$ m.
- *Near* - $0,2$ m $<$ distance < 2 m.
- *Far* - distance > 2 m.

Since the appearance of the term iBeacon along with the BLE technology, leading retailers and brands including Apple, Macy's, Coca-Cola and Procter & Gamble are exploiting proximity marketing to close the last gap between the customer and the register (G., 2016). It has led commercial brands to

launch proximity based products. For instance, a store is provided of a set of beacons, which are strategically placed to cover different areas. Along with this, a mobile application must be downloaded and installed in the user's smartphone. If this application is running, when the user approaches a beacon and enters inside its coverage area, the application starts an action related to this event (e.g. a pop-up advertisement is shown). Our proposal is to study the behavior of beacons signals and we propose to add a higher layer that, processing the RSS values from the installed beacons, to estimate the position of a user. First, we have done a calibration work, which has allowed us to understand the BLE beacons behavior in indoors enclosures. We acquired two different iBeacons models: RadBeacons (Radius Networks) and Estimotes.

The former one is a USB standalone iBeacon (B.1). This is a pluggable device feeded through the USB port where it is installed. The main disadvantage of this type of iBeacons is its dependance on the actual position of the device where it is plugged. By the other hand, the autonomy of this devices is not dependable on a battery, so that further maintenance is not necessary. Nevertheless, this is necessary a device to provide the power and the iBeacon position depends on its connection to the USB port. For instance, if a RadBeacon is plugged in a television, it must be placed behind it.

Figure B.1: Radius Networks RadBeacon

The second type of iBeacons that have been tested and have definitely been chosen to deploy the localization system are Estimotes (B.2). Unlike RadBeacons, these beacons do not depend on an external supply. They contain a button battery that offers an autonomy of approximately two years. New firmware versions allow to configure every single beacon in order to increase the iBeacons' autonomy at the expense of reduce the bandwidth communication

with other devices.

Figure B.2: Estimotes

Both types of iBeacons are provided with a mobile application that allows service developers to configure individually each beacon. A Bluetooth connection is opened and some of the parameters that can be obtained in ranging mode can be set up. For instance, the Estimote application for iPhone enables a sort of application demos such as proximity, distance or notification demo and an indoors positioning system that allows the user to calibrate the positioning system. Besides, detection of iBeacons is possible and some of the configurable parameters that can be configured are:

- *Major* and *Minor* numbers
- *Broadcasting power* - a value from -30 dBm to +4 dBm in steps of 4 dB. It is estimated a coverage range from 1-2 meters at the minimum level to 70 m, proportionally to the broadcasting power.
- *Advertising interval* - this value can be set from 50 ms to 2000 ms.

Figure B.3 shows an approximation of the autonomy of an Estimote based on different values of the advertising interval and broadcasting power.

		Broadcasting power		
		-30 dBm [low]	-4 dBm	+4 dBm [high]
Advertising interval	2000 ms [long]	3.3 years	3 years	2.3 years
	1000 ms	1.9 years	1.7 years	1.3 years
	600 ms	1.2 years	1 year	300 days
	200 ms	160 days	140 days	104 days
	50 ms [short]	40 days	35 days	26 days

Figure B.3: Estimate autonomy based on broadcasting power and advertising interval Blog (2016)

Appendix C

RSS-based solutions

In Chapter 2, we classified and described the most common techniques and technologies related to indoors positioning. On one hand, RSS techniques aim to estimate the distance between the transmitter and the receiver, in this case the beacon and the mobile device, using the RSS value. These are channel-based techniques that use either theoretical models as the one represented in Figure C.1, which describe the detected RSS due to path loss. Once the RSS is measured in the mobile device, the distance to the beacon is calculated. If three distances are known, the position on a 2D map can be inferred using trilateration.

This model is not real because signal propagation does not fit to the ideal curve represented in Figure C.1. Figure C.2 represents an actual sample of indoors signal behavior. This chart was obtained holding a mobile devices at several distances between the beacon and the receiver (the mobile device), from : $0,5m$, $2m$, $4m$ and $6m$. The represented value for each distance is the mean of 100 RSS measurements when the beacon is transmitting at $-4dBm$. The descent of the curve is not proportional to the distance, what makes it hard to calculate the distance between the beacon and the mobile device from the RSS.

Figure C.1: Theoretical Free-space path loss

Figure C.2: RSS vs distance - mean value

Besides, Figures C.3, C.4, C.5 and C.6 represent the measurements of RSS along the time of the experiments described above. In this chart, we can

observe the instability of the RSS values, what makes it hard to create an ideal relation between power loss and distance and, then, to build a channel model based system.

Figure C.3: RSS vs time at 0,5 m

Figure C.4: RSS vs time at 2 m

Figure C.5: RSS vs time at 4 m

Figure C.6: RSS vs time at 6 m

In (Besada et al., 2007)(Tarrío et al., 2007)(Tarrío et al., 2008)(Bernardos et al., 2010)(Tarrío, 2011), an optimal solution is discussed, based on the so-called circular and hyperbolic positioning algorithms. These algorithms start from the distance estimated using the RSS. In the former case, the optimal solution is found by minimizing the sum of the square errors in the set of estimated distances (Chernov and Lesort, 2005):

$$\epsilon = \sum_{i=1}^N (\sqrt{(x_i - x)^2 + (y_i - y)^2} - \tilde{d})^2 \quad (\text{C.1})$$

The hyperbolic positioning algorithm estimates the least squares solution of the set of equations that calculate the difference of the estimated distances from all the nodes (except one, which is taken as reference) to the reference node (Yang et al., 2008).

The solution proposed by (Tarrío, 2011) tries to improve these two algorithms by weighting these distances, increasing its weighting when distance is shorter. This is because, according to the log-normal channel model, RSS does not depend linearly on the distance, so that the same error in RSS produces larger

errors in the distance estimation if the distance between the node and the target is higher.

C.1 BLE signal behavior

The variability of the RSS indoors is inherent to this type of signal in this context. As mentioned above, RSS solutions try to find techniques that handle with this drawback. In this section a battery of tests to study and demonstrate these irregularities is carried out.

This variability mentioned above is space and time dependent as it is demonstrated below. Besides, the variation of RSS as a function of the transmitted signal strength, the actual placement of the beacon, the distance between the transmitter and the receiver and the type of material that could interfere between transmitter and receiver are also discussed in this section.

C.1.1 Temporal variation of RSS

In static conditions, the receiver, in this case a smartphone, is not moving while taking BLE signal measurements. Beacons are supposed to be always still for each experiment. Therefore, here we intend to study the variability of the RSS while both transmitter and receiver are not moving. In this experiment, three different beacons are used as transmitters and three curves that represent the RSS versus time curves are shown in Figure C.7.

Some statistical values corresponding to these three curves are represented in Table C.1. The RSS measured from three beacons placed in the same position differ their mean RSS value from 200 measurements as well as the deviation of these values. Apart from the outliers, which rarely happen and can be filtered, the signal in static conditions is variable along the time and it could make it hard to implement an ideal RSS-based algorithm.

Therefore, here we face two main RSS-related problems. First, this is not a value that varies smoothly, so that big changes of the RSS values could become changes in the result of the proposed algorithm. On the other hand, instead of working with point values, intervals could be useful to decide, for

Figure C.7: RSS vs time from three independent beacons

	<i>Beacon 1</i>	<i>Beacon 2</i>	<i>Beacon 3</i>
<i>Mean value</i>	-79,8	-83,8	-82,4
<i>Standard deviation</i>	4,13	6,39	5,92
<i>Minimum value</i>	-73	-53	-75
<i>Maximum value</i>	-102	-102	-101

Table C.1: Some statistical values of chart in Figure C.7

instance, that over a certain RSS value, we can ensure to be closer to a certain distance, with a given probability.

The use of a median filter could avoid some peaks in the acquired signals. In a real implementation, it is translated into a smoother curve. Figures C.9 and C.10 show the respective results of processing the original RSS signal using the last 3 and 5 values of the signal from *Beacon 1*, which is represented in Figure C.8. Median value at instant k is calculated through the median value of the current instant and the most recent $m-1$ values, where m is the type of median. This type of filtering smooths signals and eliminates some peaks out of the mean of the trend of the curve.

The minimum value of the RSS chart of the Beacon 1 is $-102dBm$ and the histogram represented in Figure C.11 demonstrates how this value is a clear outlier. $RSS = -92dBm$ could be considered as the practical minimum value but 90% of the measurements are over $RSS = -84dBm$. Then, we can ensure with a 90% of confidence that if $RSS \geq -84dBm$ the distance between the beacon and the smartphone is lower than $2m$.

Figure C.8: RSS vs time from Beacon 1

Figure C.9: Median of the 3 last values from the Beacon 1 curve (Figure C.8)

Figure C.10: Median of the 5 last values from the Beacon 1 curve (Figure C.8)

Figure C.11: Histogram of the beacon 1 RSS values (Figure C.8)

C.1.2 RSS as a function of the distance and the transmission power

The signal that a beacon broadcasts varies not only along the time, but depending on other parameters that are studied below, such as the distance between both the transmitter and the receiver, or the transmission power. The radiation pattern of an Estimote, where the signal propagation is uniform in 360°.

Figure C.12 represents the variation of the RSS as a function of the transmission power. The beacon is 1 meter high and the user is both facing at the beacon and back to the beacon. Since the human body absorbs part of the signal and interrupts the direct ray from the beacon to the smartphone, the average RSS received in the same point in the *back-type* experiment is lower than the average RSS detected when the user is facing the beacon. The difference of the RSS in both cases keeps constant and the difference is mostly below 5dB.

Figure C.12: RSS (dBm) as a function of the transmission power (dBm). Beacon at 1 m high

The same experiment was done with the beacon placed on the ceiling. If the transmission power is modified along each possible value, the mean of the RSS measured values for both front and back positions are presented in Figure C.13.

Figure C.13: RSS (dBm) as a function of the transmission power (dBm). Beacon on the ceiling

The variation of the distance between the beacon and the smartphone produces the effects represented in Figure C.14. For this experiment, the transmission power was setup at $-4dBm$. The variation of the distance was done every 30 cm, from 30 cm to 150 cm.

Figure C.14: RSS (dBm) as a function of the distance (cm) between the beacon and the smartphone

C.1.3 Spatial variation of RSS

Indoors propagation is affected by all the existing multipaths from the transmitter to the receiver as seen above, due to multiple effects such as the position of both the transmitter and the receiver and the parameters of the signal, besides the configuration of the area. Positions changes of any of these two elements are studied below to estimate the variation of the RSS statistics. Since the beacons-based infrastructure is supposed to be installed and in a fixed position, the receiver moves to study the RSS variation as a function of the position.

From the scenario represented in Figure C.15, four positions are studied to discover how the forward and backward movement of the receiver affects the RSS values.

This scenario consists on three areas, two of them are $5m \times 4,2m$ and the smaller one is $5m \times 2,8m$. According to the colors of the beacons represented in Figure C.15, let's call these three areas: *Purple*, *Blue* and *Green* areas.

Lines around each area represent a wall if the line is continuous. On the other hand, dotted lines are no obstacle areas.

Beacons are placed on the ceiling because it tries to simulate a real installation and we considered the ceiling as a strategic placement for this type of beacons. Besides, clear spaces do not allow to place the beacons anywhere except walls and the ceiling, if there are not strategic objects to place them. The ceiling allows a uniform configuration to cover more or less the same area in each zone.

Figures C.16, C.17, C.18 and C.19 represent the RSS values acquired in 200 measurements in the positions 2, 5, 8 and 11, from the beacon B5, all represented in Figure C.15.

Figure C.15: Set to study the spatial variation of RSS

Figure C.16: RSS from beacon B5 in position 2 (see Figure C.15)

Figure C.17: RSS from beacon B5 in position 5 (see Figure C.15)

Figure C.18: RSS from beacon B5 in position 8 (see Figure C.15)

Figure C.19: RSS from beacon B5 in position 11 (see Figure C.15)

These four measurements are taken by a user holding the phone in her right hand, facing at the same direction in four cases. Positions 5 and 8 are closer to the vertical projection of the B5 beacon, while positions 2 and 11 are further this projection. If three zones are differentiated in the test scenario, both positions belonging to the same area than the B5 beacon receive an average RSS higher (about -75dBm) than the measurements obtained from the other two positions out of the B5 beacon area (below -80dBm). It must be said that, from the three areas of the scenario represented in Figure C.15, the *green one*, where the G3 and G5 beacons are and the *blue one*, containing B3 and B5 beacons

On the other hand, if the user turns 180° and new 200 measurements are taken in the same conditions, results are presented in Figures C.20, C.21, C.22 and C.23. In order to differentiate two batteries of measurements in the same position but with different orientation, we use the number for the original orientation and a *b* is added to mean backward orientation (180°).

Figure C.20: RSS from beacon B5 in position 2b (see Figure C.15)

Figure C.21: RSS from beacon B5 in position 5b (see Figure C.15)

Figure C.22: RSS from beacon B5 in position 8b (see Figure C.15)

Figure C.23: RSS from beacon B5 in position 11b (see Figure C.15)

The difference of average in the RSS also exists in these four measurements. Positions in the same area than the placement of the beacon receive a higher RSS than positions out of the actual area. This assert is not always true because we are talking of mean values.

Main conclusion obtained in this scenario is that human body actually affects the RSS. If the distance between the emitter and the transmitter is the same, the orientation of the user when taking the measurements is very influential to the mean value of the RSS.

Again, we can verify how, regardless the mean value if the receiver is placed in the same area than the beacon is higher if compared with the mean value observed from other areas, at same instants, due to the huge variability of this parameter, the difference can be shorter or even reverted. The intention is to correct this effect using a pair of beacons per area and compare the RSS values from all detected beacons.

C.1.4 Smartphone position

Users do not always carry their mobile in the same way. This is, some times they are using it and carrying it on her hand. Other times, they keep it in their pocket or in their bag. Figures C.24, C.25 and C.26 show the difference of the RSS in the same conditions and position but carrying the smartphone in different places. Besides, Figures C.27, C.28 and C.29 demonstrate how the variation of the RSS does not follow any pattern, because if we change the source of the signal, the mean difference of RSS does not maintain any proportion.

Figure C.24: RSS from a beacon placed over the user (P3 in place 14 from Figure C.15) - smartphone on the hand

Figure C.25: RSS from a beacon placed over the user (P3 in place 14 from Figure C.15) - smartphone in the bag

Figure C.26: RSS from a beacon placed over the user (P3 in place 14 from Figure C.15) - smartphone in the pocket

Figure C.27: RSS from a beacon placed over the user (B3 in place 14 from Figure C.15) - smartphone on the hand

Figure C.28: RSS from a beacon placed over the user (B3 in place 14 from Figure C.15) - smartphone in the bag

Figure C.29: RSS from a beacon placed over the user (B3 in place 14 from Figure C.15) - smartphone in the pocket

The actual mean value of every set of measurements conclude that any change in the position of the smartphone also affects the RSS (see Table C.2). It confirms the spatial variation of the RSS. Anyway, there is a coherent point in this results, which is that, from a farther beacon the RSS is lower than that received from a closer one. Besides, the RSS value usually decreases if the smartphone is in the user pocket. These results are logical regardless of the variation between the RSS depending on the position of the smartphone could affect the the estimation of the user position.

<i>Beacon</i>	<i>B3</i>			<i>P3</i>		
<i>Position</i>	<i>Pocket</i>	<i>Hand</i>	<i>Bag</i>	<i>Pocket</i>	<i>Hand</i>	<i>Bag</i>
<i>Mean</i>	-90,6	-83,5	-84,8	-78,7	-75,2	-75
<i>Standard deviation</i>	2,29	4,91	2,96	2,59	4,2	3
<i>Minimum value</i>	-96	-99	-95	-89	-87	-86
<i>Maximum value</i>	-86	-77	-81	-75	-69	-68

Table C.2: Statistic values from two beacons to study the RSS vs. the position of the mobile device

C.1.5 Over the ceiling vs false ceiling

The decision to hide the beacons to the visibility only had esthetical reasons. How this decision affects the actual RSS behavior is presented in Table C.3.

	<i>Position 1</i>		<i>Position 3</i>	
	<i>Hand</i>	<i>Pocket</i>	<i>Hand</i>	<i>Pocket</i>
<i>G5 - on the ceiling</i>	-68,1	-82,7	-83,9	-92,6
<i>G5 - over the ceiling</i>	-75,1	-79,1	-80,8	-89,1
<i>G3 - on the ceiling</i>	-64,2	-77,7	-77,7	-90,3
<i>G3 - over the ceiling</i>	-73,9	-86,1	-84,3	-92,9
<i>B5 - on the ceiling</i>	-80,7	-85	-66,9	-80,3
<i>B5 - over the ceiling</i>	-84,5	-86	-74,8	-86
<i>B3 - on the ceiling</i>	-83,3	-87,3	-70,5	-83,6
<i>B3 - over the ceiling</i>	-82,3	-83,7	-75,7	-85,2

Table C.3: Comparison of RSS values between *on ceiling* and *over the ceiling* beacons configuration

We could think before carrying out this experiment that hiding the beacons over the ceiling would decrease the RSS. However, we can observe that this assert is not true. The material of the ceiling, which has not been studied in this work, does not seem to affect the signal behavior. Furthermore, the placement of the beacon changes if it is placed on or over the ceiling. There is a 180° rotation in the horizontal axis. We have not entered to study the radiation diagram too. So we are not aware of the different orientation of the beacons. So that, it is not possible to create a relation between the RSS if beacons are on the ceiling or over the ceiling.

So far, the behavior of beacons both individually and installed in a test scenario has been discussed. The instability and lack of coherence in the RSS in different situations makes it difficult to design a robust and accurate localization system.

Hereafter, the definition of the design of a series of algorithms to find a simple and low cost and effort beacons-based solution and its deployment in an actual test scenario is discussed. We are not going to consider physical points to localize, but areas. The proposed scenario consists of a set of joint areas, each with a pair of beacons installed, and the intention is to design a

set of algorithms with a robust performance. This robustness requires the algorithms not to swap between zones when localizing, but confirm the user position with any certainty.

Bibliography

- Agrawal, Rakesh and Srikant, Ramakrishnan. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, ICDE '95, pages 3–14, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-8186-6910-1. URL <http://dl.acm.org/citation.cfm?id=645480.655281>.
- Aida, H., Aoki, S., Nakazawa, J., and Tokuda, H. Real-time information distribution at a shopping mall using android phones. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, pages 1923–1928, July 2011. doi: 10.1109/IWCMC.2011.5982828.
- Alavi, B. and Pahlavan, K. Modeling of the toa-based distance measurement error using uwb indoor radio measurements. *Communications Letters, IEEE*, 10(4):275–277, Apr 2006. ISSN 1089-7798. doi: 10.1109/LCOMM.2006.1613745.
- Antoniou, A., Theodoridis, E., Chatzigiannakis, I., and Mylonas, G. Human mobility trace acquisition and social interactions monitoring for business intelligence using smartphones. In *Informatics (PCI), 2012 16th Panhellenic Conference on*, pages 1–6, Oct 2012a. doi: 10.1109/PCi.2012.9.
- Antoniou, Athanasios, Theodoridis, Evangelos, Chatzigiannakis, Ioannis, and Mylonas, Georgios. The future internet. chapter Using

- Future Internet Infrastructure and Smartphones for Mobility Trace Acquisition and Social Interactions Monitoring, pages 117–129. Springer-Verlag, Berlin, Heidelberg, 2012b. ISBN 978-3-642-30240-4. URL <http://dl.acm.org/citation.cfm?id=2340856.2340870>.
- Aparicio, S., Perez, J., Bernardos, A.M., and Casar, J.R. A fusion method based on bluetooth and wlan technologies for indoor location. In *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pages 487–491, Aug 2008. doi: 10.1109/MFI.2008.4648042.
- Argotte, L., Mejia-Lavalle, M., and Sosa, R. Business intelligence and energy markets: A survey. In *Intelligent System Applications to Power Systems, 2009. ISAP '09. 15th International Conference on*, pages 1–6, Nov 2009. doi: 10.1109/ISAP.2009.5352918.
- Aung, Htoo Htet, Guo, Long, and Tan, Kian-Lee. Mining sub-trajectory cliques to find frequent routes. In *Proceedings of the 13th International Symposium on Advances in Spatial and Temporal Databases - Volume 8098, SSTD 2013*, pages 92–109, New York, NY, USA, 2013. Springer-Verlag New York, Inc. ISBN 978-3-642-40234-0.
- Baniukevic, A., Sabonis, D., Jensen, Christian S., and Lu, Hua. Improving wi-fi based indoor positioning using bluetooth add-ons. In *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*, volume 1, pages 246–255, June 2011. doi: 10.1109/MDM.2011.50.
- Baniukevic, A., Jensen, C.S., and Lu, Hua. Hybrid indoor positioning with wi-fi and bluetooth: Architecture and performance. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, volume 1, pages 207–216, June 2013. doi: 10.1109/MDM.2013.30.
- Bartone, C. and van Graas, F. Ranging airport pseudolite for local area augmentation. In *Position Location and Navigation Symposium, IEEE 1998*, pages 479–486, Apr 1998. doi: 10.1109/PLANS.1998.670202.
- Bernardos, A. M., Casar, J. R., and Tarrío, P. Real time calibration for rss indoor positioning systems. In *2010 International Conference on Indoor Positioning and Indoor Navigation*, pages 1–7, Sept 2010. doi: 10.1109/IPIN.2010.5648231.
- Besada, J. A., Bernardos, A. M., Tarrío, P., and Casar, J. R. Analysis of tracking methods for wireless indoor localization. In *2007 2nd International Symposium on Wireless Pervasive Computing*, Feb 2007. doi: 10.1109/ISWPC.2007.342653.

- Betz, Frederick. *Managing Technological Innovation: Competitive Advantage from Change*. Hoboken, NJ: John Wiley, 2011. doi: 10.1002/9780470927564.
- Blog, The Estimote Team. How to extend Estimote Beacon battery life. <http://blog.estimote.com/post/83618039493/how-to-extend-estimote-beacon-battery-life>, 2016. [Online; accessed 06-August-2016].
- Bostanipour, B. and Garbinato, B. Improving neighbor detection for proximity-based mobile applications. In *Network Computing and Applications (NCA), 2013 12th IEEE International Symposium on*, pages 177–182, Aug 2013. doi: 10.1109/NCA.2013.43.
- Breiman, L., Friedman, J., Olshen, R., and Stone, C. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.
- Capgemini, 2014.
- Castano, J.G., Svensson, M., and Ekstrom, M. Local positioning for wireless sensors based on bluetooth trade;. In *Radio and Wireless Conference, 2004 IEEE*, pages 195–198, Sept 2004. doi: 10.1109/RAWCON.2004.1389106.
- Chang, Yao-Tang, Wu, Chi-Lin, and Cheng, Hsu-Chih. Integrated cross-correlation and genetic algorithm to improve tdoa-based locating system in preventing radio interference. In *Computer, Consumer and Control (IS3C), 2014 International Symposium on*, pages 1078–1081, June 2014. doi: 10.1109/IS3C.2014.281.
- Chen, Hsinchun, Chiang, Roger H. L., and Storey, Veda C. Business intelligence and analytics: From big data to big impact. *MIS Q.*, 36(4):1165–1188, December 2012. ISSN 0276-7783. URL <http://dl.acm.org/citation.cfm?id=2481674.2481683>.
- Chernov, N. and Lesort, C. Least squares fitting of circles. *Journal of Mathematical Imaging and Vision*, 23(3):239–252, 2005. ISSN 1573-7683. doi: 10.1007/s10851-005-0482-8. URL <http://dx.doi.org/10.1007/s10851-005-0482-8>.
- Corredor, Iván, Metola, Eduardo, Bernardos, Ana M, Tarrío, Paula, and Casar, José R. A lightweight web of things open platform to facilitate context data management and personalized healthcare services creation. *International journal of environmental research and public health*, 11(5):4676–4713, 2014.
- Cortes, Corinna and Vapnik, Vladimir. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

ISSN 0885-6125. doi: 10.1023/A:1022627411411. URL
<http://dx.doi.org/10.1023/A:1022627411411>.

Daryaei, M., Shirzad, M., and Kumar, V. Adoption of business intelligence: In hotel industry. In *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on*, pages 1–6, July 2013. doi: 10.1109/ICCCNT.2013.6726506.

Dashti, M., Ghorraishi, M., Haneda, K., and Takada, J.-i. Sources of toa estimation error in los scenario. In *Ultra-Wideband (ICUWB), 2010 IEEE International Conference on*, volume 2, pages 1–4, Sept 2010. doi: 10.1109/ICUWB.2010.5614046.

Duan, Lian and Xu, Li Da. Business intelligence for enterprise systems: A survey. *Industrial Informatics, IEEE Transactions on*, 8(3):679–687, Aug 2012. ISSN 1551-3203. doi: 10.1109/TII.2012.2188804.

Dulmage, J., Cioffi, R., Fitz, M.P., and Cabric, D. Characterization of distance error with received signal strength ranging. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, pages 1–6, April 2010. doi: 10.1109/WCNC.2010.5506590.

El-Kafrawy, K., Youssef, M., El-Keyi, A., and Naguib, A. Propagation modeling for accurate indoor wlan rss-based localization. In *Vehicular Technology Conference Fall (VTC 2010-Fall), 2010 IEEE 72nd*, pages 1–5, Sept 2010. doi: 10.1109/VETECF.2010.5594108.

Forno, F., Malnati, G., and Portelli, G. Design and implementation of a bluetooth ad hoc network for indoor positioning. *Software, IEE Proceedings* -, 152(5):223–228, Oct 2005. ISSN 1462-5970. doi: 10.1049/ip-sen:20045027.

G., Petro. How Proximity Marketing Is Driving Retail Sales. <http://www.forbes.com/sites/gregpetro/2014/10/08/how-proximity-marketing-is-driving-retail-sales/>, 2016. [Online; accessed 06-August-2016].

Gan, Mingming, Meissner, P., Mani, F., Leitinger, E., Frohle, M., Oestges, C., Witrisal, K., and Zemen, T. Calibration of indoor uwb sub-band divided ray tracing using multiobjective simulated annealing. In *Communications (ICC), 2014 IEEE International Conference on*, pages 4844–4849, June 2014. doi: 10.1109/ICC.2014.6884087.

Gao, Xiaoneng and Zhong, Jianwei. Development of application system for mobile location-based services based on tdoa technology. In *Information*

- Science and Engineering (ICISE)*, 2010 2nd International Conference on, pages 3644–3647, Dec 2010. doi: 10.1109/ICISE.2010.5689418.
- Geng, Yishuang, He, Jie, Deng, Haokun, and Pahlavan, K. Modeling the effect of human body on toa ranging for indoor human tracking with wrist mounted sensor. In *Wireless Personal Multimedia Communications (WPMC)*, 2013 16th International Symposium on, pages 1–6, June 2013.
- Giannotti, Fosca, Nanni, Mirco, Pinelli, Fabio, and Pedreschi, Dino. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 330–339, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-609-7.
- Gil, J, Tobari, E, Lemlij, M, Rose, A, and Penn, AR. The differentiating behaviour of shoppers: Clustering of individual movement traces in a supermarket. In *Proceedings of the 7th International Space Syntax Symposium*, pages 036:1–036:13, Jun 2009.
- Gu, Yanying, Lo, A., and Niemegeers, I. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys Tutorials*, 11 (1):13–32, 2009. ISSN 1553-877X. doi: 10.1109/SURV.2009.090103.
- Hallberg, J., Nilsson, M., and Synnes, K. Positioning with bluetooth. In *Telecommunications, 2003. ICT 2003. 10th International Conference on*, volume 2, pages 954–958 vol.2, Feb 2003. doi: 10.1109/ICTEL.2003.1191568.
- Hans, R.T. and Mnkandla, E. Modeling software engineering projects as a business: A business intelligence perspective. In *AFRICON, 2013*, pages 1–5, Sept 2013. doi: 10.1109/AFRCON.2013.6757849.
- Hinton, G. E. and Sejnowski, T. J. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning and Relearning in Boltzmann Machines, pages 282–317. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X. URL <http://dl.acm.org/citation.cfm?id=104279.104291>.
- Hinton, Geoffrey E. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800, August 2002. ISSN 0899-7667. doi: 10.1162/089976602760128018. URL <http://dx.doi.org/10.1162/089976602760128018>.
- Hinton, Geoffrey E., Osindero, Simon, and Teh, Yee-Whye. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006. ISSN 0899-7667. doi: 10.1162/neco.2006.18.7.1527. URL <http://dx.doi.org/10.1162/neco.2006.18.7.1527>.

- Honkavirta, V., Perala, T., Ali-Loytty, S., and Piche, R. A comparative survey of wlan location fingerprinting methods. In *Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on*, pages 243–251, March 2009. doi: 10.1109/WPNC.2009.4907834.
- Hu, Weiming, Tan, Tieniu, Wang, Liang, and Maybank, S. A survey on visual surveillance of object motion and behaviors. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(3):334–352, Aug 2004. ISSN 1094-6977. doi: 10.1109/TSMCC.2004.829274.
- Huang, Zhengyong, Xia, Jun, Yu, Hui, Guan, Yunfeng, Gan, Xiaoying, and Liu, Jing. Fusing fixed and hint landmarks on crowd paths for automatically constructing wi-fi fingerprint database. *Communications, China*, 12(1): 11–24, Jan 2015. ISSN 1673-5447. doi: 10.1109/CC.2015.7084380.
- Hui, Sam K., Fader, Peter S., and Bradlow, Eric T. Path data in marketing: An integrative framework and prospectus for model building. *Marketing Science*, 28(2):pp. 320–335, 2009. ISSN 07322399. URL <http://www.jstor.org/stable/23884266>.
- James, R.D., Sampan, S., Mendola, J.B., and Mizusawa, G.A. Neural network techniques for multistatic hyperbolic vehicle positioning. In *Vehicle Navigation and Information Systems Conference, 1996. VNIS '96*, volume 7, pages 94–102, Oct 1996. doi: 10.1109/VNIS.1996.1623738.
- Kaemarungsi, K. and Krishnamurthy, P. Properties of indoor received signal strength for WLAN location fingerprinting. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004*, pages 14–23, August 2004. doi: 10.1109/MOBIQ.2004.1331706.
- Kanaris, L., Kokkinis, A., Raspopoulos, M., Liotta, A., and Stavrou, S. Improving rss fingerprint-based localization using directional antennas. In *Antennas and Propagation (EuCAP), 2014 8th European Conference on*, pages 1593–1597, April 2014. doi: 10.1109/EuCAP.2014.6902090.
- Khare, Arpita. Influence of mall attributes and demographics on indian consumers’ mall involvement behavior: An exploratory study. *Journal of Targeting, Measurement & Analysis for Marketing*, 20:192–202, Sep 2012. ISSN 0967-3237. doi: 10.1109/TIE.2012.2230596.
- Kim, Doyoon, Park, Byungwoon, Lee, Sanghyo, Cho, Am, Kim, Jihoon, and Kee, Changdon. Design of efficient navigation message format for uav pseudolite navigation system. *Aerospace and Electronic Systems*,

- IEEE Transactionson*, 44(4):1342–1355, Oct 2008. ISSN 0018-9251. doi: 10.1109/TAES.2008.4667713.
- Kjaergaard, M.B. Cleaning and processing rss measurements for location fingerprinting. In *Autonomic and Autonomous Systems, 2007. ICAS07. Third International Conference on*, pages 12–12, June 2007. doi: 10.1109/CONIELECOMP.2007.64.
- Kohavi, Ron and Quinlan, Ross. Decision tree discovery. In *IN HANDBOOK OF DATA MINING AND KNOWLEDGE DISCOVERY*, pages 267–276. University Press, 1999.
- Koweerawong, C., Wipusitwarakun, K., and Kaemarungsi, K. Indoor localization improvement via adaptive rss fingerprinting database. In *Information Networking (ICOIN), 2013 International Conference on*, pages 412–416, Jan 2013. doi: 10.1109/ICOIN.2013.6496414.
- Köbler, F., Koene, P., Krcmar, H., Altmann, M., and Leimeister, J.M. Locatag - an nfc-based system enhancing instant messaging tools with real-time user location. In *Near Field Communication (NFC), 2010 Second International Workshop on*, pages 57–61, April 2010. doi: 10.1109/NFC.2010.20.
- Larson, Jeffrey S., Bradlow, Eric T., Fader, Peter S., Frances, Peter S. Fader Is, yuan Chia, Pei, and The, Professor Marketing. An exploratory look at supermarket shopping paths. *Internat. J. Res. Marketing*, pages 395–414, 2005.
- Learned-Miller, Erik G. Supervised Learning and Bayesian Classification. Technical report, September 2011. URL <https://people.cs.umass.edu/elm/Teaching/Docs/supervised.pdf>.
- Lee, Chunhan, Chang, Yushin, Park, Gunhong, Ryu, Jaeheon, Jeong, Seung-Gweon, Park, Seokhyun, Park, Jae Whe, Lee, Hee Chang, Hong, Keum-Shik, and Lee, Man Hyung. Indoor positioning system based on incident angles of infrared emitters. In *Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE*, volume 3, pages 2218–2222 Vol. 3, Nov 2004. doi: 10.1109/IECON.2004.1432143.
- Lee, In-Chul Jung; Young S. Kwon; Yung-Seop. A sequence pattern matching approach to shopping path clustering. In *Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management*, pages 2364–2372, Jul 2012.

- Li, Xinrong. Rss-based location estimation with unknown pathloss model. *Wireless Communications, IEEE Transactions on*, 5(12):3626–3633, December 2006. ISSN 1536-1276. doi: 10.1109/TWC.2006.256985.
- Liang, P. C. and Krause, P. Smartphone-based real-time indoor location tracking with 1-m precision. *IEEE Journal of Biomedical and Health Informatics*, 20(3):756–762, May 2016. ISSN 2168-2194. doi: 10.1109/JBHI.2015.2500439.
- Liu, Congfeng, Yang, Jie, and Wang, Fengshuai. Joint tdoa and aoa location algorithm. *Systems Engineering and Electronics, Journal of*, 24(2):183–188, April 2013. doi: 10.1109/JSEE.2013.00023.
- Liu, Hui, Darabi, H., Banerjee, P., and Liu, Jing. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, Nov 2007. ISSN 1094-6977. doi: 10.1109/TSMCC.2007.905750.
- Liu, Shu, Jiang, Yingxin, and Striegel, A. Face-to-face proximity estimation using bluetooth on smartphones. *Mobile Computing, IEEE Transactions on*, 13(4):811–823, April 2014. ISSN 1536-1233. doi: 10.1109/TMC.2013.44.
- Liu, Yanxia, Li, Shufen, and Cao, Liting. Application of bluetooth communication in digital photo frame. In *Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on*, volume 4, pages 370–373, Aug 2009. doi: 10.1109/CCCM.2009.5270428.
- Madhani, P.H., Axelrad, P., Krumvieda, K., and Thomas, J. Application of successive interference cancellation to the gps pseudolite near-far problem. *Aerospace and Electronic Systems, IEEE Transactions on*, 39(2):481–488, April 2003. ISSN 0018-9251. doi: 10.1109/TAES.2003.1207260.
- Mahtab Hossain, A.K.M., Jin, Yunye, Soh, Wee-Seng, and Van, Hien Nguyen. Ssd: A robust rf location fingerprint addressing mobile devices’ heterogeneity. *Mobile Computing, IEEE Transactions on*, 12(1):65–77, Jan 2013. ISSN 1536-1233. doi: 10.1109/TMC.2011.243.
- McGuire, M., Plataniotis, K.N., and Venetsanopoulos, A.N. Data fusion of power and time measurements for mobile terminal location. *Mobile Computing, IEEE Transactions on*, 4(2):142–153, March 2005. ISSN 1536-1233. doi: 10.1109/TMC.2005.24.

- Metola, Eduardo, Aparicio, Sofía, Tarrío, Paula, and Casar, José R. Comparison of localization methods using calibrated and simulated fingerprints for indoor systems based on bluetooth and wlan technologies. *Proc. MADRINET*, 10, 2009.
- Metola, Eduardo, Bernardos, Ana M, Martín, Henar, and Casar, José Ramón. A fusion component for location management in mobile. 2010.
- Moghtadaiee, V. and Dempster, A.G. Wifi fingerprinting signal strength error modeling for short distances. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pages 1–6, Nov 2012. doi: 10.1109/IPIN.2012.6418852.
- Nandwani, A., Edwards, R., and Coulton, P. Contactless check-ins using implied locations: A nfc solution simplifying business to consumer interaction in location based services. In *Electronics Design, Systems and Applications (ICEDSA), 2012 IEEE International Conference on*, pages 39–44, Nov 2012. doi: 10.1109/ICEDSA.2012.6507812.
- Nascimento, J.C., Marques, J.S., and Figueiredo, M.A.T. Classification of complex pedestrian activities from trajectories. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 3481–3484, Sept 2010. doi: 10.1109/ICIP.2010.5650138.
- Ni, Bingbing, Pei, Yong, Moulin, P., and Yan, Shuicheng. Multilevel depth and image fusion for human activity detection. *Cybernetics, IEEE Transactions on*, 43(5):1383–1394, Oct 2013. ISSN 2168-2267. doi: 10.1109/TCYB.2013.2276433.
- Oshin, T.O. and Poslad, S. Lals: A low power accelerometer assisted location sensing technique for smartphones. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 127–133, Dec 2013. doi: 10.1109/GLOCOM.2013.6831059.
- Ozeki, S., Sumiya, Y., and Shirakawa, M. Aoa measurement error under the antenna couplings. In *TENCON '96. Proceedings., 1996 IEEE TENCON. Digital Signal Processing Applications*, volume 2, pages 804–809 vol.2, Nov 1996. doi: 10.1109/TENCON.1996.608449.
- Prieto, J.C., Jimenez, A.R., Guevara, J., Ealo, J.L., Seco, F., Roa, J.O., and Ramos, F. Performance evaluation of 3d-locus advanced acoustic lps. *Instrumentation and Measurement, IEEE Transactions on*, 58(8):2385–2395, Aug 2009. ISSN 0018-9456. doi: 10.1109/TIM.2009.2016378.

- Priyantha, Nissanka B., Chakraborty, Anit, and Balakrishnan, Hari. The cricket location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 32–43, New York, NY, USA, 2000. ACM. ISBN 1-58113-197-6. doi: 10.1145/345910.345917. URL <http://doi.acm.org/10.1145/345910.345917>.
- Priyantha, Nissanka B., Miu, Allen K.L., Balakrishnan, Hari, and Teller, Seth. The cricket compass for context-aware mobile applications. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, pages 1–14, New York, NY, USA, 2001. ACM. ISBN 1-58113-422-3. doi: 10.1145/381677.381679. URL <http://doi.acm.org/10.1145/381677.381679>.
- Quinlan, J. Ross. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0.
- Rhee, Hongjai and Bell, David R. The inter-store mobility of supermarket shoppers. *Journal of Retailing*, 78(4):225 – 237, 2002. ISSN 0022-4359. doi: [http://dx.doi.org/10.1016/S0022-4359\(02\)00099-4](http://dx.doi.org/10.1016/S0022-4359(02)00099-4). URL <http://www.sciencedirect.com/science/article/pii/S0022435902000994>.
- Rojas, Raúl. *Neural Networks: A Systematic Introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996. ISBN 3-540-60505-3.
- Safavian, S. R. and Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674, May 1991. ISSN 0018-9472. doi: 10.1109/21.97458.
- S.B. Hiregoudar, K.S.Patil, K. Manjunath. A survey: research summary on neural networks. *International Journal of Research in Engineering and Technology*, 3:385–389, May 2014.
- Schmidl, T.M. and Cox, D.C. Robust frequency and timing synchronization for ofdm. *Communications, IEEE Transactions on*, 45(12):1613–1621, Dec 1997. ISSN 0090-6778. doi: 10.1109/26.650240.
- Schroth, G., Huitl, R., Chen, D., Abu-Alqumsan, M., Al-Nuaimi, A., and Steinbach, E. Mobile visual location recognition. *Signal Processing Magazine, IEEE*, 28(4):77–89, July 2011. ISSN 1053-5888. doi: 10.1109/MSP.2011.940882.

- Shin, Beomju, Lee, Jung Ho, Lee, Taikjin, and Kim, Hyung Seok. Enhanced weighted k-nearest neighbor algorithm for indoor wi-fi positioning systems. In *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*, volume 2, pages 574–577, April 2012.
- Siekkinen, M., Hienkari, M., Nurminen, J.K., and Nieminen, J. How low energy is bluetooth low energy? comparative measurements with zigbee/802.15.4. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*, pages 232–237, April 2012. doi: 10.1109/WCNCW.2012.6215496.
- Siira, E., Tuikka, T., and Tormanen, V. Location-based mobile wiki using nfc tag infrastructure. In *Near Field Communication, 2009. NFC '09. First International Workshop on*, pages 56–60, Feb 2009. doi: 10.1109/NFC.2009.8.
- Soloviev, A. and Dickman, J. Extending gps carrier phase availability indoors with a deeply integrated receiver architecture. *Wireless Communications, IEEE*, 18(2):36–44, April 2011. ISSN 1536-1284. doi: 10.1109/MWC.2011.5751294.
- Srikant, Ramakrishnan and Agrawal, Rakesh. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '96, pages 3–17, London, UK, UK, 1996. Springer-Verlag. ISBN 3-540-61057-X. URL <http://dl.acm.org/citation.cfm?id=645337.650382>.
- Tarrio, P., Bernardos, A. M., and Casar, J. R. An rss localization method based on parametric channel models. In *2007 International Conference on Sensor Technologies and Applications (SENSORCOMM 2007)*, pages 265–270, Oct 2007. doi: 10.1109/SENSORCOMM.2007.4394932.
- Tarrio, P., Bernardos, A. M., Besada, J. A., and Casar, J. R. A new positioning technique for rss-based localization based on a weighted least squares estimator. In *2008 IEEE International Symposium on Wireless Communication Systems*, pages 633–637, Oct 2008. doi: 10.1109/ISWCS.2008.4726133.
- Tarrio, P. *Efficient localization methods for wireless sensor networks based on received signal strength measurements*. PhD thesis, Universidad Politécnica de Madrid, 2011.

- Torteeke, P. and Chundi, Xiu. Indoor positioning based on wi-fi fingerprint technique using fuzzy k-nearest neighbor. In *Applied Sciences and Technology (IBCAST), 2014 11th International Bhurban Conference on*, pages 461–465, Jan 2014. doi: 10.1109/IBCAST.2014.6778188.
- Tran, Quang, Tantra, J.W., Foh, Chuan Heng, Tan, Ah-Hwee, Yow, Kin Choong, and Qiu, Dongyu. Wireless indoor positioning system with enhanced nearest neighbors in signal space algorithm. In *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, pages 1–5, Sept 2006. doi: 10.1109/VTCF.2006.489.
- Tyler, N. Cdma mobile station location. In *Novel Methods of Location and Tracking of Cellular Mobiles and Their System Applications (Ref. No. 1999/046), IEE Colloquium on*, pages 5/1–5/6, 1999. doi: 10.1049/ic:19990242.
- van Diggelen, F. Indoor gps theory implementation. In *Position Location and Navigation Symposium, 2002 IEEE*, pages 240–247, 2002. doi: 10.1109/PLANS.2002.998914.
- Wang, Lei and Wong, Wai-Choong. A novel rss model and power-bias mitigation algorithm in fingerprinting-based indoor localization in wireless local area networks. In *European Wireless 2014; 20th European Wireless Conference; Proceedings of*, pages 1–7, May 2014.
- Wang, Y, Yang, X., Zhao, Y., Liu, Y., and Cuthbert, L. Bluetooth positioning using rssi and triangulation methods. In *Consumer Communications and Networking Conference (CCNC), 2013 IEEE*, pages 837–842, Jan 2013. doi: 10.1109/CCNC.2013.6488558.
- Want, Roy, Hopper, Andy, Falcão, Veronica, and Gibbons, Jonathan. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, January 1992. ISSN 1046-8188. doi: 10.1145/128756.128759. URL <http://doi.acm.org/10.1145/128756.128759>.
- Wilk, P. and Karciarz, J. Optimization of map matching algorithms for indoor navigation in shopping malls. In *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference on*, pages 661–669, Oct 2014. doi: 10.1109/IPIN.2014.7275541.
- Witten, Ian H., Frank, Eibe, Trigg, Len, Hall, Mark, Holmes, Geoffrey, and Cunningham, Sally Jo. Weka: Practical machine learning tools and techniques with java implementations, 1999.

- Xiao, Zhuoling, Wen, Hongkai, Markham, A., Trigoni, N., Blunsom, P., and Frolik, J. Non-line-of-sight identification and mitigation using received signal strength. *Wireless Communications, IEEE Transactions on*, 14(3): 1689–1702, March 2015. ISSN 1536-1276. doi: 10.1109/TWC.2014.2372341.
- Xiao-qiang, Xing. Investment under uncertainty in new technology with time-to-build in perspective of learning. In *Management Science and Engineering, 2009. ICMSE 2009. International Conference on*, pages 1354–1360, Sept 2009. doi: 10.1109/ICMSE.2009.5318045.
- Xin-di, Liu, Wei, He, and Zeng-shan, Tian. The improvement of rss-based location fingerprint technology for cellular networks. In *Computer Science Service System (CSSS), 2012 International Conference on*, pages 1267–1270, Aug 2012. doi: 10.1109/CSSS.2012.321.
- Xu, Lisheng, Yang, Feifei, Jiang, Yuqi, Zhang, Lei, Feng, Cong, and Bao, Nan. Variation of received signal strength in wireless sensor network. In *Advanced Computer Control (ICACC), 2011 3rd International Conference on*, pages 151–154, Jan 2011. doi: 10.1109/ICACC.2011.6016387.
- Yang, Chouchang and rong Shao, Huai. Wifi-based indoor positioning. *Communications Magazine, IEEE*, 53(3):150–157, March 2015. ISSN 0163-6804. doi: 10.1109/MCOM.2015.7060497.
- Yang, K., An, J., Xu, Z., and Bu, X. A generalized total least-squares algorithm for hyperbolic location. In *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–4, Oct 2008. doi: 10.1109/WiCom.2008.822.
- Yang, P. Rfid-loc: A general experimental investigation strategy for high accuracy and precision in passive rfid location systems. *Internet of Things Journal, IEEE*, PP(99):1–1, 2014. ISSN 2327-4662. doi: 10.1109/JIOT.2014.2377351.
- Yang, Po, Wu, Wenyan, Moniri, M., and Chibelushi, C.C. Efficient object localization using sparsely distributed passive rfid tags. *Industrial Electronics, IEEE Transactions on*, 60(12):5914–5924, Dec 2013. ISSN 0278-0046. doi: 10.1109/TIE.2012.2230596.
- Yin, Jian, Yin, Ye, Wang, Jianwei, Wu, Xiangnong, and Li, Chuanjiang. Rfid-based real time location and safety management system. In *Wireless Mobile and Computing (CCWMC 2011), IET International Communication Conference on*, pages 85–88, Nov 2011. doi: 10.1049/cp.2011.0852.

- Yoshimura, Yuji, Sobolevsky, Stanislav, Ratti, Carlo, Girardin, Fabien, Carrascal, Juan Pablo, Blat, Josep, and Sinatra, Roberta. An analysis of visitors' behavior in the louvre museum: A study using bluetooth data. *Environment and Planning B: Planning and Design*, 41(6):1113–1131, 2014. doi: 10.1068/b130047p. URL <http://epb.sagepub.com/content/41/6/1113.abstract>.
- You, Chuang-Wen, Wei, Chih-Chiang, Chen, Yi-Ling, Chu, Hao-Hua, and Chen, Ming-Syan. Using mobile phones to monitor shopping time at physical stores. *Pervasive Computing, IEEE*, 10(2):37–43, April 2011. ISSN 1536-1268. doi: 10.1109/MPRV.2011.14.
- Zancanaro, Massimo, Kuflik, Tsvi, Boger, Zvi, Goren-Bar, Dina, and Goldwasser, Dan. Analyzing museum visitors' behavior patterns. In *User Modeling 2007*, volume 4511 of *Lecture Notes in Computer Science*, pages 238–246. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-73077-4. doi: 10.1007/978-3-540-73078-1_27.
- Zhou, Sheng and Pollard, J.K. Position measurement using bluetooth. *Consumer Electronics, IEEE Transactions on*, 52(2):555–558, May 2006. ISSN 0098-3063. doi: 10.1109/TCE.2006.1649679.