Contents

List of Figures

List of Tables

Listings

Sommario

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Parole chiave: PoliMi, Tesi, LaTeX, Scribd

Abstract

Text of the abstract in english...
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Keywords: PoliMi, Master Thesis, LaTeX, Scribd

Introduction

On average, people spend approximately 70% of their time indoors [23], such as in offices, schools, and at home. In the last years, the advent of sensors networks have enabled buildings to be retrofit for improved monitoring and automation. In particular, the use of technology is intended to increment the occupants comfort, their safety and the energy efficiency of the indoor environments. Smart Buildings are becoming a reality with the adoption of Building Management Systems (BMS), i.e. control systems that integrate the underlying monitoring infrastructure with a logic decision engine. These decisions are usually based on administrator policies or user-defined rules, other than more complex machine learning algorithms. In this context, identifying in real-time the number of occupants in the building, their location and even the identity of each one can be essential to provide the building behavior logic. For instance, Heating, Ventilation and Air Conditioning (HVAC) systems can be modulated according to the number of occupants, increasing thermal comfort while avoiding energy wastes. The problem of knowing in every instant the location of users in a building, known as the *Occupancy Detection* problem, is one of the two main targets of this work.

In order to solve the occupancy detection problem, as well as other smart building systems like thermal comfort or emergency management, it's common to employ several connected devices with sensor, actuator or network capabilities. During the testing phase of our system, but also trying to verify existing approaches from the state of the art, we experienced the difficulty of tedious, error-prone and time consuming tasks like the correct allocation of devices in the environment. During the installation of such systems, the choice of the nodes type and quantity, the proper placement of each one heavily affect the accuracy and the cost of the system. The lack of supporting tools for the design of indoor monitoring systems induced us to develop a CAD based application tailored for the purpose. The development of the mentioned application comprises the second main part of this thesis.

0.1 Motivations

Energy sustainability is one of the key target of smart (or green) buildings. From 1980 to 2010, energy in the United States has increased 53% [12]. In 2010, heating, ventilation, and air-conditioning (HVAC) consumed 42% of the energy used in the United States.

There are two different parts of HVAC systems usage that should be affected by occupancy; temperature and ventilation. Temperature control is dependent on whether a space is occupied. However, indoor environments are often conditioned assuming a constant occupancy. This leads to heating or cooling empty rooms. One method of reducing consumption is to condition based on the actual binary state of occupancy. Ventilation instead, used to control CO2 levels and air pollutants, depends on the number of people occupying a space. Also in this case, spaces that are only partially occupied could be over-ventilated, leading to loss of conditioned air and thus thermal energy. For this purpose, standards have been issued to recommend ventilation for acceptable indoor air quality depending on occupation (ASHRAE standard 62.1 [3]). The usage of energy-hungry appliances like HVAC systems can be improved not only by the real-time occupancy information, but also from occupancy probability profiles built on top of historical data. For instance, the information that in a certain office, every Friday the last employee leaves the room at 5 p.m. with a probability of 90% can be exploited to reduce energy consumption.

Another purpose of smart buildings is safety. Automated systems that respond to harmful situations limiting damages and victims have been studied and adopted in commercial and private buildings since many decades. The primary job of these systems is to detect crisis events and provide a suitable response. The ability to detect the position of users inside the building can be crucial to assist evacuations or rescue operations. For example, the safety system could trigger alarm notifications tailored on user's position; escape routes and emergency exits might be highlighted with labels and lighting panels.

Occupancy detection can be exploited to build indoor location data analytics, that measure time spent by visitors in particular locations and which routes were used. This enables shopping stores find out how many people pass a certain zone during the day or which hours of the week are the least busy. Location monitoring allows shopping malls to provide rewards, discounts and targeted advertisement to their customers. At the same way in which web analytics are used by web managers to optimize their pages, indoor location data analytics can be exploited by store owners

to increase profitability of their business.

0.2 Challenges

The problem of locate users in indoor environments is intrinsically hard, in both the technological and the user interaction sides. For outdoor applications the use of GPS has revealed to be the best solution in terms of cost, accuracy and ease of use, leading to its widespread popularity. For indoor applications, the lack of localization systems in modern buildings is caused exactly by the nonexistence of a low-cost technological solution with sufficient accuracy and good usability. In addition, the specific occupancy detection problem poses additional challenges to the generic Indoor Positioning Systems (IPS). The distinction between occupancy monitoring systems and traditional IPSs will be explained in detail in section 1.3.7, but intuitively the key difference relies in the execution time. Occupancy monitoring requires to monitor the users for many hours of the day, while the execution of an IPS is typically limited to the orientation or navigation task of the user.

First of all, long execution time leads to a huge **battery consumption** problem for solutions that involve mobile devices in the localization process. Localization applications are typically very battery hungry, and whereas this can be accepted for short usage, it becomes unfeasible for many hours of execution.

Another issue derived from the long monitoring is the **intrusiveness** on the occupants activity. Occupancy detection systems based on images coming from cameras can raise privacy issue, even in working spaces like offices or meeting rooms. Solutions that relies on the assumption that the occupants are always accompanied by a tag, a smartphone or a wearable device are also intrusive in the sense that a certain level of effort is required to the user. This level of effort can be perceived differently depending on the situations. At work, where the user is used to guarantee phone call and email availability, the effort can be nearly zero; at home instead, where some users are used to empty their pockets as soon as they are inside, the effort can be high. Again, battery consumption affect also the system intrusiveness. A system that cause a considerable increase in battery drain directly afflict the user comfort and peace of mind, and can bring the user to leave the application.

A challenging requirement of some occupancy detection systems, including the proposed one, is to provide the occupancy information in **real-time**. In order to be considered real-time, the location data should be received

from the system back-end with a maximum delay of 2-3 seconds, otherwise the automation will not be perceived as useful or responsive. This means that ambient characteristics like temperature, humidity, light, CO2 and so on, that are related to occupancy, can't be easily exploited for real-time detection due to their slow time response.

Other issues that discourage that adoption of occupancy monitoring systems are in common with traditional IPSs. When the system is composed by transmitting nodes or sensor networks, the **cost** of the hardware purchase and the **installation** of appliances can be onerous. Even for small buildings, the choice of the nodes quantity and a correct placement able to guarantee space coverage can be cumbersome. Adjust and move devices to tune the system performances is time-consuming and sometimes unfeasible. To address this specific problem, we developed a CAD application for the design of smart building systems based on the installation of hardware nodes across the indoor space. The tool, that provide cost-effective deployment of wireless localization systems, will be described in chapter 3.

0.3 Scope and Contributions

This thesis is about occupancy detection and monitoring in smart buildings. The work done in this thesis can be summarized in two main parts.

First, an occupancy monitoring system based on Bluetooth Low Energy (BLE) is proposed. the key aspects of the proposed system are: **(1)** extremely low battery consumption allows the system run continuously for many hours; **(2)** real-time occupancy detection; **(3)** non-intrusive system from the user point of view; **(4)** simplified installation and training.

The second part of this thesis is about design automation for smart building systems based on sensor and network devices. The rationale of the developed application comes from a real-world necessity that we experienced during the arrangement of wireless indoor localization systems. The key contributions of the proposed application with respect to other existing approaches are: **(1)** a CAD interface to specify building floor-plan and functional components of the smart environment; **(2)** an algorithm for hardware nodes allocation that provides to designers a cost-effective placement of devices; **(3)** a site-specific model for wireless indoor localization accuracy optimization that keeps into account the actual structure of the building; **(4)** The integration of the tool within an open-source

application framework able to extend the system by means of JavaScript or C++ plug-ins.

0.4 Document Organization

The thesis is structured as follows:

The first chapter explains and investigates the occupancy monitoring problem for smart buildings. The state of the art is analyzed reporting the most relevant existing approaches grouped by their respective enabling technologies. A particular attention is directed to wireless-based indoor positioning systems, how they have been exploited for occupancy monitoring and their limitations in this specific context. In this part are also reported some concepts of WiFi and Bluetooth Low Energy protocols useful to understand the limitations of existing approaches and the design choices of the adopted solution.

The second chapter describes the proposed BLE occupancy monitoring system. First, the feasibility study is reported, explaining also the reasons of the adopted design and the technology employed. Then will be illustrated the architecture of the system, passing through application, sensing and back-end layers. On the application side, will be discussed the potentiality and limitations of the BLE signal advertisement of recent smartphones, one of the key functionalities exploited in our system. Then, at sensors level, will be reported the different methods of synchronization and signal filtering experimented. For the back-end side, are reported the different classification methods employed for users localization, and their performances with respect to the ground truth. Here will be also illustrated how the occupancy information has being visualized in a web application.

The third chapter shows the design challenges of smart building systems based on the installation of several hardware nodes such as sensors or wireless access points. To address this problem, the developed supporting tool is presented. The application, based on a CAD interface for floor-plans specification, implements an heuristic algorithm for an optimal deployment of nodes. Although the heterogeneity and lack of standardization, is explained how we tried to provide a general-purpose tool, offering three different indoor localization techniques and the ability to extend the application with third-party plug-ins.

In the fourth chapter will be described our system deployment in a real environment and the experimental results obtained. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Chapter 1

Occupancy Monitoring Basics and State of the Art

Smart buildings are environments like offices or schools able to exploit sensors, devices and appliances to observe users activity and adapt autonomously taking decisions to achieve comfort, safety and energy efficiency. Considering the fast development of technologies for connected sensor and actuator (the so called *Internet of Things*), since few years smart buildings are becoming a reality from a technological point of view.

Both the industry and the academia have put a lot of effort in the research of smart buildings. In some commercial products the term *smart* is often referred to devices that can be remotely accessed and managed, even though there is no intelligence involved. In this thesis the term *smart building* refers to the concept of environment able to cooperate and self-organize its components observing the indoor space.

One of the main components of a smart building is the **Occupancy Monitoring system**, i.e. a system able to detect the number (or even the identity) of the occupants in every room or zone of the building. Occupants play a key role in the objectives of smart buildings. User comfort can be improved controlling automatically the environment temperature, humidity, and brightness, setting all the parameters exploiting user defined preferences. In this scenario, user identification and localization inside the building is fundamental.

In order to guarantee the building safety, the knowledge of the occupants position in real-time can be precious. In case of emergencies like fires, escape routes and emergency exits might be highlighted with notifications targeted to the specific user's location [22]. Rescue operations can be organized faster and more efficiently when the position of injured occupants is known.

Complex safety systems are even able to automatically detect dangerous situations like terrorist attacks when fast mass-movement of people happen from a room to another.

Probably, the main purpose of smart buildings, or in this case also called green buildings, is energy sustainability. In the last few years, plenty of methods have been proposed to learn the building occupants habits and to use such information to forecast the energy consumption. Many of these approach are based on the concept of *Smart Grid*. Smart Grid is a traditional electric grid enhanced with a two-way communication between the provider and its customers to respond digitally and more efficiently at quickly changing electric demand. The mentioned methods for smart buildings are able to select the most appropriate appliance usage scheduling given a energy reduction request coming from the smart grid. Also in this case an occupancy monitoring system is essential to recognize users activities.

The expected outcome of an occupancy monitoring system is the detection in real-time (or with a tolerable delay) of every user inside the building, with their position. Usually, the occupant needs to be localized in a zone or room of the building, since this information is sufficient to automatize HVAC and lighting systems. In cases of automation based on user-defined preferences also the identification of each user is required by the system. At first look, the occupancy monitoring problem can be associated at the more traditional problem of the indoor localization. Even though the two problem share some characteristics, we will see that conventional solution for indoor localization are not directly applicable for monitoring (section 1.3.7).

As we will see in this chapter, a lot of effort has been put both from the industry and the academia in terms of research and product industrialization. However, some intrinsically hard challenges of the occupancy monitoring problem bring to the lack of an accessible solution with broad adoption.

1.1 Evaluation Metrics

Occupancy Monitoring systems have the purpose to detect the number of occupants in a building, their position and in some cases also the identity of each one. Occupancy detection solutions can range to deeply different implementations, depending on the technologies exploited to detect human presence. In this section will be defined some metrics and aspects of

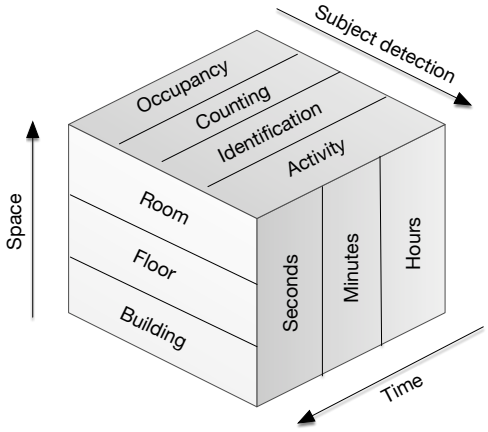

Figure 1.1: Occupancy Monitoring systems resolution.

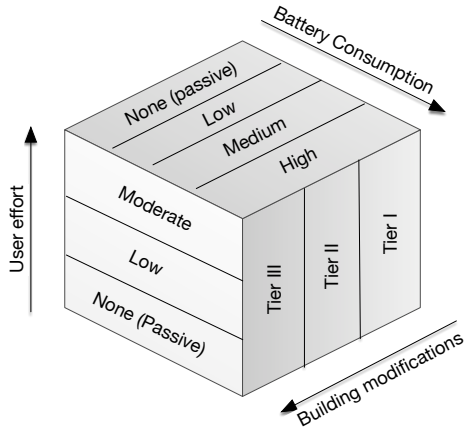

Figure 1.2: Occupancy Monitoring systems intrusiveness.

occupancy monitoring systems useful to evaluate and compare different solutions.

A feature that characterizes different occupancy monitoring systems is **Resolution** (Fig 1.1): it represents the granularity of the measurement.

Resolution can refer to:

- Subject of the detection: from low to high resolution, the system can perceive only the occupied/unoccupied status of a zone, the number of people or even the identity of each person.
- Space resolution, i.e. the smallest region in which the user can be localized. Can range from building zones to single rooms, desktop stations or cubicles.
- Temporal resolution indicates the smallest time span in which occupancy status transitions can be reported. For real-time applications, small delays are always required to avoid user waits and annoyances.

A common metric to valuate and compare any solution is the **Accuracy** of the system, an index that describe how much the resulting measurement differs from the true value (also called *Ground Truth*). In our context, detection accuracy is the capability to correctly recognize occupied/unoccupied state of the rooms; occupancy estimation accuracy reveals how much the number of estimated people in the room differs from the actual number of occupants. A good level of accuracy is essential for almost every detection systems: as instance, in order to reach energy waste

reduction, false positive presence detections can be counterproductive since appliances would be turned-on in empty rooms.

Accuracy and resolution are good parameters to summarize the system performances, but are not sufficient to describe how much the solution is desirable and attractive for real commercial applications. For instance, approaches that provide high accuracy and resolution tend to have also a higher **Cost**, complex **Installations** and higher **intrusiveness** for the occupants (see Fig. 1.2). To classify installation complexity, three levels, or *tiers*, have been identified by Akkaya et al. [1]:

- Tier I requires no modifications to the existing systems other than a data collection and processing unit. Solutions at this level make use of already existing building infrastructures, such as local network or electricity meters to measure occupancy. This approach is called *implicit* occupancy sensing, and will be discussed in section 1.2.2.

- Tier II is characterized by fast software and/or hardware installations to make occupancy related data available. Training phases are fast or immediate to execute, and can be accomplished by one or few building administrators.

- Tier III involves the addition of software and hardware that requires complex installations or long training periods. Complex installations include precise measurements of the building geometry in order to know the exact position of every installed device. Long training periods are usually required by solutions that extract occupancy information from ambient data (see section 1.2.3).

Usually higher tiers also imply higher costs for hardware purchase, system maintenance and computational resources. For occupancy monitoring systems, keeping low the installation and running costs is crucial, since spending cut is one of the main purposes of building renovations.

Another factor that need to be considered to promote the adoption of a monitoring system is the **Intrusiveness** on the users routine. Intrusiveness describe how much the user activity is negatively affected by the system. In occupancy monitoring systems, where positioning data needs to be continuously collected during many hours of the day, a low level of system intrusiveness is essential. Implicit occupancy sensing (tier I) is usually completely passive and doesn't require the occupant to use any additional device or software. Other solutions instead, relies on the assumption that the occupants are always accompanied by a tag, a smartphone or

a wearable device. This can be quantified more or less by a "low effort" required to the user. This amount of effort can be perceived differently depending on the situations. At work for example, where the user is used to guarantee phone call and email availability, the effort can be nearly zero; at home instead, where some users are used to empty their pockets as soon as they are inside, the effort can be high. However, for solutions based on smart devices, the aspect that more affects users routine is the battery consumption. Smart devices are becoming omnipresent around people and they can be considered as an extension of ourselves. A possible weak point of every smart device is the battery life, that becomes an issue also for the user daily activity. For this reasons, occupancy detection systems that rely on smart devices should carefully consider the energy efficiency of the device. A system that cause a considerable increase in battery drain directly afflict the user comfort and peace of mind, and can bring the user to leave the application. Nevertheless, we'll see in the next sections that many wireless based solutions present in literature that exploit smart devices completely ignore the energy problem.

System intrusiveness, from both the points of view of the building equipment and the user activity, is summarized in figure 1.2.

1.2 Existing Solutions

In this section are reviewed some of the most relevant approaches proposed in literature to solve the occupancy monitoring problem, grouped by their respective enabling technologies. A particular attention is directed to wireless-based indoor positioning systems (section 1.3), how they have been exploited for occupancy monitoring and their limitations in this specific context.

1.2.1 Traditional solutions

The traditional solutions used to face the occupancy detection problem are based on passive infrared sensors (PIR), door badges or radio frequency identification tags (RFID tags). PIR sensors are used to detect people movements in order to control lighting, doors and other appliances. The main issue is that overly still occupants become invisible to these sensors. In some cases, door sensors have been added to mitigate this issue. How will be explained in section 1.2.3, usually PIR are used in combination with other ambient sensors because of their low accuracy and their inability to provide estimations on the number of occupants.

In RFID systems, antennas are displaced inside the building to detect the presence of tags co-located with the occupants. However, few RFID systems can reach 10 meters of working range increasing the number of minimum antenna; in addition, tag transmissions are easily disrupted by other radio-frequency sources.

1.2.2 Implicit Sensing Solutions

Implicit occupancy sensing is the use of existing building infrastructure (that is not originally intended for occupancy detection) to measure occupancy. Implicit sensing relies on the notion that the effects passively produced by occupants on building systems can be used to determine occupancy information.

An example is those proposed by Melfi et al. in [20] that measures occupancy using existing network infrastructure: their method consists in the collection of MAC and IP addresses in routers that are successively correlated to the occupancy of a building, a zone, and/or a room. The main idea is that when a client sends a WiFi packet to an Access Point (AP), it is assumed to be located within the range of the AP. They collect data from Address Resolution Protocol tables (or ARP cache) and Dynamic Host Control Protocol (DHCP), of available access points, and derive the occupancy level of each area. The approach reveals a low occupancy accuracy (59%), caused by two main problems. First the overlap of AP coverage results in devices connecting to APs that are not the nearest available. Second, the inconsistence of mobile devices that lose connectivity causing false negatives.
Although they obtained low levels of occupancy accuracy with respect to the ground truth, they proposed one the first solutions able to re-use infrastructures common to many (if not most) building types, and does not require any additional hardware or physical modification.

Balaji et al. proposed Sentinel [4], another WiFi based occupancy detection system used to actuate on HVAC. They used the protected network of the University of California (San Diego) to collects packets coming from users' devices and infer occupancy. Also in this work, clients location has been assumed to be in the range of the receiving AP. In order to increase accuracy, building zones have been divided into personal spaces, with the constraint that a personal space cannot be occupied unless the owner is present in the space. This approach is represented in figure 1.3. If the AP does not hear from the client for a fixed period of time (1000 seconds in their network), the connection is terminated and the user is

considered out of the zone.

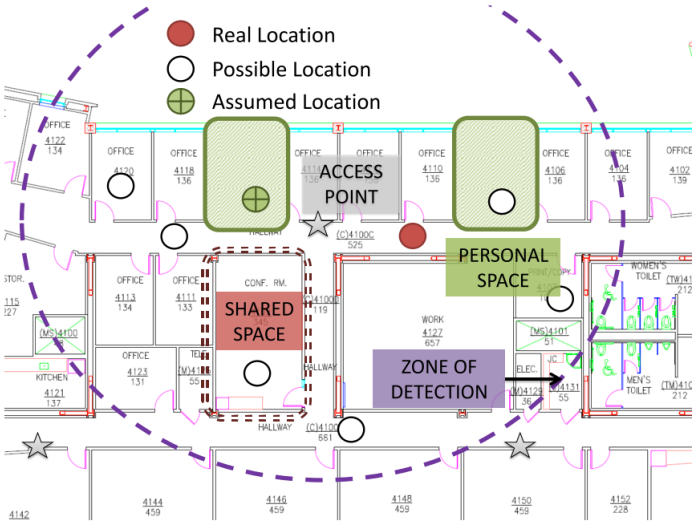

Figure 1.3: Occupancy inference of the Sentinel system using WiFi connectivity. The occupant is assumed to be in her personal space whenever she is within the associated AP's zone of detection, as denoted by "Assumed Location".

Since an occupant can have different WiFi devices, Balaji et al. decided to use the smartphone as the most representative of the user position. However, the algorithm that they implemented in order to distinguish the phone from the others devices in APs logs has a high level of inaccuracy, causing a degradation of overall results. In addition, smartphones revealed an aggressive sleep policy on WiFi activity as soon as the screen is locked, determining lack of packets and significant false negative detections of occupancy.

Implicit sensing solutions have very low costs of installation and don't require the user to do anything except their usual activity. Implicit sensing is usually characterized by low levels of accuracy and resolution; for example can be difficult, if not impossible, to distinguish if more than one connected devices belong to a single occupant of the building. Due to the low performances, implicit sensing is not a suitable source of information for real-time applications like appliance control or safety operations, but thanks to the easy and inexpensive implementation can be always used to collect historical data and to extract long-term occupancy probabilities as a function of time.

1.2.3 Ambient Sensor Networks

Thanks to the diffusion of low cost and low power sensors, and the emerging research area on Wireless Sensor Networks (WSN), solutions based on ambient sensing are currently deeply investigated. The basic idea is that human presence inside the building continuously affects ambient factors like temperature, humidity, sound, CO_2 and so on. By monitoring these factors the system can estimate the number of occupants. At this purpose, sensor motes are applied to each room of the building; data is sent to a centralized system where an algorithm estimates the number of occupants. One of the main issues of this approach is the long training period required to build the model, causing time consuming and intrusive data collection. We report two works that tried to overcome this issue.

Beltran et al. proposed ThermoSense [6], a wireless sensor network of nodes that can measure occupancy of rooms. Each node implements a combination of thermal sensor array and PIR sensor. Since occupants are typically warmer than the surroundings, the thermal sensor array, that is capable of measuring multiple temperatures with an area, is used for detecting occupancy. For distinguish between people from passive warm objects, the PIR functionality is exploited. If the PIR sensor indicates the room is empty, then the array determines the thermal background of the space, otherwise the system tries to recognize the number of hotspot corresponding to occupants.
The thermal sensor array used is able to measure temperatures in an 8x8 grid pattern within an 2.5mx2.5m area, when placed at a height of 3m (ceiling). A photo of the employed sensor mote and the resulting detection is reported in figure 1.4. The system was able to measure occupancy with a Root Mean Square Error of 0.35 person. The weak point of this implementation relies on the short coverage of single sensors, that heavily increases the cost of installation. In order to fully cover a simple 10mx10m room, 16 nodes would be necessary. Covering only hot points like desktop stations or cubicles would drastically reduce the accuracy.

Zheng et al. [30] proposed a framework for cross-space occupancy modeling in which the model is trained in a space only a single time, and then applied to other geometrically similar spaces. The objective of their study was to generalize relationships between occupancy and ambient factors, collected through sensor boxes equipped with several different sensors: light, sound, motion, CO2, temperature, humidity, PIR and door switch. Training data has been collected from six rooms for long periods of time (10 / 18 months). The occupancy ground truth was collected using ceiling-mounted cameras, which captured images every 10 seconds.

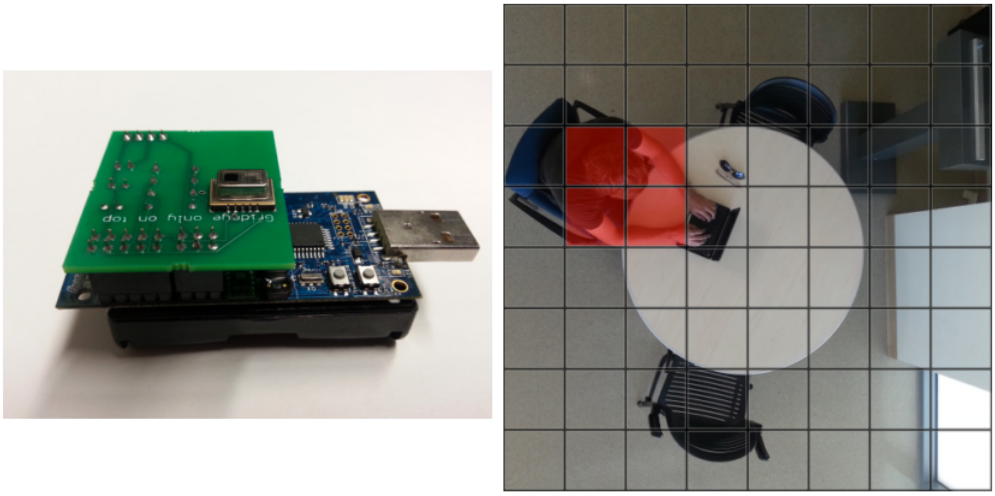

Figure 1.4: Thermal sensor array employed in the Thermosense system and its occupancy detection outcome.

Occupancy was then manually labeled for the ground truth. After a phase of selection and pre-processing, the ambient features were used by five supervised learning classifiers to build the general model.
The occupancy model was then tested applying samples from other geometrically similar rooms. The modeled occupancy was compared with the actual occupancy detection from 6:00 AM to 9:00 PM (only weekdays) to calculate the performance. They experienced a daily F-measure from 0,71 to 0,94 and a daily RMSE from 0,42 to 1,72 person. However, training rooms employed accommodate no more than four occupants, suggesting that the performances of such model could drastically drop if applied to more populated areas.

The advantage of similar solutions is that the user is not forced to keep devices with him/her to be detected by the system. Intrusiveness on the occupant is rather low, except for the employment of cameras and microphones that strongly reduce the privacy.
The principal issue of ambient sensing solutions is that ambient factors variate slowly, and provide fast reactions to users movements is difficult. Second, ambient models requires long training period in order to teach the algorithm the relationships between occupancy and ambient factors. Other complications arise because ambient factors are influenced, besides occupancy, also by HVAC and lighting systems. HVAC and lighting are often the appliances that occupancy detection outputs should regulate, creating undesired feedbacks in the control system.

1.2.4 Cameras

Cameras can be used for measuring occupancy exploiting image analysis algorithms, that recognize bodies or faces within captured images.

Erickson et al. developed OPTNet [14], an occupancy estimation system comprised of 22 node wireless camera nodes used as an optical turnstile to measure area/zone occupancies. They employed a lightweight on-board image processing algorithm along with classification techniques in order to detect occupants' transitions. In addition, they exploited a 40 node PIR wireless sensor network to improve the system accuracy. Figure 1.5 reports two photos of the deployed camera systems and an example of image analyzed. Their work claims to be capable of detecting occupants transitions with up to 94% accuracy and that the overall system was able to bound the error of occupancy within 1.83 people on average for their building.

Figure 1.5: The deployed OPTNet system. Transition areas are represented as bigger blue boxes while trigger areas are small red boxes.

Camera placement is a considerable issue. Cameras placed in offices or meeting rooms raise privacy concerns. A flexible solution for different privacy requirements (and legislations) should allow to keep track of anonymous occupants without collects sensitive data like faces pictures. Cameras can be placed in public and narrow transition points, like hallways or stairways and function as an "optical" turnstile, counting people for each direction. This strategy however is prone to cumulative error: If an optical turnstile misses even a single person entering/exiting a space, this error is propagated until another offsetting error occurs or some other mechanism is used to remove the cumulative error. To give an example, if the last person in an office leaves at 7 p.m. and this transition is missed, then the space will continue to be erroneously considered occupied.
Another problem of camera based systems is the computational effort required by the image analysis algorithms. One of the main purpose of

smart buildings is the reduction of costs through energy harvesting. In this perspective, the installation of expensive hardware with high computational resources and consumptions for image processing is not the most appealing solution.

1.3 Wireless Indoor Positioning

In order to determine the position of a user (or an object) in a limited physical space, positioning systems based on wireless technologies have been studied and employed from many years. Wireless positioning systems, even with deeply different implementation technologies, always rely on a set of reference nodes that communicates with the target object to be localized through generic radio frequency signals. GPS as instance, the well-known Global Positioning Systems, is able to provides location information anywhere on the Earth where there is an unobstructed line-of-sight with at least four GPS satellites. GPS, which is able to provide good performance levels in open spaces, has been widely adopted for outdoor applications; however, is not capable of operating indoors, due to the significant attenuation introduced by buildings walls and ceilings.

During last years, solutions for indoor localization have been proposed; in this case reference points are represented by WiFi access points or beacons enabled with different signal protocols like Bluetooth. Some existing solutions that exploit WiFi infrastructure have already been discussed in section 1.2.2. In that case however, the location estimation was performed based on access points data like MAC or IP addresses collected at the Network layer of the OSI model. In this section instead will be discussed localization techniques that analyze the wireless signals at a lower level (Physical OSI layer). For this reason, this section provides first some background concepts of radio signals, later an overview of the most common localization techniques based on signal analysis and finally some relevant related works.

1.3.1 Signal Transmission

Electromagnetic signals are classified according to their frequency. Radio frequency (RF) includes all the wave frequencies that lie in the range extending from around 3 kHz to 300 GHz, which include those frequencies used for the most common wireless communications. For this reason the term *Radio Frequency* (or its abbreviation RF) is used as a

synonym for radio, describing the use of wireless communication as opposed to communication via electric wires.

The signal transmitted by an antenna can be expressed as a function of time:

$$s(t) = a(t)\cos[2\pi f_c t + \vartheta(t)] \tag{1.1}$$

where $\vartheta(t)$ is the phase, f_c is the carrier frequency. A useful measurement of radio signals is power, or more precisely the average power, defined as the time average of energy:

$$P = \lim_{T\to+\infty} \frac{1}{2T} \int_{-T}^{+T} (|s(t)|)^2 dt \tag{1.2}$$

When an antenna transmits a RF signal, the signal is characterized by a certain power. The signal at this point is transformed by the antenna into a electromagnetic wave. During propagation the signal travels a medium that introduces some power attenuation to the signal. When the signal hits the receiving antenna this attenuation, also called *Path Loss*, can be expressed as:

$$PL = \frac{P_t}{P_r} \tag{1.3}$$

where P_t is the power of the transmitted signal while P_r refers to receiving signal. Different transmission mediums, like free space, air, walls or bodies, can introduce different attenuations. In free space the attenuation is minimal and is in fact called free-space loss. In other materials additional attenuation is introduced by effects like refraction, diffraction, reflection, and absorption. If the medium is completely composed by the free space or the air, the transmission link is referred as Line of Sight (LOS). Differently, if the signal needs to travel across other materials in order to reach the receiving antenna, the communication is called Non Line of Sight (NLOS).

Usually the path loss is expressed in decibels (dB):

$$PL_{db} = 10\log_{10}(\frac{P_t}{P_r}) \tag{1.4}$$

dB unit is a shorthand way to express the ratio of two values. To express absolute values instead, in this thesis many examples and results are expressed in dBm (Decibel-milliwatt), as returned by smartphones and sensors used for the measurements. The unit dBm denotes an absolute power level measured in decibels and referenced to 1 milliwatt (mW).

Absolute power P (in watts) is converted to dBm with the formula $P_{dBm} = 10\log(\frac{P}{1mW})$. This equation looks almost the same as that for the dB, except that the power level P has been referenced to 1 mW.

The transmission medium entails, other than the power loss of the signal, also a propagation delay. In particular, the delay depends mainly on the distance to be traveled, and secondary by the wave propagation speed of the medium. For wireless communication this propagation speed tends to $3 * 10^8$, which is equal to the speed of light in vacuum.

1.3.2 Signal Analysis for Localization

Radio Frequency signals transmitted by reference nodes can be analyzed in order to compute or estimate the position of a target object or subject. Different signal characteristics has been measured at this purpose.

- **Time of Flight (ToF)**: As we previously said, the time a signal spend traveling from transmitter to receiver depends on the path length and the materials encountered. However, in most cases the main medium is the air and its propagation speed can be considered in good approximation equal to the constant speed in space. ToF technique exploits the resulting linear relationship between travel time and distance, computing distance between two points using the time delay between the transmitted waves.

  There are two main approaches to compute ToF, *single packet* and *double packet*. The first simply requires that the reference nodes transmit a packet containing the timestamp of the transmission instant. The receiving mobile device estimates the distance between each nodes comparing the received timestamp with its time of arrival. However this approach requires a strong synchronization between the reference nodes and the target device. When synchronization between reference and target nodes is impossible or too difficult to achieve, a double packet approach can still be used to determine ToF, as shown in figure 1.6. This approach is based on the differential time difference of arrival, and eliminates time offsets between the reference and target nodes simply assuming that the path traveled by each packet is the same. Since the same time offset is added to the measured time of one packet and subtracted from the other, the correct time of flight can be calculated without any synchronization with the equation:

$$ToF = \frac{T_{b1} - T_{a1} + T_{a2} - T_{b2}}{2} \tag{1.5}$$

Where T_a and T_b represent a timestamp measured using respectively node A and B local clocks.

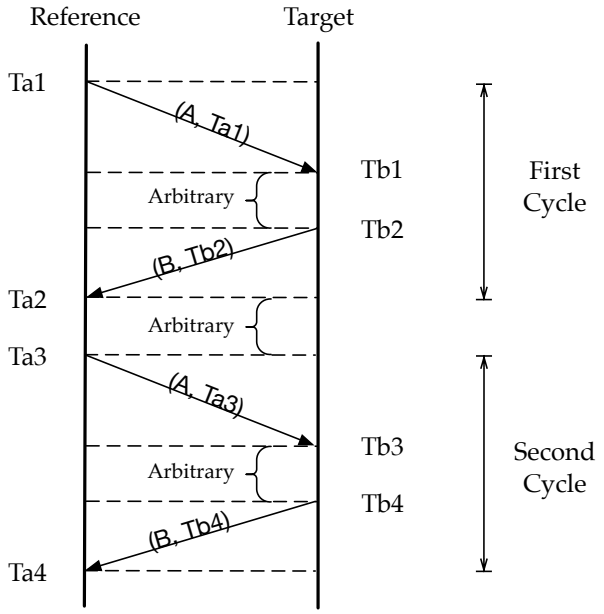

Figure 1.6: Time of Flight computation exploiting a double packet transmission.

- **Received Signal Strength Indicator (RSS or RSSI)**: is a measurement of the power present in a received radio signal. RSS can be used in two main approaches: distance estimation and fingerprinting. As introduced in 1.3.1, power loss of the signal during transmission depends on the distance, other than the medium material. The RSS perceived by a receiving node from a transmitter can be used to estimate their distance. Typically this estimation requires a prior work to be done before localization can take place. A common necessary training consists in a calibration that measures the RSS value at a fixed distance. Defining C as the received signal strength collected at one meter, the run-time distance can be estimated with the following equations:

$$RSS_{dBm} = -10\alpha \log_{10}(distance) + C) \tag{1.6}$$

$$distance = 10^{\frac{(RSS-C)}{(-10-\alpha)}} \tag{1.7}$$

Once that three or more distances between reference and target nodes have been computed, the position can be estimated using geometric properties like trilateration.

RSS values can also be used for localization without the needs of estimating each single distance between reference and target nodes, focusing instead on the aggregate RSS information. Pattern Recognition (PR) or fingerprinting techniques attempt to engage a received power levels vector obtained from multiple reference nodes with a defined calibration test, once a time without the need for geometric algorithms. Some example of these pattern recognition techniques, that will be illustrated in the next section, are K-Nearest Neighbors (KNN), Bayesian methods, and Support Vector Machines (SVM).

Depending on OS and application, signal strength is measured either as quality in percentage, or an RSSI value in dBm. RSS values usually range from 0 (zero) to -110dBm and the closer they are to zero, the stronger the signal is. RSS level lower than -80dBm may not be usable for telecommunication, but still very useful for localization.

- **Angle of Arrival (AoA)**: is a method for measuring the propagation direction of a RF signal incident on an antenna array. Usually AoA determines the direction by measuring the time difference of arrival at each single antenna of the array, or directly measures the angle exploiting highly directional sensors. Once that angles have been measured, position con be computed using geometric properties of triangles like in the case of RSS, but using triangulation instead of trilateration.

1.3.3 Localization Techniques and Algorithms

In the previous section there have been mentioned some properties and characteristics of wireless signals useful to infer or compute the target position. The way in which these signal properties are used to extract location encompass different algorithms and techniques. These algorithms can be classified in two main classes: *Geometric* and *Fingerprinting*.

Geometric algorithms are based on geometric properties of triangles. They achieve high accuracy localization, provided that the underline signal analysis is also accurate. Their biggest drawback is that they require a meticulous set-up in which distances (or angles) between the installed

reference points are precisely measured in order to identify each reference at least in a Cartesian plane, if not in space. Some of the most common geometric techniques are reported below.

- **Trilateration**: is the process of determining absolute or relative locations of points by measurement of distances between reference and target nodes, using the geometry of circles, triangles, or spheres. Representing the space in two dimensions, when receiving a signal from a single transmitter, target device can be situated on a circumference with the reference at the center. With two transmitters there are only two positions possible: the two points where the circles around the two transmitters intersect. Adding a third transmitter enables to eliminate one of these two possibilities. In figure 1.7, this condition is represented by a target node T localized by means of distances $d1, d2, d3$.

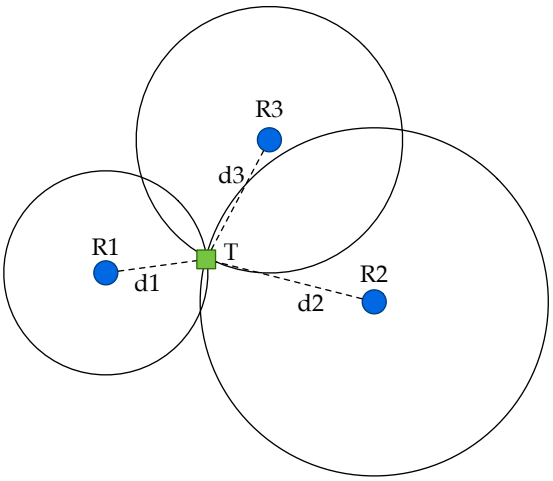

Figure 1.7: Example of a target node T localized with Trilateration by means of distances $d1, d2, d3$.

  When we extend trilateration to three dimensions, the circles become spheres. In this case it's necessary to add one additional transmitter in order to find the position of the target. This explains why GPS needs to receive at least four satellites to work.

- **Triangulation**: allows an observer (target) to calculate its position by measuring two directions towards two reference points. Since the positions of the reference points are known, it is hence possible to construct a triangle where one of the sides and two of the angles are known, with the observer at the third point. This information

is enough to define the triangle completely and hence deduce the coordinates of the observer. An example of the triangle construction is reported in figure 1.8 where the system is able to measure at run time α and β, while distance L is known since the two reference points $R1$ and $R2$ have well-known coordinates.

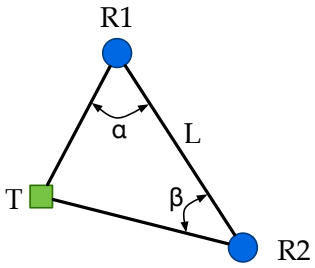

Figure 1.8: Example of Triangulation where the system is able to measure at run time α and β, while distance L is known since the two reference points R1 and R2 have well-known coordinates.

- **Min-Max algorithm**: is based on simple geometric considerations, and it is one of the most used for localization due to its easy implementation. The target node computes the distance d_i from each reference node, and for each of them it draws a square with center in the position of the reference node, and edges of length $2d_i$. Then, the algorithm calculates the intersection among the squares drawn around the anchor nodes, and the mobile node is at the center of the resulting quadrilateral. In mathematical terms, the target node computes the maximum and minimum values, and it identifies a square having coordinates

$$(max_{(x_i-d_i)}, max_{(y_i-d_i)}) \times (min_{(x_i-d_i)}, min_{(y_i-d_i)}) \tag{1.8}$$

  The center of this four-sided figure is the estimated position searched. An example of Min-Max estimation is shown in Figure 1.9.

- **Maximum Likelihood Algorithm**: is based on statistical considerations on the set of measures coming from the reference nodes. Its main aim is to minimize the Mean Square Error (MSE) of the measurements which are affected by noise. First, an estimate of distance d_i to each reference device is derived from the RSSI value. Then, the node defines the error e_i between the measured and the actual distance with the following equation:

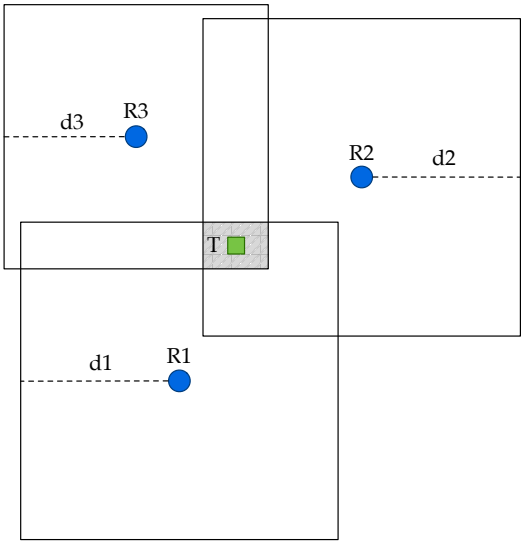

Figure 1.9: Example of MinMax computation where the target node computes the distance d_i from each reference node, draws a square with center in the reference node and edges of length $2d_i$.

$$e_i(x_0, y_0) = d_i - \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} \tag{1.9}$$

where (x_0, y_0) is the unknown position of the target node and (x_i, y_i) the position of the i-th reference node. To estimate the position this algorithm minimize the MSE, obtaining solutions with low error variance. Unfortunately the measurement pool need to be significant, and with only three reference nodes the accuracy can be poor.

Fingerprinting techniques are used to estimate the position of a mobile device based on measured signal characteristics without the constraints of geometric properties. This subsection will refer to RSS as the main signal measure, but the same could be applied to ToF or AoA. Each target receives the signal from k nodes, each one with a given power. The values are collected at run-time into a vector of k element that is later compared with a dataset of vectors, each one pre-labeled with the corresponding position. Although fingerprinting algorithms are less accurate then geometric techniques, they are more flexible since there aren't constraints on the vectors dimension. In addition, training phase are usually far more easy and fast to perform, since to construct the database is sufficient to subdivide the target area in tiles and scan for the surrounding signals in each one of them. The most common ways in which run time

and database vectors are compared to estimate target position are reported below.

- **K-Nearest Neighbors (KNN) method**: is based on the idea that even if RSS values do not depend linearly on the distance between references and target, some relation exists. Exploiting this relations, a database of locations and the radio map (the set of test vectors) can be created, containing the position of each location (as coordinates or tile) and the corresponding RSS from the reference nodes. To locate the target device, the vector of current received powers is analyzed, and then compared with the database of locations. The test vector closest to the received one is selected. A majority vote on the vector elements is used to identify the closest test vector. Later, euclidean distance between runtime and off-line vector can be used to adjust and improve the outcome position. An example has been provided in figure 1.10. For each room, three signal scans have been performed and are represented by colored shapes. The classification assigns to the runtime measure (green square) the most likely position based on the three closest values.

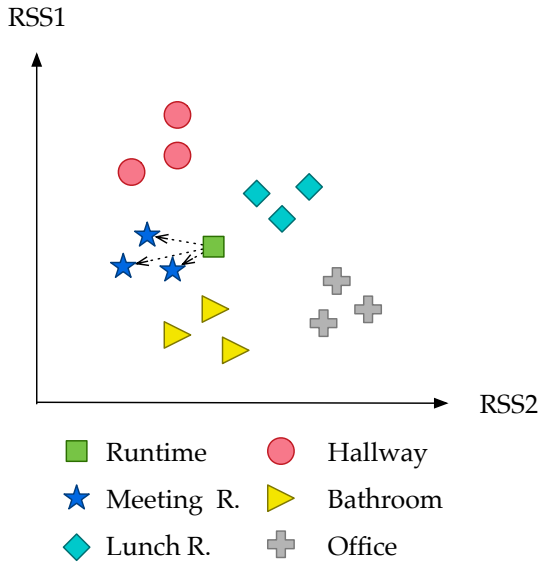

Figure 1.10: Example of KNN classification of the current position between the target sub-areas. For each room, three signal scans have been performed. The classification assigns the most likely position based on a majority vote with $K = 3$.

- **Bayesian method**: A Bayesian network containing the points of

interest is built during the training phase. In the next step the system is trained a certain time on each of the points of interest, collecting RSS samples periodically. During the training phase, a priori probability $p(L)$ is constructed as a function of location L. Then, conditioned posteriori probabilities $p(E|L)$ are defined as the function of perceived RSS from a reference nodes in a certain location, and a given event or locations history (E). These methods are able to model impossible transitions from one location to another, in similar way as in Hidden Markov Model (HMM).

- **Support Vector Machines (SVM)**: This method is a paradigm of Neural Networks, in which measure/observation vectors are processed. Processing is done by using a hypothesis space of linear functions over a space with a greater dimension (space of high-dimensional features) where the dimension of observations is induced by a kernel function with the purpose to obtain a hyperplane that separates linearly the observations and let to locate points in the most reliable as possible way. An hyperplane separates the set of training points into two subsets, each one containing a different label. From all possible planes there is only one optimal separating hyperplane (OSH) which is calculated by maximizing the distance between the optimal separation hyperplane and the closest training pattern, i.e. the maximum margin.
- **Neural Networks Methods (NN)**: this method uses a Neural Network to estimate the location processing distinct RSS emitted from fixed receptor-emitters. The NN is implemented by a multilayer perceptron in which the input entries may be the RSS of each fixed receptor-emitters, and the output is likely to be in each of the locations.

1.3.4 WiFi protocol

The midrange wireless local area network (WLAN) standard, operating in the 2.4GHz Industrial, Scientific and Medical (ISM) band, has become very popular in public hot-spots and enterprise locations during the last few years. With a typical gross bit rate of 11, 54, or 108 Mbps and a range of 50-100 m, IEEE 802.11 is currently the dominant local wireless networking standard. It is, therefore, appealing to use an existing WLAN infrastructure for indoor location as well, by adding a location server. The accuracy of typical WLAN positioning systems using RSS is approximately 3 to 10 m, with an update rate in the range of few seconds.

All the functionality of the protocol is reflected in the packet headers. RF technology and station mobility impose some complex requirements on 802.11 WLAN networks. This added complexity is reflected in the long physical layer convergence protocol (PLCP) headers as well as the data-rich MAC header. The packet compared with the Ethernet structure is showed in figure 1.11.

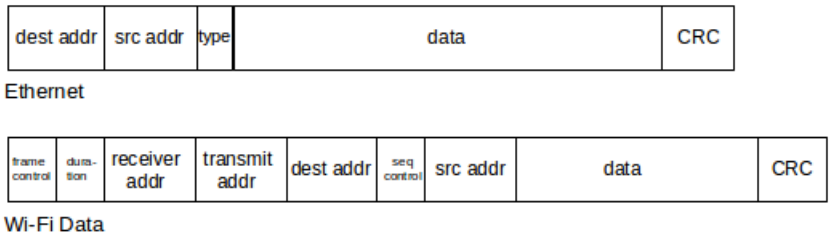

Figure 1.11

1.3.5 Bluetooth protocol

Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices, and building personal area networks. Nowadays, various mobile devices (most of commercially available phones, personal digital assistants, etc.) are equipped with Bluetooth radio transceivers. Due to its broad adoption, Bluetooth can be considered a highly ubiquitous standard. Bluetooth was originally a codename for a project lead by a Special Interest Group (SIG) consisting of major companies, like Ericsson, Intel and Nokia. The Bluetooth bit rate is lower (1 Mbps), and the range is shorter (typically 10-15 m) with respect to Wi-Fi. Moreover, Bluetooth is a lighter standard and supports several other networking services in addition to IP. For positioning, it is suggested to employ received signal strength indications (RSSI), since RSSI decreases with distance between sender and receiver. Since Bluetooth is a low-cost and low-power technology (Bluetooth tags are small size transceivers), it can be intended as an efficient solution to design IPSs. The Bluetooth positioning systems suffer from the same drawbacks of RF positioning technique in complex and full of obstacles indoor situations.

Bluetooth 4.0 or Bluetooth Low Energy (BLE)

Bluetooth v4.0 introduced Bluetooth Low Energy in 2014, officially known as BLE or Bluetooth Smart. This specification introduced a completely different wireless radio that was smaller, cheaper and lower power to meet the needs of these new applications. The wide support for it in smartphones, tablets and embedded systems makes BLE one of the most appealing protocol for the Internet of Things communications.

BLE specification divides Bluetooth devices in two classes: *Central* and *Peripheral* devices. Central devices can scan for other devices and initiate a communication. Usually, the central is a smartphone, tablet or a PC. Peripheral wait for connections or broadcast unconnected signals coming from sensors. Typically, the slave is a small device like a fitness tracker or a smart-watch that wait for a central device and broadcast the sensed data. Another type of peripheral device is the Bluetooth beacon, a small devices usually battery powered that continuously broadcast a unique identifier in the surrounding area for ranging applications. Some examples of BLE beacons on the market are the Estimote beacon or the Apple iBeacon.

BLE has two ways of communicating. The first way to communicate is to receive packets using a *Connection*, where both the peripheral and central send packets. The second way is using *Advertisements*, where a BLE peripheral device broadcasts packets to every device around it. The receiving device can then act on this information or connect to receive more information. Advertising is by design unidirectional. A Central device can't send any data to the Peripheral device without a connection. But a single peripheral can advertise to multiple masters in the area.

Apple iBeacon protocol

The iBeacon protocol, since the release of iOS 7, allows a Apple devices to receive push notifications when users enter (or exit) in (from) a iBeacon region. In addition, iOS applications can also evaluate the relative distance among the iBeacons. In iOS, regions associated with an application are continuously tracked, including when the application itself is not running; if a region boundary is crossed, the application is activated in background to handle the event. A possible application of iBeacon is a store where users receive a welcome (plus, eventually, a set of special offers) whenever they enter the store. Usually, in this kind of applications, the accuracy in determining the location of the users is not critical, and even an approximation of several meters can be acceptable.

1.3.6 Existing Wireless IPS

In this section the most relevant wireless indoor positioning exploiting WiFi and Bluetooth are reported.

WiFi IPS

ARIEL [17] is a fully automated indoor room localization system. To accurately identify rooms without extensive manual annotation, they developed a number of novel techniques. First, a zone-based clustering algorithm that accurately identifies room occupancy hotspot(s) using Wi-Fi signatures. A clustering algorithm based on motion to identify inter-zone correlation, thereby distinguishing different rooms. An energy efficient motion detection algorithm to minimize the noise of Wi-Fi fingerprints.
ARIEL has been implemented and deployed for a 10-month study with 21 participants. The evaluation results demonstrate that their automated clustering algorithm generates clusters that are high representative of individual rooms and achieves high accuracy (95%) for room localization. The accuracy is comparable to existing techniques that require manual annotation.

Beder and Klepal [5] focused on the likelihood observation function used in fingerprinting based localization. They proposed two improvements on the commonly used Gaussian approach directly addressing two well-known issues. The first issue was the differing antenna attenuation between different devices, that they addressed by explicitly estimating a global offset thereby considering only relative differences between vectors of received signal strength measurements. The second issue was dealing with environments where not every beacon is visible everywhere, which they proposed to address by explicitly modeling the pickup probability of beacons using Gibbs distributions. Both improvements of the likelihood observation function were shown to increase the performance of their localization system. A suitable distance metric between signal strength measurements is at the core of every fingerprinting based localization system, however their findings are likely to improve all systems currently relying on a purely Gaussian approach.
during the 2014 Microsoft Indoor Localization Competition, 22 different solutions to indoor localization from different teams around the world were put to test in the same unfamiliar space, and the system deployed by Beder and Klepal [5] performed better in the infrastructure-free (WiFi) category.

Bluetooth IPS

Zhu et al. [31] proposed one of the most relevant RSSI based Bluetooth positioning method. As usual, there are two phases in their positioning procedure: offline training and online locating. In the phase of offline training, they used piecewise fitting based on the log-normal distribution model to train the propagation model of RSSI for every BLE reference node. They design a Gaussian filter to pre-process the receiving signals in different sampling points. In the phase of online locating, they used weighted sliding window to reduce fluctuations of the real-time signals. In order to reduce the errors of targets coordinates caused by ordinary least squares method, they propose a collaborative localization algorithm based on Taylor series expansion. Another feature of their method is the active learning ability of BLE reference nodes.
Every reference node adjusts its pre-trained model according to the received signals from detecting nodes actively and periodically, which improve the accuracy of positioning. Experiments showed that the probability of locating error less than 1.5 meter is higher than 80%using their positioning method.

Nacci et al. [8] proposed BlueSentinel, a Bluetooth Low Energy occupancy detection system based on the Apple iBeacon protocol, with the aim to provide an energy efficient solution. In particular they proposed a modification of the iBeacon protocol, still based on BLE, in order to overcome the limitation that allows mobile devices to detect iBeacons only when entering and exiting from beacons proximity. The main idea of their implementation was to change the way the beacons advertise the region associated with them. Instead of correlate each one of them with only a single region, they let the beacons advertise more than one region, in a cyclic sequence. Thanks to this, they was able to create false events that force the operating system to wake up the mobile application more frequently.
The application was in charge of computing the position using RSS values and transmitting the location data to the BMS through HTTP requests. The results confirmed that HTTP protocol over WiFi, even if simple to use, is not the best choice when a continuous communication is required. A test performed on a Android device (using HTTP over WiFi) showed that the battery discharged completely in only 4 hours. A different test showed that the energy consumption related to BLE tasks was 5 to 10% of the consumption related to WiFI. This demonstrates that true low power state was guaranteed only for the stationary device (beacon) while the mobile device was forced to heavily affect the battery, increasing users

effort and intrusiveness. Some consideration on this issue are reported in the next section (1.3.7).

1.3.7 Indoor Localization for Occupancy Monitoring

The problem of monitoring the occupants location in smart buildings can be erroneously considered equal to the more traditional indoor localization problem. Although the same localization algorithms and techniques can be applied, their implementations requires relevant structural differences.

The main purpose of traditional indoor localization systems is to provide to the user its position on the building. The location information is exploited for navigation apps that assist the user moving from a starting point to a destination in the building, or more generally to orient the user in the indoor environment. At this purpose, signal analysis and localization algorithms are always executed on the mobile device. Scans for wireless signals and location computation usually involve high battery consumption for the mobile devices. However, the mobile energy efficiency is not taken in consideration by the majority of the existing wireless IPS because orientation and navigation tasks are usually executed by users for a limited execution time (typically few minutes) and don't affect the battery lifetime.

Occupancy monitoring for smart buildings shows requirements extremely different. First, the main purpose is to obtain users presence and position information at the infrastructure level, the so called Building Management System (BMS). Here the occupancy information can be exploited to regulate automatic systems (e.g. HVAC and lighting), build occupancy probability profiles, expose occupancy information to safety applications, and so on. Since the information is required in real-time on the infrastructure side (back-end), traditional indoor positioning systems that compute location on mobile device needs to transfer the location data through mobile connectivity.

Take in consideration WiFi based IPS as shown in figure 1.12. The mobile device compute its position exploiting signals coming from WiFi access points. In order to monitor its position in real-time, the mobile application needs to upload the current position continuously, generating an interminable connection with high energy consumption (red connection in fig. 1.12). Since occupancy monitoring is required to work potentially for many hours of the day, the approach become unsustainable for the mobile battery. In addition, an energy-hungry application is intrusive for the user

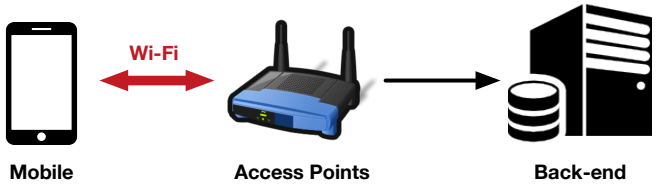

Figure 1.12: Common WiFi based IPS applied for real-time occupancy monitoring.

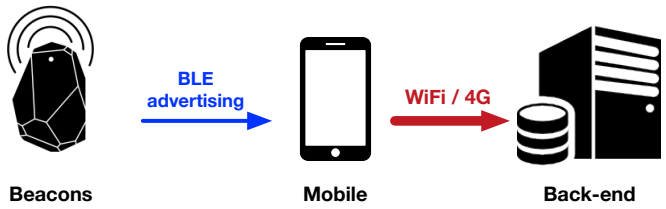

Figure 1.13: Common Bluetooth based IPS applied for real-time occupancy monitoring.

point of view and less likely to be adopted for continuous monitoring.

In case of Bluetooth Low Energy, the employment of reference nodes such as beacons enable to provide users with their location using a low power communication. The wide popularity of the Apple iBeacon or the Estimote in commercial contexts have proved that BLE beacons are a suitable technology for providing users ranging applications and location aware services. However, in the context of location monitoring, they show the same limitations of WiFi based systems, as reported in figure 1.13. In order to retrieve position in real-time at the BMS level, the device needs to send its current position continuously through mobile connectivity, generating a long connection characterized by high energy consumptions.

1.4 Comparison

In this last section, all the occupancy detection monitoring systems from existing literature that have been illustrated during this chapter are summarized and compared in table 1.1. Extra devices for building refers to any hardware installation required in common buildings such as offices or schools. WiFi hotspots are considered already present appliances in the building infrastructure. Installation complexity, as defined by Akkaya et al. in [1], is represented by three tiers from low to high complexity. User intrusiveness is an indicator of how much the its activity is negatively affected

by the system. When battery consumption is high, user intrusiveness is also considered high.

Accuracy of the system has been considered high if the system was able to localize a user within circumscribed area with an accuracy greater than 85%.

Table 1.1: Comparison between the most relevant Occupancy Monitoring systems in literature.

Approach	**Technology**	**Extra devices building side**	**Installation complexity**	**Extra device user side**	**Mobile consumption**	**User intrusiveness**	**High accuracy**	**Real-time detection**	**User identification**
[20]	Implicit network sensing	No	Tier I	No	None	Very Low	No	No	No
[4]	Implicit network sensing	No	Tier I	No	None	Very Low	No	No	Yes
[6]	Ambient sensors	Yes	Tier III	No	None	Low	No	No	No
[30]	Ambient sensors	Yes	Tier III	No	None	Low	No	No	No
[14]	Cameras	Yes	Tier II	No	None	High	Yes	Yes	No
[17]	WiFi localization	No	Tier II	Yes	High	High	Yes	Yes	Yes
[5]	WiFi localization	No	Tier II	Yes	High	High	Yes	Yes	Yes
[31]	Bluetooth localization	Yes	Tier II	Yes	High	High	Yes	Yes	Yes
[8]	Bluetooth localization	Yes	Tier II	Yes	High	High	Yes	Yes	Yes
Proposed approach	BLE sensor network	Yes	Tier II	Yes	Low	Low	Yes	Yes	Yes

Chapter 2

BlueSentinel: the Occupancy Monitoring System

2.1 Intro

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2.2 Choosing the Enabling Technology

In the previous chapter, many possible technological solutions to build indoor location monitoring systems have been reviewed. During the preliminary analysis of our work, all the available technologies have been considered in order to satisfy the occupancy monitoring requirements and a feasible solution for both users and infrastructures.

Considering the comparison summarized in table 1.1, it can be deduced that the only solutions able to guarantee a true real-time detection of users are cameras and wireless based solutions. When comparing this two approaches, cameras revealed to have many more drawbacks with respect to wireless solutions. Camera based systems require a high computational

effort to perform real-time body recognition on indoor video (or images) streams. The compute-intensive peculiarity makes the approach not suitable to scale up for large buildings, with tens or thousand rooms and users. Wireless based solutions tend to provide easily an outcome in real-time, or with a tolerable delay. Raw data produced by wireless systems is usually far more lightweight to process (for example compared to images), making the solution attractive for large indoor areas.

Inside the wireless scope, the choice is restricted to technologies that all users have already available in their mobile devices, so to avoid the employment of additional objects. The first one is certainly WiFi. WiFi based localization solutions have been deeply investigated, also because of WiFi infrastructure is already present in almost all buildings. However, for monitoring purposes, they are not able to provide real-time location data for long execution periods. This issue is explained in detail in section 1.3.7 and summarized here. There exist two ways to localize a mobile device using WiFi connectivity, an *active* and a *passive* method. The active method requires an application installed on mobile device that keeps an active connection with all the surrounding hotspots, and continuously notify collected data using an Internet connection. However, battery consumption related to WiFi operations is too high to reach many hours of execution. The passive method instead doesn't require any installed application on mobile device, but simply exploits WiFi packets sent from devices during their usual activity. Unfortunately, using this approach, is nearly impossible to obtain real-time detection since mobile operating systems apply aggressive sleep policies to WiFi module in order to limit consumptions. A WiFi connected smartphone during sleep can spend several minutes without emits any signal, making the system unresponsive to users movements [4].

The remaining available wireless technology is Bluetooth, and in particular Bluetooth Low Energy. Since 2010, BLE chips (able to work with classic and Low Energy enabled Bluetooth) have been built into in an increasing number of smartphones, tablets or wearables (as instance, Apple has incorporated them in its products since the iPhone 4S in 2011). Devices using BLE consume ten times less then WiFi and a fraction of the power of classic Bluetooth [33]. However, at the best of our knowledge, BLE has been employed for localization only through passive appliances (like beacons) that leave the burden on the mobile device, forcing it to run positioning algorithms continuously and notify the result through energy-hungry connectivities. In order to fully exploit energy efficiency of BLE, the proposed system relies on BLE mobile advertisement (section 2.3), a particular BLE signal transmission compatible with almost all mobile

devices since few years.

2.3 Mobile BLE advertisement

As explained in section 1.3.5, BLE specification divides Bluetooth devices in two classes: *Central* and *Peripheral* devices (figure 2.1). Peripheral devices generates data, like sensor readings or audio streams, while Central devices consumes it. This BLE specification was intended to classify smartphones and PC as centrals, while accessories as peripherals.

BLE has also two ways of communicating. The first way is establishing a *Connection* between a peripheral and a central device. Every connection however, requires a preliminary pairing phase between the two devices. This makes unfeasible the employment of BLE connections in monitoring applications, since a mobile device (central) would be forced to perform a pairing with every single reference node (peripheral) before monitoring can start.

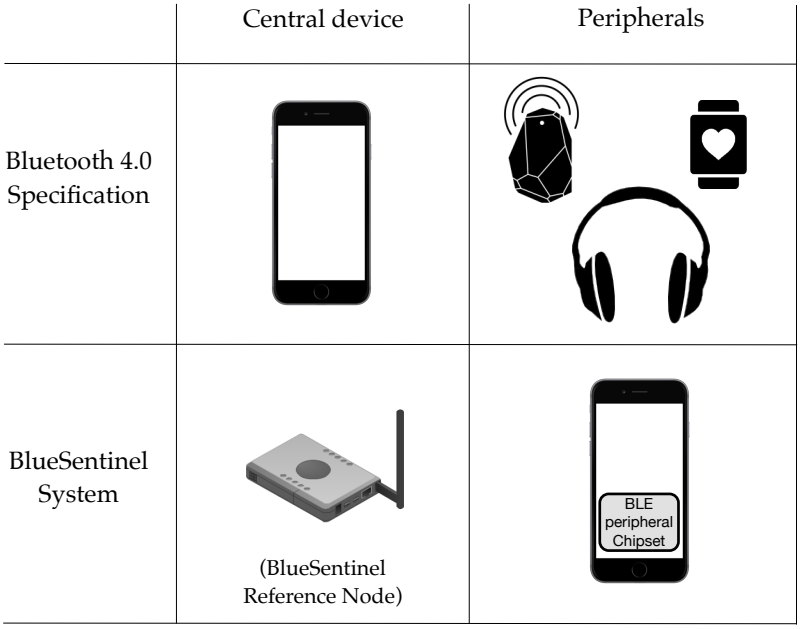

Figure 2.1: Central and Peripheral devices in BLE specification compared with the implemented system.

The second possible way to communicate is using *Advertisement*, where a BLE peripheral device broadcasts packets to every device around it. The receiving device can then act on this information. Advertising that is by

design unidirectional can be performed exclusively by peripheral devices. For localization purposes, advertisement devices known as beacons as been employed as reference nodes; however, for localization purposes, this approach leaves the burden on the mobile device, forcing it to run positioning algorithms continuously and notify the result through energy-hungry connectivities (more details in section 1.3.7).
The key idea behind the proposed system is to use smartphones as BLE peripheral devices, while reference nodes as BLE central devices (fig. 2.1). In this way, user devices are able to advertise them-self in a given position of the building without any connection or pairing, while reference nodes collect all the received advertisement packets.

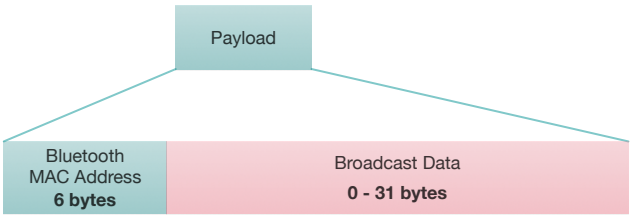

Figure 2.2: BLE advertisement packet contents.

As shown in figure 2.2, advertisement packets contain the Bluetooth MAC address of the device and a very limited custom payload of 31 bytes. Since we need to identify the user, a straightforward solution would be to associate each MAC address with a registered user. However this is no longer possible on recent versions of Android and iOS since they hide MAC addresses to developers in order to increase user's privacy. For this reason, a Universally Unique IDentifier (UUID) has been employed to recognize each user advertisement. At the same way in which UUIDs have been used to advertise regions through BLE beacons (like the Apple iBeacon), occupants of a building can be localized and identified in real-time associating each UUID with a building user.

BLE advertisement on mobile devices requires a specific hardware, the BLE peripheral mode chipset. Fortunately, almost all current devices have built-in peripheral chipset: all Apple devices since iPhone 4S and iPad 3, about 80% of all Android devices released in 2014 and about 90% in 2015 [36].

The proposed system has been developed using smartphones as advertising devices. However, The same advertising task can be easily implemented on wearable devices like smart-watch, that are natively build as BLE peripheral devices. The employment of wearable devices for occupancy

monitoring would be even more effective, since they never separate from their users.

2.4 System overview

In this section, the architecture of the proposed system will be overviewed, while in the next three sections every components will be explained in detail.

The proposed location monitoring system is composed by three main parts (figure 2.3).

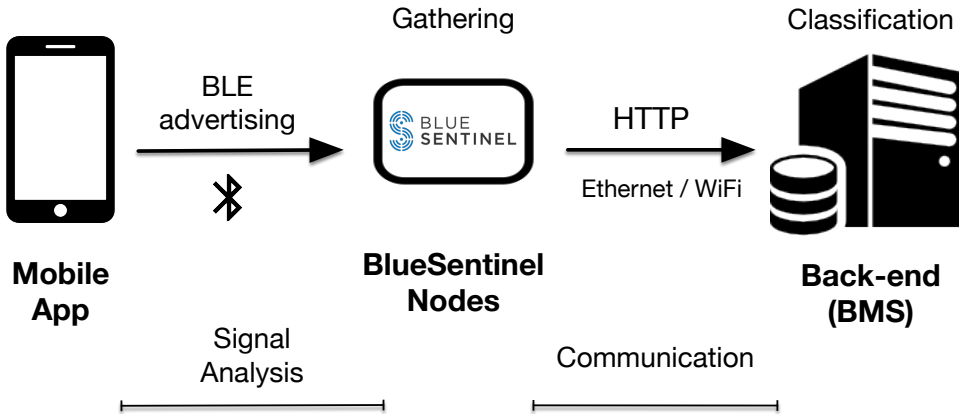

Figure 2.3: Overview of the BlueSentinel occupancy monitoring system. A mobile application perform advertising over BLE. Packets are collected by a network of sensor nodes and analyzed by a centralized system.

- First, the **mobile application** that runs on users mobile devices. The application manages registrations and logins of users (necessary to detect occupants identity) and performs the BLE advertisement. For both Android and iOS devices, it's possible to perform BLE advertisement even during device sleep and when the application is in background. During BLE advertisement the application is free from any localization algorithm, Internet connection or GPS service, to preserve battery and to bother users as least as possible.
- The component that directly interacts with occupants mobile devices is a network of Bluetooth Low Energy **receiver nodes**. This nodes, positioned by the building administrator during the system installation, are in charge of collecting BLE packets coming from

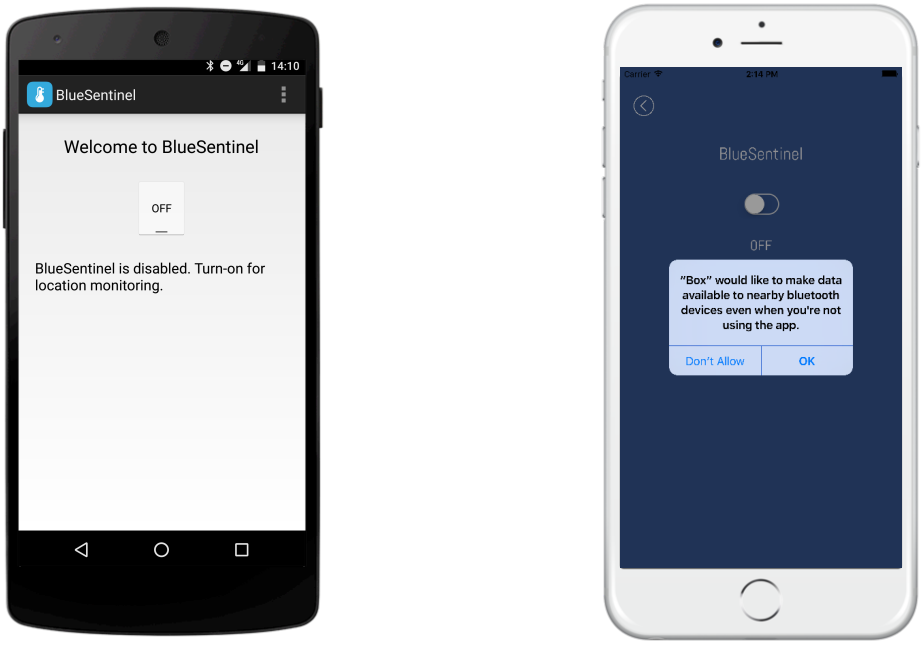

Figure 2.4: Mobile application developed for Android and iOS devices.

occupants, together with signal features (like the received power) useful to estimate their position. The information collected by each node is rapidly forwarded through an Internet connection toward a centralized Building Management System.

- All the information collected by the sensor nodes are merged at the back-end level by the **Building Management System** (BMS). Here signal features related to every occupant is reconstructed and compared to a database of training data to estimate their position. In particular, different classification algorithms have been implemented to locate each users in the best fitting zone or room. Real-time and historical data occupancy is then exposed by the BMS for a web-based monitoring and third-party applications through a REST API.

2.5 Mobile Application

A mobile application for both Android and iOS devices has been developed for BlueSentinel users and released on the respective store [34, 35] (Figure 2.4).

The application manages users registrations to the system, asking for e-mail, username and password. Registration has been set up as mandatory; in this way the system is enabled for user identification, but also to build historical data and occupancy probability profiles per single occupant.
For each registered user, the system generates an identification string of 10 characters (for example `userId=tslJp2hnqf`). This identifier will be used for both database key and advertisement UUID (Universally Unique IDentifier).

The BlueSentinel menu let the user manage the Bluetooth functionality. Here BLE advertisement can be turn on and off. In the Android environment BLE advertisement in background is supported by simply implementing the `BluetoothLeAdvertiser` class in an Android Service. The essential code for the advertisement is reported in listing 2.1.

Listing 2.1: BLE advertising of user identification for Android devices.

```
1   private void startAdvertising() {
2       // Retrieve user identification string from database
3       String userId = ParseUser.getCurrentUser().getObjectId();
4       ParcelUuid pUuid = new ParcelUuid(UUID.fromString(
    UuidFromString("BLSNTL" + userId)));
5
6       bleAdvertiser = BluetoothAdapter.getDefaultAdapter().
    getBluetoothLeAdvertiser();
7       AdvertiseSettings settings = new AdvertiseSettings.Builder()
8               // Balanced transmission frequency
9               .setAdvertiseMode(AdvertiseSettings.MODE_BALANCED)
10              // High transmission power
11              .setTxPowerLevel(AdvertiseSettings.TX_POWER_HIGH)
12              // No time limit
13              .setTimeout(0)
14              .setConnectable(false)
15
16      AdvertiseData data = new AdvertiseData.Builder()
17          .addServiceUuid(pUuid)
18              .setIncludeDeviceName(false)
19      bleAdvertiser.startAdvertising(settings, data, advCallback);
20  }
```

User identification string is retrieved from database, converted into a 128-bit UUID and added to the advertisement data. An identification token (`"BLSNTL"`) is used as prefix in the UUID; the purpose of the prefix is to discriminate BlueSentinel UUIDs from packets advertised by unrelated devices in the environment, such as BLE headphones or wristbands. Considering 6 chars for the prefix, and the 10 chars of the id, the string of 16 characters can be contained in the 16 bytes of the UUID using the 8-bit

ASCII encoding. The resulting Bluetooth packet is represented in figure 2.5.

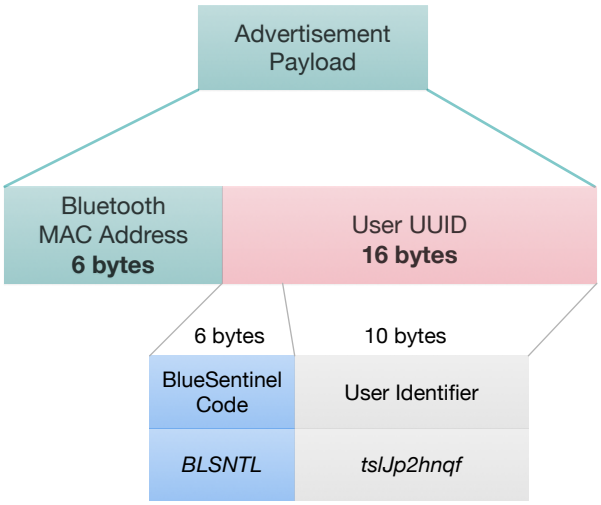

Figure 2.5: Detail of the BLE packet advertised by each BlueSentinel user.

In iOS SDK beacon functionalities are implemented in the CoreLocation framework, which supports all that it is needed to replicate beacons through an iDevice. However, the CoreLocation framework places many limits on how it can be used in the background.
The application has been implemented exploiting the CoreBluetooth framework, that has much greater background support of iOS apps. With CoreBluetooth, it's possible to broadcast as a peripheral and detect as a central while in the background. The Objective-C code that performs the advertising is similar to the equivalent 2.1, where the `CBPeripheralManager` take the place of the `BluetoothLeAdvertiser` class.

In addition to the application for building users, a separate application has been developed for the building administrator in order to perform the training of the classification algorithm. The training requires the creation of a label for each building zone; then the administrator needs to travel across each building zone, stop for some seconds and declare the correct room label. Meanwhile, on the server side, the training phase creates a set of RSS samples, each one of them labeled with the corresponding ground truth location.

2.6 Wireless Sensor Network

On the building side, the system requires a set of hardware nodes deployed in the indoor environment able to collect BLE packets and contact a centralized server. The node prototype has been developed using Raspberry Pi 3, one of the most cheapest board that supports all the necessary connections. In particular, the mentioned single-board computer provide a BLE module, and both WiFi and Ethernet for Internet connections.

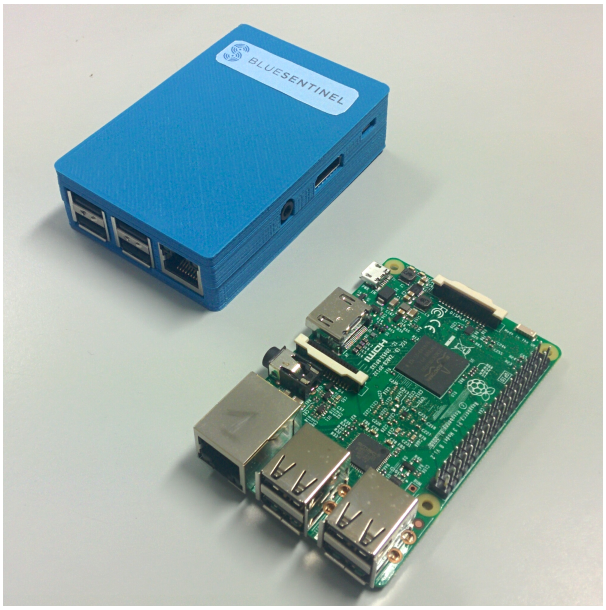

Figure 2.6: Prototype of BlueSentinel sensor nodes using Raspberry Pi 3 boards.

Raspberry boards support a Debian-based Linux distribution, that has been used for the BlueSentinel nodes. The routine that runs continuously on each board has been written in Python, since the most supported library to manage BLE connections is provided for that language. The routine is divided in three concurrent threads:

- **BLE scan**: a thread is in charge of collecting every single BLE advertising signal coming from BlueSentinel users. Received packets are first filtered, searching for the BlueSentinel token inside the packet payload. All the unidentified signals are discarded, while the useful packets are stored in a queue. In particular, for each BlueSentinel packet, the thread stores the user Id, the corresponding Received

Signal Strength (RSS) expressed in dBm (Decibel-milliwatt), and a timestamp representing the receiving instant.

- **Filtering**: Like any other indoor radio-frequency signal, BLE advertisement suffer from signal fluctuations due to multipath fading. This cause a signal to fluctuate over time even if the transmitter is stationary at a fixed distance from the receiver. To reduce the error caused by fluctuations a low-pass filter is applied to the signal received by each user. The filter implemented is the Exponentially Weighted Moving Average (EWMA) that applies weighting factors to the RSS which decrease exponentially over time. Since any filter applied to smooth the signal will always introduce a delay, the EWMA filter has been selected to prioritize more recent RSS samples. The value V_t is computed at time instant t with the following equation:

$$V_t = \alpha \cdot RSS_t + (1 - \alpha) \cdot RSS_{t-1} \tag{2.1}$$

  The weighting for each older datum decreases exponentially, never reaching zero. The α coefficient has been fixed to 0.75.

- **Data push**: the third thread is is in charge of pushing RSS values periodically to the server at regular intervals. In particular, in each cycle and for each user detected, this thread send the user id, the filtered RSS values, and the most recent timestamp within the sliding window of the EWMA filter. The choice of the timestamp is given by the weighting factor that emphasizes exponentially the more recent sample. The server is contacted with a HTTP request. After that data is pushed to the server, the thread empty the samples queue, so that new signals coming from BLE scan will be evaluated in a new filtering window. After some attempts with different intervals, the period has been set to 1.5 seconds.

Every nodes has been connected to the building LAN infrastructure and set up to be remotely accessed using SSH. Every relevant operation executed by the mentioned threads has been logged in a file, in order to check

2.7 Back-end

The system back-end has been set up using Parse-Server, an open-source API server based on the Express web application framework. The REST

API has been exploited to interact with Parse Server from the mobile applications, sensor nodes, and a web page. The JavaScript SDK provided by Parse Server (called Cloud Code) has been used to manage the creation of database objects and the localization algorithm. The system back-end has been structured to provide the building occupancy information through the REST API to any third party components of the Building Management System able to send HTTP requests.

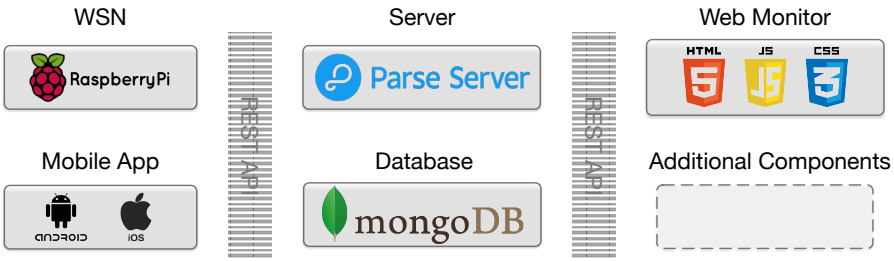

Figure 2.7: Platforms employed for the system back-end and the external components.

2.7.1 Database

The database that supports the system has been implemented using MongoDB. The database contains user objects created during registration. Each zone (or room) of the building has its own instance, created during the training phase by the administrator application. The ReceivedSignal object is created directly by the sensor nodes. It represents the reception of a BLE signal, by a given node. The object is created with the user id founded in the signal, and the received signal strength. The ReceivedSignal data is used by the fingerprinting algorithm to compute each user position. Whenever this happens, the outcome is stored using the UserPosition object.

Table 2.1: BlueSentinel back-end database

Table	Fields				
User	userId	email	username	hashedPwd	createdAt
Zone	zoneId	floor	maxCapacity	label	createdAt
Node	nodeId	zoneId	floor	label	createdAt
ReceivedSignal	userId	nodeId	RSS	timestamp	createdAt
UserPosition	positionId	userId	RSS	timestamp	createdAt
TrainingSet	zoneId	RSS[ ]	timestamp[ ]	createdAt	

2.7.2 Fingerprinting Algorithms

2.8 Integration with a Thermal Comfort System

2.9 Real-World Deployed System

2.10 Web-Based Monitoring Dashboard

2.11 implementation Decisions

2.12 Limitations

Chapter 3

CAD Deployment for Indoor WSNs

Over the last years, many smart buildings applications, such as indoor localization or safety systems, have been subject of intense research. Smart environments usually rely on several hardware nodes equipped with sensors, actuators and communication functionalities. The high level of heterogeneity and the lack of standardization across technologies make design of such environments a very challenging task, as each installation has to be designed manually and performed ad-hoc for the specific building. On the other hand, many different systems show common characteristics, like the strict dependency with the building floor plan, also sharing similar requirements such as a nodes allocation that provides sensing coverage and nodes connectivity. The work explained in this chapter consists in a CAD application for the design of smart building systems based on the installation of hardware nodes across the indoor space. The tool provides a site-specific algorithm for cost-effective deployment of wireless localization systems, with the aim to maximize the localization accuracy. Experimental results from real-world environment show that the proposed site-specific model can improve the positioning accuracy of general models from the state-of-the-art. The tool, available open-source, is modular and extensible through plug-ins allowing to model building systems with different requirements.

3.1 The Nodes Deployment Problem

Many smart buildings applications are based on indoor localization techniques, using location information to optimize the environment and provide context-aware services. Indoor localization systems often require the presence of wireless devices such as Access Points (AP), in order to

let the user identify its position by means of a mobile device. Most smart building applications have been developed in order to achieve sustainability, reducing energy waste related to energy-consuming appliances like Heating, Ventilation, and Air-Conditioning (HVAC). Some examples are [13] and [4]. Smart HVAC systems usually rely on a set of ambient sensors able to collect indoor values of temperature and humidity. This allow the control system to build thermal maps of the indoor environment, locate thermal complaint feedbacks coming from the tenants and regulate only the necessary portion of the physical system. Another target feature of complex buildings is safety, characterized by the ability to respond to crisis events limiting damages and victims. These systems are able to detect safety threats, for example from smoke detectors or heat detectors. Also in this scenario, a proper allocation of sensor nodes is essential to detect and locate the threat responsively.

The position of each node strongly affects the performance of the system, since a bad allocation could lead to unmonitored areas. The number of nodes employed, besides weighting on the installation cost, also burdens the overall energy consumption of the system, a key parameter to consider especially for energy saving systems. The choice of the hardware nodes can get more difficult by the availability on the market of several devices and components that differ in cost, power consumption and maximum range distance. Although the key role of nodes allocation, many smart building systems proposed in literature don't consider nodes amount and positioning problems in environments that differ from the original testbeds.

Without a systematic approach the design space is not well explored, which leads to inefficient solutions. In this context, the development of tools able to automatize part of the design flow of smart building systems is essential. In order to find a near-optimal allocation of nodes, the knowledge of the floor plan is required. However, for installations performed on existing buildings, administrators can encounter difficulties in obtaining the floor plan in an easily-interpretable digital format.

To address these problems, we developed a CAD tool to assist building designers during the design of smart building systems. The application manages common requirements like the building floor plan specification. We decided to implement a node allocation algorithm for three different indoor localization systems, that searches for near-optimal allocations of nodes, from mixed hardware types, with the aim of keeping low the total cost. Due to the high level of heterogeneity and lack of standardization across systems to design, we make the system extensible through plug-ins

to let new functionalities being integrated into the system. The tool[1] is developed within the QCAD[2] environment, an open-source computer-aided drafting application. The key contributions of our work can be summarized in the following:

- A traditional CAD interface to specify both the physical building floor-plan and the functional components of the smart environment.
- An algorithm for hardware nodes allocation that provides to designers a near-optimal placement of devices. The algorithm explores combinations of different types of nodes to obtain cost-effective solutions.
- A site-specific model for wireless indoor localization accuracy optimization that keeps into account the actual structure of the building.
- The integration of the tool within an open-source[3] application framework able to extend the system by means of JavaScript or C++ plug-ins.

3.2 Related work

Building Information Modeling (BIM) is a consolidate process to support building constructions and renovations. BIM softwares, and in particular CAD for buildings such as ArchiCAD [32], focus on the generation and management of digital representations of the physical aspects of places. BIM tools can coordinate architectural and structural requirements, for essential tasks such as collision detection [27]. Materials employed for a construction can be represented with extremely high levels of accuracy, thanks to the several libraries developed in many years, resulting in precise cost estimations [26]. With the diffusion of integrated smart systems built to increase comfort and efficiency, buildings require the design of aspects that go beyond the mere physical design. The concept of smart environment is becoming more and more concrete with the integration of sensors, actuators and computational elements in buildings, while tools able to model smart and interactive functionalities of modern buildings are currently lacking.

The problem of the allocation of hardware nodes in a given environment can be compared, on first approximation, by the maximal cover location

[1] A video demo of the tool has been published at https://youtu.be/6c6D6wolDBQ

[2] QCAD - Open Source CAD System: http://www.qcad.org/

[3] The source code of the system is open-source and available at https://bitbucket.org/necst/box-smartcad

Table 3.1: Comparison between proposed deployment methods and tools for indoor WSN and Access Points based systems.

Deployment	Site Specific vs General Model	Heterogeneous Nodes	Application Integrated	Extensible
Zhao et al. [29]	General Model	No	Yes	No
He at al. [16]	General Model	No	No	No
Fang et. al. [15]	Site Specific Model	No	No	No
Proposed approach	Site Specific Model	Yes	Yes	Yes

problem (MCLP), i.e. the problem of covering the maximum amount of demand locations with a given number of facilities. Similarly, the location set covering problem (LSCP) consists in finding the minimum set of facilities that covers all available demand locations. Each facility has the same coverage radius r; a demand point is assumed to be covered if it is within distance r of a facility. Daskin et al. gave a general formulation of the LSCP [10] and re-formulated it for network systems and emergency vehicle deployment [11].

The maximum sensing coverage region (MSCR) is a special case of the previous two problems that focuses on the research of an allocation of wireless nodes that guarantees both sensing coverage and network connectivity between nodes [24, 25]. In this scenario the placement need to take care not only of the sensing range, but also of the communication range of each node.

For what concern the allocation in indoor environments, only minimum literature has been published so far to the best of our knowledge. Zhao et al. in [29] proposed an access point (AP) positioning model based on the Differential Evolution algorithm, specific for fingerprinting localization techniques. Their model focuses on increasing the diversity of the received signal array along the indoor locations, and thus improving the positioning accuracy of fingerprinting schemes. However, the model does not take into account the effect of walls or other obstacles present in the target environment. He at al. in [16] made use of a genetic algorithm for APs deployment model, to study the relationship between positioning error and signal space Euclidean distance. Again, the simulation results show that

the error can be reduced increasing the Euclidean distance between the RSS (Received Signal Strength) arrays of different locations. Fang et. al. in [15] proposed a tool for linking the placement of APs and the positioning performance. Their algorithm maximizes signal-to-noise ratio (SNR) i.e. maximizes the signal and minimizes the noise simultaneously. However, the system is developed in a real-world environment, and requires measurements with different access point allocations that can be an expensive and time-consuming task.

A common limitation of many works described previously is the employment of simple and general models which doesn't take into account the actual layout and geometry of the building. The free-space path loss propagation model is often used despite the presence of fixed obstructing objects like walls. Of course, none of the cited works provide a convenient way to specify geometric layout of the indoor environment. This leads the authors to validate models simply using squared or rectangular areas to represent the indoor environment, omitting the relationship between irregular areas and system coverage. In addition, none of the existing solutions takes in consideration different hardware characteristics and costs of the nodes to be deployed.

3.3 The proposed Application Framework

Our system has been developed on top of the QCAD application framework. The QCAD Application Framework consists of programming libraries and resources that provides CAD specific functionalities. An example of module provided by the QCAD Application framework is the Math module that implements mathematical concepts such as vectors or matrices as well as basic geometrical classes like points, lines and so on. The QCAD Framework has been enhanced with a *SmartBuilding* module that provides some fundamental functionalities for the design of smart building systems. The module include abstract entities like rooms, walls, sockets, sensor nodes and gateways. User interface components are also provided in order to create and edit this entities (*tools*) and to specify parameters (*widgets*). Our module implements a node deployment algorithm for three commons indoor localization systems, that will be discussed later. The whole application rely on Qt, a framework that covers a lot of generic and low level functionality for desktop applications and not directly related to CAD.

The QCAD Application Framework offers a very complete and powerful ECMAScript interface. The SmartBuilding module, as well as the QCAD

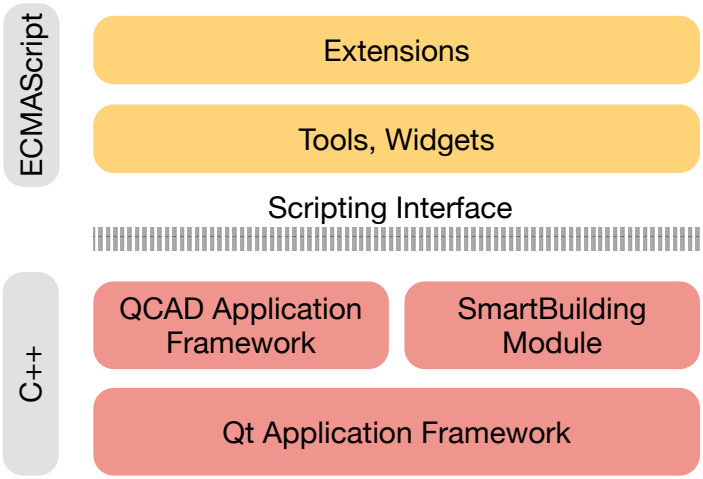

Figure 3.1: Overview of the application stack. The script interpreter features standard ECMAScript functionality and on top of that provides additional classes from the Qt API, QCAD API and the *SmartBuilding* module.

Application Framework, is accessible through that scripting interface. Through the ECMAScript interface developers will be able to extend the whole application in an easy and very efficient way. The choice of a popular script language that is easy to learn enables anyone with previous programming experience to extend the application. Such extensions can for example be CAD related interactive tools like an HVAC layout construction widget, or a temperature sensor nodes deployment algorithm.

In some situations extending QCAD through scripts alone may not be possible. This is mostly the case, if the extension is based on an existing C or C++ library. In that case, it is possible to create a C++ plug-in that wraps the existing library and adds the necessary hooks to access library functionality through the script interface. Such a plug-in will be automatically loaded by QCAD on start up to add functions and classes to the script interface of QCAD. These script extensions can then be used by a script add-ons to make that functionality available as part of the application interface.

3.4 Nodes deployment for indoor localization

Smart environments always rely on a set of hardware nodes able to collect sensing data and communicate through cabled or wireless technologies. The number of nodes employed and the position of each one strongly affect

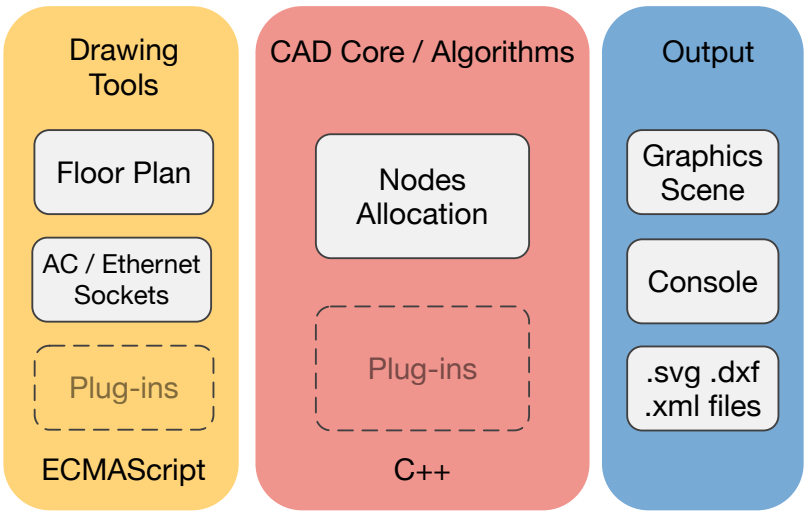

Figure 3.2: Functional overview of the system components. Drawing tools and algorithms for systems deployment and simulation are extensible through ECMAScript or C++ plug-ins.

the overall performance of the system as well as the cost of installation. In our work, indoor localization systems have been taken as the main case study for the nodes allocation, since occupants localization and monitoring is one of the most common requirements of different smart environments.

The way in which the indoor environment must be covered by the nodes depends on the particular technology implemented, however there can be identified three main manners:

(1) Single Coverage, i.e. to monitor the state of the environment with a single node for each location inside its radius. This includes for example to detect the presence of a mobile device in a proximity region [2], or to detect a RFID tag within the tags reader range [21].

(2) Trilateration, to compute the position of a mobile device. This technique requires the reception of a wireless signal of at least three reference sensors with well-known positions everywhere within the covered area. We define the term k-coverage as the minimum number of sensors (or reference nodes) required in each location by a system. Single coverage systems have k-coverage $= 1$, while for trilateration $k = 3$.

(3) Fingerprinting, where the number and the strength of the received signals is not fixed, but affect the localization accuracy.

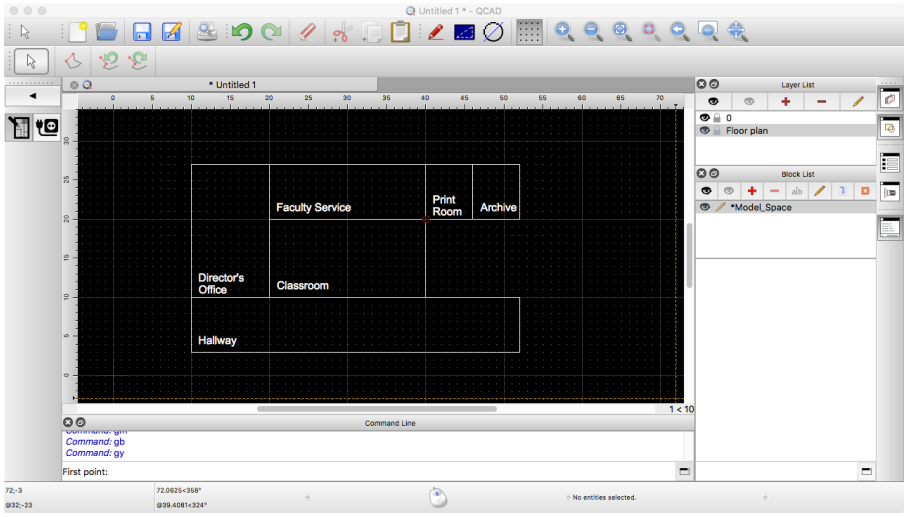

Figure 3.3: Floor-plan design tool. User can specify the layout of the rooms and a possible set of candidate sites for the node placement.

Trilateration and fingerprinting usually exploit wireless technologies as Wi-Fi or Bluetooth to establish a connection between mobile and stationary nodes. Sensing regions can refer to any type of ambient sensors, such as Passive Infrared Sensors (PIR) [14], remote thermal sensors [6], but also proximity based radio transmitters such as RFID tag readers [28] and Bluetooth Low Energy transmitters (BLE beacons) [9].

3.5 The proposed deployment tool

As we previously said, smart environments always rely on a set of sensor nodes, each one able to communicate through cabled or wireless technologies. Also for outdoor WSNs, a key challenge is how to achieve coverage of the target monitoring space and sufficient network connectivity between sensor nodes. Usually each sensor mote communicates with the rest of the network through technologies like Wi-Fi or Zig-bee. Additional issues for outdoor WSNs are the limited battery life of each node and the power consumption required for packet transmissions. Given the availability in most (also "non-smart") buildings of power outlets, Ethernet sockets and Wi-Fi signal, the mentioned limitations of WSNs can be solved in indoor application making use of the existing infrastructure. Differently

from outdoor WSN deployments, where coverage and connectivity are always treated together, our system leaves nodes connectivity optional, focusing on providing the coverage service to the indoor locations.

The design process starts with a drafting phase in which the user specify the building floor plan as a set of rooms. During this phase the designer can restrict the possible sites for nodes allocation, selecting a set of candidate points. This can be useful when the hardware devices require power supply or Ethernet connectivity. The design interface used for both map and candidate sites specification is reported in Figure 3.3.

In our model, we will refer to L as the entire set of monitoring locations to be covered, while J as the set of deployable locations where nodes can be placed. By default, $L = J$ and nodes can be positioned everywhere but as we said the set J can be restricted only to specific candidate points.

After the design phase, different parameters are provided by the administrator and used to define a domain in which search for a covering solution. The parameters are:

- The covering technique (single, trilateration or fingerprinting) that will be used to cover the locations in L;
- A cost c_t for every type $t \in T$ of node available on the market (expressed in dollars);
- A working range r_t for every type t of node (expressed in meters);
- A percentage of covered area required, called *target* (i.e. the minimum percentage of locations $l \in L$ to be covered by the solution).

The system will return to the designer a set N of nodes n_{jt} (possibly with mixed hardware types) and their position on the building map. The outcome will have the lower cost of installation among all the inspected solutions that satisfy the *target* percentage of covered area. Figure 3.4 shows an overview of the process explained so far.

3.5.1 Covering Techniques

Our tool provides three different ways to cover the floor-plan space, each one identified by the technique required by the system that will be installed.

(1) Single coverage, that guarantees from each position the presence of at least one reachable node. This is used for example to detect the presence of a mobile device in a proximity region. In our model, a

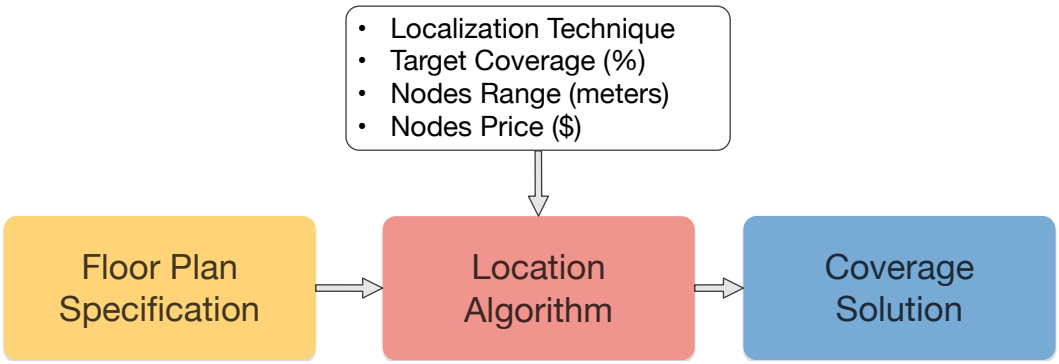

Figure 3.4: System Process. After the design of the floor plan, different parameters are used to define the search for an optimal allocation of nodes.

location l of the floor-plan is considered covered if exists at least one working node n of type t within a range r_t. An example is shown in Fig. 3.5.a.

(2) Trilateration is the process of determining the position of a point measuring its distance from three reference nodes, exploiting geometric properties of triangles. Usually, indoor trilateration systems use the strength of the signal received from a node to estimate its distance. In our model, a location l of the floor-plan is covered for trilateration if there exist at least three working nodes n_1, n_2, n_3, each one no more distant then its corresponding range r_t. A location l served for trilateration is shown in Fig. 3.5.b. Although we refer only to trilateration, the same exact result can be used also for triangulation, the technique where angles are measured instead of distances.

(3) Fingerprinting: this technique is used to estimate the position of a mobile device based on its received signal strength (rss) vector. Each location receives the signal from k nodes, where k is not the same for all locations, but depends on how many nodes are reachable from that particular location. Each one of the k signals reaches the receiving antenna with a given power (or rss). For example, the location l shown in Fig. 3.5.c perceives $k = 2$ signals so that $rss_{l,1} < rss_{l,2}$. We denote as $rss_{l,n}$ the signal strength received at location l from a node n. The vector $\mathbf{rss_l} = [rss_{l,1}, ..., rss_{l,k}]$ of the k signals received at run-time in location l is compared with a dataset of vectors, each one pre-labelled with the corresponding position.

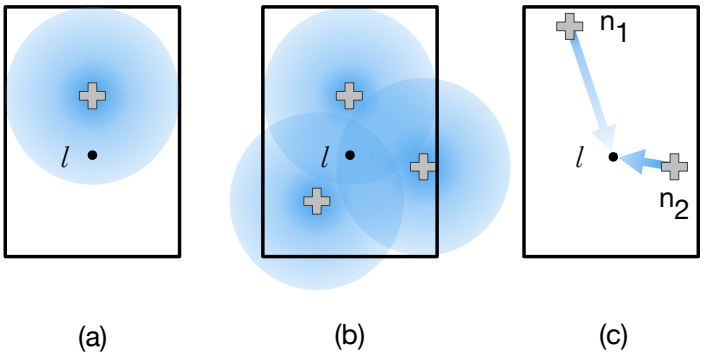

Figure 3.5: Sample floor-plans with a location l covered (a) in single mode, (b) for trilateration, and (c) for fingerprinting where $rss_{l,1} < rss_{l,2}$.

The comparison is usually performed by a classification algorithm using the Euclidean distance of the vectors, since rss vectors with a small Euclidean distance between them are more likely to be close also in the physical space. We have defined as $rss_{l,n}$ the signal strength received at location l from a node n. The Euclidean distance between $\mathbf{rss_a}$ and $\mathbf{rss_b}$, both composed by k received signals, and collected respectively in location a and b is defined as:

$$E(a,b) = \sqrt{(rss_{a,1} - rss_{b,1})^2 + ... + (rss_{a,k} - rss_{b,k})^2} \tag{3.1}$$

Consider the vector $\mathbf{rss_a}$ as the run-time sample, while the vector $\mathbf{rss_b}$ retrieved from the stored fingerprint. The smaller is the $E(a,b)$, more confident is the localization system approximating current location of a with the stored location of b.

It has been demonstrated that maximizing the Euclidean distances of the rss arrays between all sampling points, the positioning accuracy of wireless localization systems can be improved [16, 29]. In Figure 3.6 is reported a graphical demonstration of the aforementioned statement. Take as an example a dataset $(DS1, DS2, DS3, DS4)$ of stored **rss** vectors, where each vector is bi-dimensional $(K = 2)$ and coupled with the corresponding physical position. Fig. 3.6.a shows each element of the database where the Cartesian coordinates corresponds to components rss_1, rss_2. Although the plane does not represent the physical area the floor-plan, database elements that are near between them are more likely to be close also in the physical space. Given a run-time element R, each arrow represents the Euclidean distance $E(R, DS_i)$ from the surrounding dataset elements. A localization

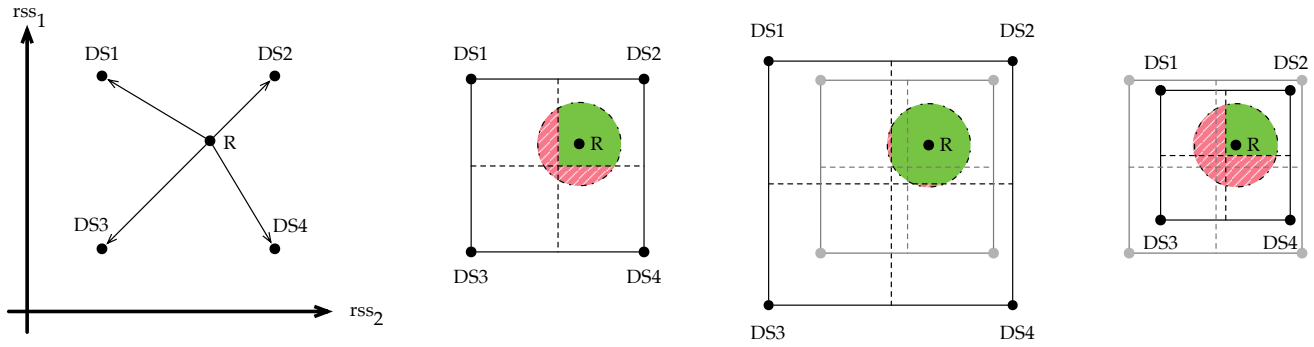

Figure 3.6: In diagram (a) bi-dimensional elements of the localization dataset are represented in Cartesian coordinates corresponding to components rss_1, rss_2. A run-time sample R is showed in (b) where its circular area delineates run-time signal fluctuations. If $DS2$ is the nearest points to R in the physical space, green area is proportional to the probability of correct localization, while red dashed area represent wrong localizations. In diagram (c) the Euclidean distance between sampling points has been increased, improving the correct localization, while diagram (d) shows the opposite effect.

algorithm can exploit the Nearest Neighbor technique to approximate the position of R with the nearest dataset element. Unfortunately, the run-time rss measurement of R will not be constant over time, but will experience continuous fluctuations due to environmental noise. These fluctuations make the sample R move randomly to the surrounding points. Suppose that $DS2$ is the nearest points to R in the physical space. Fig. 3.6.b shows with a green area the probability to assign R the correct (or more accurate) position, while a red (with line pattern) area represents the probability to get a wrong position from the system. Fig. 3.6.c demonstrates how an increase in the rss Euclidean distance between sampling points increase the red area and the accuracy of the localization, while in Fig. 3.6.d an Euclidean distance reduction will lead to poorer localizations.

The received signal strength has been estimated using the The WINNER II path loss model [19]:

$$PL = A \ \log_{10}(d[m]) + B + C \log_{10}(\frac{f_c[GHz]}{5.0}) + X \tag{3.2}$$

where PL is the signal path loss (in dB), f_c is the frequency in GHz, d is the distance between the transmitter and the receiver location in meters. Values of coefficients A, B, C, X change depending on LOS (Line-Of-Sight) or NLOS (Non-Line-Of-Sight) propagations, and are reported in table 3.2. The propagation model has been used in fingerprinting coverage to maximize the Euclidean distance of the rss vectors between a location and its surrounding points, with the aim of improve the localization accuracy of the system.

Table 3.2: Values of coefficients depending on LOS (Line-Of-Sight) or NLOS (Non-Line-Of-Sight) propagations. Values have been taken from the WINNER II path loss model [19].

Scenario	**Path Loss Coefficients**
LOS	$A = 18.7, B = 46.8, C = 20$
$NLOS$	$A = 36.8, B = 43.8, C = 20$ $X = 5(n_w - 1)$ (light walls) $X = 12(n_w - 1)$ (heavy walls)

The two-dimensional space of the floor plan is discretized with a length unit (default is $1m$) that is chosen by the user during the map specification phase.

As we have said, in addition to location coverage, also nodes connectivity has been modeled. In our model, a sensor node n is connected if exist a connected path to the gateway node. To ensure the connectivity of the whole network, the following equation must hold.

$$\forall n \in N, connected(n, gateway) = true \tag{3.3}$$

where

$$\begin{gathered} connected(n, n') \stackrel{def}{=} |(n, n')| \leq min(h, h') \\ \vee \exists\ n_1, \ldots, n_i \in N\ (1 < i), \\ |(n, n_1)| \leq min(h, h_1) \\ \wedge\ |(n_1, n_2)| \leq min(h_1, h_2)\ \wedge\ \ldots \\ \vee\ |(n_i, n')| \leq min(h_i, h') \end{gathered} \tag{3.4}$$

Connected networks are managed by our allocation algorithm in the same way of non-connected networks, with the following exception:

- first, a manual gateway nodes allocation is required;
- during nodes allocation, deployable points J are restricted to locations j' such that $connected(n_{j'}, gateway) = true$;
- during deployment optimization, nodes moves are considered feasible only within the connected area.

3.6 Covering Location Algorithm

The covering location algorithm has the purpose of placing an optimal set of nodes on the building floor plan. We have decided to implement a modified version of the Multimode Covering Location Problem [7], a generalization of the MCLP. Using a quite general and flexible reformulation of the covering problem, we have been able to adapt the algorithm at the different covering techniques described previously.

The positioning algorithm is composed by a first *Greedy* procedure, whose solution is then improved by a *Variable Neighborhood Search* (VNS) algorithm. The positioning algorithm evaluates different solutions using a reward b_l, that is defined for each location l and will be earned only for the locations covered in that particular solution. The value of the reward depends on the coverage technique:

- Single coverage: the reward b_l will be earned if there is at least one node that covers l.

Table 3.3: Notation and meaning of symbols used for the model.

Notation	Meaning
L	set of monitoring locations
J	set of deployable locations
c_t	cost of a node of type t
r_t	sensing range of a node of type t
h_t	communication range of a node of type t
$target$	coverage rate of L required by user (%)
n_{jt}	nodes of type t allocated in j
$rss_{l,n}$	signal strength received in l from n
$\mathbf{rss_a}$	vector of all the $rss_{a,n}$ values collected in a
$E(a,b)$	Euclidean distance between $\mathbf{rss_a}$ and $\mathbf{rss_b}$
D_l	set of locations no more distant than d from l
z	average signal space Euclidean distance
Z	objective function
b_l	reward earned for covering location l
w_l	reward weighted on the node cost
x_{jt}	allocation of node with type t in j (binary)
a_{ljt}	reachability of n_{jt} from location l (binary)
k-coverage	number of ref. nodes required by the system
k_l	current number of ref. nodes covering l
S	min. signal space Euclidean distance threshold
s_{min}	minimum number of node moves in *shaking* procedure
s_{max}	maximum number of node moves in *shaking* procedure
$Rmax$	number of restarts of the *VNS* algorithm

- Trilateration: the reward b_l will be earned if there are at least three nodes that cover l.

- Fingerprinting: since this technique is often considered to be a trade-off (in cost and accuracy) between single coverage and trilateration, we decided that the reward b_l will be earned if there are at least two nodes that covers l.

As we have said, in order to maximize the localization accuracy of the system it's possible to increase the signal space Euclidean distance between the target points. Consider the mean Euclidean distance between the received rss vector in a certain location l, and the surrounding locations s within a certain distance d:

$$\frac{1}{\mid D_l \mid} \sum_{s \in D_l} E(l, s)$$
$$D_l = \{s \in L \mid distance(l, s) \leq d\} \tag{3.5}$$

The distance d is used to restrict the rss comparison and diversification only to the locations that are more likely to be erroneously confuse with l by the localization system. Fig. 3.7 shows an example of how the Euclidean distance of a location is compared to a neighbor location.

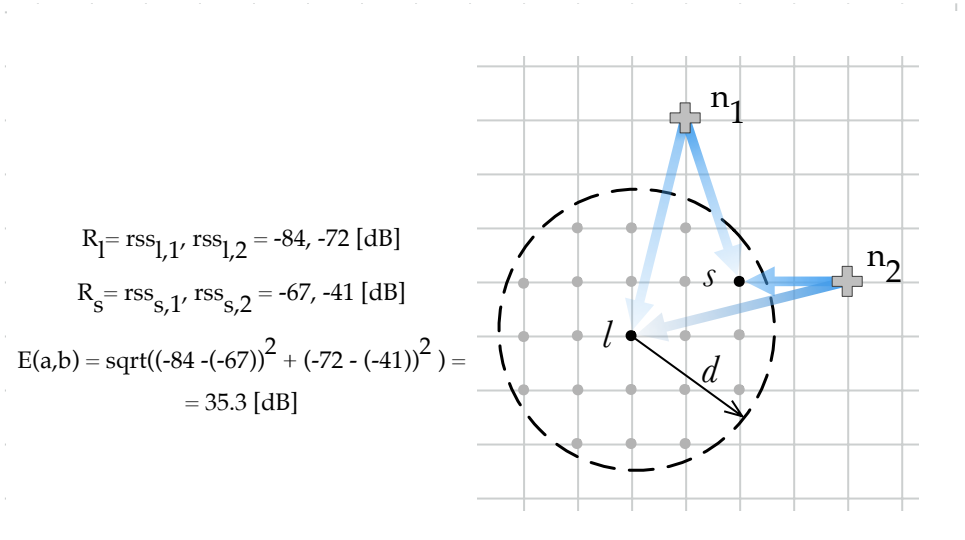

Figure 3.7: Regular grid showing how the mean Euclidean distance between the received rss vectors in a certain location l, and the surrounding locations s within a certain distance d.

We define the average signal space Euclidean distance z:

$$z = \frac{\sum_{l \in L} \sum_{s \in D_l} \frac{E(l,s)}{|D_l|}}{|L|} \tag{3.6}$$

The term z will be used by the *Greedy* procedure to produce a first solution with a reasonable allocation of nodes. Then, the value of z should be increased as much as possible to provide good localization accuracy to the system. However, maximize only the average does not seems fair enough, since a good system should provide a certain level of accuracy homogeneously among the target area. So we defined the objective function as difference between the term z and the signal space Euclidean variance:

$$Z = z - \sqrt{\sum_{l \in L} \left(\sum_{s \in D_l} \frac{E(l,s)}{\mid D_l \mid} \right)^2} \tag{3.7}$$

Maximizing the objective function Z, the intention is to provide as many target location as possible with a high signal space Euclidean distance with respect to the surrounding locations.

As we have previously introduced, we represent with L the entire set of location to be covered, while with J the set of possible positions where nodes can be placed. By default, $L = J$ and nodes can be positioned everywhere; however its possible to restrict the J set only to specific candidate points, that represent for example power outlets or Ethernet sockets. The problem of find a near-optimal set N of nodes n_{jt} (each one located in j and having a type t) with a coverage rate $f(N)$ that satisfies the *target* coverage, can be formalized as follows.

$$\max Z = z - \sqrt{\sum_{l \in L} \left(\sum_{s \in D_l} \frac{E(l,s)}{\mid D_l \mid} \right)^2} \tag{3.8}$$

$$f(N) \geq target \tag{3.9}$$

$$\sum_{t \in T} x_{jt} \leq 1 \quad \forall j \in J \tag{3.10}$$

$$x_{jt} = 1 \quad \Longleftrightarrow \; n_{jt} \in N \tag{3.11}$$

$$f(N) = |L| / \sum_{l \in L} y_l \tag{3.12}$$

$$\begin{cases} y_l \leq \sum_{j \in J} \sum_{t \in T} a_{ljt} x_{jt} & \forall l \in L \ (single) \\ 2\ y_l \leq \sum_{j \in J} \sum_{t \in T} a_{ljt} x_{jt} & \forall l \in L \ (fingerprinting) \\ 3\ y_l \leq \sum_{j \in J} \sum_{t \in T} a_{ljt} x_{jt} & \forall l \in L \quad (trilateration) \end{cases} \tag{3.13}$$

The decision variable $x_{jt} = 1$ represent the allocation of a node of type t in location j; a_{ljt} is equal to 1 if location l can be reached by a node of type t placed in j, and $a_{ljt} = 0$ otherwise. $y_l = 1$ if location l is covered, $y_l = 0$ otherwise. The constraint (3.10) set to one the maximum number of nodes that can be located in each site.

3.6.1 Greedy Procedure

The positioning algorithm starts with a *Greedy* procedure with the purpose of find a reasonable number of reference nodes, for both coverage and localization accuracy. The procedure generate a first solution N positioning a set of $k = |N|$ nodes, each one with a type $t \in T$. For all three coverage techniques, the reward b_l is weighted with the cost of the current node n^* selected for the coverage:

$$w_l = \frac{b_l}{c_t}; \quad \{n^* = n_{jt} \ \wedge \ distance(j, l) \leq r_t\} \tag{3.14}$$

The weighted reward w_l will be used by the *Greedy* algorithm so that on equal covered area, the cheapest node type has the priority over the others. We denote as L_{jt} the subset of locations that are reachable by a reference node n of type t placed at location j. At each iteration, the algorithm places a node n of type t^* at position j^* that covers the subset of locations $L_{j^*t^*}$ with the maximum reward. The term

$$1 - \frac{k_l}{k - coverage} \tag{3.15}$$

is used to prioritize the covering of locations with a lower 'temporary' k-coverage (called k_l) with respect to the k-coverage required by the current techniques. In this way, *Greedy* procedure tends to avoid the placement of nodes very close to one other which can lead, especially for trilateration systems, to poor localization accuracy. It's important to notice that the purpose of the *Greedy* procedure is to find a reasonable number of nodes for the localization service. The starting positioning is made on a best-effort basis, that will be improved by the successive *VNS*. After a node allocation, all subsets L_{jt} are updated according to the coverage

technique. In trilateration for example, a location l is removed from L_{jt} only if there exist, other than the current $n_{j^*t^*}$, other two nodes that are already covering l.

Algorithm 1 *Greedy*$(L, J, T, w, target)$

$N := \varnothing$;
$L_{jt} := \{l \in L \mid l \text{ is covered by node in } j \text{ with type } t\}$;
while $(f(N) < target) \wedge (z < S)$ **do**
  $j^* := \arg\max_{j \in J} \sum_{l \in L_{jt}} w_l \left(1 - \frac{k_l}{k-coverage}\right)$;
  $t^* := \arg\max_{t \in T} \sum_{l \in L_{jt}} w_l \left(1 - \frac{k_l}{k-coverage}\right)$;
  $N := N \cup \{n_{j^*t^*}\}$;
  $L_{jt} := L_{jt} \setminus L_{j^*t}$ for all $j \in J$;
return N;

The *Greedy* procedure ends when the $target$ coverage is satisfied, and when the average signal space Euclidean distance z reaches the threshold S. In our implementation we set the threshold $S = 4.5$ that has been proven to be the average Euclidean distance for which the positioning error is limited to 2 meters [16]. How we will see in section 3.7, the *Greedy* procedure is able to provide an average Euclidean distance not so far from the final best known. However, thanks to the low complexity of the *Greedy* procedure, additional time can be used to improve the solution. In addition, the Euclidean distance variance will be strongly improved.

3.6.2 Variable Neighborhood Search

The method called *Variable Neighborhood Search* (VNS) has been used to improve the solution coming from the *Greedy* procedure. The VNS approach empowers the classical local search framework with a restart mechanism that extends the search after a local optimum has been achieved by generating new starting solutions in progressively enlarged neighborhoods of the current best known solution. The key elements of the VNS (reported in algorithm 2) are a starting solution N with a hierarchy of size-increasing neighborhoods, and a local search procedure, i.e., the criterion to select the incumbent solution from the neighborhood. These components are used to restart the search every time that the procedure reaches a local optimum. Figure 3.8 shows an overview of the VNS process. A first local search procedure is applied to the solution produced by the *Greedy* procedure. At each iteration, the *shaking* procedure is used to

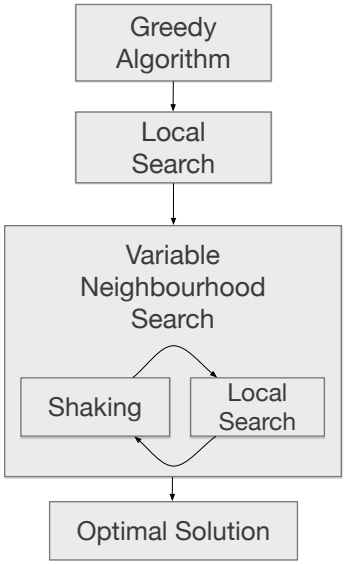

Figure 3.8: Location Algorithm. The solution found by the *Greedy* algorithm is improved applying iteratively a *Local Search* for an optimal solution and a *Shaking* procedure that perturbs the current solution.

generate a new starting solution, which is then improved by the execution of the local search. The shaking procedure perturbs s node allocations of the current solution N^* replacing them with s unused nodes. The behavior of the shaking parameter s, that depends on the result of the local search, is explained in Figure 3.9. The parameter s starts from a minimum value s_{min} (in the example $s_{min} = 1$) and every time that the local search does not improve the best known solution, s is increased by 1. Differently, when the local search succeeds, the best solution N^* is updated and s goes back to s_{min}.

The purpose of the shaking procedure is to first explore new starting solutions that are more similar to the best known result, so that the search is *intensified* in a promising neighborhood of the entire domain. If these local searches fail, the shaking procedure moves the search from intensification to *diversification*, generating starting solutions that are more and more different from the incumbent one. Whenever a new best solution is found, the shaking procedure comes back to s_{min}, to intensify the search near the just updated N^*. In principle, the shaking parameter s can be increased until $k = |N^*|$, changing all the node allocations. However, we experimented running different configurations that excessively moving away from the best known solution can be unproductive, causing a useless waste

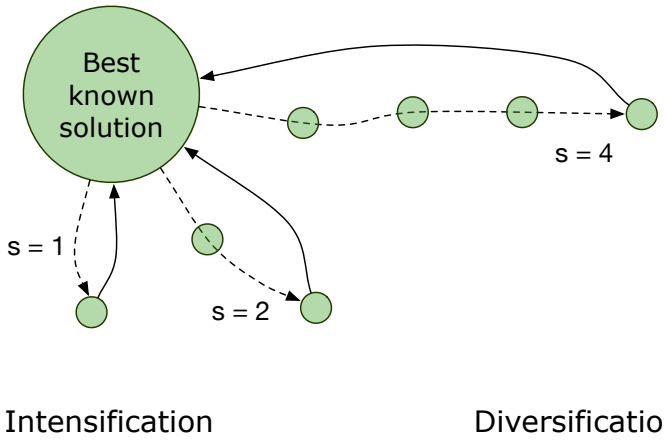

Figure 3.9: Shaking procedure: the parameter s is increased when the solution does not improve (dashed line) and restarts when a new optimum is found (continuous line).

of computational time. We have fixed a reasonable value of $s_{max} = \lfloor \frac{2}{3}k \rfloor$.

The VNS algorithm terminates when the total number of restarts reaches a given value R_{max}.

As we have said, the local search is the heuristic that proceeds from an initial solution to its neighborhood by a sequence of local changes, trying to improve each time the value of the objective function until a local optimum is found. The neighborhood of the adopted approach is given by cyclic sequences of moves, where each move consists in locating a new node, removing a node or changing the type of the node. A cyclic move is considered feasible only if the new covering rate respects the *target* coverage, and the total cost of the solution does not increase. Of course, each site must continue to hosts at maximum one node (constraint equation (3.10)). A cyclic move can be visualized on a graph $G = (N, A)$, where each node of the graph is a possible allocation of a hardware node. Each node of the graph is characterized by a location j, and a state that indicates if the node is active or inactive. A node n_{jt} currently allocated in location j, is represented on the graph with an active node n_j, labeled with its hardware type t. Note that index t does not appear because at most one type can be active in each node, and the type is specified by the label. Inactive nodes are instead left unlabeled. An arc (n_j, n_k) can represent: (1) the allocation of a hardware node in site j, if n_j is inactive and n_k is active; (2) the removal of a hardware node in site j, if n_j is active and n_k is inactive; (3) an hardware node n_j changing its hardware type, if both nodes are

Algorithm 2 $VNS(L, J, T, w, target, s_{min}, s_{max}, R_{max})$

```
N := Greedy(L, J, T, w, target);
N^0 := LocalSearch(L, J, T, w, target);
N* := N^0;
s := s_min;
for r := 1 to R_max do
    N := Shaking(N*, s, L, J, T, w, target)
    N^0 := LocalSearch(L, J, T, w, target)
    if (Z(N^0) > Z(N*)) then
        s := s_min;
        N* := N^0;
    else
        s := s + 1;
        if (s > s_max) then
            s := s_min;
return N*;
```

active. In both (1) and (2), the new node takes the hardware type of the head label (t of n_k). A cyclic exchange corresponds to a directed cycle on the improvement graph, as depicted in Figure 3.10. Each move, and so

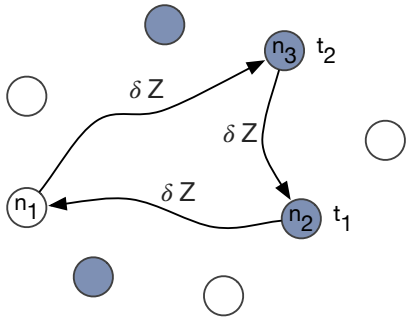

Figure 3.10: Improvement Graph: colored nodes represent current allocations, while empty nodes are possible allocations. All active nodes are labeled with their corresponding type. Each arc is a change (move) on the allocations.

each arc (n_j, n_k), determines a variation δZ in the value of the objective function Z. The purpose is to represent a group of moves so that a cyclic exchange represents an increase in the current objective function. However the total variation δZ is non additive with respect to the sequence of δZ values coming from single moves. This is caused by the interdependence

between different hardware nodes with overlapping covering regions, that lead to non-additive moves. To overcome this drawback, every cycle has been evaluated using an own temporary function Z' updated step by step from the end of the path to its starting node. In this way, all the cycles with a positive total weight bring improvements on the starting solution.

The search for the cyclic exchange with maximum weight is performed with exhaustive breadth-first exploration of the paths of graph G.

3.7 Experimental Results

Presented experimental results are initially focused on the usability of the tool, testing the ability to provide a solution in a reasonable time. Then, the performances of the model have been evaluated, in terms of localization accuracy through realistic indoor environment experiments, and in terms of cost-effectiveness of the suggested deployments.

3.7.1 Computational Experience

The tool has been been evaluated running several different configurations. Every test reported in this section has been executed with a spatial resolution of the floor plan equal to 1 meter. A first analysis can be done on the execution times of the proposed solution. Although the execution time can be tuned by the parameter R_{max}, which represent the maximum number of restarts of the VNS algorithm, an idea on the order of magnitude is given by Figure 3.11, where the time is represented as a function of the floorplan dimension. In the given example, R_{max} has been fixed to 20 restarts, the $target$ coverage equals to 95% of the total area, a single node type available with a range of 8 meters, covering floor-plans with rectangular areas. The graph shows that for single coverage, the execution time is low even for areas of 3000 squared meters. For trilateration and fingerprinting, the execution times become high from floor-plan of 2500 m^2. However, the tests represent a bad case in which the map dimension is very large while the node range available and the spatial resolution are small (respectively 8m and 1m). Increasing the range or the resolution, the instance of the problem decrease, resulting in faster executions.

A key aspect that characterizes the goodness of the proposed approach is the improvement of the objective function achieved by the VNS algorithm with respect to the first Greedy configuration. For this test we have run the tool several times with a floor-plan area of 2500 m^2 and a node range of $12m$. The number of reference nodes allocated is determined by the

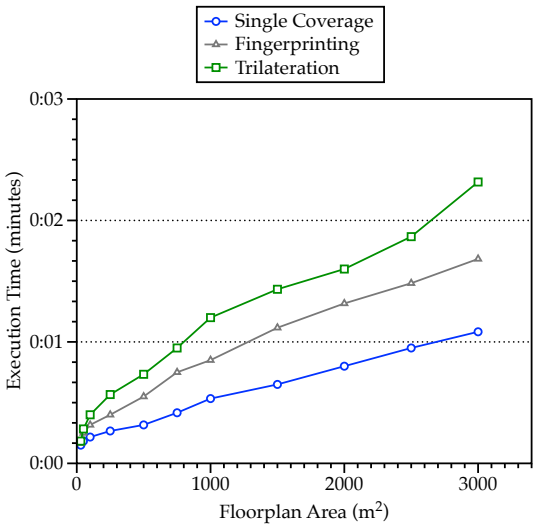

Figure 3.11: Execution time of the tool with floor plans of different areas, for each covering technique ($R_{max} = 20$, $target = 95\%$, $r_t = 8$).

Greedy procedure and increase with S, while the number of VNS restarts R_{max} has been fixed to 35. In Fig. 3.12 we reported the value of z, i.e.

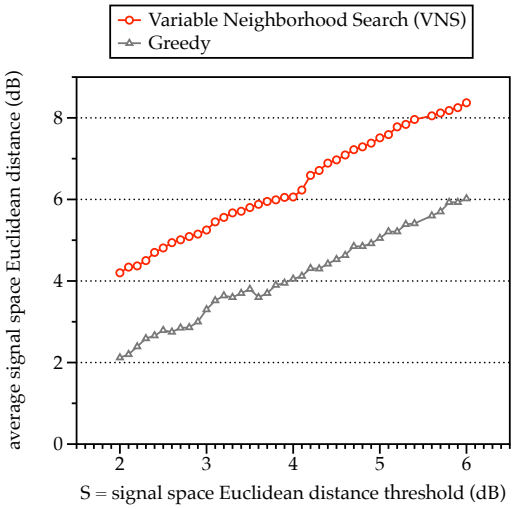

Figure 3.12: Average signal space Euclidean distance (z) obtained with the Greedy execution and compared with the z value after the VNS optimization. z values expressed as a function of the threshold S. Floor-plan area = 2500 m^2, $R_{max} = 20$, $target = 100\%$, $r_t = 12$

the average signal space Euclidean distance obtained with the first Greedy

execution, compared with the z value after the VNS optimization. The graph reports the z values as a function of the threshold S, described in section 3.6.1 as the minimum value of average signal space Euclidean distance (z) required during the Greedy procedure. The graph shows that moving the threshold within the range $(2, 6)$dB the VNS is able to improve the z value constantly around 2 dB. Although the VNS improvement is not astonishing for what regard the average value, Fig. 3.13 shows that the variance is strongly improved. This has been achieved moving from the objective function z used in Greedy procedure to the Z function of the VNS. The Z objective function has in fact the purpose to provide as many target location as possible with a high signal space Euclidean distance w.r.t. the surrounding locations.

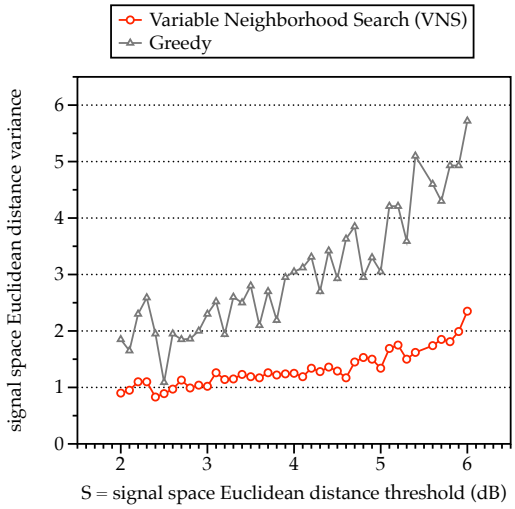

Figure 3.13: Signal space Euclidean distance variance obtained with the Greedy execution and compared with the z value after the VNS optimization. Values expressed as a function of the threshold S. Floor-plan area = 2500 m^2, $R_{max} = 20$, $target = 100\%$, $r_t = 12$

3.7.2 Experimental Setup and Accuracy Evaluation

The proposed tool was evaluated using data collected from a real-world environment, the NECST Lab, located at the basement of DEIB department at Politecnico di Milano. The dimension of the test-bed is 198 squared meters ($9x22m$). We collected Bluetooth Low Energy (BLE) signal data coming from BLE beacons with a coverage radius of 7 meters. Signal

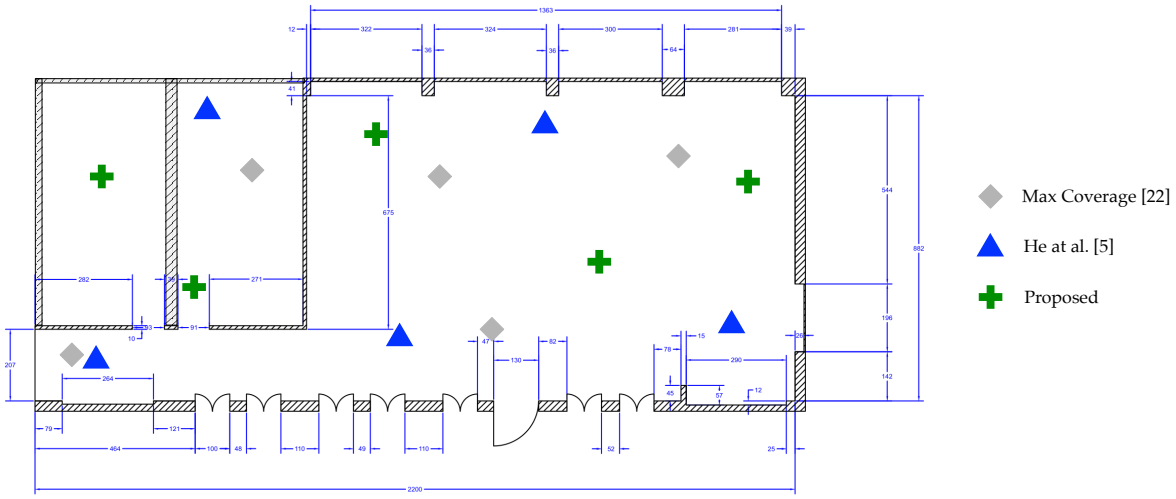

Figure 3.14: NECST Lab floor-plan, located at the basement of DEIB department at Politecnico di Milano. Each allocation corresponds to a BLE beacon with a range of 7 meters. Green crosses indicates allocations provided by our algorithm, gray rhombus represent allocations from [16] while blue triangle positions have been computed maximizing the coverage [18].

data has been collected using a Nexus 5 smartphone running Android 6.0.1. First, the NECST Lab floor-plan has been designed using our tool, obtaining the optimal number of beacons ($|N| = 5$) and their allocation for fingerprinting localization. R_{max} has been fixed to 20 restarts, the *target* coverage equals to 100% of the total area, a single node type available with a range of 7 meters, and the threshold $S = 4,5$. We collected 40 training samples for the localization algorithm using the obtained allocation. Then, the test samples were collected at distinct positions changing the phone orientation and the way in which user was keeping it, for example by hand or in a pocket. For the entire duration of training and test phase, the number of occupants and their enabled wireless devices has changed, from a minimum of 3 to a maximum of 17 people. This variation affects the accuracy performances, but at the same time contributes in obtaining realistic results. The training and test phase has been repeated with two configurations coming from different allocation algorithms: maximization of the coverage [18] and the allocation algorithm proposed by He at al. in [16]. For these two algorithms, the number of employed nodes has been fixed to 5. KNN with $K = 3$ has been employed as the fingerprinting algorithm.

A first result is showed in figure 3.15. The cumulative error distribution

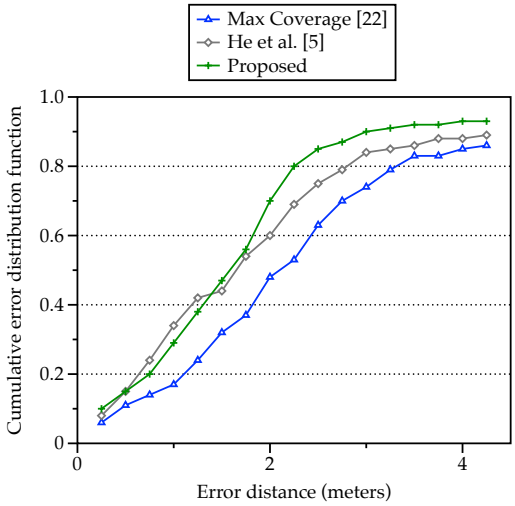

Figure 3.15: Cumulative error distribution function experienced by our approach ad compared with two different solutions from the state-of-the-art.

function shows that from 1.5 meters our approach performs better. Under 1.5 meters, He at al. approach performs better, but the difference in accuracy is marginal.

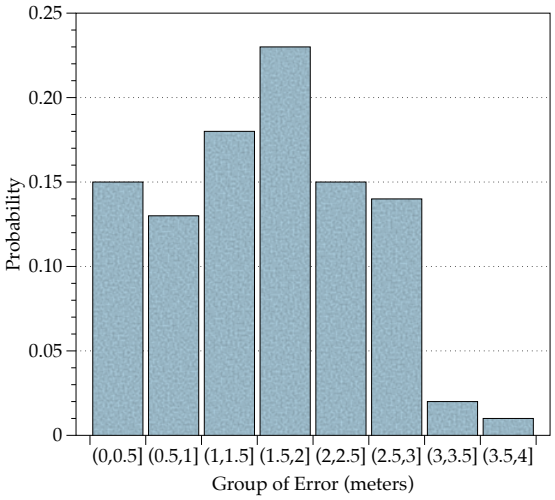

Figure 3.16: Mean positioning accuracy of the proposed allocation algorithm divided into different error ranges.

Figure 3.16 shows the mean positioning accuracy divided into different error ranges: $(0, 0.5]$, $(0.5, 1]$, $(1, 1.5]$, $(1.5, 2]$, $(2, 2.5]$, $(2.5, 3]$, $(3, 3.5]$,

$(3.5, 4]$. It's possible to notice that the majority of the localization errors appears within the $(1.5, 2]$ meters. The test-bed floor-plan, composed by 3 rooms, has been reported in figure 3.14. Green crosses indicates allocations provided by our algorithm, gray rhombus represent allocations from [16] while blue triangle positions have been computed maximizing the coverage [18].

3.7.3 Cost-effectiveness Analysis

A feature of our tool interesting for testing is the possibility to obtain solutions from mixed node types, with different characteristics and costs. In particular, given two types t_1 and t_2 characterized by two ranges r_i, and two costs c_i, it's possible to compare the total cost of a homogeneous solution with the cost of a mixed solution. Given a baseline type of node with a range $r_1 = 8\ m$ and a cost of $c_1 = 60$ \$, we can assume the presence on the market of a second type of hardware, with the half of the range distance ($r_2 = 4\ m$). The area covered by t_1 ($\approx 200\ m^2$) is four times bigger than the coverage of t_2 ($\approx 50\ m^2$). In order to obtain a fair test, the cost of t_2 should be $c_2 \geq c_1/4$, and so we set $c_2 = 20$ \$. This test has been performed with a *target* coverage of 95% on a rectangular map of 1000 m^2.

From Table 3.4 it's possible to observe that, although hardware nodes of type t_2 have a lower convenience in terms of $\frac{area}{price}$ (t_1 outperform t_2 in homogeneous solutions), the mixed strategy can use the smaller range nodes to reduce the total cost. This because less powerful nodes of type

Table 3.4: Cost of homogeneous and mixed solutions ($A = 1000\ m^2$, $target = 95\%$, $r_1 = 8m$, $r_2 = 4m$, $c_1 = 60\$$, $c_2 = 20\$$).

Node types	**Solution Costs (in \$)**		
	Single	Trilateration	Fingerprinting
$T = \{t_1\}$	480	1440	840
$T = \{t_2\}$	500	1620	880
$T = \{t_1, t_2\}$	440	1280	760

t_2 are employed to cover small portions of the floor-plan, like corners or small regions left uncovered by the larger range nodes.

The amount of saving in the total cost of the mixed solution doesn't depend only on the nodes range and price, but also on the irregularity of

the floor plan perimeter. A distinguish feature of the proposed tool respect to other works is the possibility to cover spaces that are not necessarily rectangular or squared. The level of irregularity of a floor plan can be identified by the minimum number of rectangles that compose the shape. In Figure 3.17 for example, the index of the floor plan irregularity is $I = 4$. We experimented the behavior of the tool increasing the level of irregularity,

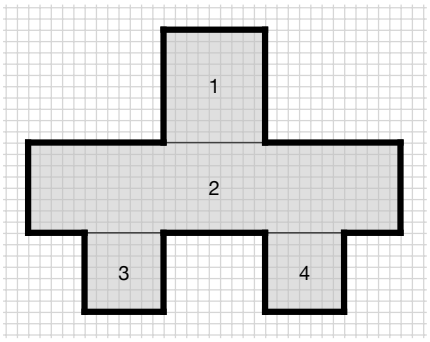

Figure 3.17: Irregularity of the floorplan perimeter summarized by the minimum number of rectangles.

while maintaining a constant total area of 1000 m^2. The test has been done with the same nodes configuration used in table 3.4 (homogeneous $T = t_1$, mixed $T = t_1, t_2$). The results shown in Table 3.5 proven that increasing the floor-plan irregularity, the cost difference between homogeneous and mixed solution becomes higher. This is caused by the increasing number of corners in the map, that can be covered with less powerful nodes.

Table 3.5: Cost differences (in $) between homogeneous and mixed solution increasing the floor plan irregularity (area fixed to 1000 m^2).

I	**Single**		**Trilater.**		**Fingerprint.**	
	homog.	mixed	homog.	mixed	homog.	mixed
1	480	440	1440	1280	840	760
2	480	440	1500	1320	840	780
4	600	500	1560	1380	900	820
8	720	580	1680	1480	1200	920

In conclusion, experimental results show that for most of the problem instances, a solution can be obtained in reasonable execution times. Depending on the available hardware types, homogeneous solutions could be improved with the employment of different type of nodes.

3.8 Conclusions and Future Work

In this paper, we tried to explain the challenges faced by designers during the installation of smart building systems that require the positioning of several hardware nodes. A common limitation of existing models is the lack of a convenient way to specify geometric information of the indoor map. This also leads to the employment of less accurate general models for signal propagation, instead of site-specific models. The design phase is get more difficult by the availability on the market of different hardware nodes, with different power transmissions and costs.

For these reasons we propose an integrated tool for both floor plan specification and node positioning, developed within an open-source CAD environment extensible through plug-ins. The tool is able to provide a near-optimal solution of node allocations, possibly with mixed types, with the aim to reduce the installation costs. The results suggest that, for most of the problem instances, a solution can be obtained in a reasonable execution time. Depending on the available hardware types, total cost of the solution could be improved moving from homogeneous to mixed type allocation.

A limitation of the proposed approach resides in the propagation model used to compute near-optimal solutions for localization systems. The model implemented is site-specific, and take in consideration walls for LOS and NLOS propagations. However the approach do not consider refraction or diffraction effects. Another limitation is the inability of the system to model the signal propagation between different floors of the building, managing each level independently. For future work, we plan to improve the system with an indoor signal propagation model able to consider refraction and diffraction effects of the indoor environment like walls and floors. In addition, we'll try to apply the model to 3D designing tools, becoming suitable also for multi-floor environments.

Chapter 4

Experimental Results

4.1 Experimental Testbed

4.1.1 Indoor Environment

4.1.2 Smartphones

4.1.3 WSN prototype

4.1.4 Back-end

4.2 Automatic Nodes Deployment Results

4.3 Localization Accuracy

4.4 Responsiveness Tests

4.5 Power Consumptions

4.6 Stress Test

Conclusions

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Acronyms

CFD Computational Fluid Dynamics

Computational Fluid Dynamics is a branch of fluid mechanics that uses numerical methods and algorithms to solve and analyze problems that involve fluid flows. Computers are used to perform the calculations required to simulate the interaction of liquids and gases with surfaces defined by boundary conditions.
www.en.wikipedia.org

HPC High Performance Computing

In informatica con il termine High Performance Computing (calcolo ad elevate prestazioni) ci si riferisce alle tecnologie utilizzate da computer cluster (insieme di computer connessi tra loro tramite una rete telematica) per creare dei sistemi di elaborazione in grado di fornire delle prestazioni molto elevate, ricorrendo tipicamente al calcolo parallelo.
www.it.wikipedia.org

OpenFOAM Open source Field Operation And Manipulation

The OpenFOAM® CFD Toolbox is a free, open source CFD software package which has a large user base across most areas of engineering and science, from both commercial and academic organisations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. It includes tools for meshing, notably *snappyHexMesh*, a parallelised mesher for complex CAD geometries, and for pre- and post-processing. Almost everything (including meshing, and pre- and post-processing) runs in parallel as standard, enabling users to take full advantage of computer hardware at their disposal.
www.openfoam.com

CINECA Consorzio Interuniversitario per il Calcolo Automatico

Cineca è un Consorzio Interuniversitario senza scopo di lucro formato da 69 università italiane e 3 Enti. Costituito nel 1969, oggi il Cineca è il maggiore centro di calcolo in Italia, uno dei più importanti a livello mondiale. Operando sotto il controllo del Ministero dell'Istruzione

dell'Università e della Ricerca, offre supporto alle attività della comunità scientifica tramite il supercalcolo e le sue applicazioni, realizza sistemi gestionali per le amministrazioni universitarie e il MIUR, progetta e sviluppa sistemi informativi per pubblica amministrazione, sanità e imprese.
www.cineca.it

Bibliography

References cited in text

Publications and Manuals

[1] Kemal Akkaya et al. "IoT-based occupancy monitoring techniques for energy-efficient smart buildings". In: *2015 IEEE Wirel. Commun. Netw. Conf. Work.* (2015), pp. 58–63. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7122529 (cit. on pp. 10, 32).

[2] Tim Andersson. "Bluetooth Low Energy and Smartphones for Proximity-Based Automatic Door Locks". In: (2014) (cit. on p. 53).

[3] ASHRAE. *ANSI/ASHRAE Standard 62-2001: Ventilation for Acceptable Indoor Air Quality*. 2004. ISBN: 4048368400 (cit. on p. 2).

[4] Bharathan Balaji et al. "Sentinel: occupancy based HVAC actuation using existing WiFi infrastructure within commercial buildings". In: *Proc. 11th ACM Conf. Embed. Networked Sens. Syst.* (2013), p. 17 (cit. on pp. 12, 34, 36, 48).

[5] Christian Beder and Martin Klepal. "Fingerprinting Based Localisation Revisited". In: November (2012), pp. 13–15 (cit. on pp. 29, 34).

[6] Alex Beltran, Vl Varick L Erickson, and Alberto E Ae Cerpa. "ThermoSense: Occupancy Thermal Based Sensing for HVAC Control". In: *Proc. 5th ACM Work. Embed. Syst. Energy-Efficient Build.* (2013), 11:1–11:8 (cit. on pp. 14, 34, 54).

[7] Fabio Colombo, Roberto Cordone, and Guglielmo Lulli. "The multimode covering location problem". In: *Comput. Oper. Res.* 67 (2016), pp. 25–33. ISSN: 03050548. URL: http://dx.doi.org/10.1016/j.cor.2015.09.003 (cit. on p. 60).

[8] Giorgio Conte et al. "BlueSentinel : a first approach using iBeacon for an energy efficient occupancy detection system". In: *1st ACM Conf. Embed. Syst. Energy-Efficient Build.* (2014), pp. 11–19 (cit. on pp. 30, 34).

[9] A. Corna et al. "Occupancy Detection via iBeacon on Android Devices for Smart Building Management". In: *Des. Autom. Test Eur. Conf. Exhib.* 1 (2015), pp. 629–632 (cit. on p. 54).

[10] Mark S. Daskin. "Maximum Expected Covering Location Model: Formulation, Properties and Heuristic Solution." In: *Transp. Sci.* 17.1 (1983), pp. 48–70. ISSN: 00411655 (cit. on p. 50).

[11] Mark S. Daskin and Edmund H. Stern. "Hierarchical Objective Set Covering Model For Emergency Medical Service Vehicle Deployment." In: *Transp. Sci.* 15.2 (1981), pp. 137–152. ISSN: 00411655. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0019565514{\&}partnerID=tZOtx3y1 (cit. on p. 50).

[12] Department of Energy DOE. *Buildings Energy Data Book*. Tech. rep. 2011, pp. 1–271 (cit. on p. 2).

[13] Varick L. Erickson, Miguel a. Carreira-Perpinan, and Alberto E. Cerpa. "OBSERVE: Occupancy-based system for efficient reduction of HVAC energy". In: *Proc. 10th ACM/IEEE Int. Conf. Inf. Process. Sens. Networks* (2011), pp. 258–269. URL: ACMBuildSys2015 (cit. on p. 48).

[14] Vl Erickson, S Achleitner, and Ae Cerpa. "POEM: power-efficient occupancy-based energy management system". In: *Proc. 12th* (2013), pp. 203–216 (cit. on pp. 16, 34, 54).

[15] Shih Hau Fang and Tsung Nan Lin. "A novel access point placement approach for WLAN-based location systems". In: *IEEE Wirel. Commun. Netw. Conf. WCNC* (2010), pp. 1–4. ISSN: 15253511 (cit. on pp. 50, 51).

[16] Ying He et al. "Rapid deployment of APs in WLAN indoor positioning system". In: *Proc. 2011 6th Int. ICST Conf. Commun. Netw. China, CHINACOM 2011* (2011), pp. 268–273 (cit. on pp. 50, 57, 65, 72, 74).

[17] Yifei Jiang et al. "Ariel: Automatic wi-fi based room fingerprinting for indoor localization". In: *Proc. UbiComp 2012, ACM Conf. Ubiquitous Comput.* (2012), pp. 441–450. URL: http://dl.acm.org/citation.cfm?id=2370282 (cit. on pp. 29, 34).

[18] Mt Kouakou, Shinya Yamamoto, and K Yasumoto. "Cost-Efficient Deployment for Full-Coverage and Connectivity in Indoor 3D WSNs". In: *Proc. 12th ACM Int. Conf. Adjun. Pap. Ubiquitous Comput.* (2010) (cit. on pp. 72, 74).

[19] Pekka Kyösti et al. "IST-4-027756 WINNER II D1. 1.2 V1. 2 WINNER II Channel Models.pdf". In: *Projectscelticinitiativeorg* 1.82 (2008), p. 82. URL: http://projects.celtic-initiative.org/winner+/WINNER2-Deliverables/D1.1.2v1.2.pdf (cit. on p. 59).

[20] Ryan Melfi et al. "Measuring building occupancy using existing network infrastructure BT - 2011 International Green Computing Conference, IGCC 2011, July 25, 2011 - July 28, 2011". In: (2011) (cit. on pp. 12, 34).

[21] Anindya S Paul et al. "MobileRF: A Robust Device-free Tracking System Based on a Hybrid Neural Network HMM Classifier". In: *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2014), pp. 159–170. URL: http://doi.acm.org/10.1145/2632048.2632097 (cit. on p. 53).

[22] Andrea Piscitello et al. "Danger-System : Exploring New Ways to Manage Occupants Safety in Smart Building". In: (2015), pp. 1–6 (cit. on p. 7).

[23] Claude Alain Roulet. "Indoor environment quality in buildings and its impact on outdoor environment". In: *Energy Build.* 33.3 (2001), pp. 183–191. ISSN: 03787788 (cit. on p. 1).

[24] a. M C So and Yinyu Ye. "On solving coverage problems in a wireless sensor network using voronoi diagrams". In: *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* 3828 LNCS (2005), pp. 584–593. ISSN: 03029743 (cit. on p. 50).

[25] Quang; Takumi MIYOSHI Vinh Tran. "An Algorithm for Sensing Coverage Problem in Wireless Sensor Networks". In: () (cit. on p. 50).

[26] Y G Xu et al. "Lean Cost Analysis Based on BIM Modeling for Construction Project". In: *Front. Mech. Eng. Mater. Eng. Ii, Pts 1 2* 457-458 (2014), pp. 1444–1447. ISSN: 1662-7482. URL: http://www.scientific.net/AMM.457-458.1444.pdf (cit. on p. 49).

[27] J. P. Zhang and Z. Z. Hu. "BIM- and 4D-based integrated solution of analysis and management for conflicts and structural safety problems during construction: 1. Principles and methodologies". In: *Autom. Constr.* 20.2 (2011), pp. 167–180. ISSN: 09265805 (cit. on p. 49).

[28] Yi Zhao, Anthony LaMarca, and Joshua R Smith. "A battery-free object localization and motion sensing platform". In: *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput. - UbiComp '14 Adjun.* (2014), pp. 255–259 (cit. on p. 54).

[29] Yongxiang Zhao, Huaibei Zhou, and Meifang Li. "Indoor Access Points Location Optimization Using Differential Evolution". In: *2008 Int. Conf. Comput. Sci. Softw. Eng.* IEEE, 2008, pp. 382–385. ISBN: 978-0-7695-3336-0. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4721767 (cit. on pp. 50, 57).

[30] Becerik-Gerber Zheng, Yang; Burcin. "Cross-Space Building Modeling by Contextual Information Based Learning". In: *Uma ética para quantos?* XXXIII.2 (2012), pp. 81–87. ISSN: 0717-6163 (cit. on pp. 14, 34).

[31] Jianyong Zhu et al. "RSSI Based Bluetooth Low Energy Indoor Positioning". In: *Int. Conf. Indoor Position. Indoor Navig.* October 27-30, 2014 (2014), p. 8 (cit. on pp. 30, 34).

Online Materials

[32] *ArchiCAD - The architectural BIM CAD software*. URL: http://www.graphisoft.com/archicad/ (cit. on p. 49).

[33] Mikel Choperena. *RFID-powered Sensors Can Play a Big Role in the Internet of Things*. 2013. URL: http://www.rfidjournal.com/articles/view?11062/ (cit. on p. 36).

[34] *NECSTbox - Android application for ThermoSense and BlueSentinel users*. 2016. URL: https://play.google.com/store/apps/details?id=polimi.necst.box (cit. on p. 40).

[35] *NECSTbox - iOS application for ThermoSense and BlueSentinel users*. 2016. URL: https://itunes.apple.com/us/app/necstbox/id1125348130 (cit. on p. 40).

[36] Radius Networks. *Device Support For Beacon Transmission with Android 5 and 6*. 2016. URL: https://altbeacon.github.io/android-beacon-library/beacon-transmitter-devices.html (cit. on p. 38).

Additional material consulted

Publications and Manuals

[37] Naeim Abedi, Ashish Bhaskar, and Edward Chung. "Bluetooth and Wi-Fi MAC Address Based Crowd Data Collection and Monitoring : Benefits , Challenges and Enhancement". In: *Australas. Transp. Res. Forum 2013 Proc. 2* October (2013), pp. 1–17. URL: http://www.patrec.org/atrf.aspx.

[38] Sina Afshari et al. "Short Paper : The Smart Conference Room : An Integrated System Testbed for Efficient , Occupancy-Aware Lighting Control". In: (), pp. 245–248.

[39] Yuvraj Agarwal et al. "BuildingDepot : An Extensible and Distributed Architecture for Building Data Storage , Access and Sharing". In: ().

[40] Apple Inc. "Getting Started with iBeacon". In: (2014), pp. 1–11.

[41] Tanvir Islam Aumi et al. "A Self - Calibrating Approach to Whole - Home Contactless Power Consumption Sensing". In: *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput. (UbiComp '14)* (2014), pp. 361–371.

[42] Alex Beltran and Alberto E Cerpa. "Poster Abstract : Model Predictive Control with Real-time Occupancy Detection". In: (), pp. 437–438.

[43] Alex Beltran, Vl Varick L Erickson, and Alberto E Ae Cerpa. "ThermoSense: Occupancy Thermal Based Sensing for HVAC Control". In: *Proc. 5th ACM Work. Embed. Syst. Energy-Efficient Build.* (2013), 11:1–11:8. URL: ACMBuildSys2015.

[44] Oded Berman, Zvi Drezner, and Dmitry Krass. "Generalized coverage: New developments in covering location models". In: *Comput. Oper. Res.* 37.10 (2010), pp. 1675–1687. ISSN: 03050548. URL: http://dx.doi.org/10.1016/j.cor.2009.11.003.

[45] Oded Berman et al. "The variable radius covering problem". In: *Eur. J. Oper. Res.* 196.2 (2009), pp. 516–525. ISSN: 03772217. URL: http://dx.doi.org/10.1016/j.ejor.2008.03.046.

[46] Jacob T Biehl et al. "LoCo: A Ready-to-Deploy Framework for Efficient Room Localization using Wi-Fi". In: *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput. - UbiComp '14 Adjun.* (2014), pp. 183–187. URL: http://dl.acm.org/citation.cfm?id=2632048.2636083.

[47] Jacob T Biehl et al. "You'Re Where? Prove It!: Towards Trusted Indoor Location Estimation of Mobile Devices". In: *Proc. 2015 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2015), pp. 909–919. URL: http://doi.acm.org/10.1145/2750858.2804284.

[48] S.-H. Gary Chan, Lei Yu, and Ning Liu. "Calibration-free fusion of step counter and wireless fingerprints for indoor localization". In: *UbiComp '15 Proc. 2015 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2015), pp. 897–908.

[49] R.L. Church and C. ReVelle. "The maximal covering location problem". In: *Pap. Reg. Sci. Assoc.* (1974).

[50] M. Collotta and G. Pau. "Bluetooth for Internet of Things: A fuzzy approach to improve power management in smart homes". In: *Comput. Electr. Eng.* 44 (2015), pp. 137–152. ISSN: 00457906. URL: http://linkinghub.elsevier.com/retrieve/pii/S0045790615000117.

[51] Fabio Colombo, Roberto Cordone, and Guglielmo Lulli. "The multimode covering location problem". In: *Comput. Oper. Res.* 67 (2016), pp. 25–33. ISSN: 03050548. URL: http://dx.doi.org/10.1016/j.cor.2015.09.003.

[52] Omar Cruz, Erik Ramos, and Moises Ramirez. "3D indoor location and navigation system based on Bluetooth". In: *CONIELECOMP 2011, 21st Int. Conf. Electr. Commun. Comput.* (2011), pp. 271–277. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5749373.

[53] M Daskin. *Network and Discrete Location: Models, Algorithms and Applications*. Vol. 48. 1997, pp. 763–764. ISBN: 047101897X.

[54] Mark S. Daskin. "Application of an Expected Covering Model To Emergency Medical Service System Design". In: *Decis. Sci.* 13.3 (1982), pp. 416–439. ISSN: 0011-7315. URL: http://doi.wiley.com/10.1111/j.1540-5915.1982.tb00159.x.

[55] Zvi Drezner and Tammy Drezner. "Multiple facilities location in the plane using the gravity model". In: *Geogr. Anal.* 38.4 (2006), pp. 391–406. ISSN: 00167363.

[56] K??roly Farkas. "Placement optimization of reference sensors for indoor tracking". In: *Acta Polytech. Hungarica* 12.2 (2015), pp. 123–139. ISSN: 17858860.

[57] Emanuele Goldoni et al. "Experimental analysis of RSSI-based indoor localization with IEEE 802.15.4". In: *2010 Eur. Wirel. Conf. EW 2010* (2010), pp. 71–77. ISSN: 978-1-4244-5999-5.

[58] Youngjune Gwon Youngjune Gwon, R. Jain, and T. Kawahara. "Robust indoor location estimation of stationary and mobile users". In: *Ieee Infocom 2004* 2 (2004), pp. 1032–1043. ISSN: 0743-166X.

[59] R Heydon and N Hunn. "Bluetooth Low Energy". In: *CSR Present.* (2012). URL: https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx.

[60] Sebastian Hilsenbeck et al. "Graph-based Data Fusion of Pedometer and WiFi Measurements for Mobile Indoor Positioning". In: *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput. - UbiComp '14 Adjun.* (2014), pp. 147–158. URL: http://www.lmt.ei.tum.de/forschung/publikationen/dateien/Hilsenbeck2014Graph-basedDataFusionof.pdf$\backslash$nhttp://dl.acm.org/citation.cfm?id=2636079$\backslash$nhttp://dl.acm.org/citation.cfm?doid=2632048.2636079.

[61] Chuan-che Jeff Huang, Rayoung Yang, and Mark W Newman. "The Potential and Challenges of Inferring Thermal Comfort at Home Using Commodity Sensors". In: *Proc. 2015 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2015), pp. 1089–1100.

[62] Yifei Jiang et al. "Hallway based Automatic Indoor Floorplan Construction using Room Fingerprints ". In: *Proc. Int. Conf. Ubiquitous Comput.* (2013), pp. 315–324. URL: papers2://publication/doi/10.1145/2493432.2493470.

[63] Chih-Wei Kang and Jian-Hung Chen. "Multi-objective evolutionary optimization of 3D differentiated sensor network deployment". In: *Proc. 11th Annu. Conf. companion Genet. Evol. Comput. Conf. - GECCO '09* (2009), p. 2059. URL: http://portal.acm.org/citation.cfm?doid=1570256.1570276.

[64] Dae-Yeob Kim et al. "Accurate Indoor Proximity Zone Detection Based on Time Window and Frequency with Bluetooth Low Energy". In: *Procedia Comput. Sci.* 56.MobiSPC (2015), pp. 88–95. ISSN: 18770509. URL: http://www.sciencedirect.com/science/article/pii/S1877050915016804.

[65] Dae-Yeob Kim et al. "Accurate Indoor Proximity Zone Detection Based on Time Window and Frequency with Bluetooth Low Energy". In: *Procedia Comput. Sci.* 56.MobiSPC (2015), pp. 88–95. ISSN: 18770509. URL: http://www.sciencedirect.com/science/article/pii/S1877050915016804.

[66] Wilhelm Kleiminger, Christian Beckel, and Silvia Santini. "Household Occupancy Monitoring Using Electricity Meters". In: *UbiComp* (2015), pp. 975–986.

[67] M.T. Kouakou et al. "Deployment planning tool for indoor 3D-WSNs". In: *Proc. 12th ACM Int. Conf. Adjun. Pap. Ubiquitous Comput.* 1 (2010), pp. 369–370. URL: http://portal.acm.org/citation.cfm?id=1864431.1864440.

[68] Seungwoo Lee et al. "Non-obstructive Room-level Locating System in Home E nvironment s using Activity Fingerprint s from Smartwatch". In: *Proc. UbiComp 2015* (2015).

[69] Fan Li et al. "A reliable and accurate indoor localization method using phone inertial sensors". In: *ACM Conf. Ubiquitous Comput.* (2012), pp. 421–430. URL: http://dl.acm.org/citation.cfm?doid=2370216.2370280.

[70] Nan Li, Gulben Calis, and Burcin Becerik-Gerber. "Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations". In: *Autom. Constr.* 24 (2012), pp. 89–99. ISSN: 09265805. URL: http://linkinghub.elsevier.com/retrieve/pii/S0926580512000283.

[71] Dimitrios Lymberopoulos, Romit Roy Choudhury, and Jie Liu. "A Realistic Evaluation and Comparison of Indoor Location Technologies : Experiences and Lessons Learned". In: Table 1 (2014), pp. 178–189.

[72] Yemao Man and Edith C.-H. Ngai. "Energy-efficient automatic location-triggered applications on smartphones". In: *Comput. Commun.* 50 (2014), pp. 29–40. ISSN: 01403664. URL: http://www.sciencedirect.com/science/article/pii/S0140366414001200.

[73] Mattei Matteucci. "An adaptive indoor positioning system based on bluetooth low energy RSSI". In: (2013), p. 147.

[74] Akshay Uttama Nambi S N, Antonio Reyes Lua, and R Venkatesha Prasad. "LocED: Location-aware Energy Disaggregation Framework". In: (2015). URL: ACMBuildSys2015.

[75] Akshay Uttama Nambi S N, Antonio Reyes Lua, and R Venkatesha Prasad. "LocED: Location-aware Energy Disaggregation Framework". In: ii (2015), pp. 45–54.

[76] Antonio J. Nebro et al. "Optimal antenna placement using a new multi-objective chc algorithm". In: *9th Annu. Conf. Genet. Evol. Comput. (GECCO '07)* (2007), pp. 876–883. URL: http://portal.acm.org/citation.cfm?doid=1276958.1277128.

[77] Stefano Piffer, Nicola Dorigatti, and Giuseppe Conti. "I-Locate - Indoor/Outdoor Location and Asset Management Through Open Geodata". In: (2014).

[78] Gerald Pirkl and Paul Lukowicz. "Resonant magnetic coupling indoor localization system". In: *Proc. 2013 ACM Conf. Pervasive ubiquitous Comput. Adjun. Publ. - UbiComp '13 Adjun.* (2013), pp. 59–62. URL: http://dl.acm.org/citation.cfm?doid=2494091.2494108.

[79] Gerald Pirkl and Paul Lukowicz. "Robust, low cost indoor positioning using magnetic resonant coupling". In: *Proc. 2012 ACM Conf. Ubiquitous Comput. - UbiComp '12* (2012), p. 431. URL: http://dl.acm.org/citation.cfm?doid=2370216.2370281.

[80] Nithyananthan Poosamani and Injong Rhee. "Towards a practical indoor location matching system using 4G LTE PHY layer information". In: *2015 IEEE Int. Conf. Pervasive Comput. Commun. Work. (PerCom Work.* (2015), pp. 284–287. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7134048.

[81] Juhi Ranjan, Yu Yao, and Kamin Whitehouse. "An RF doormat for tracking people's room locations". In: *Mobiquitous* (2013), pp. 797–800.

[82] Mohamed Er Rida et al. "Indoor Location Position Based on Bluetooth Signal Strength". In: *2015 2nd Int. Conf. Inf. Sci. Control Eng.* (2015), pp. 769–773. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7120717.

[83] Chong Shao, Shahriar Nirjon, and Jan-michael Frahm. "Years-Long Binary Image Broadcast using Bluetooth Low Energy Beacons". In: ().

[84] Zheng Sun et al. "Headio: Zero-Configured Heading Acquisition for Indoor Mobile Devices Through Multimodal Context Sensing". In: *Proc. Int. Conf. Ubiquitous Comput.* (2013). URL: papers2://publication/uuid/D025A187-CA4F-4849-88B4-3E86C4274B7D.

[85] Thiago Teixeira and Andreas Savvides. "Lightweight people counting and localizing in indoor spaces using camera sensor nodes". In: *2007 1st ACM/IEEE Int. Conf. Distrib. Smart Cameras, ICDSC*. 2007, pp. 36–43. ISBN: 1424413540.

[86] Erik Vlugt. "Bluetooth Low Energy, Beacons and Retail". In: *Verifone.com* (2013), pp. 1–12. URL: http://www.verifone.com/media/3603729/bluetooth-low-energy-beacons-retail-wp.pdf.

[87] Thomas Weng, Anthony Nwokafor, and Yuvraj Agarwal. "BuildingDepot 2.0". In: *Proc. 5th ACM Work. Embed. Syst. Energy-Efficient Build. - BuildSys'13* (2013), pp. 1–8. URL: http://dl.acm.org/citation.cfm?id=2528282.2528285.

[88] Hongwei Xie et al. "MaLoc: A practical magnetic fingerprinting approach to indoor localization using smartphones". In: *UbiComp '14 Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2014), pp. 243–253.

[89] Chenren Xu. "Device-free people counting and localization". In: *Proc. 2013 ACM Conf. ...* (2013), pp. 367–372. URL: http://dl.acm.org/citation.cfm?id=2501091.

[90] Han Xu et al. "Enhancing WiFi-based Localization with Visual Clues". In: *ACM Int. Jt. Conf. Pervasive Ubiquitous Comput.* (2015), pp. 963–974.

Online Materials

[91] *QCAD - The Open Source CAD System For Everyone - http://www.qcad.org/.* URL: http://www.qcad.org/.