ABSTRACT

WANG, YUN. Performance Analysis of Smart Space with Indoor Localization Capabilities. (Under the direction of Michael Devetsikiotis.)

Smart spaces with indoor localization capabilities can recognize the users and interact with them under the Artificial Intelligence (AI) algorithms. The intelligent, interactive systems may replace the non-interactive systems around us within the next decade, enriching the overall indoor experience, and changing many areas of our lives. The user experience greatly depends on the performance of the smart space, which can be affected by the inaccurate indoor location estimation. In this thesis, we try to quantify the indoor localization precision requirements to achieve acceptable user experience in smart spaces by varying the user arrival rate and the size of the smart space. We conduct a comprehensive survey on various indoor localization solutions, present the implementation of a cross-platform WLAN indoor localization system, and analyze the dependency between the indoor localization precision and smart space performance. Various precision levels summarized from the localization survey are used in the analysis. We furthermore take this dependency into account for optimizing smart space performance and user experience for cost minimization.

Performance Analysis of Smart Space with Indoor Localization Capabilities

by
Yun Wang

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Master of Science

Computer Engineering

Raleigh, North Carolina

2014

APPROVED BY:

Dr. Harry Perros

Dr. R. Michael Young

Dr. Michael Devetsikiotis
Committee Chair

BIOGRAPHY

Yun Wang was born in Shanxi Province, China in 1989. She graduated from Beijing University of Posts and Telecommunications and Queen Mary, University of London in June 2011 with two Bachelor of Engineering degrees in Telecommunications Engineering with Management. Following that, she began the study for pursuing her Master of Science degree in Computer Engineering at North Carolina State University under the guidance of Dr. Michael Devetsikiotis starting in August 2011. Yun is currently working towards her thesis defense and will graduate in June 2014.

ACKNOWLEDGMENTS

I would like to express sincere appreciation to my advisor, Dr. Michael Devetsikiotis for giving me the opportunity to work with him and for his persistent guidance, tremendous help and encouragement in my research work. I would also like to thank Dr. Harry Perros and Dr. Michael Young for expressing interest in my work and for serving on my thesis committee.

I would like to thank my friends and colleagues: Kostas, Andrew, J., Shalini, and Ioannis for their help and inspiration along the way. I would also like to thank Dr. Eric Rotenberg, Dr. Mihail Sichitiu, Dr. Huiyang Zhou, Dr. Yan Solihin, Shirley Rodgers, Maurice York, and Scott Williams for the opportunities to work with them, it's been a pleasure. Many thanks to Ms. Cailan for her kindly help on the administrative issues for the last two semesters.

Finally, I would like to thank my family for their trust, understanding and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 Introduction

Our daily-life activities involve spending an increasingly high amount of time indoors, either at home or at work, so improving the quality of indoor life is very important. For decades now, people have been predicting the advent of the smart space, where human activities can control and activate various ambient functions remotely and automatically [1], with the use of sensing devices deployed in the space and the advanced technologies from Artificial Intelligence. The intelligent, interactive systems may replace the non-interactive systems around us within the next decade, enriching the overall experience, and changing many areas of our lives. We focus on the smart space like libraries and museums, and how to achieve the desired level of service for users in such informal adaptive learning spaces through the available sensing inputs and resources.

1.1 Motivation

The increasing use of interactive technologies, digital media, and dynamic content in smart space like libraries and museums is improving the user experience substantially. In well-designed learning spaces, digital content can itself become a service, either with pre-designed dynamic output to energize the space, or with creative and social output to interact with users actively. In libraries, repeat users can benefit from enhanced experience with the digital content the most, because the environment can service based on their preferences, whether they normally make frequent short visits to check out materials or use computing resources, or stay

at one place for extended periods of time for research and project work. In museums, digital content can assist the users' walk-through with better learning and educational experience. For example, the digital content can guide the users through selected scenarios and interact with them dynamically based on their previous actions, which makes the learning more intriguing, and elevates their experience along the way. Besides, the automated, digital-driven systems reduce the amount of workload of giving tours and directions for staff in libraries or museums, because the tasks can be handled by the dynamic content in the space. The anonymous human mobility data set (including human behaviors and how people adapt to different situations) can also be collected from the human-space interactions for future social network research.

The intelligent and interactive technologies for adaptive learning space present new opportunities, but also new challenges because the automated, digital-driven systems rely on the sensing inputs from the space, and the available resources. In reality, the sensing inputs cannot be 100% accurate. The faulty estimate of the user's state (e.g. errors in users' physical location estimation) can lead to suboptimal resource allocation and system performance. Given the limited number of resources available, the presence of multiple users leads to conflicts regarding the resources of the space, let alone the effect from the inaccurate sensing inputs, which will even worsen the situation by assigning the limited resource to the wrong user. Therefore, the accuracy of the sensing inputs and the number of available resources play important roles in controlling the system performance. It is evident that pursuing the best sensing-input accuracy and possessing enough amount of resources can solve all the problems, but with the increasing system load (demand for resources) and limited budget, we need to

have a thorough understanding of the resource requirements and performance demand of the system to achieve the optimal solution.

1.2 Introduction to Smart Spaces

Smart spaces are highly interactive physical environments equipped with computational facilities [2]. The smart spaces should be able to autonomously acquire knowledge of their users and to use it to improve their overall experience in the space [3]. “Smart” is defined by the sensing capabilities, and the service set the space offers. Sensing capabilities refer to recognizing the user’s physical location, identity, needs and preferences by collecting various data such as room temperature, ambient noise level, wireless signal strength, and so on. The service set adapts the computational resources in the space according to the sensing inputs collected. It depends on both the resources and the users present in the space at a given time, as the inherent mobility of the users force the interactions to be transient [4]. And the presence of multiple users leads to the need of a conflict resolution mechanism regarding the resources of the space.

1.2.1 Smart Space Architecture

The smart space architecture is shown in Figure 1.1. The Artificial Intelligence (AI) algorithms control the operations of the smart space. The components on the left of the figure are the inputs to the AI algorithms. The components on the right of the figure are the resources that the AI algorithms manage. Basically, the AI algorithms should allocate the resources like the

video displays, audio playback system, and computers, etc. accordingly based on the data (like indoor locations) collected from all kinds of sensors deployed in the space.

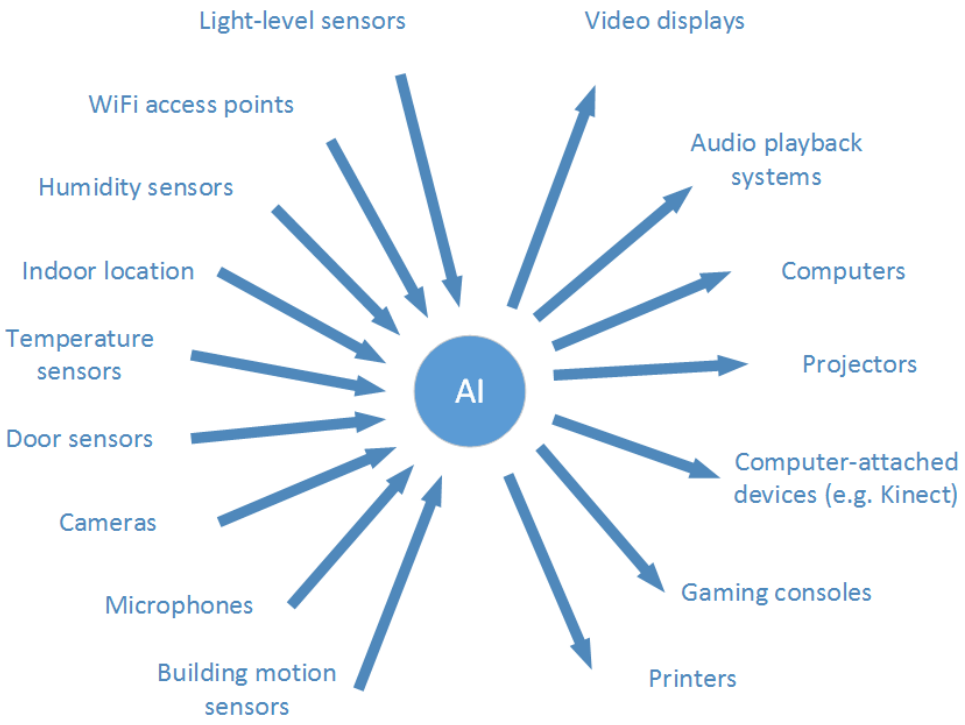

Figure 1.1 Smart space architecture

1.2.2 Resource Management

The number of resources in the multi-user smart space shown in Figure 1.1, including the video displays, computers, etc., is limited, which leads to the necessity of resource management. Good resource management allocates the different resources to needed users effectively and efficiently. When the number of users are more than the number of the resources available in the space, resource management is also responsible of resolving the conflict to keep the entire

system functioning. So resource allocation and conflict resolution are the two major responsibilities of resource management.

A lease-based method for managing various resources in smart spaces is introduced in [4]. Leasing is the process of giving users the control of the leased resources transiently. It introduces a flexible way of utilizing context information in resource management, and helps optimize the resource usage in smart spaces. Figure 1.2 illustrates the conceptual view of key components in resource management generally.

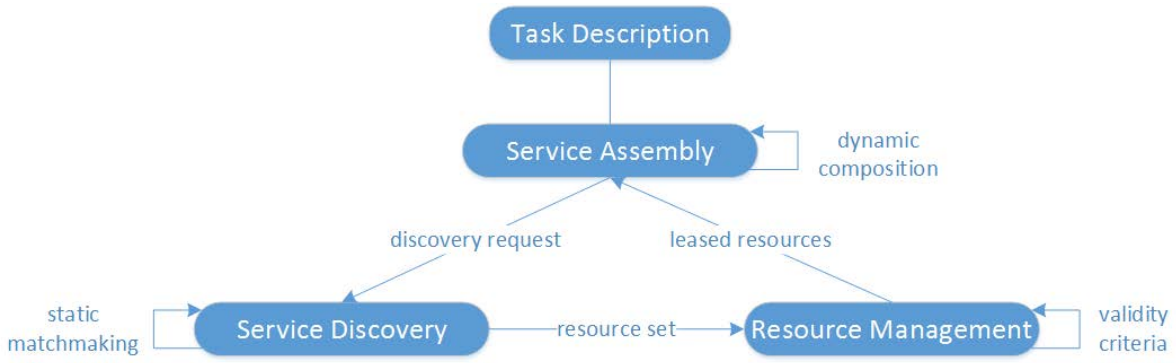

Figure 1.2 Key components in resource management [4]

In Figure 1.2, task description specifies what task the user is utilizing the resource for. Service assembly dynamically builds a service for executing the task by using the leased resources. It sends a request to the service discovery component based on the resources required by the task. Service discovery performs a static matchmaking between the received resource requirements and the discovered resources to get an eligible resource set, which is then forwarded to the

resource management component. The resource management component filters the resource set based on the resource availability and the validity criteria, such as the proximity of the user. Finally, the available resources that meet the criteria are leased to the user, and passed to the service assembly to perform dynamic service composition.

To explain the process in Figure 1.2 in more detail, consider the following use case: Kris is reading the presentation slides on his smartphone before he attends the class, going through the materials beforehand in the library. The presentation slides is the content of the task description. When he walks by a big display, he notices that he can use the display by pairing his phone and the display with Bluetooth. When he sends the Bluetooth pairing request from his phone, the discovery request for leasing the screen is sent to the service discovery component, then forwarded to the resource management component. The resource management component checks the availability of the display, and the distance between the display and the user. It turns out that the display is free to use (current content displaying has lower priority than the user), and the user is standing within the 5m-range of the display, the leasing process is then initiated.

Kris is allowed to use the display as long as the validity criteria are met, which are user proximity of 5 meters and maximum leasing duration of 30 minutes. The resource management component notifies the service assembly with an event containing the lease and the validity criteria. The service assembly component then adapts the presentation slides service to the leased resource, i.e. the display. The slides are transferred to the big display, while leaving a

user interface on the phone. Since Kris has to attend his class after 15 minutes, he leaves the proximity of the display, which violates the validity criteria of the lease. The lease is then ended, and the display stops showing the presentation slides.

Conflict resolution

Each user in the space tries to gain control on resources over other competitors, causing conflicts that need to be resolved by resource management. A priority-based scheme should handle the conflicts. Basically, users with higher priority can transiently override lower priority services. The system has several priority levels according to the smart space policy. Each user is assigned to a certain priority level per area. For example, a graduate student can be assigned to priority level 5 in his office (given priority levels from 1 to 5, 5 as the highest priority), and priority level 2 in the hallway. A user who tends to stay for a longer duration in a certain area should have higher priority over the ones who stay for shorter period of time. Users who visits the area frequently should have precedence as well. Moreover, resources can also be non-exclusive, depending on the resource type. For example, bandwidth can be shared between users simultaneously.

In summary, leasing checks the resource's validity criteria and availability status, resulting in conflict management capability. Temporal validity also enables the estimation of the resource availability, giving the possibility to prepare a resource for a user before it is actually free.

Hence, lease-based resource management method facilitates the optimization of the resource usage in smart spaces, thus helps increase the user's QoS.

1.2.3 Smart Space Performance Metrics

In smart spaces, Artificial Intelligence guides the user through multi-action sequences utilizing the computational resources available, and monitors the user status and experience along the way. The level of service (QoS) for users is crucial to define the performance of the entire system. Take a simple navigation scenario as an example, George is studying in a room on the second floor of the library, and he needs to read about materials from another book which can be found from a bookshelf on the fifth floor. The system senses his need, locates the book, and calculates a path to it. Then the system can guide George until he reaches the destination with the help of the displays and lights along the way. George's experience is dependent on whether or not the instructions that are given to him make sense, whether or not he can get to the book successfully, and when he changes the route, whether or not the system can sense the change, adapt to it, and give out new instructions. Consistency and adaptivity in the example are both human task performance metrics. Performance that can be represented in numbers is called functional performance. The key metrics for functional performance are listed below.

System blocking rate

System blocking rate refers to the ratio of the number of blocked users to the total number of users in the system. "Blocked" is defined by if the user has achieved his/her goal. The user is

considered being blocked either when he/she does not get the resource requested, or he/she cannot finish the task with the assistance from the environment. The cause of blocking can be different, unavailability of the resource, faulty instructions from the system, or task aborted by the user itself can all lead to blocking of the user.

Average delay

Delay refers to the time difference between the optimal approach and the actual approach for the task. The optimal approach is calculated in a perfect system with no resource conflict, 100% accurate sensing inputs and no irregular user behaviors. The actual approach is what really happens in the real system, where resource conflict, inaccurate sensing inputs and irregular user behaviors can all exist. Queueing management is applied upon the unavailability of the resources. Average delay is the average value of the delay for each user in the system.

1.2.4 User Privacy

Smart spaces maintain users' identity and preferences in order to keep track of the users' action sequences and make customized responses. Hence actions need to be taken to protect users' private information (including who they are, where they are, and what their preferences are) from being traced and revealed by adversaries or ill-intentioned people. Different security techniques can be applied according to the smart space policy. One approach to enhance the system security is to use authentication mechanism, which provides a reliable way of verifying the user's identity. The other approach would be using encryption when transferring raw sensor

data to the server, either from the user's portable devices or from the sensors deployed in the space. HTTPS (HTTP Secure) is a good choice for requests sent from user's devices. A more advanced cryptosystem (a revised ElGamal encryption algorithm) is introduced in [5].

1.3 Introduction to Indoor Localization

Resource management introduced in Section 1.2.2 uses location-based information to filter the resource set, i.e. the lease can only be valid if the user's position is within a certain proximity of the resource, which makes indoor location as one of the most important sensing inputs in smart spaces, as shown on the left side of Figure 1.1. Since almost everyone has a smart device in the pocket nowadays, smart device location information is used as the location of the user. Indoor localization has gained a lot of attention and research interests during the past two decades, as a result, various indoor localization solutions have been proposed.

1.3.1 Indoor Localization Solutions

Even though Global Positioning System (GPS) is extremely valuable for vehicle and even pedestrian navigation with an accuracy of around 10 meters outdoors, it is highly inefficient in the indoor environment due to signal attenuation caused by construction materials. Without line-of-sight connections between the mobile station and at least three satellites, using GPS indoors can be impossible. Given that GPS is generally not suitable for indoor localization, a variety of alternative solutions have been proposed to make ubiquitous location estimation.

Wireless-based solutions

There are two basic approaches to design a wireless localization system. The first approach is to deploy a signaling system and a network infrastructure specifically for localization purposes, such as a RFID or Bluetooth system. The advantage of this approach is that the designers have easier control of the physical specification and, consequently, the quality of the location estimation. The density of the RFID or Bluetooth transceivers can be designed based on customized localization requirements (e.g. 10-meter accuracy required) and the acceptable budgets. The second approach is to use an existing wireless network infrastructure to locate a target, such as a WLAN or cellular system. The advantage of the second approach is that it avoids the expensive and time-consuming deployment of infrastructure. However, more intelligent localization algorithms need to be used to compensate for the lack of control over the physical specification of the system.

Several types of wireless technologies are used for indoor localization. Figure 1.3 gives a rough overview of the current wireless-based technologies, and their differences in indoor localization accuracy, which is a simplified version of [6, Fig. 6]. Ultra-wideband (UWB) radio, measuring Angle of Arrival (AOA) and Time Difference of Arrival (TDOA) have the best localization accuracy. Hybrid methods of Radio Frequency (RF) and Infrared (IR), RF and ultrasonic technologies can achieve the accuracy of around 1 meter. The accuracy of other technologies like WLAN, Bluetooth, Digital Enhanced Cordless Telecommunications

(DECT), and ZigBee can be from 3 meters to 10 meters. More detailed description of various wireless technologies and their localization performance can be found in Chapter 2.

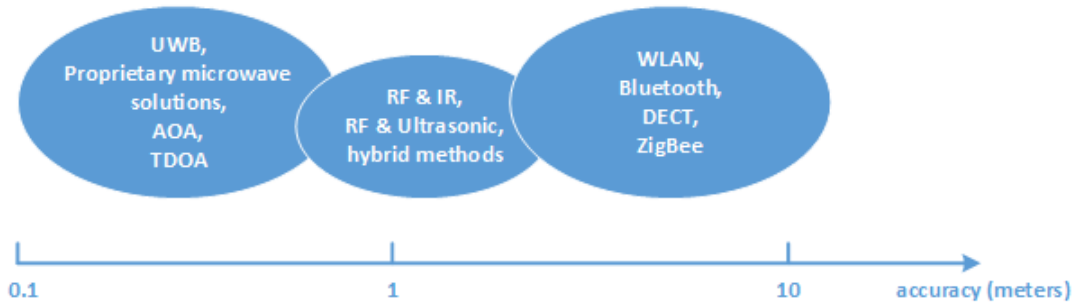

Figure 1.3 Overview of current wireless-based indoor localization technologies

Sensor-based solutions

Most smartphones or tablets have inertial sensors built in, which include accelerometer, gyroscope, altimeter, etc. Indoor localization systems can take advantage of the embedded inertial sensors to achieve localization. For example, one possible algorithm can rely on accelerometer-based step detection and step length estimation, along with gyroscope measurements for direction and altimeter or barometer measurements for floor detection to estimate location within known indoor map constraints as long as the entry point to the building is known (either from GPS or distinct landmarks in the building). Sensor-based indoor localization technology is a relatively new research area since the embedded sensors on smart phones have only been developed for a few years. Even though it can be a stand-alone

algorithm as described in the example, most of the time, inertial sensor data is used as a supplement of the wireless-based technologies to enhance the localization accuracy [7].

Ambience-based solutions

Ambience-based indoor localization utilizes ambient sound, light, color and other landmarks as signatures, which can be sensed by the smart phones' sensors and cameras. The main phases for the ambience-based solutions are ambience calibration phase and localization phase. The time-stamped ambience signatures are collected during the calibration phase. Then the smartphone collects the sensing data again, and the localization algorithm returns the user's location based on the closest match in the fingerprint database. Different types of signatures include light and acoustic signatures, and landmark signatures.

1.3.2 Effect of Indoor Localization

Various indoor location estimation solutions like wireless-based, sensor-based and ambience-based solutions, or the combinations of two or more solutions can give an accuracy of less than 10 meters, and different technologies have different estimation accuracies (Figure 1.3). However, localization technologies are not perfect, it is not possible to achieve an accuracy of 100%. Different location estimation performance may lead to different system performance.

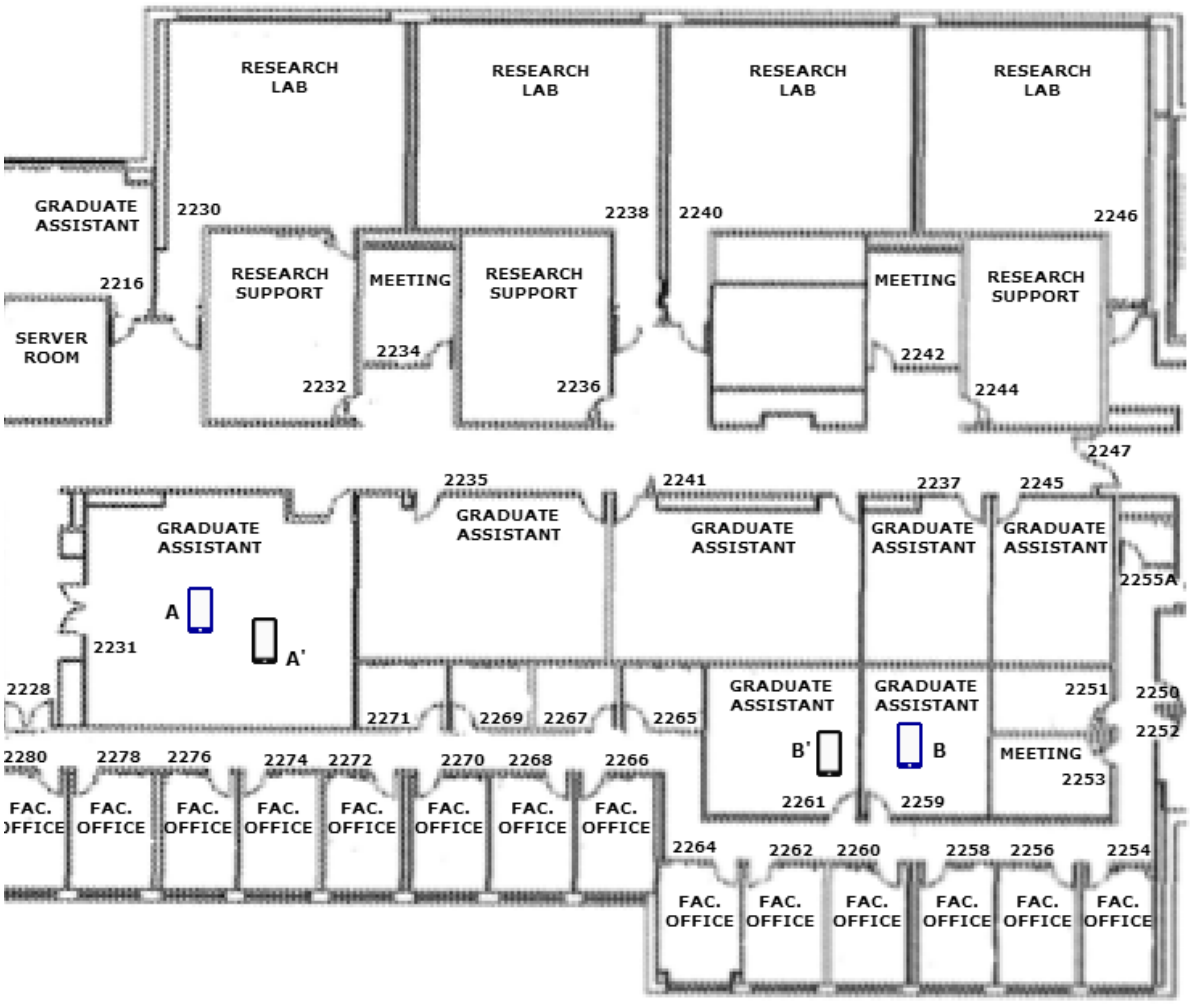

Figure 1.4 Illustration of the effect of indoor localization, in east wing of EBII, NCSU

The effect of indoor localization can be various as well, because a few meters' estimation error can put the target in a completely different room in the space. As shown in Figure 1.4, user B is in room 2259 currently, but the system (the smart space) recognizes him/her as point B' in room 2261, due to localization error. That is, the system recognizes the user on the other side of the wall, even though the localization error is only a few meters. Since the actual position

and the estimated position of user B are in two rooms, if user B wants to use the projector in room 2259 (actual position), the system won't be able to lease the projector to B even though the projector is not occupied at the moment, because it thinks B is in room 2261 (estimated position). The system fails to provide service to the user due to localization error, the user is "blocked". While in some other cases, the penalty due to localization error might be subtle. For example, user A is in room 2231 now, and his/her estimated position A' is in the same room. Even though the estimated position is still off by a few meters, this won't be a huge issue when it comes to validating the user's proximity to the resources.

1.3.3 Indoor Localization Performance Metrics

Referring to [6] and considering the differences between various technologies, the following performance metrics for indoor localization system are considered: accuracy, precision, complexity, dimension, and cost. There is often a tradeoff between accuracy, precision and other characteristics like complexity and cost. The comparison among different localization solutions is made in Table 2.1, Chapter 2.

Accuracy

Accuracy is the most important characteristic of a localization solution. It's calculated as the average estimation error (the Euclidean distance between the actual position and the estimated position). The lower the estimation error, the better the localization solution.

Precision

Precision refers to the distribution of the estimation error, i.e. it measures the variation of the accuracy over time. Precision is represented by the cumulative probability functions (CDF) of the estimation error. For example, if the localization solution has a precision of 90% within 5m, it means that the probability of estimation error being less than 5 meters is 0.9 on the CDF graph. Higher precision means that estimation errors are concentrated in small values. The higher the precision (reaches high probability value faster), the better the localization solution.

Complexity

Complexity includes hardware and software complexity. Only software complexity is considered in this thesis. Thus, complexity refers to the computing complexity of the localization algorithm. Computing complexity consists of both time complexity and space (memory) complexity. The lower the complexity, the better the localization solution.

Dimension

Dimension refers to the ability to locate a target in a 2-D or 3-D space. Some localization solutions only focus on locating in 2-D spaces, while other solutions with floor-detection capability can support both 2-D and 3-D spaces.

Cost

Cost of a localization solution depends on many factors, including money, time, space, and energy. Money refers to the amount of money needed to purchase the necessary hardware and software specifically for localization purposes. Time is related to the effort in installation and maintenance needed. Space refers to how much space the localization hardware takes. Energy is also an important cost factor. Some localization units (e.g., RFID tags, and QR code tags) are completely energy passive, and they can have an unlimited lifetime. Other units (e.g., devices with rechargeable battery) can only last for several hours without recharging.

1.4 Objectives

The objectives of this thesis are as follows:

- **Conduct a comprehensive survey on indoor localization solutions.**
    - Present an overview of the existing indoor localization solutions.
    - Classify different location estimation technologies.
    - Compare the performance of different localization solutions using the proposed metrics.
- **Implement a cross-platform WLAN indoor localization system.**
    - Design the server backend for the localization system.
    - Implement a WLAN-based location estimation algorithm.
    - Build an Android application to demonstrate the system.

- **Analyze how indoor localization performance affects smart space performance.**
    - Build a simulation platform for resource management in smart space with indoor localization capabilities.
    - Use the simulation results to analyze the performance dependencies.
    - Solve the cost optimization problems with available resources and different system load.

1.5 Overview of Thesis

The thesis is organized as follows. Chapter 2 presents various indoor localization solutions and the comparison among different solutions. Chapter 3 describes the implementation of the WLAN indoor localization system. The software development process as well as UML-based descriptions are provided. Chapter 4 gives a description of the simulation platform for the way finding case study for resource management modeling in smart spaces. The dependency between the indoor localization performance and smart space performance is evaluated based on the simulation results. The optimal solution for choosing the proper localization solution (accuracy) with available budget, resources under different system load to achieve acceptable QoS is derived. Finally, Chapter 5 concludes the thesis.

CHAPTER 2 Indoor Localization

2.1 Overview

Indoor localization has gained a lot of attention and research interests during the past two decades. Since Global Positioning Systems (GPS), the most popular outdoor localization method, does not work indoors, a variety of alternative solutions have been proposed for indoor localization (RADAR, Horus, SurroundSense, WILL, etc. [8, 9, 10, 11]). The notion of location can range from physical coordinates in 2-D/3-D maps to logical labels (such as store names or room IDs) [10]. The choice of whether to deploy physical localization or logical localization mostly depends on the requirement of the application. For example, if the application only needs room-level localization accuracy (such as the apps that send coupons to customers who are in the store), physical coordinates and logical labels are both acceptable. Most researches have been focusing on physical localization, while logical localization has also been studied lately.

Different indoor localization solutions include wireless-based solutions, sensor-based solutions, and ambience-based solutions. Wireless-based solutions have been studied extensively during the last ten years. As the mobile technology being developed rapidly, more and more sensors are now available in mobile phones or tablets. The use of cameras, microphones, and other inertial sensors such as accelerometer on the mobile devices make sensor-based solutions and ambience-based solutions possible, which brings variety to indoor localization algorithms.

2.2 Wireless-Based Solutions

As discussed in Section 1.3.1, there are two basic approaches to design a wireless-based localization system. One is to deploy a signaling system and a network infrastructure specifically for localization purposes, such as a RFID or Bluetooth system. The other one is to use an existing wireless network infrastructure to locate a target, such as a WLAN or cellular system. Both approaches have their own advantages and disadvantages, the choice of the technology should be made considering the specific localization requirements, available hardware, and the acceptable budget. Various wireless technologies for both approaches are introduced in this section.

2.2.1 Indoor Wireless Environment

According to the characteristics of the Radio Frequency (RF) signals, it is inevitable for the signals to experience absorption, reflection, scattering, refraction, diffraction, multipath fading, time delay, attenuation (free space path loss), and gain during propagation from the transmitter antenna to the receiver antenna. The RF propagation behaviors can vary drastically depending on the materials on the signal's path and the signal's frequency.

Indoor WLAN environment

Given that IEEE 802.11 is currently the dominant local wireless networking standard, an existing WLAN infrastructure can be used for indoor localization by adding a location server

and employing received signal strength indicator (RSSI) as a metric for location determination. Several studies have been performed to model the indoor received signal strength. According to [12], the indoor received signal strength is normally distributed. An experiment is conducted at one particular point for a certain Access Point (AP) in EBII, NCSU, one signal strength sample is measured every 0.8 seconds for a duration of 1 hour. The measurements are taken by a Lenovo T420 laptop. The experimental results are shown in Figure 2.1, with the RSSI distribution and its Gaussian approximation curve (μ = -59.95, σ = 0.67). The results suggest Gaussian being a reasonable approximation.

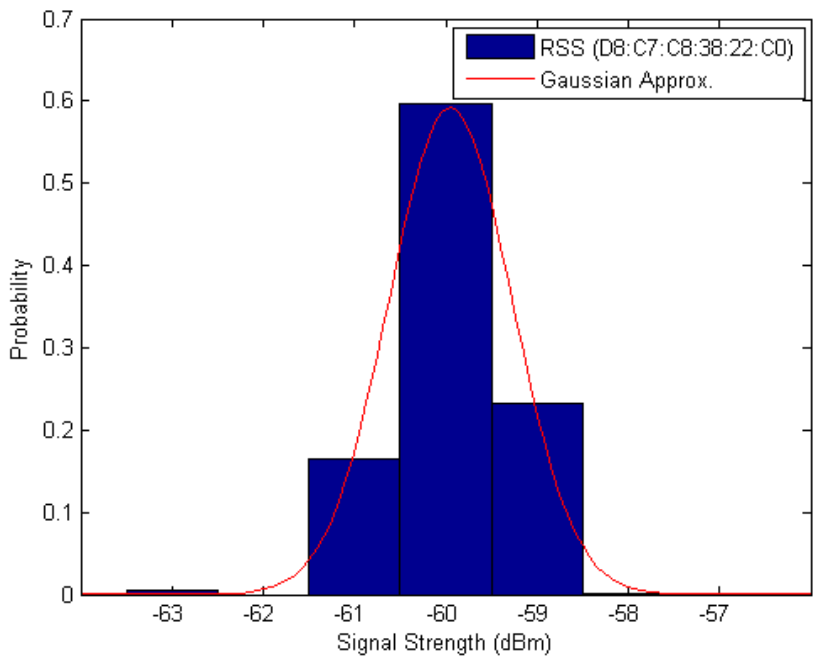

Figure 2.1 Distribution of RSSI at a particular point regarding one AP (MAC address as D8:C7:C8:38:22:C0) and its Gaussian approximation, when user is not present

RSSI can be affected by many factors like the building materials, floor layout, moving objects, and even user's presence. Human body can greatly attenuate the WLAN signals. [12], Fig. 2 shows that the presence of users can greatly spread the distribution of the RSSI. In addition, the user's orientation can also block the signal from a certain AP. With different APs in the space, the independence of RSSI values from different APs is assumed. The assumption is based on the experiments in [12] that evaluate the correlation factor among the different APs' RSSI values as almost zero, i.e. the different APs' RSSI values are uncorrelated.

2.2.2 Wireless-Based Localization Algorithms

Due to severe multipath, lack of Line-of-Sight (LOS) paths, and scattering objects in indoor wireless environment, it is very tricky to model the radio propagation. There are various wireless-based algorithms for location determination. They are divided into four major groups: proximity, distance measurements, angle measurements and fingerprinting. Distance measurements and angle measurements are classified as triangulation. A detailed classification of wireless-based localization algorithms is shown in Figure 2.2.

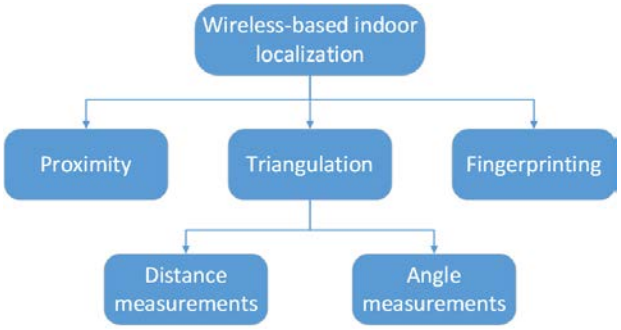

Figure 2.2 Classification of wireless-based indoor localization algorithms

2.2.2.1 Proximity

Proximity algorithms are the simplest localization algorithms to implement. The position of the mobile device is determined by the cell of origin (CoO) method with known antenna positions and limited range. When a mobile device is detected by a single antenna, it is considered to be at the location of the antenna. When more than one antenna detects the mobile device, it is considered to be at the location of the one that receives the strongest signal. The accuracy of CoO relates to the density of the antenna grid and its signal range. In practice, the systems using infrared radiation (IR) and radio frequency identification (RFID) are often based on this method [6].

2.2.2.2 Distance Measurements (Lateration)

The location of the user can be estimated based on its distances from multiple reference points. Distances are measured using time of arrival (TOA), time difference of arrival (TDOA), or received signal strength indicator (RSSI).

TOA (time of arrival)

Since the distance between the transmitter and the receiver is proportional to the radio propagation time, by measuring the one-way propagation time between the user and at least three reference points, the user's location can be determined (as shown in Figure 2.3). The one-way propagation time is calculated by the time difference between the timestamp sent with the transmitted signal and the system time upon receipt of the signal, which means that all

transmitters and receivers in the system have to be precisely synchronized for the algorithms to work correctly. Three different TOA algorithms, Closest-Neighbor (CN) algorithm, Least-Squares (LS) algorithm and Residual Weighting (RWGH) algorithm are presented in [13]. TOA algorithms can be used for ultra-wideband (UWB) measurements.

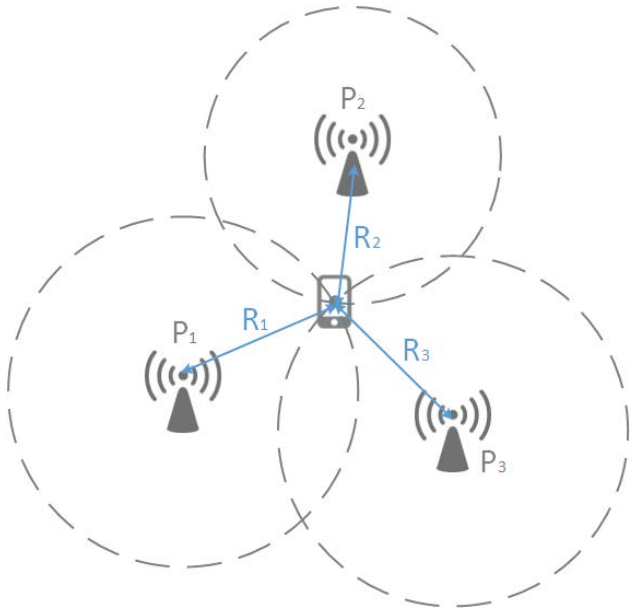

Figure 2.3 Indoor localization using TOA

TDOA (time difference of arrival)

Instead of using absolute time in TOA, TDOA uses the time differences when the signal arrives at multiple reference points to determine the relative position of the user. TDOA algorithms require the synchronization among all the reference points, but does not have any requirements

on the user. A conventional solution to TDOA algorithm can be found in [14]. Li et al. [15] present a delay measurement-based TDOA algorithm for 802.11 wireless LANs.

RSSI (received signal strength indicator)

Since it is difficult to find a LOS path from the transmitter to the receiver, and RF propagation suffers from multipath effect in indoor environments, location estimation accuracy with TOA and TDOA algorithms can be decreased significantly. An alternative approach is to estimate the RF propagation distances according to the measured RSSI values using path loss model and multipath model. However, due to the irregular and unpredictable signal propagation in indoor environment (with densely located obstructions), the distance-estimation model can get very complicated, and the parameters employed in the models are site-specific. RADAR [8] introduces a signal propagation model using wall attenuation factor (WAF) and floor attenuation factor (FAF). Madigan et al. [16] propose a Bayesian hierarchical model.

2.2.2.3 Angle Measurements (Angulation)

The location of the user can also be determined based on its angles from multiple reference points, which is called the angle of arrival (AOA) algorithm. As shown in Figure 2.4, the location of the user is the intersection of two lines formed from the reference points to the user. At least two known reference points (P_1 and P_2) and two measured angles (θ_1 and θ_2) are required to determine a 2-D location. To derive a 3-D location, three known references points

are required. Direction finding (DF) is accomplished with highly directional antennas or antenna arrays [6], which adds more hardware cost to the system.

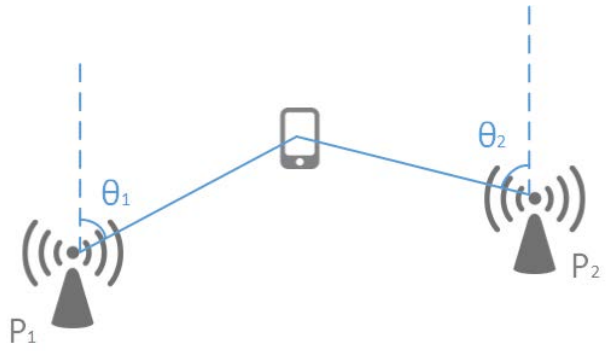

Figure 2.4 Indoor localization using AOA

A common drawback that AOA shares with TOA and TDOA is its susceptibility to multipath interference [17]. AOA, TOA and TDOA work well with direct line of sight (LOS), but suffer from decreased accuracy and precision when confronted with multipath effect. Unfortunately, in indoor wireless environment, they are barely usable because line of sight to two or more reference points is difficult to find.

2.2.2.4 Fingerprinting

Fingerprinting techniques exploits the relationship between a location and its corresponding radio signature [18]. Fingerprinting solutions can be implemented completely in software, which can reduce localization complexity and cost significantly compared to lateration and angulation solutions. Most fingerprinting solutions use received signal strength indicator

(RSSI) at the client side as the radio signature (fingerprint). A fingerprinting localization process is divided into two phases: calibration phase and localization phase. Radio signatures are collected at different locations in the area of interests during the calibration phase, which take the radio propagation effects like multipath and attenuation from objects in the space into account. The localization phase match the reported radio signature of the target against calibration-map database.

Calibration phase

In the calibration phase, the location fingerprints are collected by performing a walk-around of the space with a mobile device. The *location fingerprint*, $\mathcal{F}$, refers to the vector of RSSI values from each access point at their corresponding *location*, L in the space. A *calibration map* refers to the collection of fingerprints and their associated locations, $<\mathcal{F}, \mathcal{L}>$. Since the RSSI values fluctuate over time for a given location (estimated as a Gaussian distribution as shown in Figure 2.1), a fingerprint can be the vector of the mean RSSI values from each access point at a certain location, the size of the vector is the number of the access points can be detected. Assuming that N access points can be detected at a certain location, and the mean RSSI values from the ith access point is r_i, the fingerprint at the given location can be described as:

$$F = (r_1, r_2, ..., r_N) \tag{2.1}$$

Localization phase

In the localization phase, the user's mobile device measures RSSI values from each detected access point, and forwards the vector of the values to the location server. The location server uses a localization algorithm and the calibration-map database to estimate the location of the mobile device. The server then reports the estimated location estimate back to the mobile device. Fingerprinting localization algorithms can be classified into three groups:

- *Deterministic algorithms*

  Assume that the calibration map <$\mathcal{F}$, $\mathcal{L}$> contains m location fingerprints. The fingerprint set $\mathcal{F}$ is represented as $\{\mathrm{F}_1, \mathrm{F}_1, ..., \mathrm{F}_m\}$, and the location set $\mathcal{L}$ is represented as $\{\mathrm{L}_1, \mathrm{L}_2, ..., \mathrm{L}_m\}$. Each fingerprint is represented by the vector $F_i = (\mathrm{r}_1^i, \mathrm{r}_2^i, ..., \mathrm{r}_N^i)$, where $i \in [1, \mathrm{m}]$. In the localization phase, a sample RSSI-value vector $S = (s_1, s_2, ..., s_N)$ is measured from N access points, where s_i is the RSSI value from the ith access point. The deterministic algorithms attempt to find the fingerprint F_i that is the "closest" to the current measurement S. "Closest" is defined as the fingerprint in the database that has the minimum signal distance to S.

$$Dist(\mathrm{S}, \mathrm{F}_j) \leq \mathrm{Dist}(\mathrm{S}, \mathrm{F}_k), \quad \forall\, \mathrm{k} \neq \mathrm{j} \tag{2.2}$$

  The distance function $Dist(\cdot)$ can be computed using Euclidean, Manhattan, or Mahalanobis distance [19].

In practice, estimating the location using only the closest fingerprint can be error-prone. An improvement is to use kNN (k nearest neighbor) algorithm. Instead of using only the closest one, kNN estimates the location based on the average position returned by k nearest neighbors. According to [8], the location accuracy increases as the value of k increases, until k = 8 [19].

- *Probabilistic algorithms*

Probabilistic algorithms use probability inferences to determine the likelihood of a particular location [17]. The most popular approach is to use Bayesian probability inferences. Assume that there are m location candidates $L_1, L_2, ..., L_m$, and S is the measured RSSI vector during the localization phase. Location L_i will be chosen if:

$$P(L_i \mid \mathrm{S}) \geq P(L_j \mid \mathrm{S}), \quad \forall i, \mathrm{j} \in [1, m], \ \ \mathrm{j} \neq \mathrm{i} \tag{2.3}$$

where $P(L_i \mid \mathrm{S})$ denotes the probability that the mobile device is at location L_i, given that the measured RSSI vector is S. Assume that $P(L_i) = P(L_j)$) for i, j = 1,2,...,n. Based on Bayes' formula,

$$P(L_i \mid \mathrm{S}) = \frac{P(\mathrm{S} \mid L_i) P(L_i)}{P(\mathrm{S})} = \frac{P(\mathrm{S} \mid L_i) P(L_i)}{\sum_{k=1}^{m} P(\mathrm{S} \mid L_k) P(L_k)} \tag{2.4}$$

We have the decision rule that location Li is chosen if:

$$P(\mathrm{S} \mid L_i) \geq P(\mathrm{S} \mid L_j), \quad \forall i, \mathrm{j} \in [1, m], \ \ \mathrm{j} \neq \mathrm{i} \tag{2.5}$$

The most popular approach, kernel approach can be used to calculate $P(\mathrm{S} \mid L_i)$, i.e. the probability that the measured RSSI vector is S, given that the mobile node is located at location L_i. Assume that for each location L_i in the space, n RSSI samples are taken from each access point (N in total). Each sample is assumed to be Gaussian distributed with a mean of μ and a standard deviation of σ (Figure 2.1). Then the likelihood function can be defined as an equally weighted Gaussian kernel function:

$$P(\mathrm{s} \mid L_i) = \frac{1}{n}\sum_{j=1}^{n}[\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(\mathrm{s}-\mu_j)^2}{2\sigma^2})] \tag{2.6}$$

Since the RSSI values from multiple access points are independent from each other, and $S = (s_1, s_2, ..., s_N)$,

$$P(\mathrm{S} \mid L_i) = P(\mathrm{s}_1 \mid L_i) \cdot P(\mathrm{s}_2 \mid L_i) \cdots P(\mathrm{s}_N \mid L_i) \tag{2.7}$$

- *Other algorithms*

  Other algorithms include neural networks and support vector machine (SVM) algorithms. Neural networks assume that the RSSI is too complex to be analysed mathematically and require non-linear discriminant functions for classification [17]. Extensive training data is needed to adequately train the neural networks in optimizing the weighting parameters. SVM is a machine learning algorithm that can be applied to classification or regression. The indoor localization performance of SVM is almost the same as the deterministic algorithms. However, SVM is more suitable for determining if an area is inside or outside a room.

Even though fingerprinting algorithms are very popular for indoor localization systems due to their relatively low computing complexity, low cost and high accuracy in severe multipath environments, they do have drawbacks. One of the drawbacks is their labor-intensive site survey (calibration phase), the other one is that they fully depend on the existing infrastructure, which means that if the infrastructure changes, all the fingerprints will need to be recollected, and the fingerprint database will need to be repopulated. For example, if the manager decides to install a new set of routers from a different manufacturer to replace the existing ones, all of the existing fingerprints will become useless.

Table 2.1 Comparison of wireless-based indoor localization algorithms

Algorithm		LOS/NLOS (non-line-of-sight)	Affected by multipath	Antenna position needed	Accuracy[1]	Cost	Dimension (2D/3D)
Proximity		LOS, NLOS	No	Yes	Low	Low	2D, 3D
Lateration	TOA	LOS	Yes	Yes	High	High	2D
	TDOA	LOS	Yes	Yes	High	High	2D
	RSSI	LOS, NLOS	No	Yes	High	Medium	2D, 3D
Angulation	AOA	LOS	Yes	Yes	Medium	High	2D, 3D
Fingerprinting		LOS, NLOS	No	No	High	Medium	2D, 3D

[1] Accuracy is evaluated based on the assumption of perfect wireless environment. For example, for TOA and TDOA algorithms, we are assuming that multiple LOS paths are available, and there are no severe multipath effect.

2.2.3 Survey of Wireless-Based Solutions

Having introduced the different localization algorithms and the comparison among them (Table 2.1), we will discuss about specific wireless-based localization solutions. There are various types of wireless technology that are used for indoor localization, including radio frequency identification (RFID), ultra-wideband (UWB), WLAN, Bluetooth, infrared radiation (IR), ultrasound, ZigBee, etc.

2.2.3.1 RFID (Radio Frequency Identification)

RFID is a non-contact (direct contact or line-of-sight contact) and automatic identification technology that uses radio signals to identify, track and detect objects. An RFID system consists of several RFID readers and RFID tags. RFID readers and tags can send and receive data using a defined RF and protocol. RFID tags can be either active or passive. Passive tags operate without a battery, and they are lighter, smaller, less expensive, and more energy efficient compared to active tags. The passive tags have a communication range of 3m, while the communication range of active tags is more than 100m. A well-known indoor localization system that uses active RFID technology is LANDMARC [21]. It uses the fingerprinting (kNN) algorithm to calculate the RFID tags. In order to increase accuracy without placing more readers, the system employs the idea of having extra fixed location reference tags to help with the location calibration. LANDMARC has an error distance of 1 m for 50% of the time, and a maximum error distance of 2 m. An RFID-based localization system using passive tags is presented in [20]. It uses the nearest neighbor method in the fingerprinting algorithm for

location estimation. The experiment is conducted in a 9×9 grid with a cell size of $1m^2$. 6 trials out of the 90 trials estimated the location incorrectly, giving a precision of 93% within 1 m.

2.2.3.2 UWB (Ultra-wideband)

UWB is a radio technology that transmits ultra-short pulses (less than 1ns), with low duty cycle over a large portion of the radio spectrum (more than 500 MHz). The brevity of the emitted UWB pulses minimizes the multipath interference, because the reflected pulses are well separated in time from those taking the direct path between transmitter and receiver [22]. A Dublin-based fabless semiconductor manufacturer, DecaWave[2] has a wireless-networking chip (ScenSor DW1000) designed to provide extremely precise indoor locations using UWB. Indoor localization is achieved by installing the chips into both users' phones and all the access points. It utilizes TDOA algorithms to measure the time it takes the pulses to travel to a fraction of a nanosecond, allowing the precision of less than 10 cm, even while moving at up to 5m/s. Even though UWB has high immunity to multipath fading, the UWB signals get blocked too easily (40% of the time according to [22]) in various indoor environments, leading to the possibility of the first pulse detected not being the LOS path. The absence of the LOS signal can easily create errors of a meter or two, which gives a localization accuracy similar to the other wireless-based solutions. Other companies that provide UWB location equipment include Berlin-based Nanotron[3], Time Domain of Huntsville, Ala.[4], and Zebra Technologies of Lincolnshire, Ill[5].

[2] http://www.decawave.com/products/dw1000
[3] http://www.nanotron.com/EN/index.php
[4] http://www.timedomain.com/
[5] http://www.zebra.com/us/en.html

2.2.3.3 WLAN

The WLAN (IEEE 802.11) standard, operating in the 2.4-GHz ISM (Industrial, Scientific and Medical) band, is the most popular local wireless networking standard currently. WiFi signals can be found in almost every building now, and they can penetrate through walls and obstructions where GPS fails. Due to the excellent signal accessibility indoors and its ubiquity, WLAN (WiFi) indoor localization systems can be deployed without installing extra hardware as RSSI values can be obtained directly from NICs that are available on most handheld computing devices. WLAN infrastructures provide a cost-effective solution for localization in indoor environments.

An early attempt to adopt WLAN technologies in location determination is RADAR [8]. RADAR proposed two solutions: empirical and signal propagation modeling. The first one utilizes the kNN fingerprinting algorithm as described in Section 2.2.2.4. The second one uses the wall attenuation factor (WAF) and the floor attenuation factor (FAF) propagation model. The result for RADAR system shows that, the propagation solution provides an accuracy of about 4.3 m compared to an accuracy of 2.94 m for the empirical one. For the empirical solution, results show that 25% of the error distances is within 1.92 m, 50% of the error distances is within 2.94 m and 75% of the error distances is within 4.69 m. Horus system [9], improved upon RADAR, employs a stochastic description of the RSSI-location relationship and uses a maximum likelihood-based method to estimate locations. The experiment results show that the average accuracy of the Horus system is better than the RADAR system by more than 82%.

WILL [11] presents an indoor logical localization approach without site survey or knowledge of AP locations. Fingerprints are partitioned into different virtual rooms based on RSSI stacking difference. Fingerprints with high similarity are put into one virtual room. Different virtual rooms then construct the logical floor plans, which are mapped into physical floor plans using betweenness centrality of a vertex and shortest paths length between vertices. The implementation results show that WILL can achieve an average room-level accuracy of 86%. EZ [23] models the physics constraints of wireless propagation with path loss model and uses a genetic algorithm to solve them for localization. However, EZ still relies on occasionally available GPS information at the entrance or near a window. Besides, EZ involves complex computation and the physical localization scheme might result in lot of misdetections of rooms.

From the commercial and research systems built based on WLAN fingerprinting algorithms, it is observed that there is a trade-off between the measurement efforts and the localization performance. The more the number of samples are taken during the calibration phase, the more accurate the location estimation is. Researchers have also been working on reducing the measurement efforts, like WILL [11], but it has to trade with more complicated localization algorithm and increased location estimation latency.

2.2.3.4 Bluetooth

Compared to WLAN, Bluetooth has shorter transmission range, which makes it easier to be confined to a single room. Each Bluetooth transceiver (beacon) has a unique ID, which can be used for locating the Bluetooth tag. Devices can determine their locations based on the ID of

the Bluetooth “beacon” they are getting the signal from. Bluetooth can provide higher indoor localization resolutions than WLAN. However, there are several drawbacks of using Bluetooth technology for localization. One of the drawbacks is the extra cost for installing special hardware (beacons) in the space. Another drawback is that the beacons run the device discovery procedure upon each location request, which significantly increases the localization latency (by 10 to 30 seconds) and the power consumption. An Austria-based indoor positioning software company, indoo.rs[6] has developed a Bluetooth solution (iOS and Android SDK released) based on Bluetooth LE (Bluetooth 4.0) standard. Their solution uses the signals from beacons installed in the space to determine the real-time location, allowing an accuracy of 1-2 meters. Other companies, like the France-based company, Insiteo[7] is also working on ways to squeeze precise indoor localization out of Bluetooth beacons.

2.2.3.5 Others

Other wireless technologies have also been studied for feasibility in indoor localization. IR (infrared radiation) is one of most popular ones on the list. Most IR wireless devices require line-of-sight (LOS) communication between the transmitter and the receiver. IR devices are small in volume and light in weight, and IR-based systems can precisely locate the target. However, interference from fluorescent light and sunlight can significantly degrade the localization performance [6]. Besides, hardware and maintenance of IR devices are expensive, which makes scalability an issue. Ultrasound signals can also be used to estimate indoor

[6] http://indoo.rs/
[7] http://www.insiteo.com/joomla/index.php/en/

locations with an accuracy of around 10 cm, but they are blocked by obstructions like walls easily and suffer a lot from the reflected-signal interference. Recent research in ultrasound-based indoor localization can be found in [24]. Hu et al. [25] presents a ZigBee-based solution for indoor localization.

2.3 Sensor-Based Solutions

As introduced in Section 1.3.1, most smartphones and tablets have built-in inertial sensors that measure motion and orientation of the device. Raw data provided by the sensors can be used to detect 3-D device movement and its position indoors. According to [27], if the initial location of an object and all forces applied the object are known, its position can be tracked over time. If the initial location is not known, only the relative position to the origin can be estimated.

2.3.1 Sensors

Figure 2.5 shows the sensor coordinate system used by Android Sensor API. The sensor framework uses a standard 3-axis coordinate system. When a device is held in its default orientation, the X axis is horizontal and points to the right, the Y axis is vertical and points up, and the Z axis points to the front side of the screen [26]. This coordinate system is used by sensors including accelerometers, gyroscopes, magnetometers, etc.

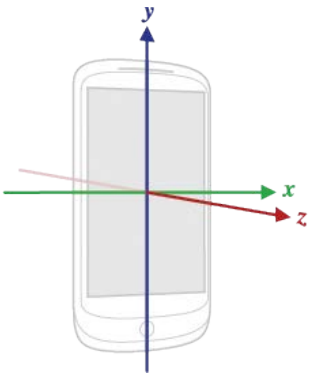

Figure 2.5 Sensor coordinate system (relative to a device)
used by Android Sensor API [26]

Accelerometer

The Accelerometer measures the acceleration applied to the device, including the force of gravity. According to [28], accelerometers provide apparent evidence of human walking patterns. An experiment is conducted by walking along the hallway of EBII, NCSU with a Nexus 7 with Android 4.4. An Android application is created to record the raw accelerometer data, the code snippet of the application can be found in Appendix A of the thesis. The sensor data is a vector of three values with each one being the reading on x, y, or z axis (Figure 2.5). The raw z-axis readings from the first 14 steps taken are shown in Figure 2.6 as the blue line. As illustrated in Figure 2.6, magnitude of about 2 m/s^2 is caused by foot lifting and about 3 m/s^2 by putting the foot down. This signature can be exploited for step detection (marked as crosses in Figure 2.6).

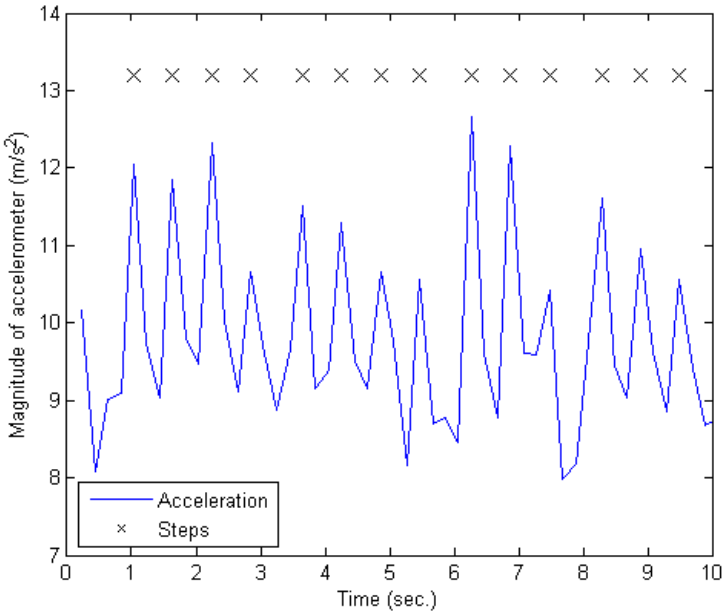

Figure 2.6 Acceleration signatures of 14 steps

Gyroscope and Magnetometer

The gyroscope measures the rotation in rad/s around a device's x, y and z axis. Together with the magnetometer that monitors changes in the earth's magnetic field, it is possible to calculate the change of angles on the device over time, which can be explored to detect the user's direction. With the knowledge of the user's average step distance, we can estimate how far away the user has walked towards which direction. Therefore, the sensors on the smartphone or the tablet can pretty much track the user's movement in the space, and determine the indoor location. The algorithm based on detecting steps and step headings is called *Dead Reckoning*.

Another approach is to use magnetic fields (collected by the magnetometer) as unique signatures in fingerprinting algorithms.

Compared to the fingerprinting algorithms, sensor-based dead reckoning algorithms can reduce the measurement efforts dramatically since constructing a dense fingerprint database, with 1-2 meters fingerprint samples distance, can be very labor-intensive. However, sensor-based dead reckoning algorithms have drawbacks. First, the algorithms need to have complete knowledge of the building plan, and mapping the model of the building into software can be very complicated. Second, the availability of multiple inertial sensors on different mobile devices can also be a constraint.

2.3.2 Survey of Sensor-Based Solutions

Even though sensor-based indoor localization technology is a relatively new research area, there has been several systems developed that attempt to utilize sensor data to determine indoor positions. Liu et al. [30] proposes an indoor positioning system based on the accelerometer and compass on ubiquitous smart phones. In the system, only limited WiFi fingerprints at the center of each room and each segments of corridors are collected. Initial estimation is derived as the most similar room among all the fingerprints. Digital compass measurements are taken at each step detected by the accelerometer. Localization solution is based on the particle filter. By eliminating the particles whose trajectory intersects with walls dynamically, positioning error decreases over time, which can achieve the accuracy of 0.33 m at the last position. The Android application “Footpath” has been developed by Bitsch et al. [29] using an accelerometer and a

compass, to follow a path in a building. The project is published as open sources software[8]. A Finland-based indoor localization software company, IndoorAtlas[9] has developed a fingerprinting-based solution using a smartphone's magnetometer measurements as location signatures, allowing an accuracy of less than 3 meters for 90% of the time.

2.4 Ambience-Based Solutions

Most ambience-based solutions use fingerprinting algorithms to pinpoint users' location indoors. As discussed in Section 2.2.2.4, fingerprinting algorithms are divided into two phases, calibration phase and localization phase. Time-stamped ambience signatures are collected during the calibration phase. In the localization phase, the smartphone collects the sensing data again, and the localization algorithm (deterministic algorithm, probabilistic algorithm, etc.) returns the user's location based on the closest match in the fingerprint database. Even though ambience-based fingerprinting localization solutions depend less on the network infrastructure compared to wireless-based fingerprinting solutions, recognizing fingerprints from raw ambience data can be labor intensive due to the huge variety of data. Another drawback of ambience-based solutions is that motion recognition may take time to collect, which increases the localization latency. Different types of ambience signatures include light and acoustic signatures, and landmark signatures.

[8] https://github.com/COMSYS/FootPath/
[9] https://www.indooratlas.com/

2.4.1 Light and Acoustic Analysis

Different places can have different light and acoustic signatures, which makes light and acoustic analysis a feasible solution for indoor localization. For example, coffee shops can have specific noise signatures from coffee machines and microwaves, which are different from the clinking sounds of forks and spoons in restaurants [10], pubs can have high noise level and dark indoor environment.

SurroundSense [10] considers audio sample amplitude distribution as the ambient sound fingerprint for acoustic processing, so it distinguishes between stores with different loudness characteristics. And the system translates pixels of the floor pictures to a hue-saturation-lightness (HSL) space, and considers the light intensity on L-axis as the ambient light fingerprint. These two fingerprints alone do not give good localization accuracy, but they work well as the sub-stages in the multi-sensor localization method. Tarzia et al. [31] introduces the Acoustic Background Spectrum (ABS), an ambient sound fingerprint technique. The system first calculates the ABS of the room, and then classifies the room by comparing its ABS with the existing, labeled ABS values in the database. The system yields 69% correct fingerprint matches.

2.4.2 Landmarks

Landmarks can be the different signatures that naturally exist in the environment. Similar to the way in which people provide directions using easily recognizable landmarks: turn left at

the red wall and keep going until you reach the help center. These landmarks can be detected by a smartphone's sensors or cameras. Example landmarks can be the distinct motion signature of an elevator (abrupt change in altitude), or certain dead points where there are no WiFi or cellular signals, or a distinct color of the carpet or the wall [32]. SurroundSense [10] considers human movement pattern as a fingerprint. Two simple states from the accelerometer readings, stationary and motion are used as the input to the support vector machines (SVM). The sequence of stationary or moving state are viewed as an abstraction of the user's movement pattern. This fingerprint works as the second last stage of the multi-sensor filter. Putting all the stages together, SurroundSense can achieve an average room-level accuracy of 87%.

2.5 Summary

In this chapter, we reviewed different indoor localization solutions (wireless-based, sensor-based, and ambience-based solutions). One or two solutions picked from each indoor localization technology are listed in Table 2.2. Each solution has its advantages and disadvantages. The choice of localization solutions is mainly dependent on the requirement of the specific project and the available network infrastructure and mobile devices. However, if possible, employing hybrid schemes that take advantage of multiple solutions can help achieve better localization performance, which is a common choice for most commercial and research location-aware products. For example, combining WLAN together with cellular signals, Bluetooth, even ultra-wideband sources, if these are available. Wireless-based solution or sensor-based solution alone has the problem that the estimation error of several meters can

result in the misdetection of rooms which are separated by walls. However, if ambience data that can separate adjacent rooms logically is also collected, the pairs of adjacent rooms can be distinguished much easier (as [10] and [31] introduced).

Several hybrid localization systems are proposed. WILL [11] combines WiFi fingerprints with user movements detected by the accelerometer. The system in [7] combines WiFi fingerprints with the orientation of the phone obtained from gyroscope and magnetometer data. The system in [30] combines sensor data with limited WiFi fingerprints to achieve faster initial estimation. Indoor location startup WiFiSLAM (recently acquired by Apple) develops their localization solution based on the combination of many technologies. It uses pattern recognition and machine learning to draw correlations between data gathered by all of the sensors in a device, and combines with WiFi lateration (TDOA) and fingerprinting. It even uses image recognition together with studies of human psychological decision-making to predict the routes the users would take through a building [33]. From all of the discussions in this chapter, hybrid solutions with mobile devices sensing their surroundings in every way possible seem to be the best approach towards indoor localization.

Table 2.2 Indoor localization solutions

Solution	Technologies	Localization algorithms	Accuracy	Precision	Complexity	Dimension	Cost
[20]	Passive RFID	Fingerprinting (deterministic)	1m	93% within 1m	Low	2D	<$1/tag, $1000/reader, energy passive
LANDMARC [21]	Active RFID	Fingerprinting (deterministic)	<2m	50% within 1m	Medium	2D	$15/tag, $1000/reader
DecaWave[10]	UWB	TDOA	10cm	N/A	Medium	2D, 3D	$33/chip, high installation cost
Q-Track[11]	Active RFID	Fingerprinting, TDOA	40cm	N/A	Medium	2D	$0.5/ft^2 installation cost, power hungry
RADAR [8]	WLAN	Fingerprinting (deterministic), RSSI	3-5m	50% within 3m, 75% within 4.69m	High	2D, 3D	$0[12]
Horus [9]	WLAN	Fingerprinting (probabilistic)	2m	90% within 2.1m	Medium	2D	$0
WILL [11]	WLAN, Sensors	Fingerprinting (deterministic)	3m	86% within 3m	High	2D, 3D	$0
indoo.rs[13]	Bluetooth, WLAN	RSSI	1-2m	N/A	Medium	2D, 3D	$25/beacon[14], $9,995 for software SDK
[28]	Sensors	Dead Reckoning	1m	75% within 50cm, 95% within 73cm	High	2D, 3D	$0
IndoorAtlas[15]	Sensors	Fingerprinting	1-2m	90% within 3m	Medium	2D, 3D	N/A[16]
SurroundSense [10]	Ambience	Fingerprinting (deterministic)	3m	87% within 2m	Medium	2D, 3D	$0, high calibration cost

[10] see footnote 2
[11] http://q-track.com/your-q-track-solution/
[12] no extra cost other than the existing WLAN infrastructure, same for the other $0 cost solutions
[13] see footnote 6
[14] https://www.sticknfind.com/store.aspx
[15] see footnote 9
[16] currently in Beta, commercial plans coming soon

CHAPTER 3 WLAN Localization Implementation

This chapter describes the implementation of the WLAN indoor localization system. The system consists of both the server and database backend and the app frontend. The fingerprinting algorithm introduced in Chapter 2 is implemented to localize the user, which includes two phases, calibration phase and localization phase. Both the system design and the details of the localization algorithm will be presented.

3.1 System Design

The design of the WLAN localization system is shown in Figure 3.1. The two major parts are the server and database backend and the app frontend.

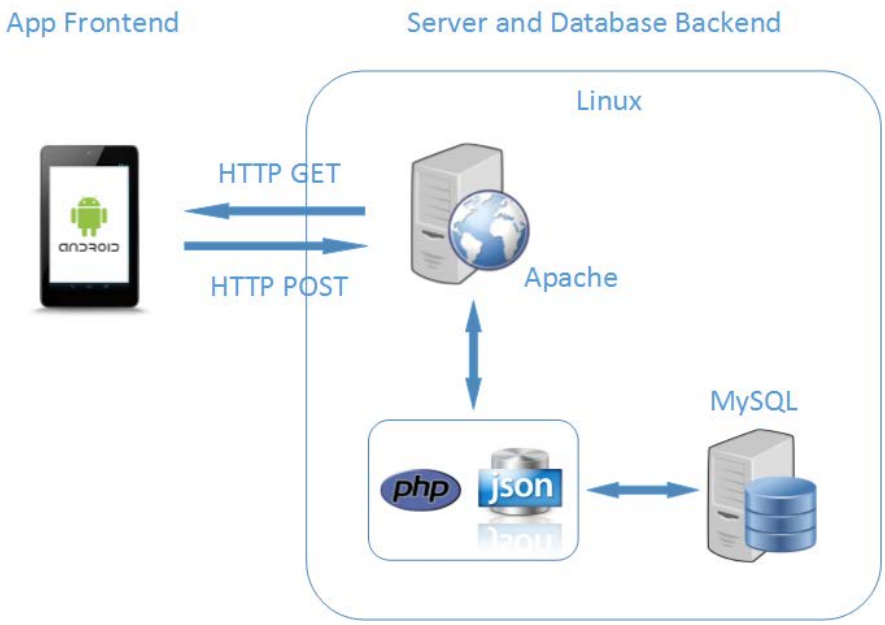

Figure 3.1 WLAN localization system design

3.1.1 Server and Database Backend

The server and database utilizes a LAMP stack[17] (Apache + MySQL + PHP), as shown in Figure 3.1. Both location information and user information are stored in the MySQL database. Location information includes the location “signatures” of the pre-recorded reference points in the smart space. User information includes the user's login information, frequent locations, preference, etc. Its functionality can also be extended according to the specific requirements of the AI algorithms. Since the MySQL database cannot communicate with the application directly, we use PHP as an interpreter between the database server and the application. JSON format is used to parse data from the PHP scripts to the application. Besides, the localization algorithm is implemented in PHP at the server backend as well. The application fetches the device’s location periodically from the server by using HTTP GET requests.

3.1.2 App Frontend

The application frontend is implemented on a Nexus 7 with Android 4.4. Android is chosen over iOS because the access to WiFi data is not supported in the public iOS API. The application utilizes Google Maps Android API v2. A Google Maps Android API v2 key is needed to add a map to the application. The key obtaining process is described in [34]. Google indoor maps are used for floor plans, which can decrease the error caused by doing Google Map 2D overlay manually. Every 4 seconds, the application collects a list of RSSI data from all of the access points it can sense, sends the data to the server using HTTP POST requests,

[17] inherited from Andrew Williams

gets the estimated location of the device back, and update the location on the map. The localization system introduced can be cross-platform because the location is estimated entirely on the server side, the app frontend can be implemented on Android, iOS, HTML or any other platform as long as it can communicate with the MySQL server using PHP.

3.2 Localization Algorithm

Fingerprinting algorithm is implemented, as introduced in Section 2.2.2.4. The implementation details of the two phases, calibration phase and localization phase, are explained below.

3.2.1 Calibration Phase

An Android application is implemented to collect the location “signatures” in the calibration phase. The screenshot of the application is shown in Figure 3.2.

The details of the application are listed below:

- The application utilizes Google Maps Android API v2.
- The floor plans can be switched by clicking the floor numbers on the right.
- The blue markers indicate the locations that already exist in the database, which can avoid duplicate measurements.
- The red markers indicate the new measurements taken. The user can go around the building and click at his/her location to take measurements (a list of signal strength from different APs). One red marker is dropped on each click.

- The measurements are saved to the Android device when the “Write to file” button on the bottom is pressed. The measurements are encoded into the MySQL format during the writing process, so that they can be stored in the database directly.
- The measurements are limited to only ncsu and ncsu-guest APs.

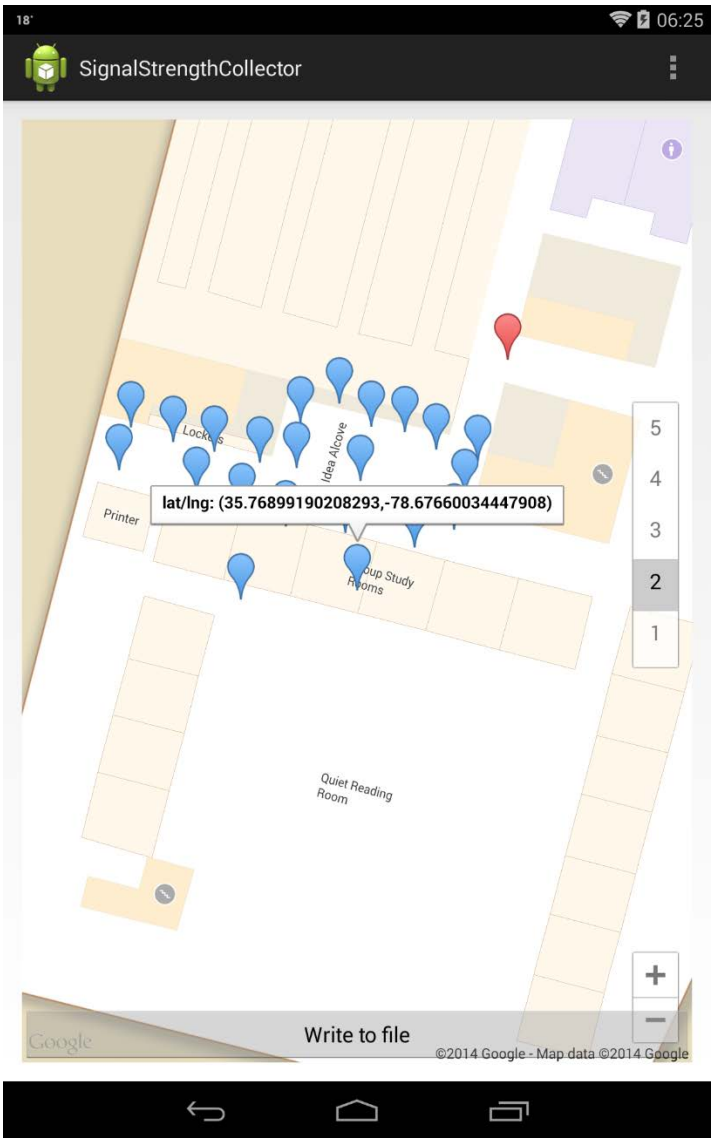

Figure 3.2 A screenshot of SignalStrengthCollector (2nd floor of Hunt Library, NCSU)

The location-related MySQL tables are shown in Figure 3.3. In the LOCATIONS table, LATITUDE and LONGITUDE are stored as varchar(20) in degree format. In the SIGNAL_STRENGTHS table, L_ID is the foreign key, and SIGNAL_STRENGTHS is a weak entity of LOCATIONS. Each L_ID can be associated with a list of S_IDs, because a list of signal strengths from different APs are collected from each location. The code snippet for setting up the database can be found in Appendix B.

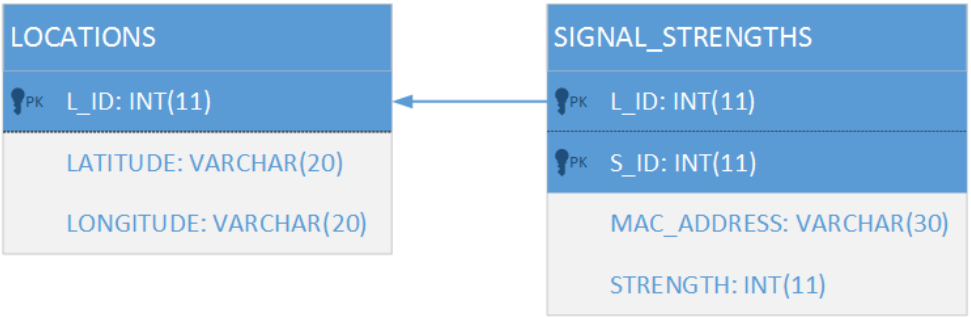

Figure 3.3 Database UML diagram

3.2.2 Localization Phase

The nearest neighbor deterministic algorithm is implemented for the localization phase. The algorithm attempts to find the the fingerprint F in the database that is the “closest” to the current measurement S (a list of <mac, strength> pairs). The flowchart of the algorithm is shown in Figure 3.4. The algorithm retrieves all database entries for each MAC address in the current measurement. For each MAC address, it goes through each database entry and makes a list of all the locations associated with the MAC address that have similar signal strengths (the difference is within a threshold i). If no similar signal strength is found, the threshold is

increased and the process is repeated until i reaches 100. It keeps a counter for each potential location, and the one with the highest count is the best-matched location.

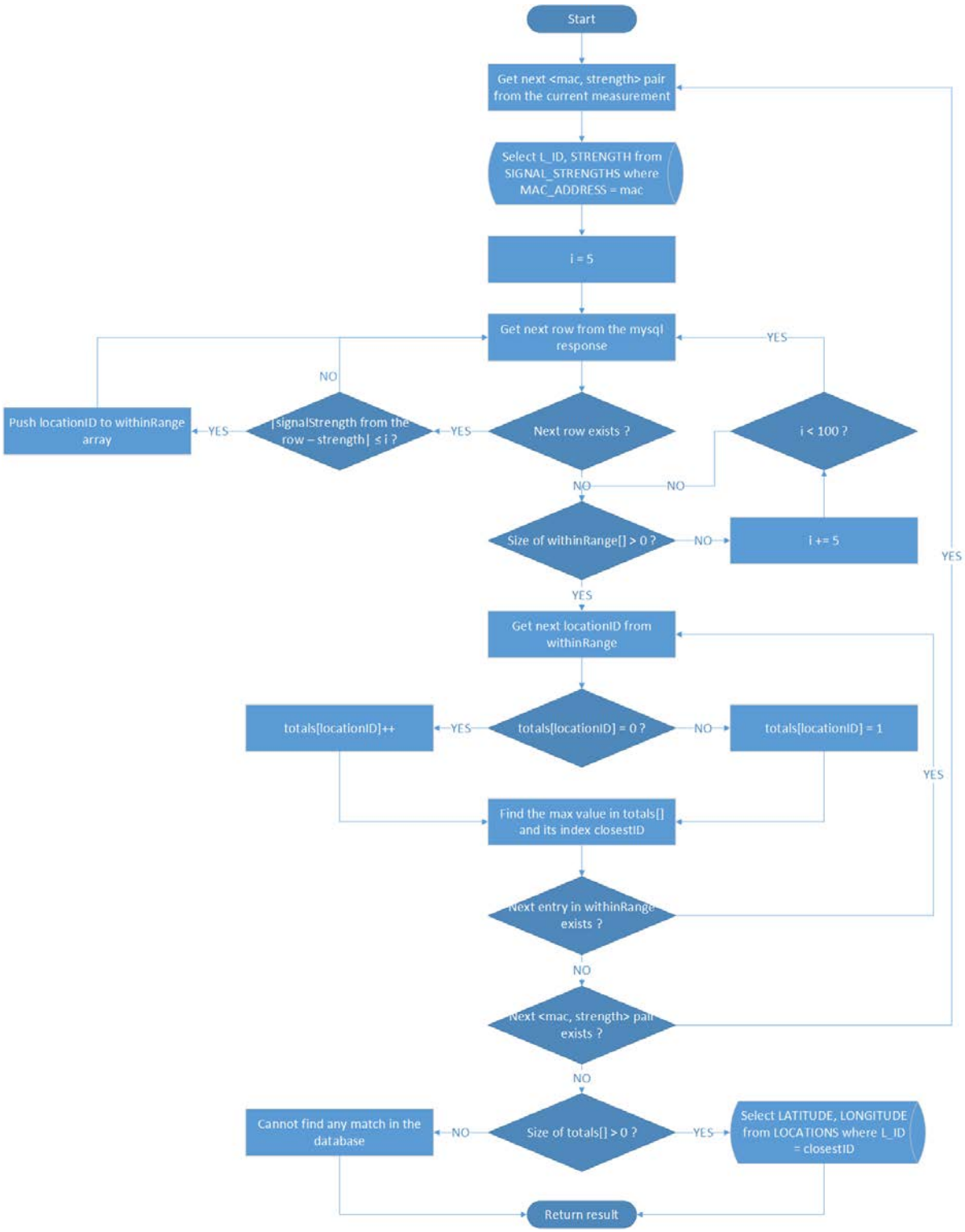

Figure 3.4 Flowchart of the localization algorithm

CHAPTER 4 Smart Space Simulation and Case Study

This chapter presents the simulation and modeling of a way finding case study for resource management in a smart space, where the space can recognize the users and interact with them under the Artificial Intelligence (AI) algorithms. The system allocates the resources to incoming users based on their physical locations. Different indoor localization accuracy as discussed in Chapter 2 can lead to different smart space performance introduced in Chapter 1. Here, we evaluate the dependency between the indoor localization performance and smart space performance, based on the simulation results. We furthermore take into account this dependency for optimizing smart space performance and user experience for cost minimization with various space size and user arrival rate.

4.1 Simulation Setup

The smart space layout used for simulations is shown in Figure 4.1. The space is modeled as an $M \times N$ grid. The size of each room is $5m \times 5m$. All the rooms have the same size. Each incoming user picks a room in the grid randomly. Arrival process is Poisson with parameter λ, i.e. Inter-arrival times are exponentially distributed with parameter λ. Figure 4.1 shows a grid of M = 4 and N = 5. The ith user A arrives at room (1, 1) with coordinates (x_i, y_i), and the jth user B arrives at room (2, 3) with coordinates (x_j, y_j).

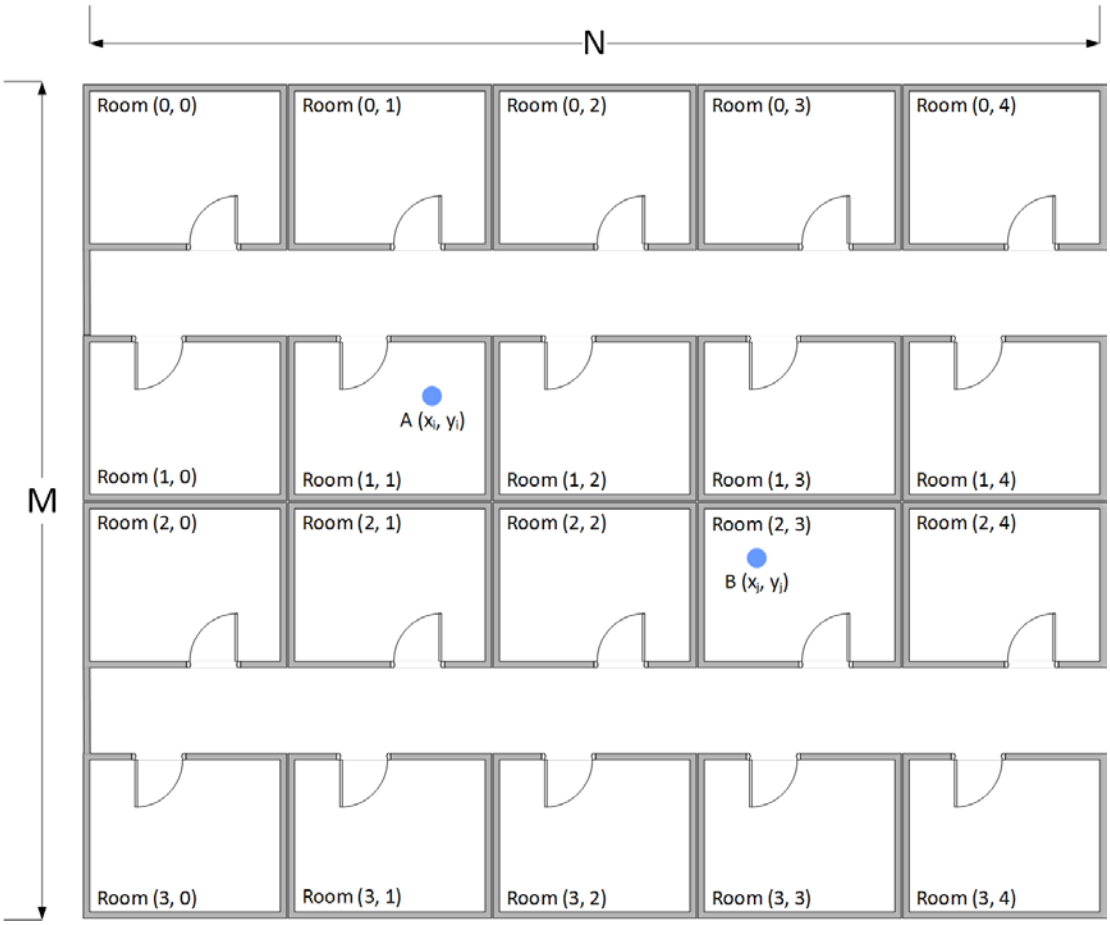

Figure 4.1 The grid layout of smart space simulation

4.1.1 Indoor Localization Error

Different indoor localization solutions have different performances. The performance of various indoor localization solutions are compared in Table 2.2 based on the localization performance metrics. Localization accuracy and precision are used to simulate the indoor localization error. As introduced in Section 1.3.3, accuracy is the average estimation error measured as the Euclidean distance between the actual position and the estimated position. Precision is the distribution of the estimation error, represented by the cumulative distribution

functions (CDF) of the estimation error. In the simulation, the location estimation error (ε_x, ε_y) is simulated using a zero-mean Gaussian distribution based on [12], which compares the error performance of an indoor localization system and a Gaussian model. The results indicate that the Gaussian model provides values on the same order as a real system.

Assume that the ith user (i ≥ 0) arrives at the space with coordinate (x_i, y_i), its estimated coordinate in the space should be ($x_i^{'}$, $y_i^{'}$), where

$$\begin{aligned} x_i^{'} &= x_i + \varepsilon_x^i \\ y_i^{'} &= y_i + \varepsilon_y^i \end{aligned} \tag{4.1}$$

With the effect of the indoor localization errors, the estimated coordinate ($x_i^{'}$, $y_i^{'}$) and the actual user coordinate (x_i, y_i) may belong to different rooms. As shown in Figure 4.2, a few meters' error (represented with a circle range) can put user A on the other side of the wall. It is called a mis-prediction of rooms, which can degrade the performance of the smart space in some scenarios. The degradation caused by room mis-predictions will be explained later in this chapter.

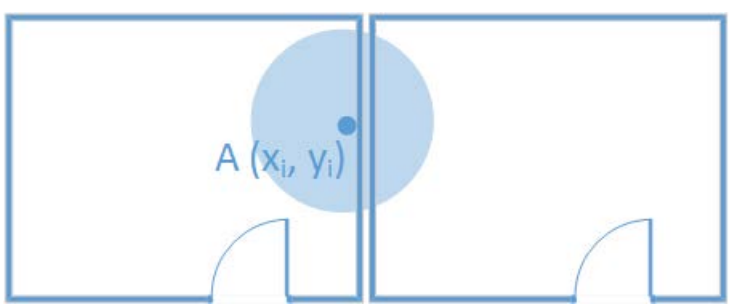

Figure 4.2 Indoor localization error range

4.1.2 Resource Management in Smart Space

In the simulation, there are certain amount of resources $c_{(i, j)}$ available in room (i, j). Each room is simulated as a M/G/c queueing system, where c is the number of the predefined resources in the room, i.e. $c_{(i, j)}$. The incoming user looks for an available resource in the room upon arrival, if all of the resources are allocated at the moment by other users with higher or equal priorities, the user won't be able to get the resource, and he/she will be dropped by the system. The service policy is FIFS (First-Input First-Served), and the system size can be up to $c_{(i, j)}$ (all the resources are occupied).

Resource management decisions in the rooms are made based on the lease-based method introduced in Section 1.2.2. The validity criteria used in the simulations are the user's proximity to the resource, and the maximum duration of the lease. Any violation of the validity criteria such as being outside of the resource proximity will cause the lease to be terminated. The space keeps monitoring all the users' estimated positions. When a user comes by a resource (e.g. a large display), the space notices that he/she is within the user proximity and initiates the leasing process for the resource (the large display) automatically without him/her doing anything. After the lease begins, the resource can perform certain operations based on the specific use case and the AI algorithms. It can be displaying a greeting message to the user on a display and letting the user take control of the display afterwards, or changing the background color on the display as the user walks by, or setting the room temperature and light brightness based on the user's pre-saved preferences. The key features of the resource management algorithm are listed below:

- Automatic service discovery.
- The system initiates the resource allocation process.
- Resources are allocated based on the users' *estimated* position.
- Double penalty with localization error.

  Because the localization error can put the user outside of the resource proximity which he/she should be eligible for, while close enough to a whole different set of resources, the user can actually take up other users' resources but still not being able to get what he/she wants at the same time. For example, two screens A and B have the same user proximity of 5 meters. John is standing 3 meters away from screen A and 10 meters away from screen B, Amy comes after John, and stands 2 meters away from screen B and 15 meters away from screen A. The space estimates that John is 8 meters away from screen A and 4 meters from screen B, and Amy is 3 meters away from screen B and 12 meters away from screen A. The localization error puts John within the proximity of screen B but outside the 5 meters' range from screen A. Since John is actually far away from screen B, he won't notice any changes on screen B that the space allocated for him, while at the same time screen A is not displaying any customized messages for him either. So John is considered a dropped user since he didn't get the resource he wanted. When Amy comes close to screen B, even though the system recognizes her within the proximity of screen B, the resource is not available because it's allocated to John. So Amy is considered dropped as well, which causes double penalty.

4.2 The Way Finding Case Study

Way finding is a basic functionality for the smart space. The way finding algorithm, as a sub-algorithm of the Artificial Intelligence (AI) algorithms that control the smart space operations, should guide the user from point A to point B by giving instructions using the displays, lights and other resources along the way.

4.2.1 Description of the Case Study

The specific work flow of this case study is listed as below:

1. The simulation takes place in the grid layout as shown in Figure 4.1. Each room in the grid has 5 digital displays for the user to use, i.e. each room can accommodate up to 5 users simultaneously at a time.
2. Each incoming user selects a start point (room A) and a destination point (room B) randomly in the grid upon arrival, a path (a list of rooms) is built based on the two rooms selected.
3. The smart space guides the user to go through each room on the path, stays at each room en route for one minute to watch a short video that involves using the digital display in the room.
4. When the user arrives at a room, the space tries to allocate the first available display in that room for him/her. If all of the 5 displays in the room are allocated at the moment by other users with higher or equal priorities, the user won’t get the needed display.

5. At any room along the path, if the user fails to get a display to watch the video, the way finding task is considered failed, the user is considered blocked.
6. If the space senses that the user is off the path, the space will try to allocate the display in the user's detected current room to inform him/her to get back onto the original path. If the display is allocated successfully, it will be leased to the user for 15 seconds to display the instructions about how to get back on path. The way finding task continues afterwards. If the resource allocation fails, the way finding task is terminated, the user is blocked.

4.2.2 Assumptions for the Simulations

Here are a couple of assumptions for the simulations:

- All the users are assumed to follow the instructions given one hundred percent. That is to say, the user can only be considered off-the-path because of the mis-prediction of rooms due to inaccurate indoor localization. After displaying the 15-second instructions after a mis-prediction, the user is assumed to be back on track.
- If the start position is estimated wrong by the space, the user can correct the estimated position by claiming his/her current room, so wrong start position is not considered in the simulations.
- All the users are assumed to have the same priorities in the grid.

4.2.3 Simulation Inputs

The simulation accepts 10,000 arrivals in total, in an $M \times N$ grid-building (Figure 4.1). Each room in the grid has 5 digital displays (resources). The area of each room is $5m \times 5m = 25m^2$. The various grid size is shown in Table 4.1.

Table 4.1 Various grid size used in the simulation

(M, N)	(5, 5)	(10, 10)	(30, 30)	(50, 50)
Area (m^2)	625	2,500	22,500	62,500

Arrival process is Poisson with parameter λ, which varies from 0.0167 to 0.2, as shown in Table 4.2. Constant service time of 60 seconds is used.

Table 4.2 Arrival rate vs. Average inter-arrival time for simulation

Arrival rate (λ)	0.017	0.022	0.033	0.067	0.2
Average inter-arrival time (sec)	60	45	30	15	5

Each arrival in each room is associated with a zero-mean (μ = 0) Gaussian distributed location estimation error (ε_x, ε_y). The standard deviation σ of the Gaussian distribution are shown in Table 4.3. The various localization precision is selected based on the different precision level of the current localization solutions listed in Table 2.2. For example, a standard deviation of 0.25 gives a location estimation error of 0.5m for 95% of the time.

Table 4.3 Gaussian-distributed location estimation

Localization precision (95% within)	0.5m	1m	2m	3m	4m
Standard deviation (σ)	0.25	0.5	1	1.5	2

4.2.4 Performance Metrics

Three performance metrics are measured during the simulations, they are listed as below:

Room mis-prediction rate

As explained in Section 4.1.1, room mis-predictions occur when the estimated location and the actual location are in different rooms. Room mis-prediction rate is calculated as the ratio of number of mis-predictions to the total number of location prediction/estimation made. Since mis-predictions are caused entirely by the inaccurate location estimations, the mis-prediction rate and the system arrival rate are independent.

Average time spent per room

Without location estimation error, the average time each user spends in each room along the path should be exactly 60 seconds because of the constant service time. However, if a room mis-prediction occurs, the space will consider the user as off-the-path and occupy an extra resource for 15 seconds before the user can get back on track. The 15 seconds add a delay to

the total time the user needs to finish the task. Average time spent per room T_{avg} is calculated as

$$T_{avg} = \frac{\sum_{i=1}^{10000} \frac{\text{total time user i used to finish the task}}{\text{number of rooms on path i}}}{10000} \tag{4.2}$$

Similar as room mis-prediction rate, the value of the average time spent per room doesn't change with different system arrival rate, because they are independent.

System blocking rate

With the assumptions listed in Section 4.2.2, a user can only be blocked due to limited resources in the simulations. The system blocking rate P_b is calculated as

$$P_b = \frac{\text{number of blocked users}}{10000} \tag{4.3}$$

4.3 Simulation Results and Smart Space Performance Analysis

Simulations are run with various arrival rates, indoor localization precisions, and grid sizes, as indicated in Table 4.1, 4.2 and 4.3. Three performance metrics introduced in Section 4.2.4 are measured, the results are shown in the following sections.

4.3.1 Room Mis-prediction Rate

Table 4.4 shows the room mis-prediction rate results. Note that the blocked users are not counted in the calculation, only the active users are considered. σ is the standard deviation of the Gaussian distribution used to model the indoor localization precision.

Table 4.4 Room mis-prediction rate
(different localization precision and grid size)

Grid size (M, N)	Standard deviation (σ)				
	0.25	0.5	1	1.5	2
(5, 5)	6.65%	12.90%	25.18%	36.56%	47.19%
(10, 10)	7.37%	14.31%	27.75%	39.90%	50.94%
(30, 30)	7.71%	15.09%	28.89%	41.49%	52.76%
(50, 50)	7.75%	15.16%	29.11%	41.78%	53.10%

The simulation results from Table 4.4 are plotted in Figure 4.3. For a given grid size, the room mis-prediction rate increases as the standard deviation for the Gaussian distribution increases. Since the larger the standard deviation is, the less accurate the indoor localization is (Table 4.3), it makes sense that a larger standard deviation leads to a higher mis-prediction rate. For a given standard deviation for the Gaussian distribution, the room mis-prediction rate increases as the grid size increases. When the standard deviation is 2 meters, the mis-prediction rate can reach around 50%. Four curves that represent different grid sizes show similar trends in the figure.

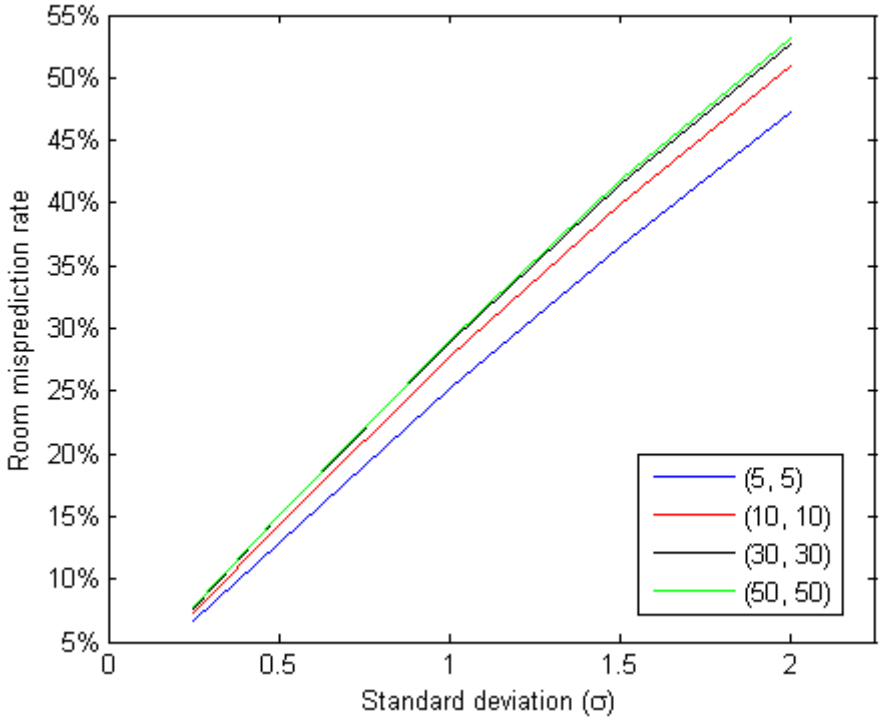

Figure 4.3 Room mis-prediction rate vs. Standard deviation and grid size

4.3.2 Average Time Spent Per Room

The average time spent per room results are shown in Table 4.5. Similar with the room mis-prediction rate results, the blocked users are not counted in the calculation, only the active users are considered.

The simulation results from Table 4.5 are plotted in Figure 4.4. For a given grid size, the average time spent per room increases as the standard deviation for the Gaussian distribution increases, which coincides with the results shown in Figure 4.3. A larger room mis-prediction rate gives a longer delay to the task due to the extra 15 seconds to display the instructions to

get the user back on track. For a given standard deviation for the Gaussian distribution, the average time spent per room increases as the grid size increases. Four curves in the figure show similar trends.

Table 4.5 Average time spent per room
(different localization precision and grid size)

Grid size (M, N)	Standard deviation (σ)				
	0.25	0.5	1	1.5	2
(5, 5)	61.064	62.224	65.032	68.624	73.318
(10, 10)	61.194	62.526	65.786	69.994	75.618
(30, 30)	61.256	62.67	66.118	70.658	76.812
(50, 50)	61.26	62.684	66.172	70.79	77.026

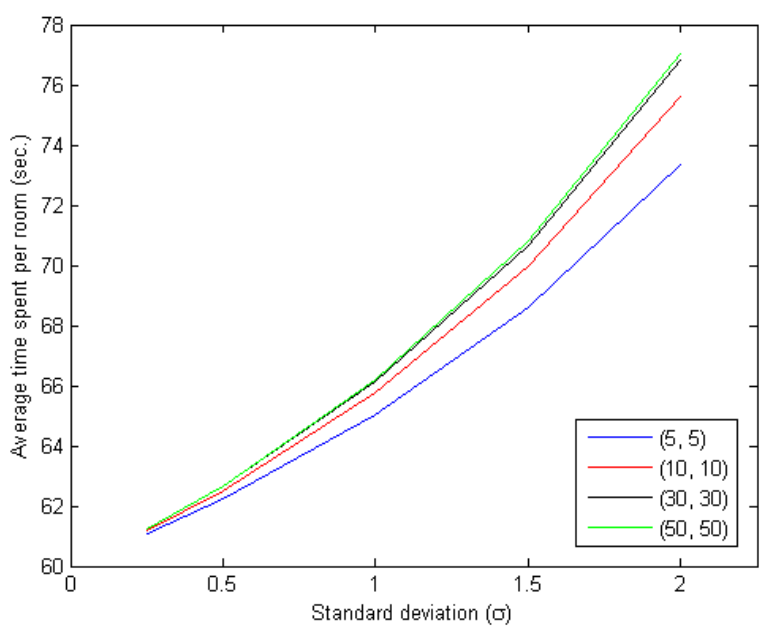

Figure 4.4 Average time spent per room vs. Standard deviation and grid size

4.3.3 System Blocking Rate

The system blocking rate results with different grid sizes, arrival rates, and different indoor localization precisions are shown in Table 4.6 to Table 4.9.

Table 4.6 System blocking rate (M = 5, N = 5)

	Arrival rate (λ)				
	0.0167	0.022	0.033	0.067	0.2
No localization error	0.00%	0.01%	0.00%	0.40%	16.34%
Standard deviation (σ)					
0.25	0.00%	0.01%	0.02%	0.43%	16.93%
0.5	0.00%	0.00%	0.04%	0.70%	18.15%
1	0.00%	0.00%	0.06%	0.96%	22.12%
1.5	0.00%	0.00%	0.12%	1.25%	25.54%
2	0.01%	0.02%	0.10%	1.84%	30.31%

Table 4.7 System blocking rate (M = 10, N = 10)

	Arrival rate (λ)				
	0.0167	0.022	0.033	0.067	0.2
No localization error	0.00%	0.00%	0.00%	0.07%	2.40%
Standard deviation (σ)					
0.25	0.00%	0.00%	0.00%	0.04%	2.91%
0.5	0.00%	0.00%	0.00%	0.07%	3.08%
1	0.00%	0.00%	0.00%	0.07%	4.04%
1.5	0.00%	0.00%	0.00%	0.08%	5.48%
2	0.00%	0.00%	0.00%	0.12%	8.71%

Table 4.8 System blocking rate (M = 30, N = 30)

	Arrival rate (λ)				
	0.0167	0.022	0.033	0.067	0.2
No localization error	0.00%	0.00%	0.00%	0.00%	0.07%
Standard deviation (σ)					
0.25	0.00%	0.00%	0.00%	0.00%	0.07%
0.5	0.00%	0.00%	0.00%	0.00%	0.07%
1	0.00%	0.00%	0.00%	0.00%	0.13%
1.5	0.00%	0.00%	0.00%	0.00%	0.12%
2	0.00%	0.00%	0.00%	0.01%	0.32%

Table 4.9 System blocking rate (M = 50, N = 50)

	Arrival rate (λ)				
	0.0167	0.022	0.033	0.067	0.2
No localization error	0.00%	0.00%	0.00%	0.00%	0.03%
Standard deviation (σ)					
0.25	0.00%	0.00%	0.00%	0.00%	0.03%
0.5	0.00%	0.00%	0.00%	0.00%	0.03%
1	0.00%	0.00%	0.00%	0.00%	0.01%
1.5	0.00%	0.00%	0.00%	0.00%	0.02%
2	0.00%	0.00%	0.00%	0.00%	0.04%

The simulation results from Table 4.6 to Table 4.9 are plotted in Figure 4.5. In Figure 4.5 (a) to (d), each figure contains six separate curves, each one for a different standard deviation for the Gaussian distribution. The magenta-color curve represents the one without any localization

error. From the figures, we can see that the “no error” curve has the lowest system blocking rate as expected, because location estimation error degrades the system performance and user experience. For a given standard deviation for the Gaussian distribution, the system blocking rate increases as the arrival rate increases, because a higher arrival rate leads to more resource conflicts in the space, and more users are blocked due to the limited number of resources. For a given arrival rate, the system blocking rate increases as the standard deviation for the Gaussian distribution increases, so the larger standard deviation leads to a worse system performance. Besides, the system blocking rate differences among different curves becomes larger as the arrival rate approaches 0.2, so the effect of localization error increases as the arrival rate increases.

Comparing the four subplots in Figure 4.5, we can see that the larger the grid size is, the lower the system blocking rate is. For the 50×50 grid (Figure 4.5 (d)), the system blocking rates are almost 0 even for the $\sigma = 2$ curve. From Figure 4.3, we know that larger grid size gives a higher room mis-prediction rate, which raises an interesting question of why a higher room mis-prediction rate leads to a lower system blocking rate. The reason is that the way finding case study chooses a random start point and a random destination point for each user. If the grid has more rooms, it’s less possible for users to have the same rooms on their paths. Therefore, even though the larger grid size suffers from a higher room mis-prediction rate, the user can manage to get the resources, with some extra time (15 second delay) to display extra instructions in the wrong rooms.

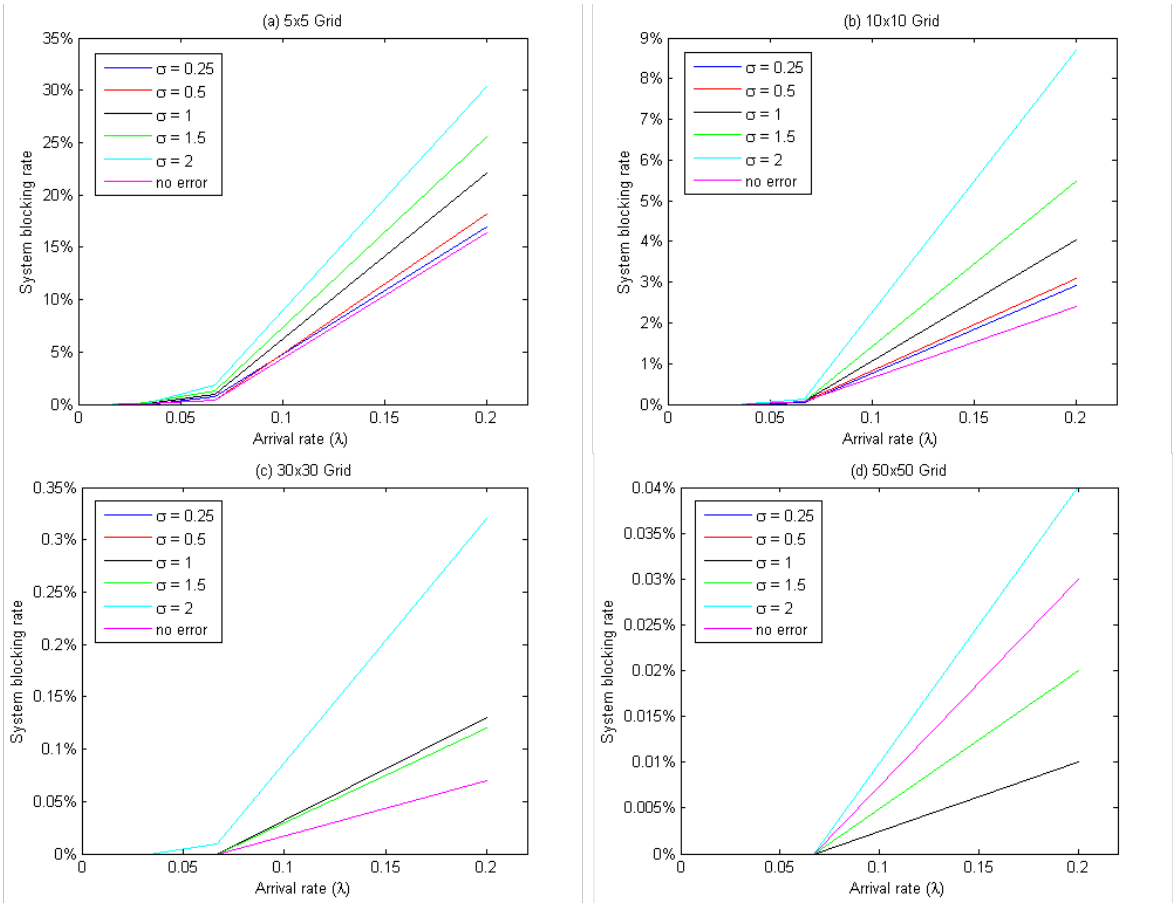

Figure 4.5 System blocking rate vs. Standard deviation and arrival rate in (a) 5×5 grid, (b) 10×10 grid, (c) 30×30 grid, (d) 50×50 grid

4.4 Cost Optimization

Given that different indoor localization precisions can lead to different smart space performance (in terms of system blocking rate) with different inputs of grid size (M, N) and arrival rate λ, our goal is to find the indoor localization solution that can minimize the total cost of the localization system, while keeping an acceptable smart space performance. Hence, the cost optimization problem can be stated as follows,

$$\min_{(M,N),\lambda} Cost((M,N),\lambda) \quad \text{subject to } P_b \le \rho \tag{4.4}$$

where P_b is the system blocking rate and is the maximum blocking rate needed for the system to function and maintain certain quality of service.

The grid size and the arrival rate can be any value shown in Figure 4.5. The indoor localization solutions are the ones listed in Table 4.10, their costs are estimated based on the system parameters listed in Table 4.11 and the unit costs listed in Table 2.2. RFID-based and UWB-based solutions need both tags and readers for the system to work. The number of tags should be the number of users in the space, and the number of the readers are listed in Table 4.10. We assume that WLAN-based solutions have a hardware cost of $0 since the existing WLAN infrastructure is used.

Table 4.10 Indoor localization solutions and their precisions[18]

Standard deviation for Gaussian distribution (σ)	Localization precision (95% within)	Indoor localization solutions
0.25	0.5m	DecaWave (UWB)
0.5	1m	Q-Track (active RFID) [20] (passive RFID)
1	2m	indoo.rs (Bluetooth, WLAN)
1.5	3m	Horus (WLAN) IndoorAtlas (sensors) SurroundSense (ambience)
2	4m	RADAR (WLAN)

[18] refer to Table 2.2 for the performance of specific indoor localization solutions

From Figure 4.5, we can see that different grid sizes can have significantly different smart space performance in terms of the system blocking rate. We divide the different grid sizes into two subgroups. One is for the small-size smart spaces, which includes the 5×5 grid and the 10×10 grid. The other one is for the large-size smart spaces, which includes the 30×30 grid and 50×50 grid.

Table 4.11 Parameters of indoor localization solutions

Grid size (M, N)	(5, 5)	(10, 10)	(30, 30)	(50, 50)
# of users	100	500	1000	3000
# of passive RFID readers	25	100	900	2500
# of beacons/readers (UWB, active RFID)	4	16	144	400

4.4.1 Small-Size Smart Spaces

5×5 Grid

Table 4.12 shows the cost table of different indoor localization solutions in a 5×5 grid. The cost optimization results are shown in Table 4.13. Only the maximum arrival rate λ_{max}, i.e. the heaviest system load is considered in the optimization problem. When ρ = 0.5%, we can see that $\lambda \leq 0.067$ and $\sigma \leq 0.25$ are needed based on the results in Table 4.6. From Table 4.10, we can derive that only DecaWave can meet these criteria, so the cost will be $8000. When ρ = 1%, the criteria change to $\lambda \leq 0.067$ and $\sigma \leq 1$, which means that DecaWave, Q-Track, passive

RFID and indoo.rs are all eligible. Q-Track has the lowest cost of $7500 among the four solutions. When ρ = 2%, all of the solutions listed in Table 4.10 can achieve the requested system blocking rate when $\lambda \leq 0.067$. Among the four zero-cost solutions, IndoorAtlas becomes the optimal one because of its 2D/3D location estimation capability and low software complexity (Table 2.2). Note that the 5×5 grid will suffer from more than 16% blocking rate even when there is no localization error in the space if $\lambda \geq 0.2$.

Table 4.12 Costs for indoor localization solutions (M = 5, N = 5)

Indoor localization solutions	Hardware cost	Software cost	Total cost
DecaWave	$8,000	N/A	$8,000
Q-Track	$7,500	N/A	$7,500
Passive RFID [20]	$25,100	N/A	$25,100
indoo.rs	$100	$9,995	$10,095
Horus, IndoorAtlas, SurroundSense, RADAR	$0	N/A	$0

Table 4.13 Results for cost optimization (M = 5, N = 5)

Maximum blocking rate (ρ)	Maximum arrival rate (λ_{max})	Maximum standard deviation (σ_{max})	Eligible indoor localization solutions	Best indoor localization solution
0.5%	0.067	0.25	DecaWave	DecaWave
1%	0.067	1	DecaWave, Q-Track, [20], indoo.rs	Q-Track
2%	0.067	2	ALL	IndoorAtlas

10×10 Grid

Table 4.14 shows the cost table of different indoor localization solutions in a 10×10 grid. The cost optimization results are shown in Table 4.15. $\rho = 1\%$ can only be achieved when $\lambda \leq 0.067$, the maximum standard deviation $\sigma_{max} = 1.5$. The zero-cost solutions are the optimal ones and IndoorAtlas is the best among them. When $\rho = 3\%$ or 5% and $\lambda = 0.067$, all of the solutions are eligible, similarly IndoorAtlas is the optimal solution. When $\rho = 3\%$ and $\lambda = 0.2$, the corresponding maximum standard deviation is 0.25, DecaWave is the only solution and the cost is $36000. $\rho = 5\%$ and $\lambda = 0.2$ gives $\sigma \leq 1$ from Table 4.7, among the eligible solutions, DecaWave, Q-Track, passive RFID and indoo.rs, indoo.rs has the lowest cost of $10,395.

Table 4.14 Costs for indoor localization solutions (M = 10, N = 10)

Indoor localization solutions	Hardware cost	Software cost	Total cost
DecaWave	$36,000	N/A	$36,000
Q-Track	$30,000	N/A	$30,000
Passive RFID [20]	$100,500	N/A	$100,500
indoo.rs	$400	$9,995	$10,395
Horus, IndoorAtlas, SurroundSense, RADAR	$0	N/A	$0

Table 4.15 Results for cost optimization (M = 10, N = 10)

Maximum blocking rate (ρ)	Maximum arrival rate (λ_{max})	Maximum standard deviation (σ_{max})	Eligible indoor localization solutions	Best indoor localization solution
1%	0.067	1.5	ALL but RADAR	IndoorAtlas
3%	0.067	2	ALL	IndoorAtlas
3%	0.2	0.25	DecaWave	DecaWave
5%	0.067	2	ALL	IndoorAtlas
5%	0.2	1	DecaWave, Q-Track, [20], indoo.rs	indoo.rs

4.4.2 Large-Size Smart Spaces

30×30 Grid

The cost table of different indoor localization solutions in a 30×30 grid is shown in Table 4.16.

Table 4.16 Costs for indoor localization solutions (M = 30, N = 30)

Indoor localization solutions	Hardware cost	Software cost	Total cost
DecaWave	$184,000	N/A	$184,000
Q-Track	$216,000	N/A	$216,000
Passive RFID [20]	$901,000	N/A	$901,000
indoo.rs	$3,600	$9,995	$13,595
Horus, IndoorAtlas, SurroundSense, RADAR	$0	N/A	$0

50×50 Grid

The cost table of different indoor localization solutions in a 50×50 grid is shown in Table 4.17. From the results in Table 4.8 and Table 4.9, we can see that in large-size smart spaces, the system blocking rate is less than 0.4% when $\lambda \leq 0.2$. For the 50×50 grid (Figure 4.5 (d)), the system blocking rates are almost 0 even for the $\sigma = 2$ curve. As shown in Table 4.16 and Table 4.17, the cost of the UWB-based and RFID-based indoor localization solutions can get very high due to their poor scalability, while the performance of these solutions are not much better than the zero-cost ones. Therefore, in large-size smart spaces, the zero-cost indoor localization solutions can provide acceptable QoS and keep the cost minimized at the same time. Zero-cost solutions are suitable for large size smart spaces.

Table 4.17 Costs for indoor localization solutions (M = 50, N = 50)

Indoor localization solutions	Hardware cost	Software cost	Total cost
DecaWave	$520,000	N/A	$520,000
Q-Track	$750,000	N/A	$750,000
Passive RFID [20]	$2,503,000	N/A	$2,503,000
indoo.rs	$10,000	$9,995	$19,995
Horus, IndoorAtlas, SurroundSense, RADAR	$0	N/A	$0

4.5 Summary

In this chapter, we proposed a simulation platform for the way finding case study for resource management in smart spaces. We analyzed the dependency between the indoor localization precision and the smart space performance including the room mis-prediction rate, the average time spent per room, and the system blocking rate. Our results show that a less accurate indoor localization solution gives a higher room mis-prediction rate, a longer average time spent in each room, and a higher system blocking rate. Even though a larger smart space size leads to a higher room mis-prediction rate and a longer average time spent per room, it gives a lower system blocking rate. Finally, we solved the cost optimization problems by deciding the least-cost indoor localization solution under different smart space sizes and system loads that can achieve various maximum system blocking rate allowed. The optimization results can be used for budget estimation in smart spaces.

CHAPTER 5 Conclusion

The intelligent and interactive technologies for smart spaces present great potentials to improve user experience indoors substantially, but also new challenges because the automated, digital-driven systems rely on the sensing inputs from the space. In reality, the sensing inputs cannot be 100% accurate. The faulty estimate of the user's indoor location can lead to suboptimal resource allocation and system performance. The presence of multiple users leads to conflicts regarding the resources in the space, let alone the effect from the inaccurate sensing inputs, which will even worsen the situation by assigning the limited resource to the wrong user. Hence getting a comprehensive understanding of the indoor localization solutions and finding out how faulty location estimation affects the performance of the smart space and the user experience are necessary on deciding which indoor localization solution is the optimal one given a limited budget and limited number of resources available.

We reviewed different indoor localization solutions, including the wireless-based solutions, the sensor-based solutions, and the ambience-based solutions. Wireless-based solutions utilize either proximity, triangulation, or fingerprinting algorithms. Different wireless technologies used for indoor localization like UWB, RFID, WLAN, Bluetooth, etc. are also discussed. Different solutions have different advantages and disadvantages. The choice of localization solutions is mainly dependent on the requirement of the specific project and the available network infrastructure and mobile devices. Employing hybrid schemes that take advantage of multiple solutions can help achieve better localization performance. A summary of the

performance comparison of different localization solutions with regard to their accuracy, precision, complexity, dimension, and cost is conducted at the end of Chapter 2.

Following the detailed survey of indoor localization solutions, we present the implementation of the fingerprinting-based WLAN indoor localization system. The calibration phase uses an Android application to collect fingerprints in the space. The localization phase algorithm is implemented on the server side, which gives flexibility to the choice of app frontend platforms. Current frontend demo is implemented on Android.

After the descriptions of the localization system implementation, we propose a simulation platform for the way finding case study for resource management in smart spaces. We analyze the dependency between the indoor localization precision and the smart space performance including the room mis-prediction rate, the average time spent per room, and the system blocking rate. Our results show that a less accurate indoor localization solution gives a higher room mis-prediction rate, a longer average time spent in each room, and a higher system blocking rate. Even though a larger smart space size leads to a higher room mis-prediction rate and a longer average time spent per room, it gives a lower system blocking rate. Finally, we solve the cost optimization problems by deciding the least-cost indoor localization solution under different smart space sizes and system loads that can achieve various maximum system blocking rate allowed. The optimization results can be used for budget estimation in smart spaces with different sizes and system loads.

REFERENCES

[1] Aarts, E., & Marzano, S. (2003). The new everyday: views on ambient intelligence. Rotterdam, Netherlands: 010 Publishers.

[2] Moussa, S., & Agha, G. (2009). A multi-agent simulator for resource management in smart spaces. *Pervasive Computing (JCPC), 2009 Joint Conferences on , vol., no.*, pp.765-770.

[3] Cook, D., & Das, S. (2004). Smart environments: technology, protocols and applications. Hoboken, NJ: John Wiley & Sons, Inc., ISBN 0-471-54448-5.

[4] Jurmu, M., Perttunen, M., & Riekki, J. (2007). Lease-based resource management in smart spaces. *Pervasive Computing and Communications Workshops '07. Fifth Annual IEEE International Conference on , vol., no.*, pp.622-626.

[5] Chen, L., Xu, Y., Fang, W., & Gao, C. (2008). A new ElGamal-based algebraic homomorphism and its application. *Computing, Communication, Control, and Management, 2008. CCCM '08. ISECS International Colloquium on* , vol.1, no., pp.643-648.

[6] Liu, H., Darabi, H., Banerjee, P., & Liu, J. (2007). Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* , vol.37, no.6, pp.1067-1080.

[7] Martin, E., Vinyals, O., Friedland, G., & Bajcsy, R. (2010). Precise indoor localization using smart phones. In *Proceedings of the international conference on Multimedia* (MM '10). ACM, New York, NY, USA, 787-790.

[8] Bahl, P., & Padmanabhan, V.N. (2000). RADAR: An in-building RF-based user location and tracking system. *Proc. IEEE INFOCOM*, vol. 2, pp. 775-784.

[9] Youssef, M., & Agrawala, A. (2005). The Horus WLAN location determination system. *Proc. Third Int'l Conf. Mobile Systems, Applications, and Services*, pp. 205-218.

[10] Azizyan, M., Constandache, I., & Choudhury, R.R. (2009). Surround-Sense: mobile phone localization via ambience fingerprinting. *Proc. 15th Ann. Int'l Conf. Mobile Computing and Networking*, pp. 261-272.

[11] Wu, C., Yang, Z., Liu, Y., & Xi, W. (2013). WILL: wireless indoor localization without site survey. *IEEE Transactions on Parallel and Distributed Systems*, vol.24, no.4.

[12] Kaemarungsi, K., & Krishnamurthy, P. (2004). Properties of indoor received signal strength for WLAN location fingerprinting. *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on*, vol., no., pp. 14-23.

[13] Kanaan, M., & Pahlavan, K. (2004). A comparison of wireless geolocation algorithms in the indoor environment. *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE* , vol.1, no., pp. 177-182.

[14] Torrieri, D.J. (1984). Statistical theory of passive location systems. *Aerospace and Electronic Systems, IEEE Transactions on* , vol.AES-20, no.2, pp. 183-198.

[15] Li, X., Pahlavan, K., Latva-aho, M., & Ylianttila, M. (2000). Comparison of indoor geolocation methods in DSSS and OFDM wireless LAN systems. *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd* , vol.6, no., pp. 3015-3020.

[16] Madigan, D., Einahrawy, E., Martin, R.P., W.H. Ju, W.H., Krishnan, P., & Krishnakumar, A.S. (2005). Bayesian indoor positioning systems. *Proc. IEEE CS INFOCOM*, vol. 2, pp. 1217-1227.

[17] Wi-Fi location-based services 4.1 design guide. (2008). San Jose, CA: Cisco System, Inc.

[18] Patmanathan, V. (2006) *Area Localization using WLAN* (Master of Science thesis). Royal Institute of Technology, Stockholm, Sweden.

[19] Prasithsangaree, P., Krishnamurthy, P., & Chrysanthis, P.K. (2002). On indoor position location with wireless LANs. *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on* , vol.2, no., pp.720-724.

[20] Ting, S.L., Kwok, S.K., Tsang, Albert H.C, & Ho, George T.S. (2011). The study on using passive RFID tags for indoor positioning. *International Journal of Engineering Business Management*, vol.3, no.1, pp. 9-15.

[21] Ni, L.M., Liu, Y., Lau, Y.C., & Patil, A.P. (2003). LANDMARC: indoor location sensing using active RFID. *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on* , vol., no., pp.407-415.

[22] Schneider, D. (2013). You are here. *Spectrum, IEEE* , vol.50, no.12, pp.34-39.

[23] Chintalapudi, K., Padmanabha Iyer, A., & Padmanabhan, V.N. (2010). Indoor localization without the pain. *Proc. 16th Ann. Int'l Conf. Mobile Computing and Networking*, pp. 173-184.

[24] Medina, C., Segura, J.C., & Holm, S. (2012). Feasibility of ultrasound positioning based on signal strength. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN '12)*, pp. 1–9.

[25] Hu, X., Cheng, L., & Zhang, G. (2011). A Zigbee-based localization algorithm for indoor environments. *Proceedings of the International Conference on Computer Science and Network Technology (ICCSNT '11)*, pp. 1776–1781.

[26] Android sensor coordinate system. Retrieved April 21, 2014 from Sensors Overview, Android Developers: http://developer.android.com/guide/topics/sensors/sensors_overview.html#sensors-coords

[27] Titterton, D.H., & Weston, J.L. (2004). Strapdown inertial navigation technology, 2nd ed. Herts, United Kingdom: The Institution of Engineering and Technology.

[28] Woodman, O., & Harle, R. (2008). Pedestrian localisation for indoor environments. *Proc. 10th Int'l Conf. Ubiquitous Computing*, pp. 114-123.

[29] Link, J.A.B., Smith, P., Viol, N., & Wehrle, K. (2011). FootPath: Accurate map-based indoor navigation using smartphones. *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on* , vol., no., pp. 1-8.

[30] Liu, Y., Dashti, M., & Zhang, J. (2013). Indoor localization on mobile phone platforms using embedded inertial sensors. *Positioning Navigation and Communication (WPNC), 2013 10th Workshop*, pp. 1-5.

[31] Tarzia, S.P., Dinda, P.A., Dick, R.P., & Memik, G. (2011). Indoor localization without infrastructure using the acoustic background spectrum. *MobiSys'11*.

[32] Quick, D. (2012, June 27). UnLoc app uses "invisible" landmarks for precise indoor localization. *gizmag*. Retrieved from http://www.gizmag.com/unloc-indoor-navigation-app/23106/

[33] Panzarino, M. (2013, March 26). What exactly WiFiSLAM is, and why Apple acquired it. TNW. Retrieved from http://thenextweb.com/apple/2013/03/26/what-exactly-wifislam-is-and-why-apple-acquired-it/

[34] Obtain a Google Maps API key. Retrieved May 6, 2014 from Google Maps Android API v2, Android Developers: https://developers.google.com/maps/documentation/android/start#obtain_a_google_maps_api_key

APPENDICES

Appendix A

Code Snippet for Collecting Sensor Data on Android

```
// Data structures to hold the sensor data
ArrayList<Long> time = new ArrayList<Long>();
ArrayList<Float> x_value = new ArrayList<Float>();
ArrayList<Float> y_value = new ArrayList<Float>();
ArrayList<Float> z_value = new ArrayList<Float>();

public void onSensorChanged(SensorEvent arg0) {
	if (arg0.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
		getAccelerometer(arg0);
	}
}

private void getAccelerometer(SensorEvent event) {
	float[] values = event.values;
	// Movement
	float x = values[0];
	float y = values[1];
	float z = values[2];

	long actualTime = System.currentTimeMillis();

	// Take measurements every 0.2 sec
	if (actualTime - lastUpdate < 200) {
		return;
	}

	time.add(actualTime - startTime);
	z_value.add(z);
	x_value.add(x);
	y_value.add(y);

	Log.d("sensor", "Time: " + (actualTime - startTime) + "; Reading: " + z);
	lastUpdate = actualTime;
}
```

Appendix B

Code Snippet for Setting up the Database

```
// Create LOCATIONS table
$sql = "CREATE TABLE LOCATIONS (L_ID INT(11) NOT NULL AUTO_INCREMENT,
                               LATITUDE VARCHAR(20) NOT NULL,
                               LONGITUDE VARCHAR(20) NOT NULL,
                               PRIMARY KEY (L_ID)) ENGINE=MyISAM";

mysql_query($sql) || die("Error creating table: " . mysql_error() . "");
echo "Created table 'LOCATIONS'<br />";

// Create SIGNAL_STRENGTHS table
$sql = "CREATE TABLE SIGNAL_STRENGTHS (S_ID INT(11) NOT NULL AUTO_INCREMENT,
                                       MAC_ADDRRESS VARCHAR(30) NOT NULL,
                                       STRENGTH INT(11) NOT NULL,
                                       L_ID INT(11) NOT NULL,
                                       FOREIGN KEY (L_ID) REFERENCES LOCATIONS ON
                                   DELETE CASCADE,
                                       PRIMARY KEY (S_ID, L_ID)) ENGINE=MyISAM";

mysql_query($sql) || die("Error creating table: " . mysql_error() . "");
echo "Created table 'SIGNAL_STRENGTHS'<br />";

// Add in the signal strength measurements
$fileName = "data_files/measurements.sql";
$handle = fopen($fileName, "rb");
if($handle == FALSE){
	//Couldn't open the file
	echo "Error, couldn't open file $fileName<br />";
}
else{
	while($data = fgets($handle)){
		$sql = chop($data);
		mysql_query($sql) || die(mysql_error());
		echo "Executed: $sql<br />\n";
	}
	fclose($handle);
}
```