IMPROVING INDOOR POSITIONING VIA MOBILE SENSING

By

Chen Qiu

A DISSERTATION

Submitted
to Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2017

ABSTRACT

IMPROVING INDOOR POSITIONING VIA MOBILE SENSING

By

Chen Qiu

Accurate indoor position and movement information of devices enables numerous opportunities for location based services. Services such as guiding users through buildings, highlighting nearby services within shopping malls, or tracking the number of steps taken are some of the opportunities available when accurate positioning information is computed by devices. GPS provides accurate localization results in an outdoor environment, such as navigation information for vehicles. Unfortunately, GPS cannot be applied indoors pervasively due to the various interferences.

Indoor localization has been a challenging and significant topic in recent decades. Although extensive research has been dedicated to this field, accurate indoor location information remains a challenge without the incorporation of expensive devices or sophisticated infrastructures within buildings. We explored one practical approach for indoor map construction and four representative resolutions for indoor positioning.

Considering most indoor localization approaches are based upon indoor maps, we develop techniques to construct indoor maps. Since indoor maps might be dynamic and updated regularly, we present iFrame, a dynamic approach that uses mobile sensing techniques for constructing 2-dimensional indoor maps. We abstract the unknown map as a matrix and use mobile devices that incorporate three mobile sensing technologies - accelerometers to support dead reckoning, Bluetooth RSSI detection, and WiFi RSSI detection. The layouts of rooms and hallways can be constructed automatically within 5-10 minutes.

Based on the indoor map, we propose four indoor localization approaches by leveraging mobile sensing techniques.

First, although GPS can not help indoor localization directly, we propose iLoom, which

adopts a user's motion behaviors built by GPS information to enhance indoor localization. iLoom leverages an Acceleration Range Box to improve a user's acceleration value used for computing dead reckoning. By transfer learning the information from users' motion behaviors to the Acceleration Range Box, iLoom improves the Acceleration Range Box to achieve more accurate indoor positioning results.

Second, we introduce CRISP - a prototype that leverages opportunities of the interaction of multiple smartphones to enhance indoor positioning. When mobile devices cooperate and share position information iteratively, the localization accuracies of mobile devices increases gradually. In addition, based upon the obtained location information, CRISP provide a pedometer that can avoid the accumulative errors caused by accelerometers.

Third, we present SilentWhistle, a mobile prototype that incorporates acoustic information and motion traces on smartphones to locate users. When users encounter each other or related beacons, by measuring the relation between sound strength and distance, the initial location information obtained by dead reckoning can be enhanced by triangulations transferred from sound strength. Centralized and distributed models of SilentWhistle can avoid the incorrect location messages spreading based on the fault tolerance property.

Finally, by employing mobile robots in indoor scenarios, we propose AirLoc to improve the indoor positioning accuracies on smartphones. When a robot is near a smartphone, the robot sends accurate location information to users' smartphones via Bluetooth. In AirLoc, we design dynamic programming algorithms to generate the optimal serving routes for a single robot. Then, we extend the single robot model to multi-robot model. In our simulation, the multi-robots are organized by an unbalanced tree and serve areas by the Distance/Density First Algorithm.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Professor Matt W. Mutka for his support, the time he spent to advice, guide, and teach me how to seek academic problems and accomplish research projects. This dissertation cannot be completed without his insights in numerous discussions, patience in manuscript reviews, encouragement and inspiration in the whole Ph.D. study. I am honored to be one of his students. This rewarding experience will benefit my entire life. I also thank the rest of my guidance committee members: Dr. Li Xiao, Dr. Guoliang Xing and Dr. Mi Zhang, for their helpful comments and creative suggestions.

I would thank the labmates in eLANS including Yu Wang, Tian Hao, Wahhab Albazrqaoe, Chin-Jung Liu, Fernando Cintron, Aiping Dou, Sayeed Hyder, Le Tam Dan, etc. I thank to Dr. Rongliang Zhou and Mr. Bruce Fleming at HP Labs broaden my view in reinventing research works to industry applications. I also thank my great friends, Chaoyue Liu, Di Kang, Chao Huang, Jiao Chen, and all other graduate students in Michigan State University for their help and company throughout the journey.

Finally, I would express my profound gratitude to my beloved families, who always believe in me and give me encouragements. They have always been a rock behind me, loving, and supporting me unconditionally.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Challenges and Motivation

Obtaining location information is fundamental, significant and challenging in mobile computing and pervasive computing areas. For outdoor scenarios, GPS [49] has been widely used. The accuracy and performance of GPS are satisfactory for most applications. Nevertheless, GPS is generally not suitable to locate certain objects indoors, since microwaves will be attenuated and scattered by roofs, walls and other objects [34].

In the past decades, extensive research has attempted to locate people in indoor environments with the high-accuracy, convenience, and low-cost, but the results of these approaches are not satisfactory [41, 30]. Traditional device-based indoor localization approaches can achieve high accuracy, however, the cost of these systems and the inconvenience constrain their further development [14, 55, 51, 87, 43, 27]. Although device-free indoor localization, such as fingerprint indoor positioning, does not depend on expensive devices, the complex data training and high computational complexity are still challenging [91, 4, 83]. More researchers employ smartphones to localize people by using some inertial sensing techniques. Nevertheless, the computed locations are not accurate due to the errors of sensing, such as the deviations of accelerations and the noise of visible lights and sounds [71, 3, 32, 90]. Therefore, we ask the question: can we provide approaches to improve smartphones' localization accuracies with affordable complexity? By leveraging mobile sensing techniques, we design, implement, and evaluate accurate indoor localization approaches and the related location based applications.

Most indoor localization mechanisms depend on known indoor maps [15, 2, 54]. In reality, for some situations, users of smartphones cannot obtain the indoor floor plan. Even if users have the indoor plan, the layouts of rooms and hallways often change. We design and

implement an approach to construct the dynamic indoor map with inertial sensing techniques.

Based on the above analysis and discussion, in this chapter, we introduce one approach for constructing indoor floor plan and four prototypes for improving indoor positioning.

1.2 Indoor Floor Plan Construction via Mobile Sensing

Many pervasive computing applications depend upon maps for navigation and support of location based services. Maps are commonly available for outdoor pervasive applications from a variety of sources. An individual can determine their location outdoors on these maps via GPS. Indoor pervasive applications may also need to know the layout of rooms, doorways and hallways of buildings, and the objects and obstacles within them, however indoor maps of buildings are less prevalent [15, 2, 69]. Moreover, indoor maps may need to be dynamic and updated regularly since the layout changes when objects and obstacles are added or removed by people within the building. We propose iFrame [58], a dynamic approach that leverages existing mobile sensing capabilities for constructing indoor floor plans. We explore how iFrame users may collaborate and contribute to constructing 2-dimensional indoor maps by merely carrying smartphones or other mobile devices, and to allow their mobile devices to share information with other users' devices.

1.3 Improving Indoor Localization Accuracy by Learning Outdoor Motion Behaviors

Smartphones are equipped with many low-cost sensors. As a result, opportunities open for smartphones to serve as a platform for many challenging applications, including indoor localization [77, 3, 75, 39]. By employing accelerometers on smartphones, dead reckoning is an intuitive and common approach to generate a user's indoor motion trace [71, 9]. Nevertheless, dead reckoning often deviates from the ground truth due to noise in the sensing data. We propose iLoom, an indoor localization approach that benefits by transferring learning from

tracking outdoor motions to the indoor environment [60]. Via sensing data on a smartphone, iLoom constructs two datasets: relatively accurate outdoor motions from GPS and less accurate indoor motions from accelerometers. Then, iLoom leverages an Acceleration Range Box to improve a user's acceleration value used for computing dead reckoning. After using a transfer learning algorithm to the two datasets, iLoom boosts the Acceleration Range Box to achieve better indoor localization results. In addition, iLoom exploits indoor GPS exception cases, pedometer, and average speed estimation to further improve dead reckoning.

1.4 Cooperation among Smartphones to Improve Indoor Positioning

Modern smartphones are equipped with sensors and radios that can detect movement and can be used to predict location. Dead reckoning applications on a smartphone may attempt to track a person's movement or locate a person within an indoor environment [71, 9]. We propose CRISP [57] - CoopeRating to Improve Smartphone Positioning, which assumes that dead reckoning have inaccuracies, but leverages opportunities of the interaction of multiple smartphones. Each smartphone computes its own position, and then shares it with other nearby smartphones. The signal strengths of multiple radios that are used on smartphones estimate distances between the devices. While individual smartphones provide inaccurate positioning information, accuracy may increase when several smartphones cooperate and share position information through multiple iterations. CRISP is an inexpensive means to improve position information and possibly lead to better results for a number of applications, including exercise profiling.

1.5 Indoor Positioning with Assistant from Acoustic Sensing on Smartphones

Although existing mobile sensing techniques can improve indoor localization accuracies, the time of scanning and building connections is time consuming. Sometimes, when samples are collected by mobile devices, the users of devices move to other locations. We introduce SilentWhistle[59], a passive indoor localization system with real time and fault tolerance features based on the daily uses of smartphones. By detecting and filtering the received audio frequencies, we decode the frequencies to location information. When users encounter each other or related beacons, by measuring the relation between sound strength and distance, the initial location information obtained by dead reckoning can be enhanced by triangulations transferred from sound strength. Centralized and distributed models can avoid the incorrect location messages spreading based on the fault tolerance property.

1.6 Multi-robots assisted Indoor Localization

Since most existing smartphone applications cannot avoid accumulative errors when calculating position and movement, we propose a novel approach, AirLoc - Adopting mobile robots to assist indoor Localization of smartphones [56]. A moving robot employs a Bluetooth adapter and a known map to assist a smartphone to reduce its localization error. When a robot is near a smartphone, the robot sends accurate location information to users' smartphones via Bluetooth. We design a path planning strategy for a robot to enhance the localization accuracies of smartphones over extended time periods. Moreover, in order to promote the single robot approach, we extend it to the multi-robot assisted indoor localization by simulation. The multi-robots are organized by an unbalanced tree and serve areas by the Distance/Density First Algorithm.

1.7 Structure of the Content

This rest of this dissertation proposal is organized as follows: Chapter 2 reviews the related work in indoor localization and map construction areas. We present the automatic indoor map construction approach in Chapter 3. Chapters 4-7 introduce four indoor localization prototypes: Chapter 4 studies improving indoor localization by profiling outdoor movement on smartphones; A prototype that leverages cooperation among smartphones to provide accurate indoor positioning is introduced in Chapter 5; We enhance the existing indoor localization approaches via acoustic sensing techniques in Chapter 6; Chapter 7 explores mobile robot assisted indoor localization. We presents work in progress and proposes the future work in Chapter 8.

CHAPTER 2

BACKGROUND SURVEY

Location services are essential and important for mobile computing and wireless networks. For outdoor positioning, Global Positioning System (GPS) is widely accepted and has satisfactory performance. Unfortunately, GPS signals are rarely accessible indoors. Therefore, indoor localization has been a challenging and significant topic in industry and academia. Within the ubiquitous deployment of wireless networks and devices, the past decades have witnessed extensive indoor localization techniques. In this chapter, we introduce four types of classical indoor localization approaches. In addition, within the development of indoor localization and other location based services, the means to construct an indoor map to support these services is important. We will also show the related work for this issue.

2.1 Device Based Indoor Localization

Traditional indoor positioning approaches, aiming to provide LBS indoors, can be categorized to two types: device-based and device-free. Device-based indoor localization relies on special devices, such as ultrasound [14, 79, 55], RFID [51, 20, 25, 87, 43], and infrared [27, 6] devices. Whereas some of these systems can obtain accurate location information, the costs of the devices and their inconvenience limit their further development. For example, Cricket [55] is a typical device-based systems. As Cricket, it has location beacons attached to the ceiling of a building and receivers. People carry receivers to obtain RF signal transmitted from the location beacons on the celling periodically. Cricket can locate the position of a user within $3cm$. Although people often carry smartphones, the current smartphones do not integrate these devices. SpotON [20] uses RFID tagging technology for three dimensional location sensing based on radio signal strength analysis. The specific tag hardwares are required for

the implementation.

2.2 Device Free Indoor Localization

Apart from device-based indoor localization, device-free approaches do not need users to carry extra devices and can provide users' locations by analyzing different types of signals. For most device-free approaches [91, 4, 89, 83, 67, 84, 8, 81, 88], RSSI, Channel State Information (CSI), and WiFi backscatter are used to construct a fingerprint map. A fingerprint approach selects the best-matching position in the radio map as the mobile object's position. Nevertheless, building the off-line map and signal processing can be a significant challenge.

In RADAR [4], one of classical device-free method, multiple WiFi Access Points (APs) are heard at each location. Some stations provide overlapping coverage of the area of interest. RADAR makes use of a signal propagation model to estimate the object location to a great accuracy. LiFS [78] observe the results of WiFi signal variations caused by body interferences and power fading. By analyzing the signal variations in different sites, LiFS localize users without bounded devices.

In fact, for most of device-free approaches, they can only achieve room-level accuracy. In addition, training the radio-map and signal analysis are labor-intensive and time-consuming.

2.3 Simultaneous Localization and Mapping (SLAM)

Some indoor environments, such as convention centers, museums, and hospitals provide various location services. Mobile robots may be able to supply certain services to users, such as advertisement and music. These robots have the abilities to establish their own positions and orientations within the frame of reference. By using a camera or an infrared sensor, the robot can find its route in an indoor environment and avoid dangerous situations by SLAM (Simultaneous localization and mapping) approaches. Besides, the robots can calibrate their positions computed by SLAM to ensure their high localization accuracies [13, 73, 44]. In

(a) (b)

Figure 2.1 Indoor map generation by Google Tango.

addition, robots use SLAM [13, 1, 72] to create a new map within an unknown environment, or to update a map within a known environment. For example, Google Tango combines 3D motion tracking with depth sensing to give your mobile device the ability to know where it is and how it moves through space [22]. Based on the extensive off-line training and image collections, the Google indoor map team has build the indoor floorplan of more than 1000 sites, such as famous shopping malls and airports. Also, the robots can compute and calibrate their current location information. For example, Turtlebot [16], one of the common robots, has two application (gmapping and amcl) to help it find a reasonable route in one room and compute correct positions.

But SLAM highly relies on the sensors, such as laser or camera. Besides, current SLAM approaches, the robots can only compute location information for themselves, other mobile devices (as smartphones, PDA) cannot obtain location information by interacting with the robots.

2.4 Smartphone Based Indoor Positioning

With the proliferation of smartphones, more researchers use sensors on smartphones to localize people. Dead reckoning [71, 9] is a common method to estimate user's current location by physical formulas. But the accelerations obtained from sensors often includes some noise, for example, the direction obtained from accelerometer is different from the people's real direction. The accumulative errors grows sharply within the time increase. To address the

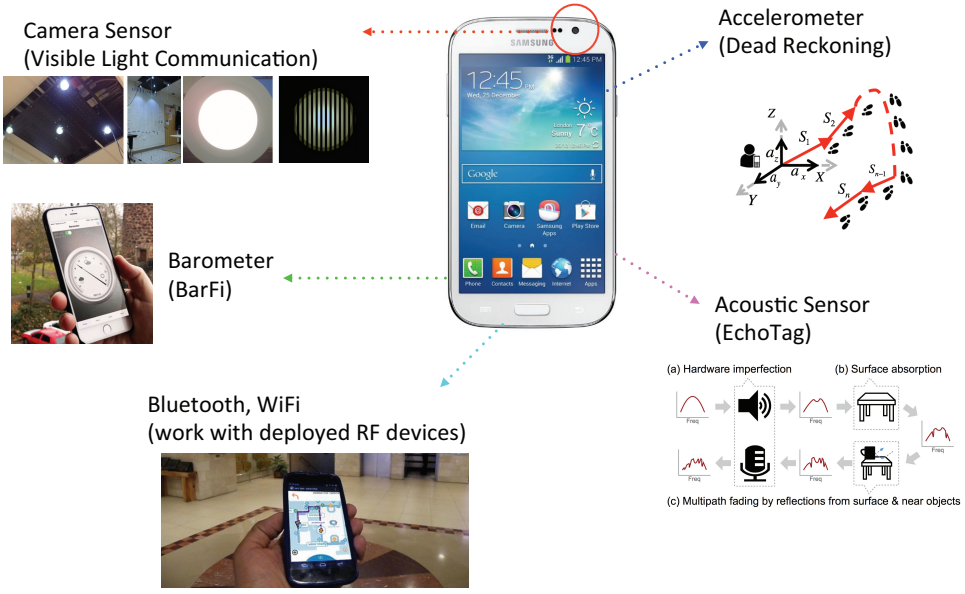


Figure 2.2 Smartphone based indoor localization.

problem, UnLoc [77] employs a virtual landmark to assist dead reckoning. The landmark is the sensing signatures naturally existing in indoor environment. Dead reckoning tracks locations between different landmarks. However, the deviations still occur sometimes.

CUPID [68] utilizes WiFi physical layer information to extract the signal strength and uses geometry to locate an object, it achieves the accuracy around 3 meters in complex office environments. SAIL [46] employs the propagation delay of the signal traversing between single WiFi AP and smartphone to eliminate the errors caused by dead reckoning. Nevertheless, the localization results for these two approaches are not highly accurate.

By using computer vision and other sensing technologies [3, 32, 85, 62, 75, 42, 39, 80, 40, 7, 33], some researchers fuse the data from accelerometers, camera sensors, and acoustic sensors to provide solutions for indoor localization. Through employing the Visible Light Communication devices, PIXEL [90] provides a localization mechanism by just using the strength of visible lights. It can achieve the sub-meter level results. LiTell [93] and Luxapose [33] analyze the fixtures and frequencies of lights, by distinguish the features of light sources in different locations, smartphone can predict the locations where they are. However, the signal

processing and recognition in indoor environments are still challenging for these systems.

By applying the geometry algorithms, Beep [45] uses PC microphones connected to desktops serving as acoustic sensors to compute the location of a roaming device. Current smartphones also can generate an acoustic signal for geometry computation. They provide capacities to localize smartphone users.

2.5 Indoor Map Reconstruction

Many researchers have designed mobile systems to rebuild indoor maps. Since people use smartphones and other mobile devices in their daily lives, some approaches propose to build indoor maps by smartphones. In the computer vision area, researchers activate camera sensors on smartphones to construct the map [15, 74, 29, 66]. JigSaw [15] leverages camera sensors to obtain images and adopts point cloud algorithms to piece together images to reconstruct the indoor floor plan. Sextant [74] combines the photos obtained by smartphones with gyroscope information to draw the map. Though these vision based approaches can build the indoor map by smartphones or other specific devices, the procedure of image processing is still challenging. Some work uses the dead reckoning by accessing data from accelerometers and gyroscopes to generate the trajectories of users [2, 54]. They analyze these trajectories and sensing data of elevators and entrances to construct the map. Nevertheless, the deviations and errors of dead reckoning often reduce the accuracies. Other approaches use WiFi signatures and a series of algorithms to discover the layout of buildings, but the WiFi noise is difficult to process [24, 69].

CHAPTER 3

AUTOMATIC INDOOR FLOOR PLAN CONSTRUCTION VIA MOBILE SENSING

3.1 Introduction

Map availability is essential for Location Based Services [34], such as tracking and localization services. In outdoor environments, Google Map, Yahoo Map and other services provide outdoor maps for the users of mobile devices. Unfortunately, maps are much less commonly available for indoor environments. In order to provide indoor floor plans, many people may need to dedicate significant time to construct the floor plans and make them available to commercial channels. Google Indoor Map [94] has collected over 1,000 indoor floor plans of airports and shopping malls in U.S. and Japan. These maps are built manually, which may be tedious and high cost. Although some approaches for building indoor floor plans have been proposed [54, 2, 15], the accuracies of generated indoor maps and the complexity of computation need to be improved. Moreover, even if the indoor maps are constructed, the layouts within these buildings may often change. For example, furniture within one room might be moved. This leads to our question: *Can we generate a dynamic indoor map automatically and accurately without complex data training?*

In this chapter, we design and evaluate iFrame, an light-weight approach that leverages mobile sensing data to construct dynamic floor plans of complex indoor environments automatically. iFrame does not require commercial negotiation with providers of indoor maps. By opening the iFrame application on smartphones and by walking around the targeted indoor environments, users passively construct the floor plan. We abstract the indoor map as an $m \times n$ matrix. Each grid in the matrix has the same shape and area. Grids in the matrix are described by a value between two extreme states: 0 - the grid is vacant, 1- the grid is

completely occupied by objects. The value between 0 and 1 represents the percentage of the area that an object occupies the grid.

As illustrated in Figure 1, users leverage dead reckoning to obtain their motion traces. For locations a user visits that are identified by dead reckoning, the grid location will be marked vacant. Because dead reckoning may deviate from the ground truth, we seek to enhance its accuracy and therefore the proper identification of the status of grids in the matrix by employing 1) a Markov chain and average speed prediction to calibrate the trace deviation and 2) Bluetooth and WiFi radios to compensate for the deviation errors. We assume the Bluetooth RSSI received from another device to be related to distance [70]. We suppose abrupt changes in WiFi signal strength is related to obstacle detection [21]. When a user turns on the Bluetooth option and if a Bluetooth adapter can detect other Bluetooth devices, then we set the grids on the link between Bluetooth adapters to 0 (vacant). Since WiFi signal strength is more sensitive to obfuscation rather than to distances, and if the WiFi signal strength decreases abruptly, then the grids that on the link are recorded as 1. Because the common discovery procedures for Bluetooth and WiFi components are time consuming, it may cause time delay to receive sensing information. We speed up the discovery procedure by interrupting the scanning mechanism at the program level. Considering the imperfections of each method, Curve Fit Fusion (CFF) is introduced to combine the methods for improving the output matrices. By executing iFrame on 3-5 users' smartphones iteratively within 5-10 minutes, we can obtain a two dimensional floor plan for a 12m $\times$ 6m office room. Even if iFrame users walk daily by an unattended mode, the system is able to generate the floor plan of a room within 7 hours. Since the generated map is built by real time sensing information, even if the layouts of rooms change, the changes can be represented on the map. By introducing Anchor Points (such as entrances, stairs, and elevators), the traces generated by dead reckoning can be initialized and calibrated. As numbers of iFrame users who share their information increases, the constructed floor plans generated from a larger data set drawn from a larger set of users will have a lower error rate and will be generated

within a shorter amount of time.

Our work therefore makes following contributions: First, although some approaches have adopted crowd sourcing as a means to rebuild the indoor floor plans, iFrame is the first of its kind to measure RSSI values with other scanned mobile devices to help construct the layout of indoor environments. Second, we design a sensor fusion approach to combine and enhance the indoor maps computed by dead reckoning, Bluetooth, and WiFi RSSI detections. Third, iFrame is convenient to deploy and may be used passively by smartphones' users. The mobile sensing mechanism allows iFrame to represent the layout changes of buildings. As more users collaborate, improved resolution and clarity of the maps are generated. In addition, iFrame can cooperate with common wireless Access Points to further enhance the accuracies of indoor map. Some special elements in an indoor map, such as doors, chairs, elevators can be represented on the generated floor plan. The Bluetooth and WiFi detection speeds satisfy the real-time requirement of mobile computing.

3.1.1 System Design

3.1.1.1 Overview of iFrame

iFrame leverages existing smartphone and mobile device technology without the need of complex training to construct floor plans within buildings. Before constructing an indoor map, we formulate the unknown map as a matrix. Each element in the matrix represents a grid on the map. Each grid is marked in one of two states: empty or object. As shown in Fig. 3.1, the procedure of map construction is composed of three phases:

1. iFrame is a system based on mobile sensing. Smartphone users adopt the sensing techniques to set the status of each element in matrix iteratively: i) dead reckoning, ii) Bluetooth, and iii) WiFi detections. By interrupting the scanning periods, we improve the speed that both Bluetooth and WiFi detect other devices.

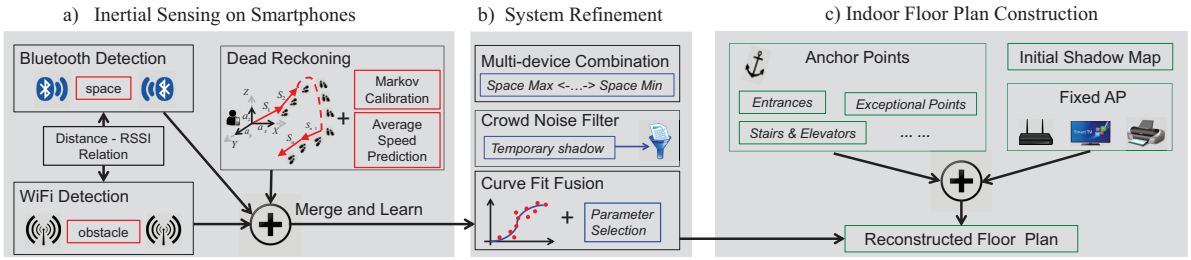


Figure 3.1 System overview of iFrame.

2. Based upon the data collected by sensors on smartphones, iFrame fuses the output matrices of the three methods via Curve Fit Fusion (CFF), which is supported on the cloud server. Then, iFrame combines the matrices built from multiple mobile devices and filters the noises caused by the *temporary shadow*.

3. iFrame draws the layout in each room and hallway. By introducing Anchor Points (such as entrances, stairs, and elevators), the map of a indoor building is constructed completely.

Table 3.1 Main notations in iFrame

Terms	Definition
(x_α, y_α)	Position of users α
$RecRSSI$	RSSI values the users received
$\mathbb{M}$	Matrix of abstracted map
T_b	Threshold of Bluetooth Detection
T_{w1}, T_{w2}	Thresholds of WiFi Detection
$Dist(A, B)$	Euclidean distance between A and B
$V_{i,j}$	Shadow value of element (i,j) in M
V_{BL}	Shadow values on the Bluetooth link
V_{WL}	Shadow values on the WiFi link
M_f	Matrix generated by sensing approach f
S_f	Shadow rates computed by sensing approach f
M_{d_i}	Matrix generated by device i
EBV	Error of Block Value
$B_{m\times n}$	EBV for element $M(m, n)$ in M

3.1.1.2 Map Matrix

Our task is to build the two dimension indoor floor plan. However, the initial two dimension map is unknown except for the size of the area. We divide the map into grids, and each grid

is processed as an element in a matrix $\mathbb{M}$. All grids are squares with the same size. If a grid is vacant, it is defined as an *empty* grid. We set the value of corresponding element in the matrix to 0. If a grid is totally occupied by an object, it is defined as an *object* grid. The corresponding element in the matrix is 1. The Shadow Rate is the amount that an object partially occupies the matrix element. If the size of the grid is $1m^2$ and the objects in the grid occupy $0.7m^2$, we define the shadow rate of the grid as 0.7. $V_{i,j}$ refers the value of a element in matrix $\mathbb{M}$. The matrix $\mathbb{M}$ is as follows:

$$\mathbb{M} = \begin{bmatrix} V_{1,1} & . & . & V_{1,n} \\ . & . & & \\ . & & . & \\ V_{m,1} & & & V_{m,n} \end{bmatrix} \tag{3.1}$$

3.1.1.3 Dead Reckoning Detection

When a user enters a grid on the map, the grid has empty space for the user to occupy. The grids on the user's motion traces will have their status marked as *empty* ($\underline{0}$). Hence, the user's motion traces in a building are able to describe the layout of the building.

iFrame uses dead reckoning [71, 9] to obtain the user's motion traces. Based on the accurate initial position, the application executing on the mobile device computes movement distance in each segment continuously.

An accelerometer is an inertial sensor that is suitable for a user's activity recognition. Mobile devices sense the acceleration on three axes orthogonal to one another periodically. We set the time length of each period to be 1 second. The formula to compute acceleration is in equation (3.2). The symbol g refers to the earth gravity, a_x, a_y and a_z refer to the acceleration received on the Ox, Oy and Oz. By the obtained acceleration in each period, the movement distance of a mobile device in time period n is based on the equation (3.2). Parameter v_{n-1} and a_{n-1} refer to the velocity and acceleration from the previous time

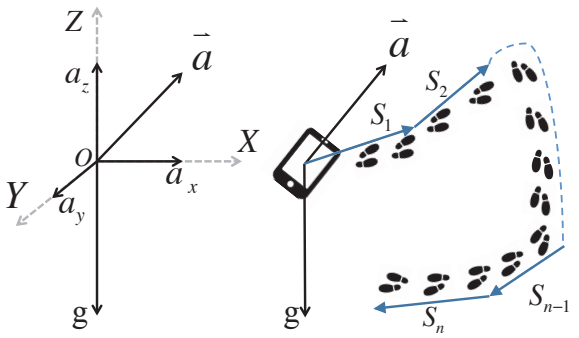


Figure 3.2 Dead reckoning.

period, t_n refers to the time length of the current period. S_n refers to the vector of movement distance in the current period. As shown in Fig. 3.2, if the application on a smartphone computes movement distance in each segment by (3.2) continuously, it can obtain the whole trace of mobile device.

$$\vec{a} = (a_x, a_y, a_z - g),\ \vec{S}_n - \vec{S}_{n-1} = \frac{1}{2}\vec{a}_{n-1}t_n^2 + \vec{v}_{n-1}t_n \tag{3.2}$$

For dead reckoning, as shown in Fig. 3.3, we leverage sensors on the smartphone (accelerometer, magnetometer, and gyroscope) to estimate the user's step length and obtain heading direction.

Unfortunately, dead reckoning has limitations: if the acceleration and orientation values from the smartphone do not reflect the human body's acceleration, for example, when the smartphone is held in a user's hand, and the hand shakes, the obtained accelerations are incorrect. The generated trace will deviate from the ground truth. Furthermore, the accelerometers on most smartphones are not highly accurate. A common sensor used for indoor localization [63] may have an error of 308 meters within one minute due to a 0.5 degree deviation occurs on the orientation sensor.

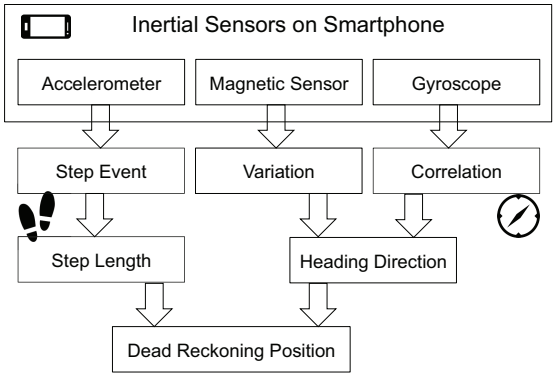


Figure 3.3 Workflow of dead reckoning.

To improve the accuracy of dead reckoning, we proposed a trace prediction approach relying on a Markov chain. A Markov chain is a sequence of random variables X_1, X_2, ..., X_n. Given the present state, if both conditional probabilities are well defined, the future is conditionally independent of the past as formula (3.3).

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = P(X_{n+1} = x | X_n = x_n) \tag{3.3}$$

In our model, Markov chain is abstracted as a directed graph:

1) Each grid in the map refers to a node, the edges in graph are labeled by the probabilities of moving from one grid at time n to the other grid at time n+1.

2) As Fig. 3.4, the user can only move to a neighbor grid/node from time n to time n+1. This action is represented by the transition matrix from time n to time n+1. For each grid, since the number of potential transition grids are 8, the size of transition matrix is 8×8. If the user goes to a non-neighbor node, it needs more than one step. The value k is the number of steps for transferring grids.

$P\left\{X(n_m + k) = j | X(n_m) = i_m\right\}$ is the transition probability by moving k steps at time n. It is recorded as $P_{i,j}(n, n+k)$ for short. It indicates the probability that the user is in the state/grid i at time n, and after the transition of k steps, it goes to the state/grid j. $P_{i,j}(n, n+k)$ depends on initial state/grid i, the final state/grid j, the number of steps k,

instead of time n. Therefore, the transition probability of k steps can be defined as:

$$P_{i,j}(k) = P_{i,j}(n, n+k) = P\left\{X(n+k) = j | X(n) = i\right\} \tag{3.4}$$

When $k = 1$, $P_{i,j}(1)$ is a one step transition probability. The matrix of one step transition is recorded as $P(1)$. For n steps transition probability, the transfer probability is $P_{i,j}(n)$. The corresponding matrix is the n step transition probability matrix, named $P(n)$. By applying C-K equation [17], we can obtain:

$$P(n) = P \cdot P(n-1) = P(n-1) \cdot P => P(n) = P^n \tag{3.5}$$

For $P(1)$, the corresponding $P_{i,j}$ value refers to the probability of moving from state/grid i to state/grid j. Since we can count the times from i to j as $num_{i,j}$, then

$$P_{i,j} = \frac{num_{i,j}}{\sum_{j=1}^{n} num_{i,j}} (1 \leqslant i, j \leqslant n) \tag{3.6}$$

After obtaining the one step transfer probability matrix, we are able to compute the transition probability matrix of k steps by $P(k) = P^k$.

To predict the user's selection for the next step, it is necessary to consider both the historical and the current information of the user's traces. The information that is closer to the current time period will have more influence on the user's decision for the next step. According to this strategy, we provide the prediction formula as follow:

$$X(t) = a_1 H(t-1)P + a_2 H(t-2)P^2 + ... + a_n H(t-n)P^n \tag{3.7}$$

In formula (3.7), t is time period of next grid, t-1 refers to the previous time period of t. $X(t)$ is the prediction probability for the next grid/state. $H(i)$ denotes the states of the next grid's previous i grids and the factors $a_1, a_2, ..., a_k$ from 1 to k are used to decide their next grids ($a_1 \geq a_2 \geq ... \geq a_k$). Then, we select the maximum element in $X(t)$, and take the maximum element's corresponding grid as the prediction grid for the next step.

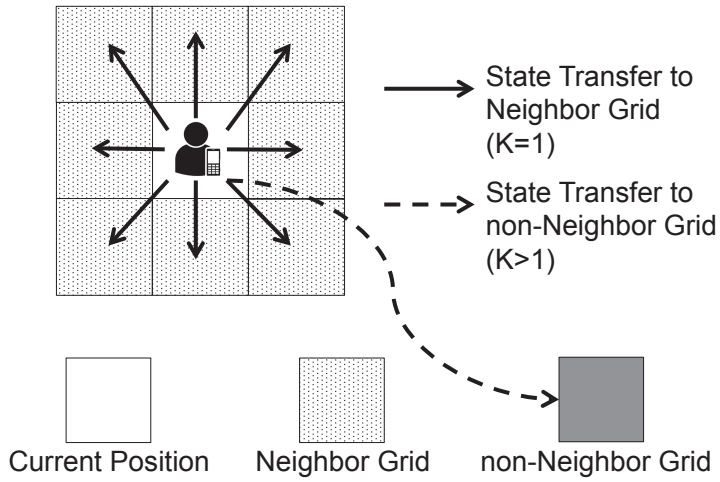


Figure 3.4 Markov chain state transition. The user moves from the current grid to the neighbor grid.

When the user predicts the grids that are his/her next moving targets, if the motion trace computed by formula (3.2) does not include the predicted grids, the acceleration values for this time period will be replaced by the retrieved accelerations in the previous time period. Then, our system recalculates the current segment of motion trace.

3.1.1.4 Bluetooth Detection

Received Signal Strength Indicator (RSSI), in units of "dBm", is a measurement of the power present in a received radio signal. RSSI can be recorded from the components of most mobile devices, such as WiFi adapters and Bluetooth adapters. Bluetooth RSSI values received from the other devices are related to the distance between the devices. Shorter distances often represent stronger RSSI values [70]. We evaluated the RSSI-distance mapping relations for the Bluetooth adapters on Samsung Galaxy smartphones and Samsung Tablets. These mapping relations are pre-trained and stored as hash maps.

If two devices can detect each other, we assume that the link between the two devices has open space. We estimate the distance between the two devices from the initial map. The positions of the two devices are obtained through dead reckoning. If the corresponding RSSI

values from RSSI-distance relation are close to the received RSSI values (the error range is 3dBm), namely, we assume no interference on the link. The elements V_{BL} between the link in $\mathbb{M}$ are set as 0. When the received RSSI is less than the corresponding RSSI values (beyond the error range), there might be some interference on the link. It is unreasonable to set all the values of elements on the link in M as 0. The procedure of Bluetooth detection is described in Algorithm 1.

Algorithm 1 Bluetooth Detection Algorithm

Input:
$(x_\alpha, y_\alpha), (x_\beta, y_\beta)$, $RecRSSI_\alpha, RecRSSI_\beta$, $V_{BL}(x, y)$;
Output:
The latest $V_{BL}(x, y)$;
// Use the prepared Hashmap to compute
// the estimated RSSI by known positions
$Dist(\alpha, \beta) \leftarrow \sqrt{(x_\alpha - x_\beta)^2 + (y_\alpha - y_\beta)^2}$
$EstRSSI \leftarrow$ HashMap($Dist(\alpha, \beta)$);
if $(RecRSSI_\alpha + RecRSSI_\beta)/2 - EstRSSI < T_b$ **then**
  for each $V_{BL}(x, y)$ **do**
    // Once the RecRSSI is close to Estimated RSSI,
    // we add 0 for the elements on Bluetooth link
    $V_{BL}(x, y) \leftarrow V_{BL}(x, y) + 0$;
    cnt $\leftarrow$ cnt+1 & $V_{BL}(x, y) \leftarrow V_{BL}(x, y)$ / cnt;
  end for
end if

3.1.1.5 WiFi Detection

Wi-Fi Direct enables devices to connect with each other without requiring a wireless access point. It also can provide RSSI values for each of the connected devices. Wi-Fi Direct is supported by most current smartphones. In contrast to Bluetooth RSSI, WiFi RSSI values are not as sensitive to the distance within a short range. However, the value of WiFi RSSI is susceptible to the interference between the link [8]. It is thus desirable to uses WiFi to describe the *object* on the map. Figure 3.5 illustrates a preliminary observation. Alice and Bob are two user of iFrame. They build connections via Wi-Fi Direct first. Then, Alice walks from position 1 to position 2. On her way from position 1 to position 2, there is a

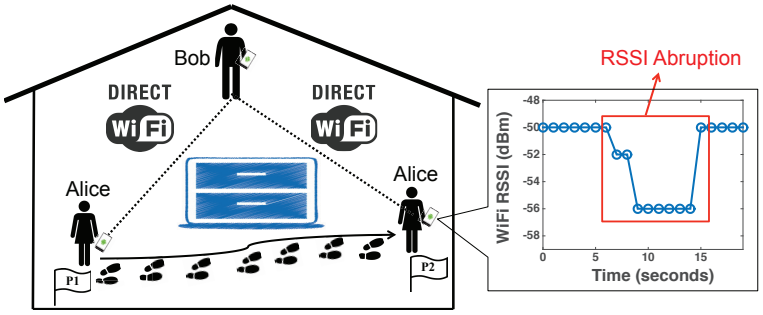
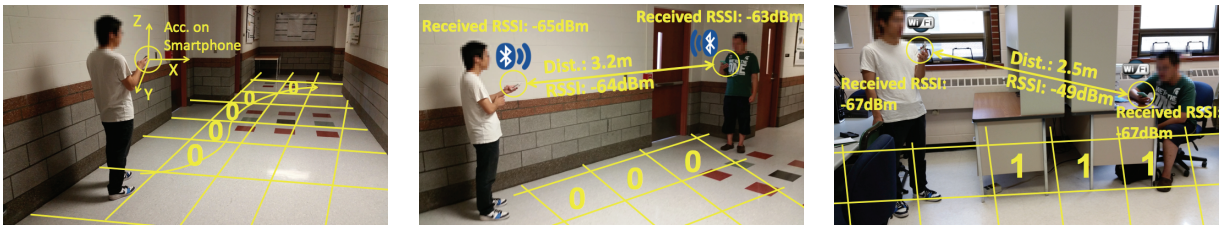


Figure 3.5 Direct WiFi Observation. When a user passes an obstacle between the WiFi link, the RSSI value decreases sharply.



(a) Dead reckoning detection. (b) Bluetooth link detection. (c) WiFi Direct link detection.

Figure 3.6 Approaches for constructing the map matrix $\mathbb{M}$: (a) Dead reckoning trace determines the space areas; (b) Bluetooth link determines the space areas; (c) WiFi Direct connection detects the obstruction.

cabinet between user Alice and Bob. Once Alice crosses the cabinet, the received WiFi signal strength from Bob has an obvious abrupt signal change. Based upon this phenomenon, we formulate the WiFi Detection in Algorithm 2.

In Algorithm 2, two events are pre-defined as follow:

Event (I): The differential value between the received RSSI value and the RSSI estimated by the distance is greater than the threshold T_{w1}.

Event (II): The WiFi RSSI between the two users drops abruptly (more than the threshold T_{w2}) from the previous period.

Only when the users want to make contributions to indoor map construction, the WiFi and Bluetooth adapters are broadcasting and scanning. The adapters will be idle when users do not build an indoor floor plan.

Fig. 3.6 demonstrates how the three sensing approaches work and perform.

Algorithm 2 WiFi Detection Algorithm

Input:
$(x_\alpha, y_\alpha), (x_\beta, y_\beta)$, $RecRSSI_\alpha, RecRSSI_\beta$, $V_{WL}(x, y)$;
Output:
The latest $V_{WL}(x, y)$;

```
1: for each time period i do
2:   // Use prepared Hashmap to compute the estimated RSSI by known positions
3:   Dist(α, β) ← sqrt((x_α − x_β)^2 + (y_α − y_β)^2)
4:   EstRSSI ← HashMap(Dist(α, β));
5:   if Event (I) and Event (II) are satisfied then
6:     // Event (I):
7:     // |EstRSSI − (RecRSSI_α + RecRSSI_β)/2| > T_w1
8:     // Event (II):
9:     // |(RecRSSI_α + RecRSSI_β)_i − (RecRSSI_α
10:    // +RecRSSI_β)_{i−1}| > T_w2
11:    for each V_WL(x, y) do
12:      // When RecRSSI is close to Estimated RSSI,
13:      // we add 1 for the elements on WiFi link
14:      V_WL(x, y) ← V_WL(x, y) + 1;
15:      cnt ← cnt+1 & V_WL(x, y) ← V_WL(x, y) / cnt;
16:    end for
17:  end if
18: end for
```

3.1.2 Enhance real-time property

For most BLE devices detection, the discovering period of a BLE adapter is more than 20 seconds. However, user's motion may change continuously. For example, once a certain round of BLE detection does not finish, the user's of smartphone may have left the position or even the environment where he was. Namely, the BLE detection should be synchronized with smartphone's users' motion behaviors. In our approach, once our program has detected three BLE devices, we interrupt the discovering program and restart the new procedure. Based on our observation from Android BLE detection, as shown in Fig. 3.7, the average discovery round for 15 BLE beacons can be reduced from 40 seconds to 5 seconds.

In addition, for WiFi detection, there exists the similar problem. We also "interrupt" the process of WiFi detection. By detecting three nearby WiFi devices, we restart the new

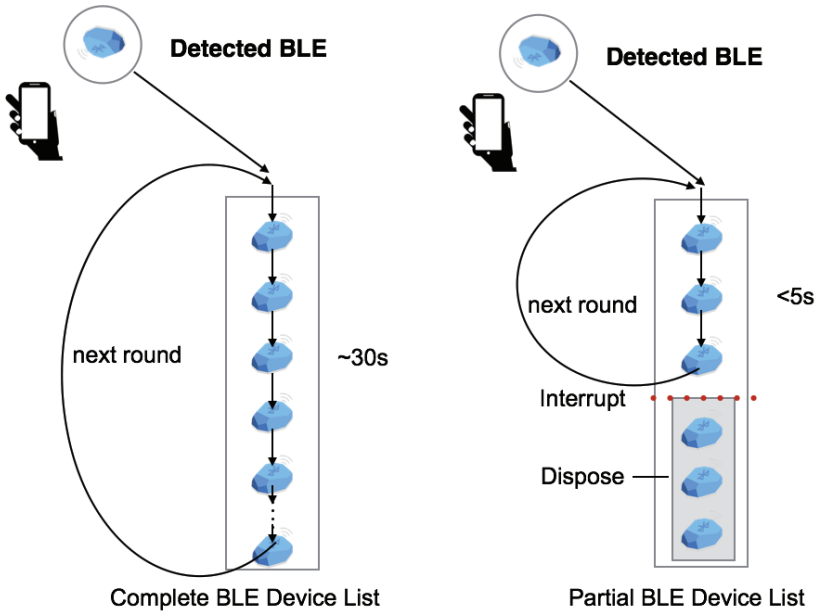


Figure 3.7 Reduce the scanning period by interruption.

scanning procedure and drop the following processing. The average discovery round for 10 WiFi devices can be reduced from 20 second to 5 seconds.

3.1.2.1 Matrix Fusion Mechanism

We introduced three techniques: dead reckoning, Bluetooth detection, and WiFi detection to set the values on the map matrix. However, there are some shortcomings for each of them: 1) dead reckoning can describe the user's motion traces, but the estimated positions often deviate the ground truth. For example, the accelerometer on a smartphone represents the acceleration of the smartphone rather than a human's movement. Once a user's body motion is different from the smartphone's motion, dead reckoning computation will fail due to the deviation. Though we designed the prediction mechanism to correct such errors, the deviations still occur sometimes; 2) Bluetooth detection approach may ignore some interferences that are lower than the threshold between the link; 3) WiFi detection cannot represent some vacant grids on the link that have detected objects. Therefore, the matrix computed by the three approaches includes errors. In order to refine the results, iFrame merges the three methods

and generates improved results.

We proposed Curve Fit Fusion (CFF) to combine the matrices computed by the three methods. According to the samples from the training data, we select a proper proportion to assign different weights to the three methods.

$$\begin{cases} a+b+c=1 \\ a \times M_D + b \times M_B + c \times M_F = M \\ a:b:c = {B_D}^{-1} : {B_B}^{-1} : {B_F}^{-1} \end{cases} \tag{3.8}$$

$$B_{m \times n} = \sqrt{\sum_{j=1}^{n} \sum_{i=1}^{m} (\Delta V_{ij})^2}, \Delta V = |V_g - V_e| \tag{3.9}$$

As given in equation (3.8), M_D, M_B, M_F denote the matrices computed after using dead reckoning, Bluetooth and WiFi detections. M is the matrix computed by combining the three techniques. The factors a, b, and c decide the proportion of M_D, M_B, M_F. S_D, S_B, S_F refer to the shadow rates of the three approaches in a room.

A special metric is introduced in equation (3.9). Error of Block Value (EBV) is the error value of the estimated shadow rate in each grid. V_g refers to the real shadow rate in each grid. V_e is the shadow rate computed by our approach. If we use an $m \times n$ matrix, the Error of Block Value is defined as $B_{m \times n}$, which is the average error value of all the grids.

We collect a small-scale training dataset: by changing the layouts in three rooms, for various types of layouts, each has a different shadow rate. Then, we compute the shadow rate and error value for each case. By applying different approaches, in Fig. 3.8, each sample on the figure is a case we tested. The figure illustrates the relation between the shadow rates and the error values for these cases.

As given in equation (3.10), B_D, B_B, B_F refer to the error values caused by dead reckoning, Bluetooth and WiFi detections. After applying the curve fit tool, we can compute the values of α and β. By knowing α and β, once we obtain the shadow rates, it is easy to

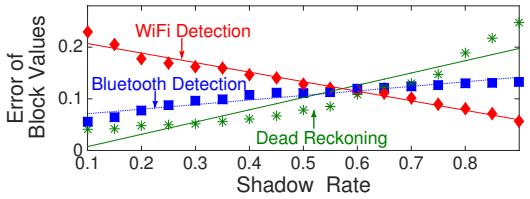


Figure 3.8 Curve Fit Fusion for the sensing data collected by the three proposed methods.

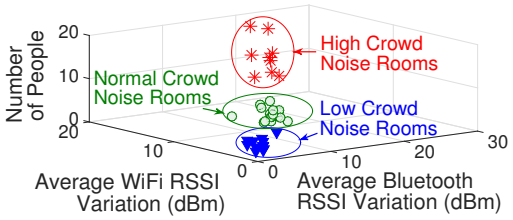


Figure 3.9 Different levels of crowd noise for rooms.

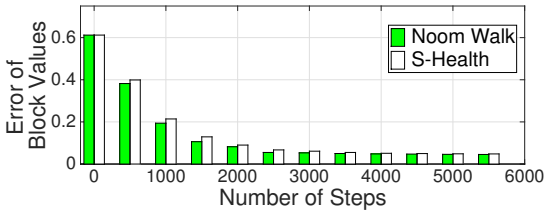


Figure 3.10 The relation between the number of steps and map rebuilt accuracy.

calculate the proper a, b, and c values from equation (3.8).

$$(B_D, B_B, B_F) = (\alpha_1, \alpha_2, \alpha_3) \cdot (S_D, S_B, S_F) + (\beta_1, \beta_2, \beta_3) \tag{3.10}$$

3.1.2.2 Multi-device Combination

Since iFrame is a crowd sourcing mechanism, all the users of mobile devices collect and upload sensing data. Therefore, it is significant how we organize data computed from each mobile device. Our system provides three types of organizations:

1) Maximum Space: $M_{d_1} \bigcup M_{d_2} ... \bigcup M_{d_{n-1}} \bigcup M_{d_n}$

2) Minimum Space: $M_{d_1} \bigcap M_{d_2} ... \bigcap M_{d_{n-1}} \bigcap M_{d_n}$

3) Mean Value: $\sqrt{(M_{d_1}^2 + M_{d_2}^2 + ... + M_{d_{n-1}}^2 + M_{d_n}^2)/n}$

M_{d_i} denotes the matrix computed by device i. The Space Maximum represents the combination matrix that has the most space. The Space Minimum is the combination matrix that remains the least space. The Mean Value is between these two extremes. We adopt the Mean Value as the default mechanism to merging sensing data from different devices.

3.1.2.3 Crowd Noise Filter

One situation is non-negligible: if there exist so many users who are stationary in one room or a hallway, the generated shadow map might include some errors. The people's bodies may tend to yield some shadows in the generated map. To reduce these "temporary shadows," iFrame checks the motion trace of each user periodically and considers if a user does not change his/her position within 5 minutes, iFrame will not use his/her data until he/she leaves the position.

In addition, if many people are in a room or hallway, regardless if the people's bodies are stationary or not, there might be interference for the smartphones' radio signals. The presence of large numbers of people often cause variations of RSSI values [82, 92]. Based on this phenomenon, we need to detect the room with high crowd density and amend the variations of the RSSI caused by the presence of many human's bodies.

Three steps are taken as a heuristic solution for filtering crowd noises:

1) We divide the matrix $\mathbb{M}$ into sub-matrices from M_1 to M_n. Each keeps the same size. For M_i ($0 \leq i \leq n$), in a certain time period, we focus on three features: the number of users, the sum of variation of Bluetooth RSSI for all the devices, and the sum of variation of WiFi RSSI for all the devices.

2) By the three proposed features, we assume each M_i is a sample on a three dimensional space. The three features represent the three dimensions. By employing the K-means algorithm, we divide the sub-matrices into three types: high crowd noise level, normal crowd

noise level, and low crowd noise level as Fig. 3.9.

3) For the matrices tagged with high crowd noise level, we take actions to reduce the noise: since the matrices M_B and M_F made by Bluetooth and WiFi RSSI detections are interfered by human bodies, these data are will not be adopted. Only dead reckoning is acceptable for the matrices with high crowd noise level. Matrices with normal or low level crowd noise still use the three key techniques together.

3.1.3 Extend Rooms to a Building

3.1.3.1 Anchor Points Analysis

In addition to the rooms and hallways of a building, there are some special places, such as entrances, elevators, and stairs. When people walk in a building, these places should be recognized as the initial positions of dead reckoning and the joints of rooms/hallways. These places are named Anchor Points. Our existing approach can detect the layout of a room and hallway, nevertheless, it has not recognized Anchor Points.

In order to detect the Anchor Points in indoor buildings, first, we employ the techniques found in CrowdInside [2] and note that the motions of users have their own acceleration ranges: 1) stationary: 0-5m/s^2; 2) elevators: 0-4.1m/s^2; 3) walking: 0-13.2m/s^2; 4) stairs: 0-20.2m/s^2. iFrame analyzes the types of Anchor Points via their acceleration signatures. Then, we adopt 1) correlation between the acceleration values on different axes and 2) variance of acceleration magnitude to further classify Anchor Points. According to the obtained accelerations and analysis patterns proposed in CrowdInside, we identify Anchor Points on our map approximately.

Considering the entrances of a building are the initial positions of users' motion traces, we not only need to recognize them but also need to obtain their locations. When a smartphone receives a GPS sample, the user can assume he/she is in an outdoor environment. When a user enters a building, the GPS samples will disappear. A common method of estimating the

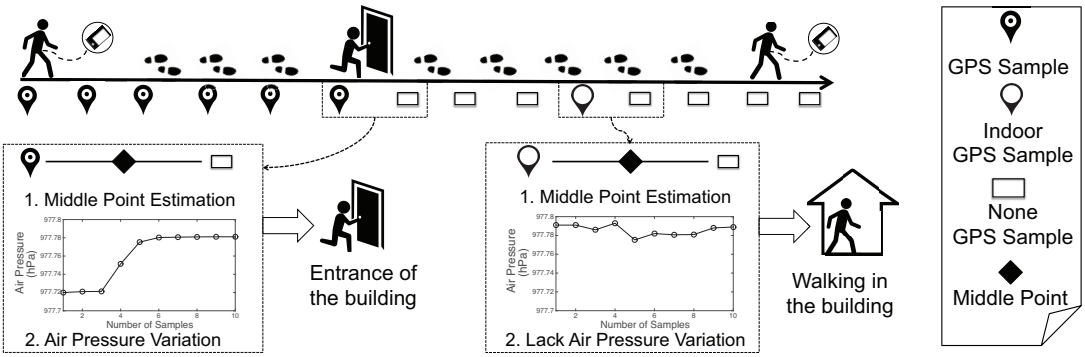


Figure 3.11 Leverage GPS information and barometer to detect the building entrances.

entrance positions is recording the position where the latest GPS signal was received and the position where the GPS disappeared, and then compute the middle point as the position of the entrance. Actually, while a user is walking in a building, e.g., when he/she is close to a window, he/she might receive GPS samples on his/her smartphone occasionally. These GPS samples are named indoor GPS samples (iGPS for short). Therefore, the middle point between the position of receiving the latest GPS and the position where the GPS disappeared may contain errors.

Since the different temperatures and layouts in indoor and outdoor environments, the values of air pressure between indoor and outdoor environments may be different. When a user of smartphone enters a building, the values of air pressure will vary on the barometer that is integrated on the smartphone. As Fig. 3.11 shows, we adopt this phenomenon to detect the entrance of the building. When we compute the middle point between the position where the user received the latest GPS and the position where the GPS disappeared, and there exist a variation in air pressure, the middle point can be recognized as the entrance of the building. Based upon our observation, the variation of air pressure should be greater than $0.03 Pha$. An additional explanation is necessary: this approach only can be applied for the places where indoor and outdoor environments have differential air pressures caused by temperature or other factors.

To identify stairs and elevators in a building, we leverage two types of information to

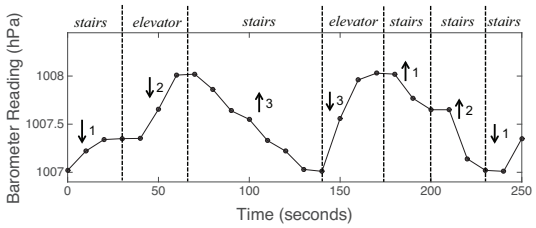


Figure 3.12 The air pressure recordings of a smartphone user goes up and down in a building.

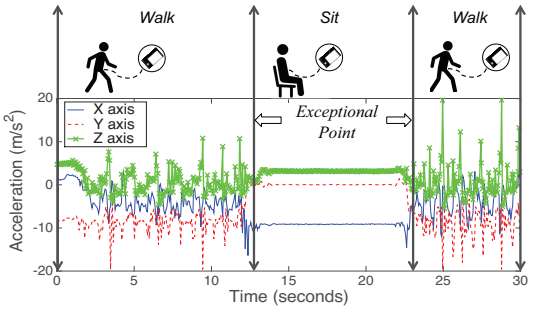


Figure 3.13 The accelerometer recordings of a smartphone user passed by a chair (smartphone in left hand).

recognize them: acceleration and air pressure. First, as the measurement in CrowdInside [2], when users of smartphones go on stairs and stand on elevators, the accelerations represent their own ranges. Second, when the users of iFrame cross levels in a building, the air pressure will change continuously. Since most modern smartphones integrate a barometer, the values of air pressure can be detected. As in Fig. 3.12, the variations of air pressure caused by changing levels in a building can be recorded by the barometer on smartphones. The numbers near the arrows refer to how many levels the user crossed. Therefore, when both of the two conditions are satisfied, we can identity the positions are elevators or stairs.

3.1.3.2 Hallway Assembling

In a building, hallways are separated by walls. Computer vision based approaches often use complex image algorithms to gather the layouts separated by walls and the walls themselves. iFrame solves this problem by: 1) users cannot cross the wall, therefore, dead reckoning detection will not set the related element in $\mathbb{M}$ as 0; 2) if a user in a room and a user out of

a room can detect each other, WiFi and Bluetooth detections cooperate together to mark the wall related elements as 1. More samples can rebuild the wall more clearly. This method does not need extra computation and can avoid image gathering.

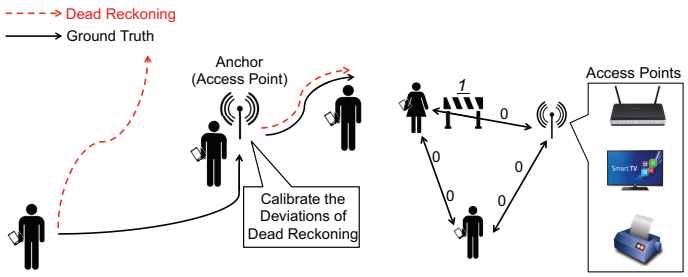
3.1.3.3 Exceptional Points

In real environments, there exists some grids that are occupied by objects but still can be passed by users. These points includes three features: 1) from the perspective of dead reckoning, it can be passed through; 2) for Bluetooth and WiFi detections, it can be detected as an obstruction; 3) in a certain time periods, as illustrated in Fig. 3.13, the values of accelerations keep a relatively stable state. The parameters that define the stable state are discussed in the evaluation section. If these three conditions can be satisfied concurrently, we recognize the grid as an "Exceptional Point" and mark the corresponding grid as "1".

For example, if a chair is in the grid, even if the grid can be described as "occupied" on the shadow map, the user can sit on the chair and leave the chair at any time. For a certain chair, we observe the detecting results by using the three proposed conditions. We tested the case 100 times. Our approach could detect the chair 92 times as an "Exceptional Point". Some other objects on the floorplan can be treated as "Exceptional Point". For example, a shopping cart, mobile cabinet (cabinet including wheels) satisfy the three conditions because of their movable features. Although the chair can be detected by the definition of "Exceptional Point", the "Exceptional Point" are not limited to chairs. The shopping carts and mobile cabinets (cabinet including wheels) also satisfy the three conditions because of their movable features.

3.1.4 Leverage Deployed Infrastructures to Enhance Shadow Map

Although iFrame does not depend on the pre-installed devices (such as WiFi Access Points and Bluetooth Beacons) in a building, iFrame is able to leverage these devices to improve



(a) Reduce the error of dead reckoning by using the anchor.

(b) Fixed device communicates with mobile devices.

Figure 3.14 Use pre-installed infrastructures to improve the indoor map.

dead reckoning. If the pre-installed wireless devices can share their positions, once a fixed device is close (within one grid) to a user of a smartphone, the user's computed position from dead reckoning will be replaced by the device's accurate position. The distance between the wireless device and the user's smartphone is computed by the prepared RSSI-distance relations. Considering the position of the installed wireless device is known and accurate, the device also can be treated as an Anchor Point in Fig. 3.14(a).

In addition, even if the pre-installed devices are stationary, they are able to communicate with mobile devices via Bluetooth and WiFi. As presented in Fig. 3.14(b), the wireless links between the pre-installed devices and mobile devices can describe the states of the grids on the shadow maps. Namely, although iFrame is not an infrastructure-based system, it is able to cooperate with the fixed infrastructures that are deployed in indoor environments.

3.1.5 Energy Saving

Most smartphone sensing approaches consume much energy. For example, the smartphones that use a camera sensor to generate street scenarios have to face the challenge of high energy consumption if the camera is always running. However, if the camera is turned off, important images may be lost. iFrame improves the energy issue by the follows aspects: First, for dead reckoning, considering the sampling frequencies are highly related to energy consumption, when the smartphone detects that the acceleration variation on any one of axises changes

more than $2m/s^2$ within 1 minute, we assume the smartphone is not stable. Namely, dead reckoning may include more errors and the sampling frequency of the smartphone will be reduced. Once the smartphone detects that the acceleration variation all the axises changes within $2m/s^2$ in 1 minute, the sampling frequency will increase. For Bluetooth and WiFi detection, because the proposed "interrupt mechanism" is more energy consuming than default device discovery, if the radio adapters can discovery any nearby device within 3 minutes, they will adopt the "interrupt mechanism" to detect devices. Otherwise, they will use the default duty cycle to detect devices.

3.1.6 Evaluation and Discussion

3.1.6.1 Experiment Setup

We prototype iFrame on Samsung Galaxy S5 smartphones and Google Nexus 7 tablets, which support various types of inertial sensors. The version of Android is 4.4. In the experiments and simulations, users leverage Bluetooth adapters, WiFi adapters, and accelerometers. The corresponding sampling frequencies of these components are 0.2-0.06Hz, 0.25-0.5Hz, and 5Hz respectively. We implement dead reckoning to generate a user's motion trajectory: 1) we calculate the movement direction and the motion distance in each segment and 2) merge computed segments in each time period. When the smartphone detects the acceleration variations on all the of axises change less than $2m/s^2$ within 1 minute, to save energy, the sampling frequency of accelerometer will adjust to 0.5HZ. Once the smartphone detects the acceleration variations on any axis change more than $2m/s^2$ within 1 minute, the sampling frequency will return 5Hz. After turning on these sensors, the users carry the mobile devices and walk freely in the indoor scenarios. Initially, we set the shadow value of each grid as 1. For the size of each grid, if the size is too large, the accuracy of the rebuilt map will be constrained. On the contrary, once the size of grid is too small, the complexity of computation will increase sharply. In our evaluation, we set the size of each grid as 0.5m×0.5m.

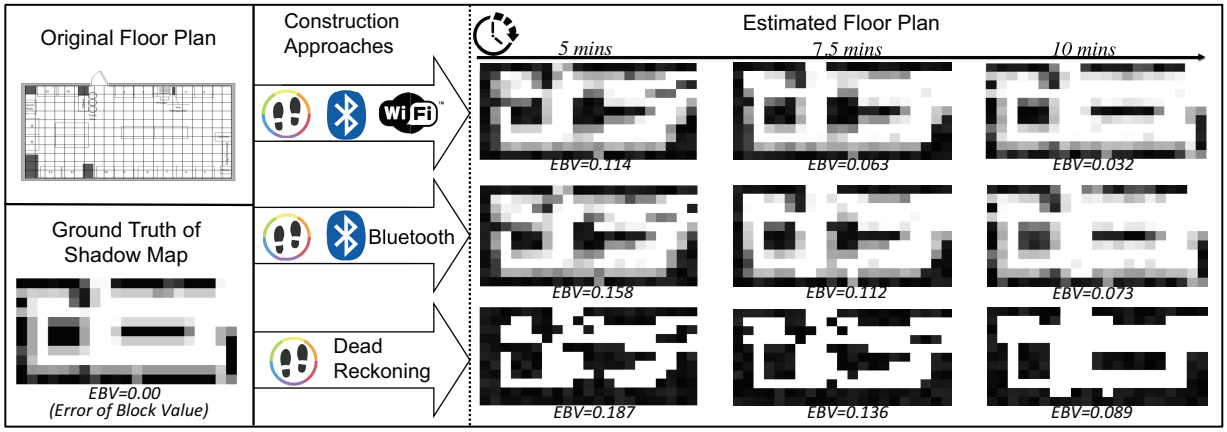


Figure 3.15 The case study of indoor floor plan reconstruction. Within the time increase, the generated shadow map is closer to the ground truth. The group including the three methods outperforms other two groups.

For Bluetooth and WiFi detections, the distance-RSSI relations for the smartphones are trained off-line and stored in hash tables. We estimate the layouts between each of the connected smartphones via the proposed algorithms. Empirically derived thresholds T_b= 3dBm, T_{w1}= 4dBm, T_{w2}= 4dBm are proposed in Algorithms 1 and 2. When we combine the three sensing approaches, the initial values of a, b and c are 1/3. We adopt Mean Value pattern to merge the matrices computed from all the mobile devices.

In our evaluation, we seek to answer these questions: 1) Does iFrame construct the indoor layout of a building successfully? 2) Can iFrame detect the change of layout of a room? 3) Can our matrix fusing mechanism improve the output results? 4) Can an increase in the number of users enhance the accuracy or speed up the rebuilding process? 5) Can iFrame cooperate with deployed wireless Access Points in indoor environments? 6) Are anchor points recognized successfully in the generated floor plan?

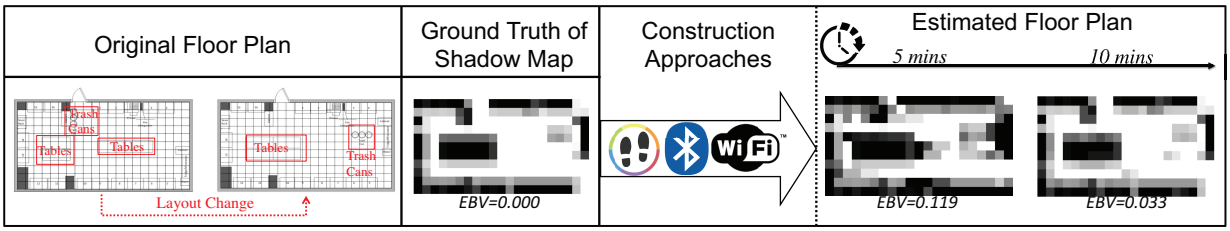


Figure 3.16 The case study of rebuilding the changed floor plan. After we changed the positions of trash cans and tables, iFrame still can build the accurate indoor shadow map within 5 to 10 minutes.

3.1.7 Indoor Environment Measurement

3.1.7.1 Room Measurement

We conducted an experiment in the eLANS Lab of Michigan State University. As shown in Fig. 3.15, we transform the floor plan of a room into a two dimension map described by a shadow based matrix. The gradient color (from black to white) in each grid represents the shadow state. Three users carry smartphones and walk freely in the room. The smartphones are put into the users' pockets. It is easier to detect the objects between wireless links. We only leverage the samples collected in the pockets to build a motion trace. To distinguish the relative position of smartphones, two types of sensors are used: embedded proximity (IR) and light sensors. By using the detection mechanism proposed by Yang [86], we are able to detect the smartphone in pocket or out of pocket.

By executing our approach for 10 minutes, iFrame constructs the indoor floor plan. With the time increasing, the estimated map reflects the ground truth better. For example, the generated shadow map in the 10th minute is more accurate than the shadow map in the 5th minute.

In order to verify our approach is able to represent the latest version of the indoor map, we changed the layout of the room. The proposed algorithm in iFrame ran continuously with unattended mode. Then, we conducted the same experiment in the changed scenario. Even if the trash cans and tables have been moved, as Fig. 3.16, the new generated map is still close to the ground truth.

Note that when people work or live in a room, there is little possibility to walk continuously. Therefore, we did an experiment that more closely resembles people's daily life. There were three users of iFrame in a room. They can walk, stop, sit, and chat in the room for 1 hour.

In this case, each user adopts Noom Walk [50] and S-Health [31] Pedometer applications on their smartphones, and record the number of steps they have walked. Fig. 3.10 depicts that even if there exists some differences for the step numbers counted by the two applications, within the increasing of the number of steps, the Errors of Block Values are reduced gradually. Then, we executed a similar experiment within 30 minutes, the experimental results are close to what we obtained in the 1 hour's group.

To analyze the effectiveness of each technique, we did the following experiments. First, we only used dead reckoning approach to build the map. Then, we added Bluetooth detection and did the experiment. In the end, we combined the three technologies together. As shown in Fig. 3.15 and Fig. 3.18(a), we conclude each technique can improve the accuracy of the map. By repeating the comparison 10 times, Fig. 3.18(b) provides the confidence interval for each approach and shows that our conclusion is not coincidental.

There are three participants in the above experiments to collect samples to construct the map. We observe two other control groups for 10 minutes: 1) One participant has iFrame. Since no other person can establish a connection to him/her, only the dead reckoning technique is available; 2) Five participants use iFrame, combining the three approaches together. We repeat the experiments 12 times. Fig. 18(c) shows that with an increasing number of users, namely, more samples of matrices computed by iFrame can boost the accuracy of map construction. Additionally, the groups with more users cost less time to achieve a low error rate.

In our measurement, since the users of iFrame cover the rooms fully, the indoor maps are generated completely. However, in certain cases, the users may not pass by some areas in a room. These areas (named "blind area") cannot be described by our sensing approaches. To reduce the "blind area" on the map, once iFrame cannot receive the data samples from

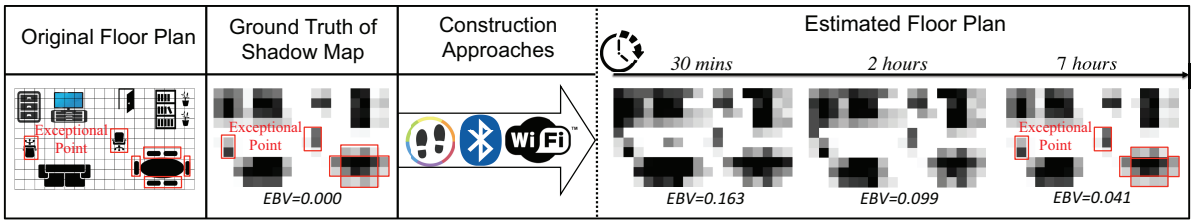
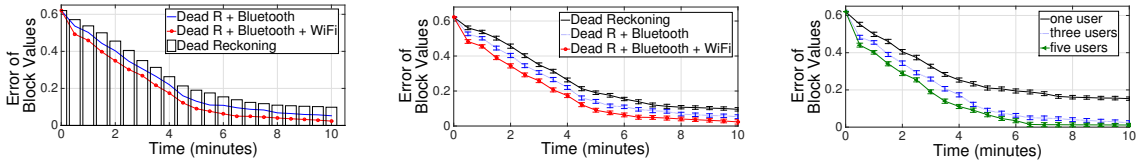


Figure 3.17 Floor plan case study for the rooms over a long time.



(a) One time measurement by three approaches.

(b) Repeated measurements by three approaches.

(c) Control Group for Different Number of users.

Figure 3.18 Experimental results for single room case study.

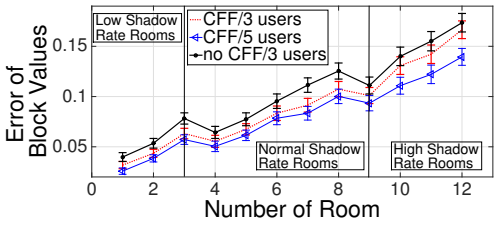


Figure 3.19 Results of the extended experiments.

certain areas for 1 hour, iFrame sends a message to users to suggest to visit the "blind area".

In contrast to letting volunteers walk frequently and cover the full space of the eLANS lab within 10 minutes, we did an experiment in a living room of apartment (the size is known), which is closer to a real world scenario. Three users of iFrame enter and leave the room freely. Our system collected the data from three iFrame users in one day (from 10AM to 5PM), as shown in Fig. 3.17, by applying the "blind area" messages, the shadow map can achieve a low Error of Block Value (0.041).

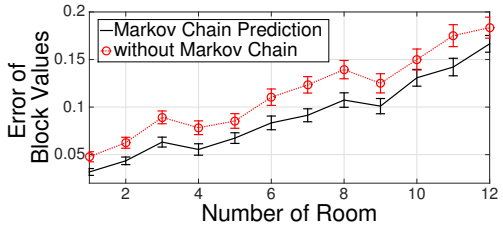


Figure 3.20 Markov chain prediction for multiple rooms.

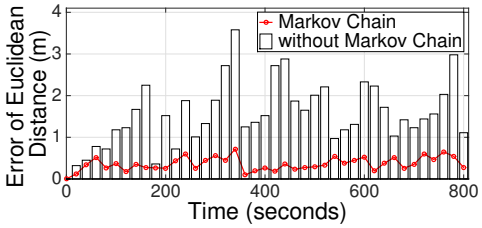


Figure 3.21 Markov chain prediction in one room.

Table 3.2 Different groups of a, b, and c values

	a	b	c
Low Shadow Rate	0.430	0.353	0.217
Normal Shadow Rate	0.412	0.324	0.264
High Shadow Rate	0.345	0.225	0.430

3.1.7.2 Building Measurement

In Fig. 3.22, we extend our approach to a building with multiple rooms and hallways. When iFrame builds the map of each single room, we also collect a, b, c values (Low/Normal/High shadow rate) for assigning weights to dead reckoning, Bluetooth and WiFi detections. The three groups are shown in Table 3.2. Once a user enters a room, after the first time period scanning, if the average shadow rate of a room is less than 0.1, it will choose a low shadow rate related to a, b, c values. If the average shadow rate is more than 0.25, it will use a high shadow rate related values. Other cases will adopt the normal shadow rate related values. As shown in Fig. 3.19, although all of these experimental scenarios stay at a low error levels, the rooms with lower shadow rates have less errors.

iFrame obtains improved results from combining the three types of output matrices. The selection of a, b, c is based on CFF. To verify the effectiveness of CFF, we did the following

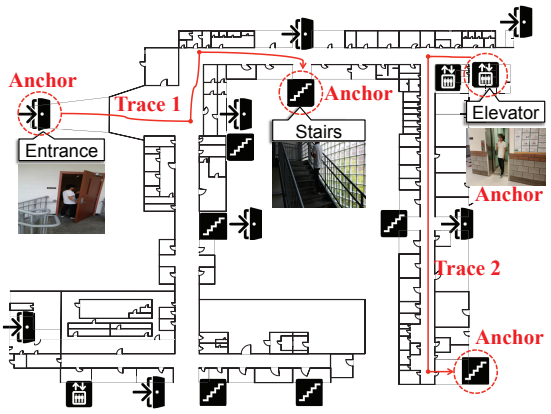


Figure 3.22 Extend our evaluation from single room to the whole building via recognizing the anchor points.

comparisons: under the same condition as the previous evaluation, we set the values of a, b and c as $1/3$, rather than selecting the values of a, b and c by the proposed method. Fig. 3.19 shows the Errors of Block Values of 12 places without changing the initial values of a, b, c and the control group using CFF. We can make two interesting observations from the comparisons. First, the evaluation results confirm CFF enhances the accuracy of map construction effectively. Another observation is that the group with five users has better performance than the group having three users.

Next, we study the benefit of the Markov chain and Average Speed predictions, which aim to correct the deviations from dead reckoning. We keep the above experiment conditions, but delete the Markov chain prediction mechanism and conduct iFrame 10 times in 12 rooms. As represented in Fig. 3.20, we observe that the group with the Markov chain prediction outperforms the other one. Then, we only concentrate on room 1 and run iFrame for 800 seconds. Two groups of results are illustrated in Fig. 3.21. With the time increase, the group with Markov Chain prediction has less error of deviation distance. Error of Euclidean Distance is the distance (in meters) between the ground truth and the estimated position. Besides, we concentrate on how ASM improves the dead reckoning. We curve fit the historic average speeds and predict the average speed in the current time period. Each time period is 30 seconds. As presented in Fig. 3.23, we set $5m/s$, $2m/s$, $0.5m/s$, and $0.1m/s$ as the error

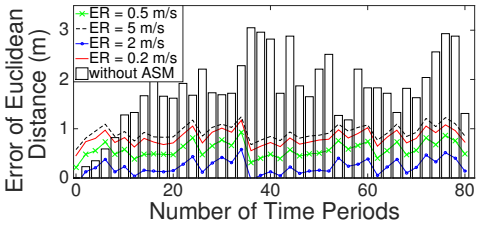


Figure 3.23 ASM comparison with different thresholds.

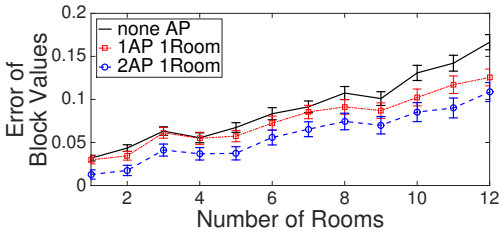


Figure 3.24 Experimental results of using fixed AP.

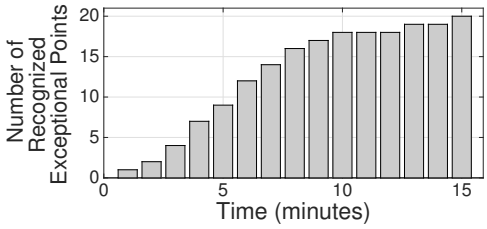


Figure 3.25 Number of recognized Exceptional Points.

ranges of predicted average speeds and compare these control groups. We conclude 1) AAS improves the accuracies of dead reckoning and 2) $2m/s$ is the most suitable error range of AAS.

We select the Mean Value mechanism to merging sensing data from different users. When we use Maximum Space and Minimum Space organizations to collecting data, the average Errors of Block Values in the building are reduced by 0.012 and 0.009.

The Anchor Points (e.g., entrances, stairs, and elevators) were identified in our system as shown in Fig. 3.22. Table 3.3 lists the recognition results for these Anchor Points. Only one stair and one entrance on our map were mis-classified.

Table 3.3 Confusion matrix for classifying different anchor positions

	Elevator	Stairs	Entrance	Other	FP	FN
Elevator	9	0	0	0	0%	0%
Stairs	0	20	0	1	0%	4.8%
Entrance	0	0	20	1	0%	4.8%
Other	0	0	0	40	5%	0%

The above evaluations do not contain fixed wireless Access Points (AP), such as wireless routers and bluetooth beacons. To validate iFrame is able to work with fixed access points, we add Access Points in the experimental scenario as in Fig. 3.24. In the first group, each room does not include any AP. In the second group, there is one fixed AP in each room. There are 2 AP in each room for the third group. As presented in Fig. 3.24, we show 1) the accuracies of reconstructed map can be improved by adding fixed AP and 2) the accuracies will increase by using more APs.

We focus on the recognition of Exceptional Points. Within 10 seconds, if the variations of accelerations on each axis are within certain thresholds, the state is defined as "stable state". Based upon the above experimental scenario, we deploy 20 chairs as the Exceptional Points. Once a user of iFrame passes by a chair, if the "stable state" can be detected by our system, the chair will be distinguished as an Exceptional Point. We tested different threshold of defining the "stable state" ($0.1m/s^2$, $1m/s^2$, and $10m/s^2$). When we choose $1m/s^2$ as the variation range of accelerations on each axis, the results are more accurate. Fig. 3.25 illustrates that three users of iFrame walk freely and ran our approach within 15 minutes. All the Exceptional Points on the map are recognized gradually. Although there is only one false positive case, it is the empty grid that is close to a chair that is distinguished as a real chair.

In the end, we discuss the real-time features of existing approaches. For the above experiments, the Bluetooth and WiFi adapters resume when the whole discovery round finishes. By keeping the same experimental conditions, we modified the original detection models by interrupting the duty cycle of device discovering, once the program obtains a certain number of devices, the serving round of device discovery will resume immediately. As shown in Table. 3.3, we compare the control groups with different number of detected

Table 3.4 Different groups of a, b, and c values

	a	b	c
Low Shadow Rate	0.430	0.353	0.217
Normal Shadow Rate	0.412	0.324	0.264
High Shadow Rate	0.345	0.225	0.430

Table 3.5 Control groups of using different number of detected devices

EBV	2 mins	4 mins	6 mins	10 mins
One Device	0.409	0.233	0.126	0.031
Three Devices	0.356	0.185	0.090	0.015
Five Devices	0.365	0.189	0.093	0.017
Finished	0.376	0.192	0.096	0.019

devices in a partial list. When we set three as the number of devices in the detection list, we obtain the best results.

3.1.7.3 Discussion

In contrast to other map construction approaches, iFrame is a light-weight and low complexity application.

Users of iFrame upload their accelerations and received RSSI values from cooperating with other devices periodically. The prepared distance-RSSI hash maps stored on the cloud server transfer the received RSSI values to distances continuously. Aiming to reduce computational burden on each smartphone, we deploy four critical tasks on the server instead of running it on smartphones: i) Forming the initial shadow map for a single smartphone, ii) Combine the matrices collected from various devices, iii) Refine the shadow map via Curve Fit Fusion (CFF), and iv) Filter the temporary shadow to enhance the accuracy of the indoor map.

Although iFrame deployed tasks to a cloud, the cloud server does not need to process complex images or simulate people's micro-activities, such as shaking hands. In this subsection, we analyze the computational complexity of each corresponding task: i) iFrame adopts formula (3.2) and Markov chain prediction to generate the motion trace. Since the complexity of formula (3.2) is $O(n^2)$, the complexity of Markov chain prediction is $O(n^3)$ (the transfer matrix is $O(n^2)$, after the k steps' transitions, the complexity of Markov chain prediction is $O(k \times n^2)$). ii) We adopt the Mean Value model to merge the matrices built by different

devices, the complexity of the Mean Value model is $O(n^2)$. iii) Solving the equation of CFF to compute the B value for each grid on the shadow map, the complexity of the CFF is $O(n^2)$. iv) Conducting the crowd noise filter for each sub matrix, because the filter relies on k-means clustering, the complexity of this approach is $O(n^{(di \times u+1)} \times \log n)$ (u is the number of cluster, di is dimension of each point to be clustered, n refers to the number of rooms to cluster). When we apply iFrame, the number of u, di are both set to three. Considering the number of rooms in iFrame is not beyond 500 for most of buildings, iFrame can manage to accurately construct the indoor map with such computational complexity.

Based on our experimental conditions, our experiments are conducted from 5 minutes to 7 hours. The number of users are less than 6. For short-term data collection, users may not cover some corners on the floorplan. This phenomenon may cause some errors. In the future, we can prolong the time period and let more of experiments. Also, more users can participate the procedure of indoor map construction. If users in a building can share the motion data of their daily routines, using crowdsourcing, the additional data samples could improve the accuracy of indoor shadow map.

3.1.8 Conclusion

We introduce a light-weight and high-speed indoor map construction approach called iFrame. After abstracting the unexplored map as a matrix, and by combining dead reckoning and RSSI detection techniques, iFrame judges whether the subareas in an indoor environment are empty or not. Each of proposed technologies compensates the shortcomings of others by adopting a matrix fusing mechanism. Our approach selects proper parameters for merging data automatically and yields a clear shadow map for each room within 5-10 minutes.

CHAPTER 4

IMPROVING INDOOR LOCALIZATION BY PROFILING OUTDOOR MOVEMENT ON SMARTPHONES

4.1 Introduction

Indoor localization is a fundamental service for various location based applications. Despite the extensive research and development of indoor positioning systems [41, 55, 51, 87, 43, 91, 4], localization service is not yet pervasive indoors. Since the capabilities of smartphones have become more powerful, many researchers use smartphones to locate people. Apart from the traditional device-based and device-free indoor localization approaches, smartphone-based approaches capture people's motions and traces by analyzing the acceleration, light, sound and other signals [71, 3, 90, 9, 75, 39, 62].

Although inertial sensing on smartphones can capture people's movement via the sensing data, there are some shortcomings: the sensing information, such as the 3-D acceleration from a smartphone does not always reflect features of a person's movements; the data training task is difficult: the size of data is small for statistical location accuracy and the learning algorithm is significantly complex for a smartphone. Based on this point of view, we ask the question: *Can we enhance smartphone users' capabilities to locate themselves accurately without complex indoor training and an extra, perhaps expensive, infrastructure?*

In this chapter, we propose iLoom (indoor Localization through transferring learning of outdoor motion), an accurate and low-cost indoor localization system that integrates an off-the-shelf dead reckoning, GPS information, and a transfer learning mechanism. Our idea is inspired by the observation that, when users walk indoors or outdoors, some features of their walking patterns, such as the average speed and acceleration are not greatly affected by the different environments. Hence, we use outdoor walking behaviors to assist users' indoor

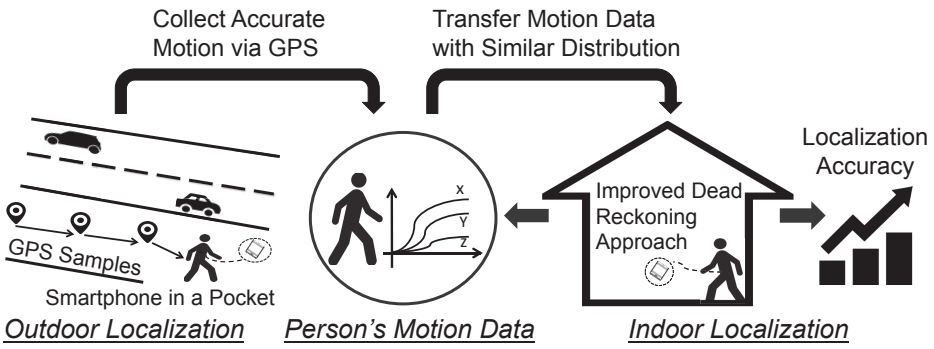


Figure 4.1 iLoom in Action: employ outdoor data to improve indoor positioning

localization.

Initially, iLoom provides a sensing service on a smartphone that detects whether the smartphone is indoors or outdoors. Then, iLoom uses a dead reckoning approach [71]. An Acceleration Range Box is introduced to filter the accelerations that do not represent the user's movement. To determine the range of the Acceleration Range Box, iLoom collects the average speed and acceleration from the indoor and outdoor environments. Since the outdoor motion data using GPS is more accurate than movement determined by accelerometer data, iLoom not only uses indoor datasets but also uses the outdoor GPS datasets. In the outdoor dataset, we employ Transfer Learning [52] to select the parts of accelerations for which people's outdoor movement behaviors are similar to indoor motions and add the chosen outdoor data to the indoor datasets for boosting the effectiveness of the Acceleration Range Box.

Three additional techniques for using outdoor/indoor information are proposed to enhance the original dead reckoning method: iLoom adopts a pedometer to construct other types of Acceleration Range Boxes that reduce the errors of indoor localization; indoor GPS exception cases are used to decrease deviations; since people's average speeds of movement typically do not change sharply, we eliminate some incorrect accelerations by average speed prediction.

We prototype iLoom and conduct a set of experiments in indoor and outdoor scenarios. Fifteen volunteers' cases have been studied. The evaluation results demonstrate iLoom

enhances the original dead reckoning approach effectively. The error of indoor localization is within 0.35 meter. Also, iLoom does not request users to do special off-line training. By opening iLoom and the GPS option for daily walking, iLoom can estimate indoor position more accurately.

In summary, we make the following contributions:

- We employ outdoor GPS information and other sensing data obtained from smartphones to detect whether the smartphone is indoors or outdoors.
- While many researchers have used dead reckoning as a means to specify a user's position, to the best of our knowledge, iLoom is the first of its kind to transfer the outdoor motion information to the indoor dataset for boosting indoor localization automatically.
- Indoor GPS Exception, Pedometer Measurements, and Average Speed filter are implemented to assist the dead reckoning method.

4.2 System Design

4.2.1 System Overview

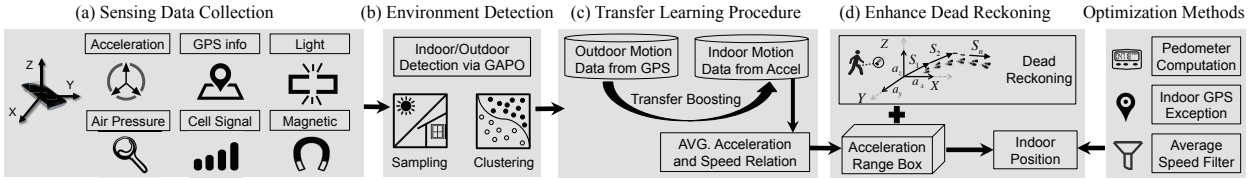


Figure 4.2 System architecture of iLoom.

Fig. 4.2 presents the system architecture of iLoom. iLoom has four steps: a) leveraging the inertial sensors on a smartphone to obtain the acceleration, GPS, air pressure, cell signal, light and magnetic information; b) using the acquired sensing data, iLoom detects the indoor/outdoor environments with high accuracy by applying *k-means* clustering algorithm;

c) iLoom proposes the Acceleration Range Box, a range of accelerations in different directions, to filter the incorrect accelerations that lead to the errors in the dead reckoning. In order to characterize the Acceleration Range Box, we construct a relation from the user's average speed to the average acceleration in each time period. Taking the average speed as the bridge, iLoom chooses the transfer learning approach to transfer the worthwhile outdoor GPS information to the dataset that stores acceleration samples that were received indoors. d) by the Average Acceleration Range Box constructed by transfer learning and other optimization technologies, iLoom calibrates the errors of dead reckoning to archive accurate indoor localization results.

Our system involves three other approaches to assist the indoor localization: 1) iLoom adopts the third-party pedometer to modify and boost the Acceleration Range Box; 2) although a user walks indoor, he/she may receive the GPS signal occasionally. Such GPS samples cannot be used for indoor positioning and may cause a false positive when a localization system detects the user is indoors or outdoors. In iLoom, we design an approach that not only avoids such mistakes but also improves indoor localization accuracy; 3) since the user's average speed will not likely vary sharply in a short time slot, we define a mechanism that filters the average speeds that change suddenly.

4.2.2 Indoor and Outdoor Detection

Before transferring the useful motion data from outdoors to indoors, we need to identify the samples obtained from the smartphones that belong to either the indoor or outdoor environments. An intuitive detection scheme estimates the positions via GPS. When the smartphone receives a GPS sample, the user can assume he/she is in an outdoor environment. In reality, while a user is walking in a building, e.g., when he/she is close to a window, he/she might receive GPS samples on his/her smartphone occasionally. However, these samples do not represent the user when outdoors.

To tackle this problem, IODetector [95] adopts three types of information on the smartphones: light intensity, cell signal strength, and magnetic sensor values. Even if each of them

cannot determine the environment, IODetector aggregates them and provides the solution. Via IODetector, researchers in the University of Edinburgh constructively introduced a semi-supervised learning model to analyze the indoor/outdoor location of smartphones [61]. They used more than three types of sensors on smartphones to collect physical signals. By applying the semi-supervised learning model, the accuracy of IODetector increases to 92.5%.

In this section, we introduce a novel approach, GAPO. By leveraging the GPS, Air Pressure, and Other cyber-physical information on the smartphones (light intensity, cell signal strength, magnetic sensing values), we distinguish the indoor/outdoor context for the smartphones.

Apart from the two above approaches, the proposed purpose of iLoom is to transfer outdoor GPS information to improve indoor positioning. Hence, GPS samples can be borrowed to detect environments. Considering both the current and historical information, a parameter tsi ($t$ime $s$equence $i$ndex of GPS) is defined as formula (4.1):

$$tsi = (\sum_{i=1}^{t} \lambda \times 2^i)/(\sum_{i=1}^{t} 2^i) \tag{4.1}$$

where t is the number of time periods, i refers to the time period. λ can be set as 0 or 1 (if the smartphone gets the sample in time period i, λ equals to 1, otherwise, it is set as 0). For tsi, the obtained GPS samples that are closer to current time period will be assigned more weight.

Additionally, modern smartphones include barometers. Highly accurate air pressure can be easily accessed. The accuracy of the barometer, such as the barometers on the Samsung Galaxy smartphones, can achieve within 0.1hPa. Although air pressure is determined by many factors, the differences of temperatures in indoor and outdoor scenarios often cause the variations of air pressure. Therefore, we add air pressure as a feature for distinguishing indoor and outdoor environments. After leveraging K-means algorithms, GAPO categorizes these samples into indoor and outdoor datasets.

We conducted a preliminary observation to explore GAPO: one user of a smartphone

Table 4.1 Confusion matrix for the samples representing the indoor/outdoor detection results.

	Indoor (ground truth)	Outdoor (ground truth)
Indoor (estimate)	N_{ii}	N_{io}
Outdoor (estimate)	N_{oi}	N_{oo}

Table 4.2 Successful rates of distinguishing indoor/outdoor environments in different scenarios.

	GPS Only	IODetector	GAPO
Sports Center	82.50%	78.50%	94.00%
Laboratory Building	77.00%	70.50%	92.50%

walks freely, receives 1000 samples from outdoor/indoor environments, and conducts GAPO to detect the environments. We compare the estimated indoor/outdoor results with the ground truth. Table 4.1 is a confusion matrix for representing the detection results. N in Table 4.1 denotes the number of samples. The metric P_e in formula (4.2) refers to the successful rate of estimating the indoor/outdoor environment. As shown in Table 4.2, by comparing the P_e values within other approaches in two different buildings, GAPO archives better performance.

$$P_e = (N_{ii} + N_{oo})/(N_{ii} + N_{io} + N_{oi} + N_{oo}) \tag{4.2}$$

4.2.3 Dead Reckoning is Not Enough

Dead reckoning has been widely used in indoor localization, especially for the smartphone based approaches [71]. After receiving the acceleration values on smartphones periodically, the motion distance of a mobile device in time period n is generated by formulas (4.3) and (4.4). The parameters a_x, a_y, a_z are the projections of acceleration $\vec{a}$ on the x, y, and z axes. $\vec{D_n}$ is the movement distance in the current period. v_{n-1} and a_{n-1} are the velocity and acceleration from the previous time period. t_n refers to the time length of the current period. A smartphone user can obtain his/her motion trace by calculating movement distance in each segment continuously.

$$\vec{a} = (a_x, a_y, a_z - g) \tag{4.3}$$

$$\vec{D}_n = \frac{1}{2}\vec{a}_{n-1}t_n^2 + \vec{v}_{n-1}t_n \tag{4.4}$$

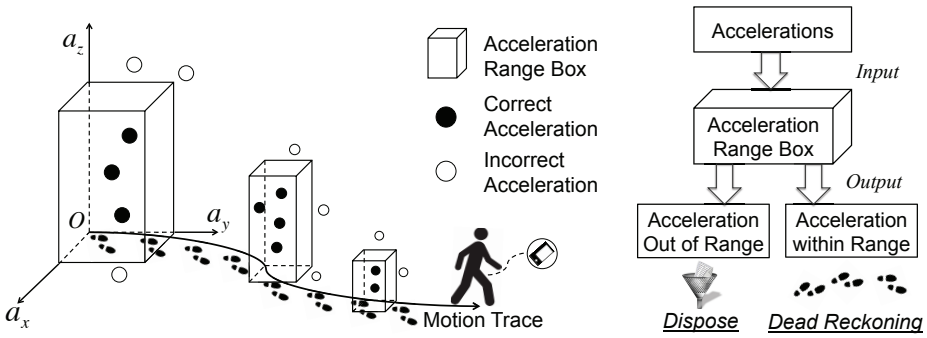
Although dead reckoning is easy to implement, a major difficulty in dead reckoning is that a smartphone only records its own accelerations rather than the accelerations of the human body's motion. In practice, when users of smartphones collect their motion data via smartphones, some cases often occur, such as giving a phone call to a friend, sending messages via typing on the screen, and swing the hands holding the smartphones. These behaviors incur serious deviations from a person's walking pattern. Moreover, such errors grows with time because the next motion segment is calculated from the current one with inaccuracy. Due to these reasons, the dead reckoning trajectories are accurate in the beginning, but diverge from the ground truth over time.

4.2.4 Initial Noise Filtering

When we employ dead reckoning as a means to locate people, it is necessary to filter obtained accelerations that cause serious errors. In iLoom, even thought we do not detect the place of a smartphone (in a pocket, on a user's hand, near an ear of user) and recognize human's activities in detail, we still set basic constraints for collected accelerations. Because the reasonable range of human bodies' motions is within 0-5m/s^2 on x, y and z axes [37], we preliminarily eliminate the acceleration beyond the range while collecting the data from accelerometers.

4.2.5 Acceleration Range Box

Every user has his/her own motion features. For example, when people walk regularly (not considering jumping, running, and other special movements), the values of acceleration and average speed on the x, y, and z directions should be in certain ranges. Inspired by this point, we propose a technique to enhance dead reckoning: if we can estimate the maximum accelerations on x, y, and z directions, they can be abstracted as the three sides of a cuboid.



(a) Motion trace generation. (b) Workflow.

Figure 4.3 Acceleration Range Box. The value on each axis represents the average maximal acceleration in each direction. The acceleration samples obtained from a smartphone that is out of the Acceleration Range Box will be disposed.

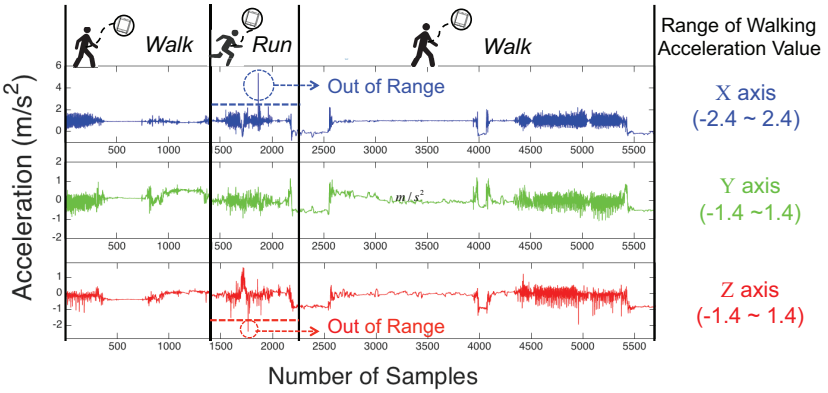


Figure 4.4 A user's abnormal accelerations can be detected when he is not walking by using the Acceleration Range Box.

The cuboid is called Acceleration Range Box (a_{rb}). As Fig. 4.3 shown, when we adopt dead reckoning to generate the a user's motion trace, if the acceleration value is out of a_{rb}, we assume the value is invalid. We will utilize the acceleration in the previous period to replace the invalid acceleration. Fig. 4.4 represents a user's accelerations on x axis. Once the values of accelerations is beyond the boundary of a_{rb}, the user's motion cannot be recognized as *walking*.

Since we introduced the Acceleration Range Box (a_{rb}), a big challenge is how to build an efficient a_{rb} for each user. A brute-force approach is 1) recording all the accelerations on x, y, and z axes of a smartphone; 2) finding the maximum value in each direction as the side of a

box. However, such a_{rb} cannot reflect people's motion feature and some invalid acceleration values will be accepted.

In iLoom, although we do not categorize people's movements in detail, a practical metric to classify people's motions is introduced. The metric is Average Speed ($\overline{v}$) of people's movement in a certain time period. For a certain Range of Speed, we assume there is a specific a_{rb} for a user. For example, a user's Average Speed in 1 minute is $1.4m/s$, the related a_{rb} is $2.4m/s^2$ on x axis, $1.4m/s^2$ on y axis, and $1.4m/s^2$ on z axis.

We then construct an a_{rb} for each certain $\overline{v}$. In an indoor environment, as in formulas (4.3) and (4.4), we can capture the $\overline{v}$ in each segment through dividing the moving distance by time t. For the outdoor localization, the $\overline{v}$ in each time period can be obtained via GPS. For each time length of t, we record the maximum values of the acceleration on the three directions and form the a_{rb}. If there are more than one a_{rb} in a speed range, we will compute the average value of maximal acceleration on each direction and make use of it as the side of a cuboid. The newly generated box is named Average Acceleration Range Box ($\overline{a_{rb}}$).

For a certain Range of Speed ($R_{\overline{v}}$), it also has its corresponding $\overline{a_{rb}}$. Therefore, we can create the relation between Range of Speed and Average Acceleration Range Box. This relation in outdoor environments is named $\mathbb{R}_o(R_{\overline{v}}, \overline{a_{rb}})$ ($\mathbb{R}_o$ for short), and it is named $\mathbb{R}_i(R_{\overline{v}}, \overline{a_{rb}})$ in indoor environments ($\mathbb{R}_i$ for short).

4.2.6 Can Outdoor Localization Help Indoor Localization

For dead reckoning based indoor localization, because the accelerations obtained from the accelerometer may not be consistent with the human body's motion, the indoor $\overline{v}$ might be computed incorrectly. However, since GPS has a relatively accurate performance in outdoor environments, the outdoor $\overline{v}$ dose not have such a problem. The corresponding $\mathbb{R}_o$ is often more accurate than $\mathbb{R}_i$.

Therefore, we propose an audacious conjecture: *could we transfer the useful data from $\mathbb{R}_o$ to $\mathbb{R}_i$, and build a better relation to improve the dead reckoning approach?*

In fact, for one person, his/her walking style changes little whenever he/she is indoor or outdoor. For every speed range of each person, there is a particular distribution, e.g, a male adult whose age is 30, his speed range is mainly distributed from $1.2m/s$ to $1.7m/s$ [**?**]. If we select the useful and highly accurate samples from $\mathbb{R}_o$ and combine them to $\mathbb{R}_i$, it is probable to build a larger and more accurate relation. For each speed range, we will re-compute the corresponding $\overline{a_{rb}}$. If the new $\overline{a_{rb}}$ is more suitable for dead reckoning, it can boost the localization results.

4.2.7 Transfer Learning from Outdoor to Indoor

In this subsection, we start to transfer the worthwhile information from the outdoor motion dataset to the indoor motion dataset. In this paper, we employ *Transfer Learning* [52]. It stores knowledge obtained from solving one problem and use it to a similar problem.

In iLoom, we study the useful instances from $\mathbb{R}_o$ and apply them on $\mathbb{R}_i$, which are different but similar to $\mathbb{R}_o$. In each instance, it consists two features: Average Speed ($\overline{v}$) and Average Acceleration Range Box ($\overline{a_{rb}}$).

When we apply transfer learning, the main challenge of transition is: for a certain user, even if his/her walking behavior is similar whenever he/she is indoor or outdoor, there is a small amount of differences in the speed distribution between $\mathbb{R}_i$ and $\mathbb{R}_o$. On the perspective of $\mathbb{R}_i$, we need to 1) choose the instances that keep the same-distributions as $\mathbb{R}_i$ from $\mathbb{R}_o$, and 2) transfer these instances to $\mathbb{R}_i$.

First, we define S and T to represent the test dataset (indoor information) and the training dataset (outdoor information). SVM [26] is the default classifier. We select part of the labeled training data having the similar distribution as the test data (indoor information) to build a better classifier. These data are named same-distribution training data (T_s), the size of T_s is m. The training data, whose distribution is different from the test data, are named diff-distribution training data (T_d), the size of T_d is n.

X and Y are two instance spaces. X_s and X_d represent same-distribution instance space

Table 4.3 Main notations in the design of iLoom

Symbols	Definition
$\overline{v}$, $R_{\overline{v}}$	Average Speed, Range of Average Speed
a_{rb}, $\overline{a_{rb}}$	Acceleration Range Box, Average Acceleration Range Box
$\mathbb{R}_i(R_{\overline{v}}, \overline{a_{rb}})$	Relation between $R_{\overline{v}}$ and $\overline{a_{rb}}$ in $\underline{i}$ndoor environment
$\mathbb{R}_o(R_{\overline{v}}, \overline{a_{rb}})$	Relation between $R_{\overline{v}}$ and $\overline{a_{rb}}$ in $\underline{o}$utdoor environment
$\mathbb{R}_c(R_{\overline{v}}, \overline{a_{rb}})$	Relation between $R_{\overline{v}}$ and $\overline{a_{rb}}$ in $\underline{combined}$ dataset
X_s,X_d	same-distribution / different-distribution instance space
S_d,S_s	diff-distribution / same-distribution as sample space
L	set of category labels, L= $(0, 1)$
mf	boolean mapping function from X to Y
S,T	test dataset and training dataset
k	size of the test set S that is unlabeled
T_s, T_d	same-distribution / diff-distribution training dataset
n, m	size of T_s and T_d
w	weight vector for dataset
h_t	hypothesis from $X \to Y$
ϵ_t	error of h_t on same-distribution training dataset
p	probability of instances transferred from T to $\mathbb{R}_i(R_{\overline{v}}, \overline{a_{rb}})$

and different-distribution instance space. $Y = \{0, 1\}$ is the set of category labels. Concept mf is a boolean mapping function from X to Y, and let $X = X_s \cup X_d$. $mf(x)$ is the return value of label for the data instance/sample x.

From $\mathbb{R}_o$, we can obtain 1) inadequate labeled same-distribution training data T_s, 2) diff-distribution training data T_d, and 3) some unlabeled test data S.

Our task is to train a classifier $mf' : X \to Y$ that minimizes the prediction error on the unlabeled dataset S. In the proposed approach, the *prediction* operation is defined as: if we use the $\overline{a_{rb}}$ and dead reckoning to localize people, and if the deviation distance is within $1m$, the prediction is successful; otherwise, the prediction is a failure.

To achieve this goal, we adopt the TrAdaBoost approach [10]: for diff-distribution training instances, when they are wrongly predicted due to the distribution modified by the learned model, these instances could be recognized as the most dissimilar instances to the same-distribution instances. TrAdaBoost provides a mechanism to decrease the weights of these instances in order to weaken their impacts.

Algorithm 3 illustrates the procedure of TrAdaBoost. In each iteration round, once a diff-distribution training instance in $\mathbb{R}_o$ is not predicted successfully, the instance may conflict with the same-distribution training data. Hence, it is necessary to reduce its training weight w to decrease its effect. We multiply the weight by the factor $\beta^{|h_t(x_i) - mf(x_i)|}$, which is in the

Algorithm 3 Algorithm of Transfer Boosting

Input:

T, S, $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$, $\mathbb{R}_o(\overline{v}, \overline{a_{rb}})$

Output:

The updated $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$ including the transferred instances

1: Set weight vector $w^1 \leftarrow (w_1^1, ..., w_{n+m}^1)$.
2: **while** $N>0$ **do**
3: $N--$;
4: Let $p^t \leftarrow w^t/(\sum_{i=1}^{n+m} w_i^t)$.
5: Call SVM/SVMt;
6: (T with distribution p^t over T) $\cup$ S.
7: Get back to hypothesis: $h_t : X \rightarrow Y$.
8: Estimate the error of h_t on T_s:
$\epsilon_t \leftarrow \sum_{i=n+1}^{n+m} \frac{w_i^t \times |h_t(x_i) - mf(x_i)|}{\sum_{i=n+1}^{n+m} w_i^t}$
9: Set $\beta_t \leftarrow \epsilon_t/(1-\epsilon_t)$, ($\epsilon_t < 0.5$) and $\beta = 1/(1+\sqrt{2lnn/N})$.
10: Update the new weight vector:
$$w_i^{t+1} \leftarrow \begin{cases} w_i^t \beta_t^{|h_t(x_i) - mf(x_i)|}, 1 \le i \le n \\ w_i^t \beta_t^{|-h_t(x_i) - mf(x_i)|}, n+1 \le i \le n+m \end{cases}.$$
11: sort instances in $\mathbb{R}_o(\overline{v}, \overline{a_{rb}})$ by the latest w_i^{t+1}
12: **end while**
13: transfer p% instances with higher weights in T to $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$

range of (0,1]. In the next round, the misclassified diff-distribution training instances will have less effect for the transfer learning procedure than the current round. By times of iterations, the diff-distribution training instances in $\mathbb{R}_o$ that are proximate to the same-distribution instances will have higher training weights, whereas the diff-distribution training instances that are dissimilar to the same-distribution ones will have lower weights. Thus, the instances having large training weights in $\mathbb{R}_o$ can help the learning algorithm to train better classifiers.

After executing the transfer boosting, we only transfer the $p\%$ instances with higher weights to assist the classification approach. The probability of transferred instances in $\mathbb{R}_o$ is determined by the experience. The theoretical analysis and mathematical proof of transfer boosting algorithm are in the literature [10].

Via transferring the same-distribution instances from $\mathbb{R}_o$ to $\mathbb{R}_i$, our system obtains a new relation $\mathbb{R}_c(R_{\overline{v}}, \overline{a_{rb}})$ ($\mathbb{R}_c$ for short) in the combined dataset. Thus, a user can employ the

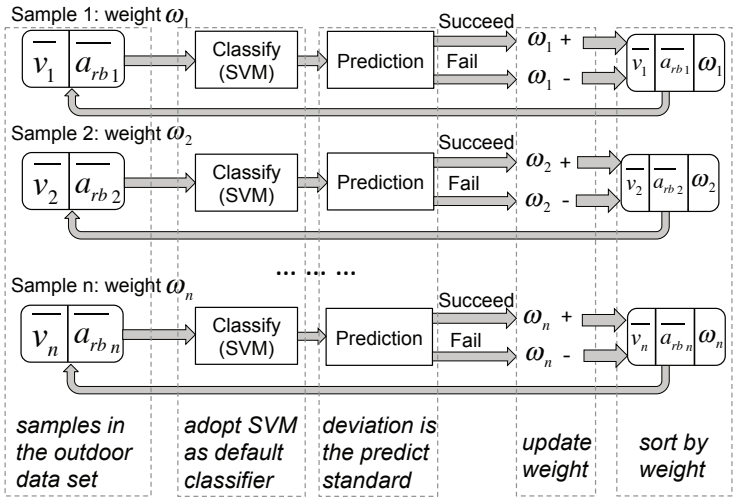


Figure 4.5 The procedure of transfer learning in iLoom.

constraint made by the newly generated $\overline{a_{rb}}$ to enhance dead reckoning localization.

4.2.8 Employing Pedometer to Improve Dead Reckoning

Most common smartphones can support and run a pedometer application. Samsung Galaxy Smartphones provide an off-the-shelf application named S Health to count the number of a person's walking steps. Some brands of wearable devices, such as Fitbit, Jawbone, etc, also contain electronic pedometers. The inaccuracy of current pedometer monitors has been shown to be around 9% [37].

In the procedure of transfer learning, the average speed ($\overline{v}$) is the bridge to connect the outdoor and the indoor information. In an indoor environment, the average speed not only can be computed by acceleration, but also can be obtained by leveraging the third-party pedometers. Based upon the two types of obtained average speeds, iLoom calculates two types of corresponding $\overline{a_{rb}}$. For each speed range, we update the Acceleration Range Box by averaging the values of the two $\overline{a_{rb}}$. Then, we employ the updated Acceleration Range Box to calibrate the dead reckoning.

Algorithm 4 illustrates how iLoom exploits pedometers to optimize the existing Accelera-

Algorithm 4 Pedometers Improve Acceleration Range Box

Input:

$\overline{v}_i$, L_i - average speed, length steps of the pedometer user i

n_p - number of pedometers

n_A - number of Acceleration Range Boxes

Output:

The updated $\overline{a_{rb}}$ for each sample

1: **for** i=0; i $< n_p$; i++ **do**
2: $L_i \leftarrow$ a$\times F_i$ + b (*or* $F_i \leftarrow$ a$\times L_i$ + b)
// step frequency F_i and step length L_i has a linear relation
3: Initialize: $\overline{v}_i \leftarrow F_i \times L_i$, j $\leftarrow n_A$, cnt $\leftarrow$ 0;
4: **while** j $>$ 0 **do**
5: **if** $|\overline{v}_i\text{-}\overline{v}_j| < \Delta SR$
// if the average speeds are in the same range, merge the different $\overline{a_{rb}}$. ΔSR is the threshold.
then
6: $\overline{a_{rb}}_i \leftarrow \overline{a_{rb}}_i + \overline{a_{rb}}_j$, cnt $\leftarrow$ cnt+1;
7: **end if**
8: // Caculate the average value of different a_{rb}
$\overline{a_{rb}}_j \leftarrow \overline{a_{rb}}_j$ / cnt , j $\leftarrow$ j-1;
9: **end while**
10: **end for**

tion Range Boxes. Li, et al. proposed that step frequency and step length has a linear relation [39], we conduct curve fitting for collected data and compute the factor values of a and b. Thus, if we obtain the step frequency of people by pedometers, we can estimate the step length of people. Also, once a user inputs his/her known step length, he/she could calculate his/her step frequency according to the linear relation. Based on these information, we can modify the $\overline{a_{rb}}$ via pedometers on the mobile devices by algorithm 10. The experimental performance of this approach will be displayed in the evaluation section.

4.2.9 Indoor GPS Exception

Anchor Points were applied in location service systems [2]. They are the positions in the environment with unique sensing signatures. Anchor Points can be used to reset the motion traces if a user reaches one of them. They are classified to two categories: 1) the points can be recognized by inertial sensors, such as stairs, elevators, etc; 2) the points could receive GPS on smartphones, as building entrances and windows.

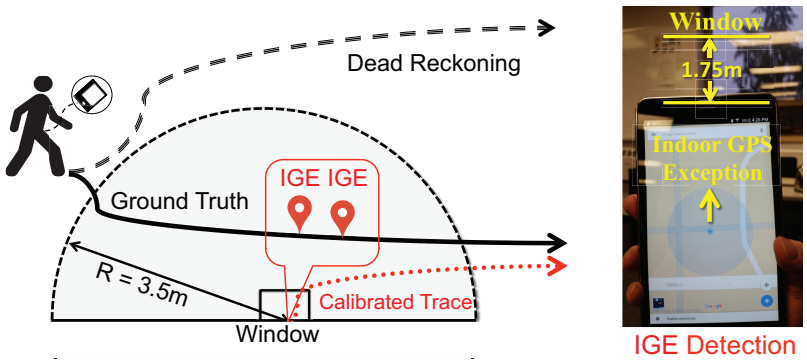


Figure 4.6 Indoor GPS Exceptions calibrate the deviations caused by dead reckoning. The generated motion trace using IGE is more close to the ground truth.

In iLoom, we focus on the second type of Anchor Point. For some entrances in an indoor building, by obtaining the position information through GPS, they are often marked as the initial positions of motion traces.

We have discussed the indoor/outdoor detection, there exists an interesting phenomenon: although within indoor rooms, people still may receive some GPS samples from the windows or other places near the outdoors. When we predict the environments are indoor or outdoor, these samples are seen as false information and should be disposed. These samples, named Indoor GPS Exception (IGE), provide an accuracy of better than 3.5 horizontal meters [**?**]. They also can be employed for calibrating some obvious deviation caused by dead reckoning: while a user is conducting dead reckoning for building the motion trace, when he/she is near the window and gets a IGE, if the estimated position by dead reckoning is out of the range of IGE, we can assume the estimated position has a serious deviation, thus, we will adopt the position of IGE to replace it.

Here we conducted an experiment: a user of iLoom walks and stops arbitrarily in a room for 10 minutes, he get 12 indoor GPS samples from the window. The GPS has a range of errors within 3.5 meters. As Fig. 4.6, 2 samples in the 12 IGE are helpful for dead reckoning. Therefore, IGE samples can calibrate the obvious deviation.

4.2.10 Filtering Noises by Average Speed

Note that the accelerometers on smartphones are not highly accurate, when we compute indoor average speed, the average speed values may still include some errors. The method to detect and rectify the incorrect indoor average speed is important.

When people walk and stop, the speeds of movements are continuous. This implies the speeds of people's movement have temporal features: for a certain time period, the speed of people should be correlated to the speeds of previous time periods and following time periods. We propose an approach to describe the correlation.

Once the detected average speed does not comply with the correlation, we assume it is average speed noise. The average speed is divided into m time periods. Each time period contains 20 seconds. In every m segments, we curve fit the average speed values as the linear function (5.5):

$$f_i(x) = k_i \cdot x + b \tag{4.5}$$

where x is the value of average speed; k_i is the slope in segment i; b refers to the intercept. In linear function (4.5), we set a margin range for b, and the margin area is from $b - \Delta b$ to $b + \Delta b$. If the obtained average speed values are in the area between $b - \Delta b$ to $b + \Delta b$, we will accept them as valid values. Otherwise, the as values will be disposed as noise.

The critical step is estimating the margin area for period n. To compute the margin area, we need to know three factors: the slope k_i, the values of $b - \Delta b$, and $b + \Delta b$. For slope k_i and b values, we strike a balance between the historical and current information of users' speeds. The average speed, which is close to current time period n, will have more weight on the user's decision for the next step. We provide the formula (4.6) for computing k_i and b values. Θ in formula (4.7) is the factor for assigning different weights.

$$\begin{bmatrix} k_n \\ b_n \end{bmatrix} = \begin{bmatrix} k_{n-1} \\ b_{n-1} \end{bmatrix} \times \Theta_{n-1} + \begin{bmatrix} k_{n-2} \\ b_{n-2} \end{bmatrix} \times \Theta_{n-2} + ... + \begin{bmatrix} k_1 \\ b_1 \end{bmatrix} \times \Theta_1 \tag{4.6}$$

$$\Theta_i = \frac{2^{i-1}}{2^{i-1} + 2^{i-2} + ... + 2^1} \tag{4.7}$$

The value of Δb is the based on experience, in our design, the default value of Δb is $0.5m/s$. After detecting the invalid average speed, we use compensating $\overline{v}$ to replace them. iLoom adopts an directive and practical method to compute the compensating value: calculate the average speed in the previous 10 valid samples.

4.2.11 Energy Saving

iLoom leverages multiple sensors on smartphones. Sensing procedures on smartphone, such as GPS, obtaining accelerations, and light sensing cause obvious energy consumption. In this subsection, we provide a strategy to save energy. For outdoor GPS sensing, we realize that if a user keeps stationary for long time, it is not helpful for building the model of user's walking behaviors. Therefore, we define the condition: if the user obtains GPS location information that the variations on x, y and z axes are less than 0.2 meters within 3 minutes, we can dispose corresponding GPS samples and turn off the accelerometer, camera sensor, and barometer. Only the GPS component is still working. Once the condition is not satisfied, the three sensing components will resume. For indoor data collection, we turn off the sensing components (including camera sensor, barometer, and GPS) for the time periods that a user are keep stationary. We define that when the user's variation of accelerations on the three axes are within $0.1m/s^2$ in 1 minute, the sensing components will turn off. Once the variation of accelerations are beyond $0.1m/s^2$ on each axis, the sensing components will turn on again. Algorithms 5 and 6 describe our energy saving strategy.

Algorithm 5 Algorithm of Indoor Energy Saving

Input:
$\overline{a_{rb}}$, initial motion state of the smartphone
Output:
Updated motion state of the smartphone, updated $\overline{a_{rb}}$.
Set $State \leftarrow Active$.
for each time slot (1min) **do**
  if $\overline{a_{rb}} < 0.1m/s^2$ **then**
    Disable barometer, camera sensor, and GPS
    $State \leftarrow Stationary$
  else if **then**
    slot $\leftarrow$ slot + 1
    Enable barometer, camera sensor, and GPS
    $State \leftarrow Active$
  end if
end for

Algorithm 6 Algorithm of Outdoor Energy Saving

Input:
initial location (x,y), initial motion state of the smartphone
Output:
Updated motion state of the smartphone, updated location.
Set $State \leftarrow Active$.
for each time slot t (3min) **do**
  if $Dist((x,y)_t - (x,y)_{t-1}) < 0.2m$ **then**
    Disable barometer, camera sensor, and accelerometer
    $State \leftarrow Stationary$
  else if **then**
    slot $\leftarrow$ slot + 1
    Enable barometer, camera sensor, and accelerometer
    $State \leftarrow Active$
  end if
end for

4.2.12 Reduce the Training Burden

So far, our analysis is based on a single user. Before a person uses the iLoom to obtain his/her locations, a procedure of light-weight training is required. If we extend our approach to more users, the training task can be further reduced.

In the multi-user model of iLoom, we provide an approximate solution for reducing the training load: People with the similar ages and heights often have the similar movement habits [65]. We categorize users into different groups by ages and heights. For each group of

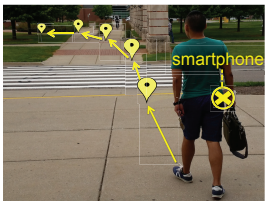
people, after data collection and transfer boosting we construct a special $\mathbb{R}_c$. The relation is stored in a hash map on the remote server. When the user logs in the iLoom system, after inputing his/her age and height, he/she will get a correlated $\mathbb{R}_c$ to assist the indoor localization.

4.3 Implementation and Evaluation

In this section, we attempt to answer the following questions: 1) Whether Acceleration Range Box, Average Speed Filter, Pedometer, and IGE can help dead reckoning indoor localization? 2) After applying transfer boosting algorithm, does the newly generated Acceleration Range Box have better performance than the original one? 3) Does GAPO pre-process the sensing data successfully? 4) Can iLoom be applied on both the single user and the multi-user models?

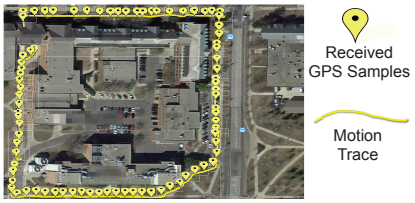
4.3.1 Experiment Setup

We implement iLoom on the Android platform (version 4.4) and evaluate its performance on two types of smartphones (Samsung Galaxy S5 and Google Nexus 5). All of the smartphones are equipped with standard sensors that include GPS, accelerometer, barometer, light and magnetic sensors. Initially, we focus on the single-user model. A user of iLoom walks arbitrarily and stores sensing data in both the outdoor and the indoor environments. He holds the smartphone in his hand or puts it in the pocket. Fig. 4.7 depicts the scenarios and routes that the user collects data. For outdoor scenario, the user carried the smartphone walked and stopped for 4 hours, the time period of each sample is 10 seconds. For indoor scenario, the user walked 10 minutes with the smartphone. The sampling frequency is 0.2 HZ. After collecting data indoor and outdoor, we built the $\mathbb{R}_o$ and $\mathbb{R}_i$ for people's motion behaviors. The Euclidean distance between the ground truth and estimated position is defined as the metric of localization error.



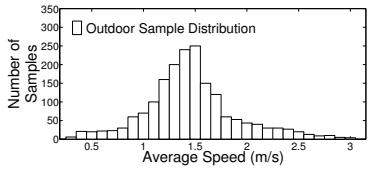
(a) Data collection in an outdoor environment.

(b) Data collection in an indoor environment.

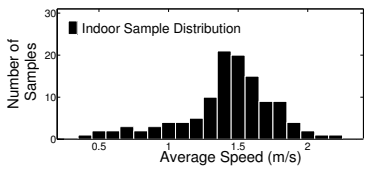


(c) A 200×200 area for data collection. Each yellow mark represents a sampling of GPS.

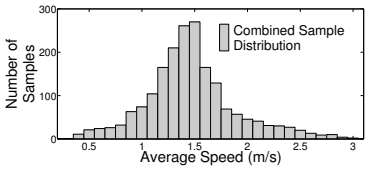
Figure 4.7 Collect data in different scenarios while a user of smartphone is walking.



(a) Sample distribution in outdoor environments



(b) Sample distribution in indoor environments



(c) Combined distribution after transfer boosting

Figure 4.8 The procedure of transfer learning. The bars in each figure represent the average speed range and the associated acceleration range box. After adding parts of samples from (a) to (b), the combined samples in (c) have more useful samples.

4.3.2 Acceleration Range Box Evaluation

Fig. 4.8 (a)-(b) represent the distributions of average speed for outdoor and indoor collections. Although most of samples distribute between the range from $1.2m/s$ to $1.6m/s$, the two distributions still have some differences. After applying the transfer learning approach on them, the combination data distribution varies as Fig. 4.8 (c).

We first validate the effectiveness of the Acceleration Range Box. When the user is walking in the indoor scenario, iLoom records the average error of distance within the growth of time. The user adopts the Average Acceleration Range Box ($\overline{a_{rb}}$) as a constraint while computing the motion trace by formula (4.3). As Fig. 4.9 shows, the localization accuracy is greatly and consistently improved by 60.4%. We repeat the comparison 10 times and the results remain the same. The shadow areas in Fig. 4.9 refer to the confidence intervals.

Based upon the results in Fig. 4.9, we measure the performance of transfer boosting. We use the transferred $\overline{a_{rb}}$ to replace original $\overline{a_{rb}}$ trained from the indoor environment. Figure

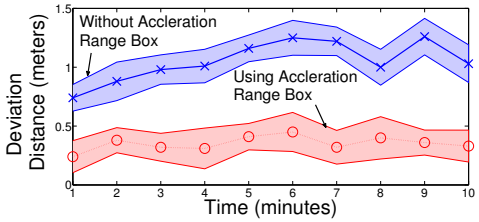


Figure 4.9 Acceleration Range Box improves the accuracy of dead reckoning.

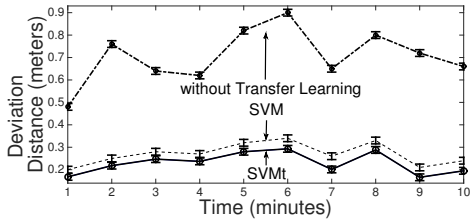


Figure 4.10 Transfer Learning improves the accuracy of dead reckoning.

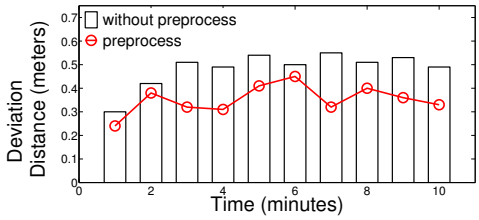


Figure 4.11 The comparison result of preprocessing and not taking preprocessing.

4.10 provides the experimental results of the comparison: although all of the three groups can enhance dead reckoning, the two groups using the $\overline{a_{rb}}$ combined with transferred outdoor and indoor data outperform the group just using $\overline{a_{rb}}$ from indoor environments. In iLoom, we choose SVM and SVMt [26] as the classifiers. In Fig. 4.10, both of the two classification algorithms fit the transfer learning approach, and SVMt performs better than SVM. By repeating the experiments 10 times, the final error of indoor localization is less than 0.35 meter.

When we transfer the instances with higher weight in outdoor dataset T to $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$, the proportion of the transferred instances ($p\%$) is significant. If we do not transfer enough instances in T to $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$, iLoom can not achieve the optimal localization accuracies. If

Table 4.4 The relation between each user's localization errors and the duration of outdoor data collection.

User / Duration	0.2 Hour	1 Hours	2 Hours	4 Hours	8 Hours
User A	0.532m	0.432m	0.399m	0.388m	0.384m
User B	0.570m	0.424m	0.404m	0.392m	0.387m

Table 4.5 Successful rates of two indoor/outdoor detection methods under different scenarios.

	Dining Hall	Residence Hall	Library
IODetector	87.24%	88.61%	91.43%
GAPO	94.32%	95.05%	95.15%

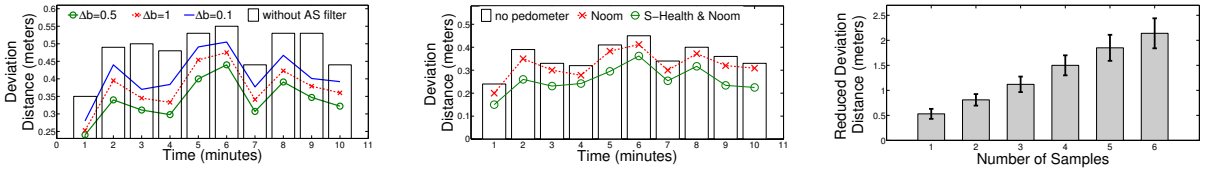
iLoom transfers excessive instances to $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$, the instances that are not similar to the instances in $\mathbb{R}_i(\overline{v}, \overline{a_{rb}})$ may include noise. In our evaluation, the optimal $p\%$ value is 74.8% for SVMt and 73.5% for SVM.

Depending on the above experiment conditions, we concentrate on the relations between indoor localization errors and the duration of outdoor data collection. By measuring the average indoor deviations of two iLoom users under different time lengths of outdoor data training, Table 4.4 supports our claim 1) within the duration of outdoor data collection increasing, each user's indoor localization accuracy improves gradually; 2) iLoom is able to enhance indoor positioning without long-term and extensive pre-training.

Figure 4.13 illustrates that, in a real indoor scenario ($160m \times 40m$), a user's dead reckoning trace is closer to the actual one via adopting transfer learning approach.

4.3.3 Performance of GAPO

The existing evaluations rely on preprocessing. In this section, we analyze the approach of preprocessing (GAPO) in detail. The function of GAPO is to distinguish data samples' environments. To validate GAPO is efficacious, we provided a control group. One group is the experiment result adopting GAPO and the other group does not use it. As shown in Fig. 4.12, we can conclude the group without preprocessing the false data cannot achieve the improved performance. To further explore the performance of GAPO, we execute the environment detection for three different buildings by receiving data samples both indoors and outdoors. Table V provides the comparison of P_e values between GAPO and the other classical detection



(a) The comparison results of average speed filtering.

(b) Using the third-party pedometers to reduce errors.

(c) IGE reduces the errors caused by dead reckoning.

Figure 4.12 Three other techiques to improve indoor localization: Average Speed Filter, Pedometer approach, and Indoor GPS Exception.

approach (IODetector). GAPO attains higher successful rate than IODetector. Although GAPO may cost more energy for smartphones due to the usage of GPS, the GPS is not working all the time for a user. The energy consumed is within a reasonable range.

4.3.4 Evaluation of Ancillary Approaches

4.3.4.1 Pedometer

As mentioned in the design section, we introduce the third-party pedometers on smartphone to help build the $\overline{a_{rb}}$. In our experiment, we install the S-Health [31] and Noom [50], two pedometers are highly praised on Google Play Store, on the Samsung Galaxy S5 smartphones. First, the user provides his step length to the pedometers. According to the known step length, the user runs iLoom with the two pedometers by the proposed algorithm. The threshold value of ΔSR is set as $0.1m/s$. The two additional datasets for $\overline{a_{rb}}$ are formed in the database on the smartphone. Next, we leverage the two additional datasets to assist the existing dataset and build the new $\overline{a_{rb}}$. One is $\overline{a_{rb}}$ from S-Health, the other is the $\overline{a_{rb}}$ from both Noom and S-Health. From Fig. 4.12(b), by applying the $\overline{a_{rb}}$ from Noom and S-Health, the improvements are much larger than what we observed without pedometers.

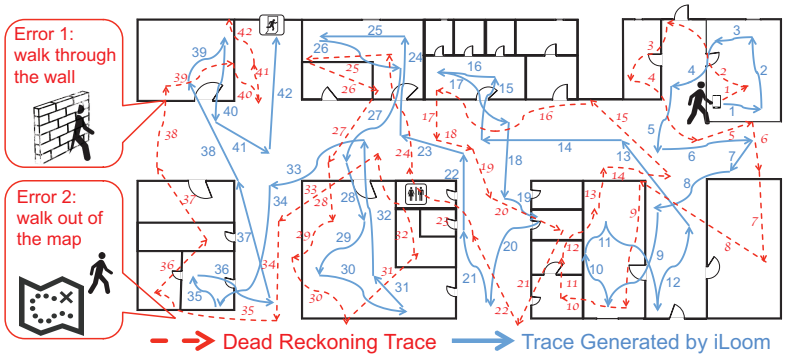


Figure 4.13 Motion traces are generated by dead reckoning and iLoom. iLoom reduces the distance deviations caused by dead reckoning.

4.3.4.2 Indoor GPS Exception (IGE)

Indoor GPS Exception can be utilized to reduce the obvious error caused by dead reckoning when a user receives a GPS sample in an indoor environment. In the above experiment, we received 23 such exceptional samples when the user walked in the indoor building. By assuming an error range of 3.5 meters, we find that when people are close to the windows, there are 6 samples that are out of the range. Then, we recompute the motion trace for these 6 segments. Such motion trace is generated 20 times. The Fig. 4.12(c) plots the reduced errors of the 6 segments by IGE (95% confidence interval). The positioning results containing more IGE outperform the results with less IGE.

4.3.4.3 Average Speed Prediction

We have designed the Average Speed filter to enhance the indoor localization accuracy. Fig. 4.12(a) compares the positioning results by using the average speed prediction and without the prediction. In this comparison, we test different values of Δb ($0.1m/s$, $0.5m/s$, and $1m/s$), we believe 1) Average Speed filtering yields the results with less errors; 2) when Δb equals to $0.5m/s$, the Average Speed filter performs best. If the range of estimated speed is too large, it could receive some invalid samples. However, when the estimated speed range is not enough, it might loose some worthwhile information. Considering the trade-off

between correctness and effectiveness, the default value of Δb ($0.5m/s$) is the best case in real measurement.

4.3.5 Energy Saving Measurement

We have compared two groups for measuring proposed energy saving strategy. We recorded the energy consumption information of one user in previous experimental observation by an Android application (Battery Widget Pro). We repeated our experiments by applying the energy saving approach. The time slot of energy recording is 1 second. For outdoor training procedure, by using the energy saving approach, iLoom reduces energy consumptions from 13.3% to 6.8%. For the indoor training and localization procedure, the power consumption of iLoom reduces from 14.3% to 7.2%. For the whole test case, the power consumption of iLoom reduces from 13.1% to 7.1%.

In our evaluation section, the sampling frequency of acceleration is 5HZ. Actually, the energy consumption is related to the frequency of accelerometer readings. Lower acceleration frequencies cause less energy consumption. However, the lower frequencies reduce the number of data samples. We build a control group for the case study. We changed the sampling frequency from 5HZ to 2HZ. Other experimental conditions are not modified. For the whole study, the power consumption of iLoom decreases from 7.1% to 6.6%. But the accuracy of indoor localization is 2.13m. This result is not as good as a high frequency control group.

4.3.6 Multi-User Model

iLoom supports two working models: single-user and multi-user. We have evaluated the single-user model in the above discussion. For multi-user model, the users' heights are highly related to the users' average speeds [65]. In this paper, we did such an experiment: we collected 15 volunteers' average speeds, accelerations and heights. The heights of these volunteers are approximately categorized to five levels: $1.65m$, $1.70m$, $1.75m$, $1.80m$, $1.85m$.

Table 4.6 The relation between users' heights and accelerations.

Height / Sides	X side	Y side	Z side
165cm±2.5cm	$1.821m/s^2$	$1.034m/s^2$	$0.525m/s^2$
170cm±2.5cm	$1.914m/s^2$	$1.122m/s^2$	$0.567m/s^2$
175cm±2.5cm	$2.011m/s^2$	$1.234m/s^2$	$0.610m/s^2$
180cm±2.5cm	$2.140m/s^2$	$1.304m/s^2$	$0.630m/s^2$
185cm±2.5cm	$2.327m/s^2$	$1.453m/s^2$	$0.679m/s^2$

Table 4.7 The comparison of localization deviation in different sites.

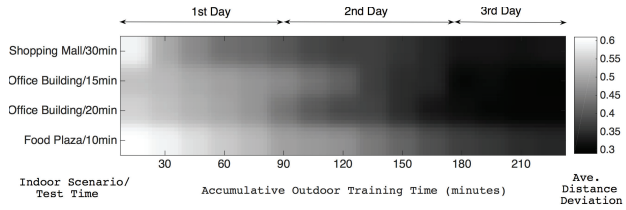
Model / Site	Entrance	Exit	Hallway	Office	PC Lab
Single-user	0.424m	0.476m	0.392m	0.287m	0.398m
Multi-user	0.454m	0.503m	0.422m	0.314m	0.423m

All the volunteers are 20-30 years old. For each user, we obtained the corresponding $\mathbb{R}_c$ for them. The person with higher height has a larger range of accelerations. The relation from height to $\mathbb{R}_c$ is stored in the iLoom system. Once a user inputs his/her height, iLoom will choose the approximate height for him/her and provide a related $\mathbb{R}_c$ for the user.

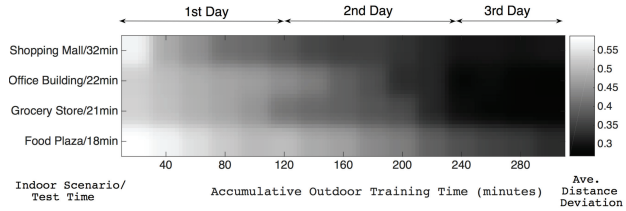
In practice, the multi-user model is not as accurate as the single user model. For example, a person with $1.69m$ will be assigned the $1.70m$ type's $\mathbb{R}_c$, but there still exists some differences. We compare the differences of the two models: for single-user model, we adopt the above experiment (the height of user is $1.73m$); for multi-user model, we let the user with the height of $1.73m$ choose the type of $1.75m$'s information and did the control group measurement as what the user did in single-model. We choose 6 observation points to record the deviations of the two models. The differences between the single-model and multi-model are listed in Table 4.7. Even though the multi-user model has more errors than single-user model, the localization results of multi-model is still convincing.

4.3.7 Long-Term Observation of iLoom

Finally, we evaluate the feasibility of extending iLoom as a daily use application. On the perspective of indoor dataset, when a certain user walks in different indoor environments, his/her walking style may vary sometimes. For example, if iLoom users walk in a shopping mall, they may walk slowly and stop frequently when they are browsing some products. Even if the proposed filters can reduce some useless samples, we still need to focus on the



(a) Indoor localization results of user1 walking in different indoor/outdoor scenarios.



(b) Indoor localization results of user2 walking in different indoor/outdoor scenarios.

Figure 4.14 Evaluation results of iLoom in three days.

localization results in various indoor scenarios. Additionally, for the existing experiments, the volunteers walked and collected the outdoor data continuously in a fixed time period. In this subsection, we extended the procedure of outdoor data collection to 3 days seamlessly. Users walked, stopped, and kept smartphones out of their pocket in their daily life. By applying iLoom, Fig. 4.14 indicates 1) the indoor localization accuracies increase with profiling more outdoor motion data and 2) the indoor positioning accuracies are within $0.4m$ even in challenging conditions where a user walked in different indoor and outdoor scenarios. We believe the transfer boosting approach reduces the influence of the samples that cannot represent user's normal walking styles.

4.4 Conclusion

In this chapter we propose one key conjecture: could we transfer a user's worthwhile outdoor motion information to indoor movement data and enhance indoor localization ? We presents iLoom, an indoor localization mechanism that utilizes the users' outdoor walking features. iLoom selects the dead reckoning to locate people indoors and introduces an Acceleration

Range Box to optimize the user's received accelerations. For building an accurate Acceleration Range Box, the sensed data from indoor and outdoor environments are processed: since certain people's moving behaviors are proximate in indoor and outdoor environments, we combine the data describing the outdoor motion by accurate GPS and the data of the indoor movements via transfer boosting. Experiments and simulations from 15 users and 3 real buildings demonstrate iLoom not only improves dead reckoning but also does not need extra infrastructure and data training for certain scenarios. The accuracy of indoor localization reaches up to 0.35 meter.

CHAPTER 5

COOPERATION AMONG SMARTPHONES TO IMPROVE INDOOR POSITIONING

5.1 Introduction

Modern smartphones or tablets are equipped with sensors, such as accelerometer, gyroscope, rotation vector, and orientation sensors, and multiple types of radios, which can detect movement and can be used to predict location. Dead reckoning [71] can calculate a person's current positions by using a previously determined position. The parameters that dead reckoning needs are obtained by the accelerometer and orientation sensors on the smartphones. The performance of dead reckoning relies on the measurement accuracy of these sensors. In fact, the accumulative errors caused by the inertial sensors are difficult to avoid. As a common sensor used for localization, UM6 [63], small errors of the orientation estimate causes serious deviation of the computed location. With only 0.5 degree error of the orientation sensor, an error of 308 meters can occur within a minute.

Furthermore, new devices are introduced regularly for health monitoring and exercise profiling, which include detecting the movement of people for the purposes of counting the number of steps a person takes on a daily basis. It is said that walking 10,000 steps a day is important exercise that the human body needs to stay fit. Therefore, by building a pedometer using the accelerometer on the smartphones, the application on smartphones can provide health and medical information to users, such as number of steps and burning calories. These pedometers count users' steps by using their own algorithms. However, since the data obtained from accelerometers are not accurate and the algorithms are not perfect, the accuracy of such pedometers is not ideal. Smartphones may be paired with such pedometers, or may use their internal sensors.

We propose a new approach, CRISP - CoopeRating to Improve Smartphone Positioning, which assumes that dead reckoning have inaccuracies, but leverages opportunities of the interaction of multiple smartphones to improve accuracy. Each smartphone computes its own position, and then shares it with nearby smartphones. Furthermore, the signal strengths of multiple radios are used on smartphones to estimate distances between the devices. The idea is that while individual smartphones may provide some positioning (possibly inaccurate) information, opportunities of accuracy improvement occur when several smartphones cooperate and share position information. Accuracy may improve as multiple iterations of information sharing and computations are made. Via indoor experimentation and simulation, we evaluate our approach and believe it is promising as an inexpensive means to improve position information and possibly lead to better results for exercise profiling.

The main contributions of CRISP include the following aspects:

1. While many researchers have used RSSI as a means to measure distances between positions [70], to the best of our knowledge, CRISP is the first of its kind to interact with other scanned mobile devices held by other users in order to improve a user's own localization accuracy.

2. We design and evaluate an approach to improve the accuracy of a pedometer application on a smartphone by RSSI measurement rather than only judging accelerometer data.

3. We combine the RSSI from Zigbee and Bluetooth detected on mobile devices, and design a WiFi filter to reduce the noise.

5.2 Overview of Design

Before introducing details about our design, we provide a short overview of the components used in design. Figure 5.1 shows the overall architecture.

Our system has two mechanisms on a user's mobile device:

1) Our system periodically measures the accelerometer on a user's mobile device. By simulating a user's walking mode as a formula, we compute the user's position by the improved dead reckoning including Average Speed Prediction (ASP).

2) When a user encounters other users, CRISP periodically broadcasts Bluetooth, Zigbee and WiFi signals to the other users' devices that are nearby. By receiving the RSSI values from other detected devices, a mobile analyzes the variation of the RSSI in each period. Since the WiFi signal is sensitive to interference, if the variation of the WiFi RSSI is beyond a threshold, we assert that the RSSI values received in this period are invalid because of interference and recompute using historical data.

A practical challenge is that how to use RSSI values to help a user locate himself accurately without any extra devices. In our system, after obtaining RSSI from detected mobile devices and other Access Points (AP), the user uses the mapping relation between RSSI and the Euclidean Distance to estimate the distance between these devices. These relations of different mobile devices are trained off-line and can be accessed on the cloud. After obtaining the distance between each pair of devices, all devices in the detected range can form a triangle or polygon. The initial position of each vertex is generated by dead reckoning. The user computes its own position by using the distances to other devices and other devices' locations. By iteration, the errors of estimated positions decrease effectively. A mechanism of choosing the estimated positions between dead reckoning and geometry computation is executed in each period. In addition, we employ triangle inequality theorem to filter some interferences [76].

CRISP also designs a model for counting the user's walking steps. This model can reduce the errors caused by common pedometers on the smartphones.

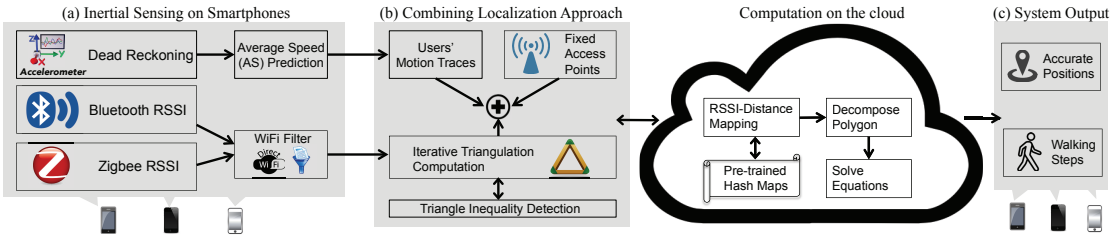


Figure 5.1 Framework of CRISP.

5.3 Design of CRISP

5.3.1 Dead Reckoning

In CRISP, we design a mechanism named Average Speed Prediction (ASP) for calibrating the deviations caused by dead reckoning. In reality, when users of smartphones walk or stop, their motion speeds are continuous. Namely, the speeds of users' motions have temporal features: for a certain time period, the speed of user is correlated to the speeds of previous time periods and following time periods. Based upon this perspective, we provide an approach to enhance the original dead reckoning. If the computed average speed does not comply with the temporal correlation feature, the speed is an *exceptional speed*. Then, as shown in Fig. 5.2, there are n time periods. Each time period contains 25 seconds. For every m segments, we curve fit the average speed values as the linear function (5.1):

$$E_t(x) = k_t \cdot x + b \tag{5.1}$$

where x is the value of average speed; k_t is the slope in segment t; b refers to the intercept. For the proposed linear function, we assume a margin range for b, and the margin area is from $b - \Delta b$ to $b + \Delta b$. Once the average speeds are not in the margin ranges between $b - \Delta b$ and $b + \Delta b$, the average speeds will be disposed as *exceptional speed* and the received accelerations will be replaced: CRISP leverages a practical approach to obtain the compensating value: 1) calculate the average speed in the previous 10 valid samples and 2) use the corresponding accelerations to replace the disposed items. The average speed values in the margin area will

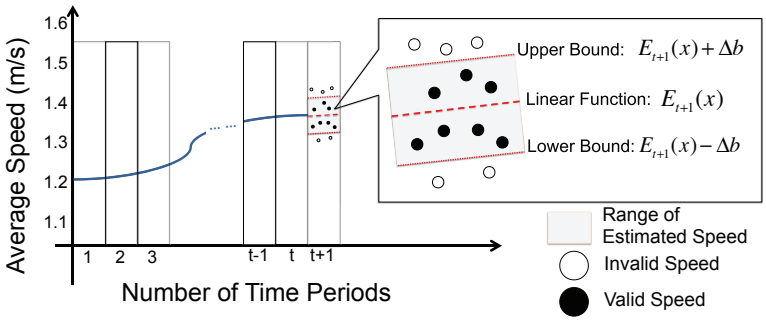


Figure 5.2 Average Speed prediction for the time period t+1.

be accepted by CRISP.

Then, the key step is estimating the margin range for time period t. When we predict the margin range, we make balance between the historical and current information of users' speeds: although all the average speeds before time period t will influence the margin range of time period t, the average speed that is closer to the current time period t will have more weight on the determination. In order to compute the margin area, we need to calculate three factors: 1) the slope k_t, 2) the values of $b - \Delta b$, and 3) $b + \Delta b$. We provide the formula (5.2) for computing k_t and b values. Θ in formula (5.3) is the factor for assigning different weights. In our design, the default value of Δb is $0.5m/s$.

$$\begin{bmatrix} k_t \\ \delta_n \end{bmatrix} = \begin{bmatrix} k_{t-1} \\ \delta_{t-1} \end{bmatrix} \times \omega_{t-1} + \begin{bmatrix} k_{t-2} \\ \delta_{t-2} \end{bmatrix} \times \omega_{t-2} + ... + \begin{bmatrix} k_1 \\ \delta_1 \end{bmatrix} \times \omega_1 \tag{5.2}$$

$$\omega_t = \frac{2^{t-1}}{2^{t-1} + 2^{t-2} + ... + 2^1} \tag{5.3}$$

5.3.2 Distance and RSSI

Received Signal Strength Indicator (RSSI) is a common measurement of the power present in a received radio signal, with "dBm" as the unit of RSSI. RSSI is easy to collect on most mobile devices. Although the RSSI values often vary due to interference and path loss, RSSI values obtained from the other devices are highly related to the distance between the devices.

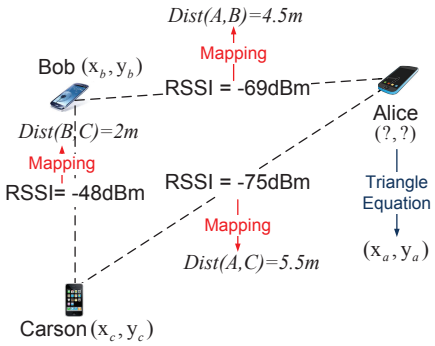


Figure 5.3 Triangulation model.

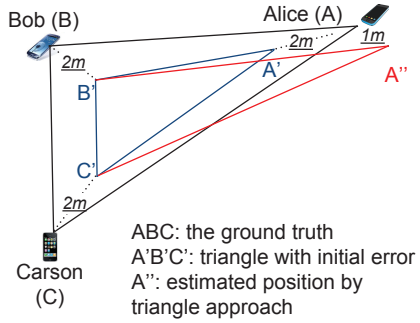


Figure 5.4 Triangulation calibration example.

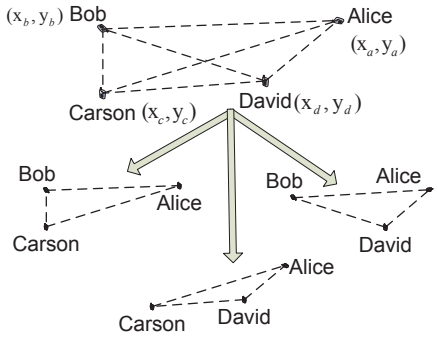


Figure 5.5 Decompose quadrilateral to triangles.

Shorter distances often represents stronger RSSI. In CRISP, we build the RSSI-distance mapping relation by collecting the data that represents the distances and RSSI values for different types of popular mobile devices.

In our preparation phase, we evaluate the RSSI-distance mapping relation for the Samsung Galaxy S5 smartphone, Samsung Tablet 4, and Google Nexus 5 tablet. The RSSI is obtained from the Bluetooth Adapter. For example, if the distance between Samsung S5 smartphone and Samsung Tablet 4 is 5 meters in an empty room, the RSSI is -66 dBm. The training relation does not consider interference and other factor fading the RSSI values. These noises and exceptions will be handled by the WiFi filter. These mapping relations are stored in the database on a cloud server. In addition, even if training the mapping relation may bring labor and time costs, since the types of mobile devices in our work are popular, the obtained relations can serve common Android based mobile devices.

5.3.3 Triangular Calculation Localization

5.3.3.1 Triangular Calculation Model

In CRISP, the goal of triangular calculation is to locate a user's position by knowing other detective devices' locations and RSSI values. To illustrate this idea, we provide an example: as shown in Figure 5.3, three users (Alice, Bob, and Carson) hold mobile devices that have Bluetooth adapters. In each time period, we assume they form a triangle. After turning on the Bluetooth option, each receives Bluetooth RSSI values from the other two users. Then, we can obtain the length of the three sides of the triangle by the distance-RSSI mapping relation. If Alice hopes to locate herself and she knows positions of Carson and Bob (Bob and Carson's positions are computed by the dead reckoning and sent to Alice when they encounter), Alice can compute her position by the equations (5.4), (x_a, y_a) denotes the the device a's position on a two dimension plane. AB and AC denotes the distances between Alice and Bob, Alice and Carson. This example explains how the triangulation calculation

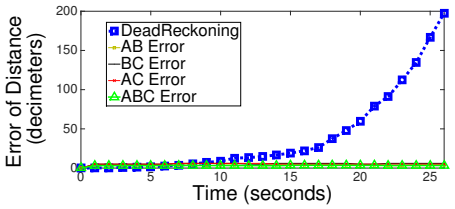


Figure 5.6 Distance errors of dead reckoning and triangle computation approaches.

model helps one user to locate his/her position. In our design, because the range of Bluetooth detection is 10 meters, the upper bound of each slide in a triangle is 10 meters.

$$Mapping(RSSI_{AB}) = |AB| = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} \tag{5.4}$$
$$Mapping(RSSI_{AC}) = |AC| = \sqrt{(x_a - x_c)^2 + (y_a - y_c)^2}$$

Since the dead reckoning is not enough to provide satisfactory location information, another case is proposed in Figure 5.4. There are three users (**A**lice, **B**ob, and **C**arson) carrying smartphones. The vertices on triangle ABC refer to the real positions of the three users. We assume the three users evaluate their initial locations by the dead reckoning apps, which are not accurate. The estimated positions are A', B', and C'. The distances between A and A', B and B', and C and C' are two meters. By using the triangular calculation, Alice obtains RSSI values from Bob and Carson, and by the RSSI-distance mapping relation, Alice evaluates the estimated distance from Bob and Carson.

Then, by the two computed distances (AB' and AC') and the distances between B' and C' (B'C'), we can compute the position A". A is closer to A" rather than A'. We can also compute the position B" and C". Thus, the new formed triangle A"B"C" is able to reduce the distance errors caused by dead reckoning.

We apply the triangle calculation to a dynamic scenario. The preliminary observation is: user Alice carries the mobile device and enters an empty room; Bob and Carson are already in the room. The three people walk freely. In the beginning, we assume they do not have any

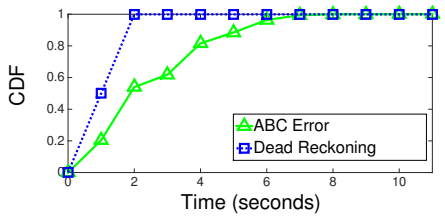


Figure 5.7 CDF error of dead reckoning and triangle computation approaches.

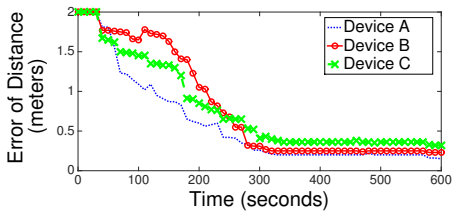


Figure 5.8 Distance errors of triangle computation by three cooperating devices.

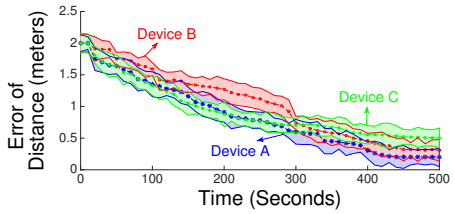


Figure 5.9 Repeated triangle computation by three cooperating devices.

initial error of distance. Then, we record the Alice's distance error, which is caused by dead reckoning in the next 14 seconds.

As illustrated in Figure 5.6 and Figure 5.7, since there are inaccurate values obtained from accelerometer, the distance errors due to dead reckoning increase rapidly. However, the localization errors of the triangle approach stay at a low level because the triangle calculation errors are caused by the differences between estimated mapping distances and the real distances.

The above analysis is from the perspective of Alice. We turn focus to all three devices in the triangle. We also execute the experiment as above. The only change is that we set each user to an initial deviation from their real starting position (the deviation is 2 meters). The initial deviations of their locations are caused by the dead reckoning application on the

smartphone. Then, we use the triangle calculation to compute the users' locations. As Figure 5.8, after running the triangle computation for 600 seconds, the distance errors of A, B and C are all reduced effectively. To validate this conclusion, we repeat the same experiment 3 times. Then, we simulate the experiments 47 times. As the Figure 5.9 depicts, the data samples on the two dimensional plane refer to the average values of distance errors of the 50 experiments (or simulations) at different time points. The shadow areas refer to the confidence interval for each data point. In this paper, confidence intervals are typically stated at the 95 percentage confidence level. Although it is a preliminary observation, by adopting triangle computation, with the time increasing, errors of distance can be reduced within 1 meter, which is reasonable and acceptable for many indoor positioning applications.

5.3.3.2 Triangulation Computation Refinement

Although we introduce triangulation model to localize users of smartphones, in practice, the original triangulation model is not perfect: 1) the interferences that occur on the link between each pair of devices yield incorrect RSSI values, and 2) the pre-trained RSSI-Distance Mapping relations still include minor errors. Therefore, we propose a directive and practical mechanism to boost the original triangulation approach.

In trigonometry, the triangle inequality states the sum of the lengths of any two sides must be greater than the length of the remaining side. For our triangulation computation model, the sides that are obtained by the RSSI-distance relation should satisfy triangle the inequality theorem. If the three sides computed by the RSSI-distance mapping do not comply with triangle inequality theorem, we assume these sides are mis-calculated. Then, we dispose the incorrect sides and adopt the sides that satisfy the triangle inequality theorem in the previous time slot to replace them. Algorithm 7 shows this mechanism in detail.

Algorithm 7 Triangle Inequality Detection

Input:

$RSSI_{AB}$, $RSSI_{AB}$, $RSSI_{AB}$ in each time period

Output:

Estimated position of each mobile device

```
for i=1;i < num of periods; i++ do
    |AB|=Mapping(RSSI_AB);
    |AC|=Mapping(RSSI_AC);
    |BC|=Mapping(RSSI_BC);
    if (|AB|+|AC| > |BC|) and (|AB|+|BC| > |AC|) and (|AC|+|BC| > |AB|) then
        // Once triangle inequality is satisfied,
        // CRISP will conduct triangulation computation.
        Start Triangulation Model;
    else
        // Use the correct sides in previous time slot instead.
        (|AB|, |BC|, |AC|)_n = (|AB|, |BC|, |AC|)_{n-1}
    end if
end for
```

5.3.3.3 Extension from Triangle to Polygon

Based on the above example that includes three users, Alice can obtain her position by triangle computation. In a real scenario, there might be more than three devices in a room or in a hallway. As the above example, if David enters the room, we can form a quadrilateral. User devices can be treated as the vertices of a quadrilateral. Then, three are three triangles in the quadrilateral including the node Alice, namely, triangles ABC, ABD, ACD as shown in Figure 5.5. The new location of Alice is defined as the mean value of estimated Alice's locations from the three triangles:

$x_a = (x_{a(abc)} + x_{a(abd)} + x_{a(acd)})/3$

$y_a = (y_{a(abc)} + y_{a(abd)} + y_{a(acd)})/3$

Where $x_{a(abc)}$, $y_{a(abc)}$ are the Alice's (a's) x and y values computed from triangle ABC. If the room contains more than 4 devices, all the devices can be abstracted as the vertices of a polygon. For each of the devices, we can use the triangles that are in the polygon to help localize itself. Then, by computing the mean value of the position obtained from different triangles, the user of a device can compute its position. If one user encounters more mobile

devices and forms more complex polygons, the localization results may be more accurate.

5.3.4 Combine Different Types of Signals: Bluetooth, Zigbee, WiFi

5.3.4.1 The Features of Three Types of Signals

Most smartphones and tablets support applications of Bluetooth and WiFi. Bluetooth RSSI is not only sensitive to interference but also sensitive to the distance between two detective devices. Bluetooth RSSI values often vary from maximum to minimum within its 10 meters' range. Shorter distance reflects stronger signal strength. WiFi RSSI values are sensitive to interference such as the human body or wall between the sender and receiver, but for most wireless routers that provide WiFi for mobile devices, the RSSI values do not vary much by changing the distance from 1 to 10 meters.

A Bluetooth adapter operates using a procedure of scanning and inquiring. It often costs 5-15 seconds for current mobile devices. Therefore, the sampling frequency of Bluetooth RSSI is limited. Sometimes, if mobile devices move rapidly, the user might lose the chance to record the Bluetooth RSSI values from them. To remove this defect, we introduce the RSSI received from the ZigBee Protocol. The feature of RSSI using ZigBee is similar to Bluetooth RSSI, but ZigBee does not require a long time to scan and connect to other devices. Also, Zigbee [5] is able to set the RSSI sampling frequency by the programmer, with 1HZ or 0.5HZ as common RSSI sampling frequencies. Although most current smartphones are not integrated with ZigBee, we consider adding ZigBee to have more RSSI samples to improve the localization accuracy, and consider that future generations of smartphones may have similar capabilities.

5.3.4.2 WiFi and Direct-WiFi Filter

RSSI is known to perform poorly in indoor environments. Some variations of RSSI values may cause errors in RSSI-distance mapping. For example, if a moving object is between

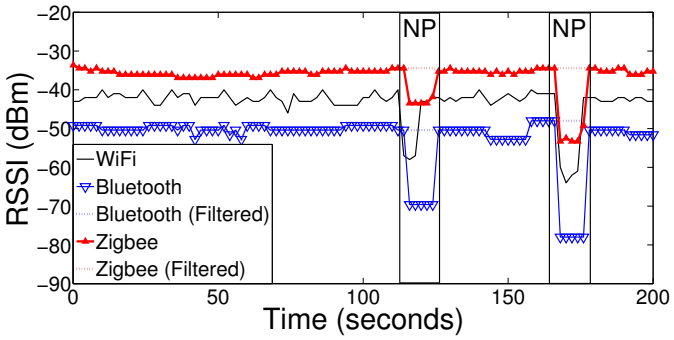


Figure 5.10 WiFi filter detects two periods which are Noise Periods (NP).

the two Zigbee sensors (or Bluetooth adapters), the received value of RSSI will decrease. Then, if we use a RSSI-distance mapping in training datasets, the corresponding distance will increase.

Therefore, we need to filter this interference (noise). As mentioned in the previous section, although WiFi is not sensitive to the distance, WiFi is sensitive to the interference. By this feature of WiFi, we design a filter to reduce the effect of noise caused by interference.

Since a user's movement cannot change abruptly, if the received WiFi RSSI varies each 10 seconds more than 5dBm, we assume such obvious change of RSSI is caused by interference. We define the 10 second time period as a "Noise Period (NP)". We use the average RSSI value in the closest previous period that is not a NP to replace the RSSI values in the NP. As shown in Figure 5.10, when the WiFi signal encounters interferences at two NPs (116-119 seconds, 166-169 seconds), the values of RSSI decrease sharply. After using the WiFi filter to detect NPs, the noise samples of Bluetooth and ZigBee RSSI are corrected by the average RSSI value in the closest previous time periods.

In most of indoor scenarios, people receive WiFi signals by wireless routers. However, some indoor environments do not have such infrastructures. Wi-Fi Direct is a Wi-Fi standard that is adopted on most of popular mobile devices, such as iPhone, iPad, and Android smartphones. This technology enables devices to connect with each other without requiring a wireless access point, such as a wireless router. Each smartphone/tablet can open the

Algorithm 8 WiFi-Filter Algorithm

Input:

The RSSI samples collected from Bluetooth, Zigbee, WiFi adapters, threshold of WiFi filter

Output:

Filtered RSSI values of Bluetooth, Zigbee

1: **for** i=1;i < num of periods; i++ **do**
2: **if** variation of WiFi RSSI value > threshold **then**
3: // Find RSSI values of the closest previous period
4: call WiFi-Filter(i-1);
5: // Replace the abnormal RSSI values in the NP
6: **for** j=1; j < number of Bluetooth samples in i (n_b); j++ **do**
7: $BluetoothRSSI[i][j] = \frac{\sum_{j=1}^{n_b} BluetoothRSSI[i-1],[j]}{n_b}$
8: **end for**
9: **for** k=1; k < number of Zigbee samples in period i (n_z); k++ **do**
10: $ZigbeeRSSI[i][k] = \frac{\sum_{k=1}^{nz} ZigbeeRSSI[i-1][k]}{n_z}$
11: **end for**
12: **else**
13: Return $BluetoothRSSI$[i][n_b];
14: Return $ZigbeeRSSI$[i][n_z];
15: **end if**
16: **end for**

Table 5.1 Using APs to improve the localization results of smartphones

Devices \User's Deviation	A	B	C
3 Smartphones (3S)	0.251±0.003 (m)	0.283±0.004 (m)	0.337±0.004 (m)
3 Smartphones (3S) + Wireless Router (WR)	0.240±0.003 (m)	0.266±0.003 (m)	0.315±0.003 (m)
3S + WR + Bluetooth Beacon + Zigbee	0.217±0.002 (m)	0.223±0.003 (m)	0.293±0.003 (m)

WiFi-direct option, which means each mobile device can detect others by WiFi and receive the RSSI values from these devices. If the WiFi values obtained by Directed-WiFi changes sharply, it is also seen to be a NP and be handled by the WiFi filter.

5.3.5 Leverage the Fixed Access Points

In our design, we have leveraged the cooperation between mobile devices to enhance the localization accuracies. In reality, there often exists some Access Points (AP) in indoor buildings, such as wireless routers, Bluetooth beacons, Zigbee senders, etc. Considering these devices are usually deployed in a fixed position, once a mobile device can detect the device information of AP, the mobile device can access AP's position to enhance its own position

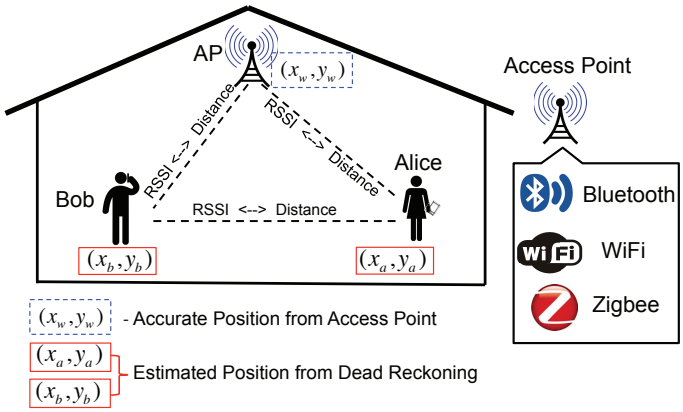


Figure 5.11 Adopt fixed access points to improve the localization of smartphones.

accuracy via the triangulation approach. Different from the mobile devices whose positions change continuously, the positions of APs are fixed and correct. Therefore, if the APs allow to share their location information with other mobile device, the AP could be used to boost smartphones' localization results. Figure 5.11 illustrates the working model of fixed APs. To verify our design, we did the following comparison: Based upon the previous observations, we add two control groups to repeat the evaluation. One is adding a wireless router, which support WiFi function and its position for mobile devices. The other is adding one wireless routers, one Bluetooth beacon and a Zigbee sender. All of them can provide their fixed positions and communication functions. After conducting the related simulations, as shown in the Table 5.1, the group including more APs outperforms other groups.

5.3.6 Step Benefits

Steps are counted by the pedometer applications on smartphones, however, most pedometers are highly inaccurate. One intuitive reason is that the pedometer integrated on the smartphone relies on the accelerometer. The accelerometer values on smartphone do not equal to human bodies' accelerations, it is difficult to identify an acceleration signature in human walking pattern without errors.

Different people have different lengths of steps. To enable CRISP to count steps, it is

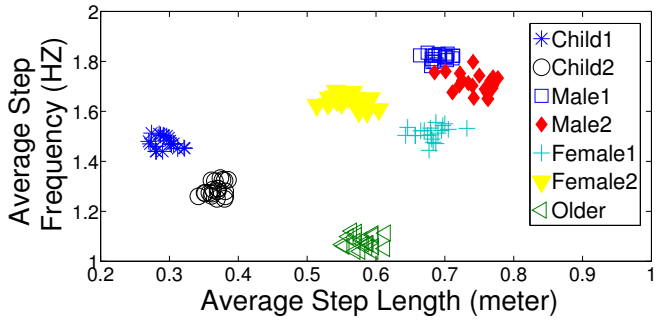


Figure 5.12 Relation between step length and step frequency.

necessary to estimate the length of the step for each user. We employ a linear step-frequency model as equation (5.5), which is described by Li [39] and Hilsenbeck [21]. The symbol f_k denotes the step frequency that can be counted manually in a short training period k, its minimum time length is 20 seconds. The symbol d_k is the length of the step. Then, we develop a two dimension data set containing the average step length and average step frequency of different people as illustrated in Figure 5.12. Seven groups of volunteers present their own features. We fit the linear model by using the least square to set a and b. Thus, by conducting a lightweight training phase, the user can get his/her own step length for counting steps.

In our approach, users obtained the location information continuously in different periods. Within a short time period i, we may assume people walk straight. Computing by equation (5.6), it is simple to count steps a user have walked within a certain time period. N_s denotes the number of steps. By adding the number of steps that have recorded in each time period, the user can determine the number of steps they walked in total.

$$d_k = a \times f_k + b \tag{5.5}$$

$$N_s = \frac{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}{d} \tag{5.6}$$

5.3.7 Running CRISP in the Cloud

CRISP is a light weight application. Considering the limited computational capabilities and power consumption on smartphones, we hope to deploy more tasks on a remote server rather than smartphones. In our design, users upload their received RSSI and location messages from other cooperating devices periodically. Thus, the prepared distance-RSSI maps stored on the server transfer received RSSI values to distances continuously. We deploy the tasks 1) forming polygons of devices, 2) decomposing polygons to triangles, 3) solving equations to compute the location of a user's device on the server instead of running it on a mobile device.

5.4 Implementation and Evaluation

5.4.1 Experimental Setup

We built a prototype of CRISP on Android mobile devices using the version KitKat. In the experiments and simulations on each device, we combine Bluetooth, ZigBee and Direct-WiFi Filter together to do the triangle calculation. Although current mobile devices, such as the Samsung Galaxy and the Google Nexus smartphones do not integrate ZigBee on them, in our experiment, we bound the TelosB ZigBee sensors [11] on these mobile devices and run the application programs on TinyOS [38]. Since the ZigBee model is not supported by Android OS, we record the ZigBee and Bluetooth data synchronously by sharing the timestamps. The frequencies of ZigBee and direct WiFi samples are 1HZ and 0.25HZ. The sampling frequency of Bluetooth RSSI is around 0.1 to 0.2 HZ. The format of the data sample is shown in Fig. 5.13. For each data sample, after receiving Bluetooth and ZigBee RSSI values translated by the trained mapping relation, the estimated distance between each pair of devices is determined.

Before our evaluation, we assume: 1) devices in our experiments (smartphones, tablets, APs) allow their positions, identifier, and timestamps can be accessed by other devices; 2)

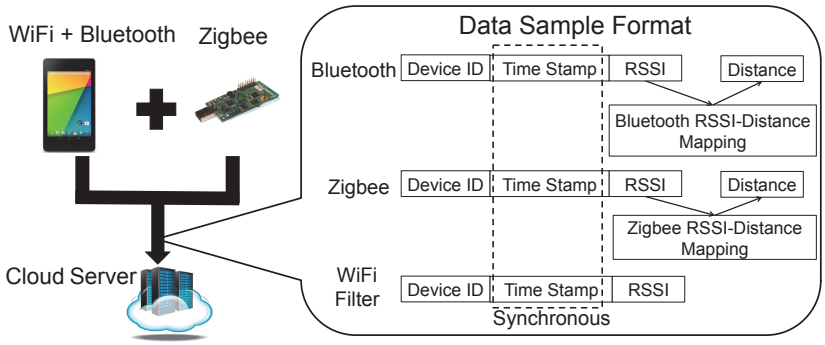


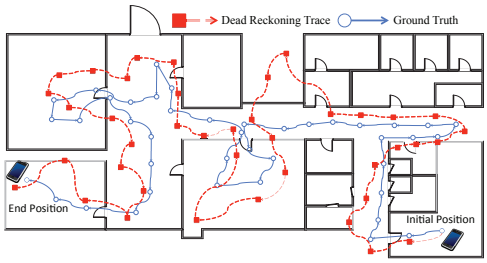
Figure 5.13 The equipment structure and data sample format in our experiment.

the scenario in our experiment has known map; 3) for each mobile device, the initial position of dead reckoning has been obtained by GPS or other methods.

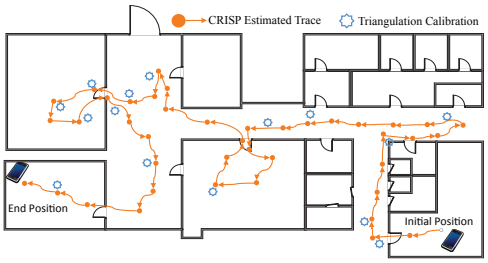
The dead reckoning is implemented as follow: we set each slot period to be 1 second. Then, we compute the movement direction and the movement distance in each segment. By collecting the computed segments in each time period, we generate the user's trace. The users carry the devices and walk freely in rooms or hallways.

For our evaluation and discussion, we mainly seek to answer five questions:

- Does our approach improve the mobile devices' localization accuracies in different environments?
- Does our approach count walking steps for users effectively?
- How does the Zigbee model and WiFi filter assist the Bluetooth model?
- For one user, does encountering a greater number of users who also use CRISP improve his/her own position accuracy?
- Can other assistant techniques (AS Filter, WiFi Filter, and Fixed AP) contribute to the localization results?



(a) Comparison between the trace generated by dead reckoning and the ground truth.



(b) The trace generated by CRISP.

Figure 5.14 Case study in a complex building with multiple rooms and hallways.

5.4.2 Metric of Measurements

Two metrics are introduced in the evaluation:

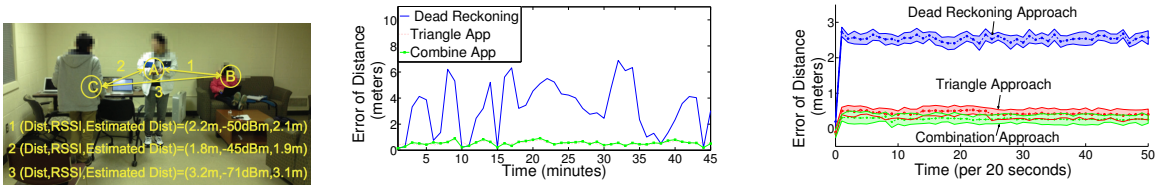
1) *Error of Accumulative Steps*: the different number of steps counted between third-party application and CRISP;

2) *Error of Distance*: the distance (in meters) between the ground truth and the estimated position.

5.4.3 Scenario Measurements

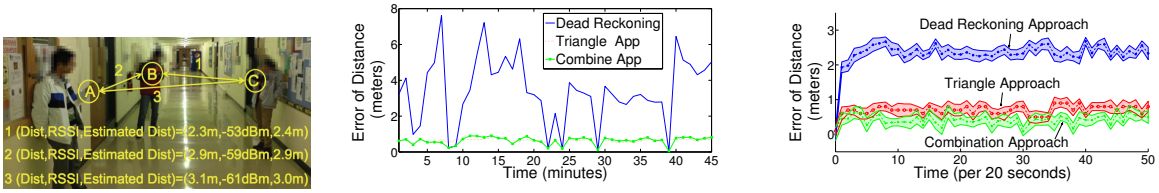
5.4.3.1 Case Study in One Room

As shown in Fig. 5.15 and Fig. 5.16, we conduct the experiments for 45 minutes in a room of the Engineering College at Michigan State University. There are three users who carry mobile devices and walk freely. All use CRISP and interact with others frequently. As illustrated in Figure 5.15(b), the X axis refers to the time of the experiment. The Y axis refers to the error of distance of a user A. The blue line, red line, and green line refer to the distance errors of dead reckoning, triangle approach and the approach combining both of them. As shown in Figure 5.16, we conduct the similar experiment in a hallway. From the two types of experiments, after cooperating with the triangle computation, the combination approach performs best. The average deviation from dead reckoning is reduced to 0.5 meters.



(a) Room environment (b) Room measurement for one time (c) Room measurement repeated 200 times

Figure 5.15 Experimental measurement in a room. Three users walk, stop, and sit to form a triangle to enhance dead reckoning.



(a) Hallway environment (b) Hallway measurement for one time (c) Hallway measurement repeated 200 times

Figure 5.16 Experimental measurement in a hallway. Three participants walk and stop to form a triangle to improve dead reckoning.

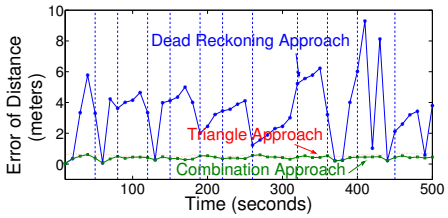


Figure 5.17 CRISP measurement in the complex indoor environment.

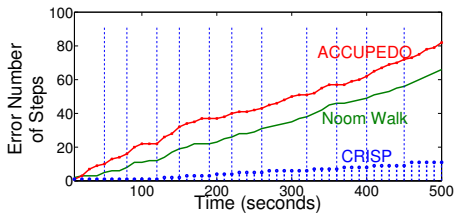


Figure 5.18 Comparison of accumulative step errors for three applications.

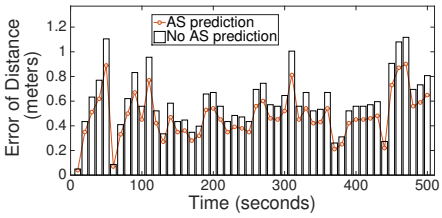


Figure 5.19 Distance error of the group without Average Speed Prediction (ASP).

We repeat our experiments 200 times by simulations. When simulating the dead-reckoning, we obtain the acceleration values from the accelerometer on the smartphone periodically. Then, we add random errors of acceleration on the x, y, z axes, the error range is from $-1m/s^2$ to $1m/s^2$. For the triangle computation, we add -10 to 10 percentage distance errors for each side of the triangle, randomly. The time of each experiment group is reduced from 45 minutes to 1000 seconds. As Figure 5.15(c) and Figure 5.16(c) display, at a specific time point, the data samples on the each line refer to the average values of distance errors obtained in 50 times of simulations. The shadow of each line is the confidence interval of computed values. The two figures indicate that after many simulations, the combination approach has more accurate results than the dead reckoning and triangle calculation. It achieves the error range that is within 1 meter.

5.4.3.2 Case Study in a Complex Building

We extend our experiment from one place to the building of Engineering College in Michigan State University. This building includes multiple rooms and hallways. A user walks with the mobile device and communicates with other devices. All devices are installed and running CRISP. Figure 5.14 provides the overview of the two estimated traces in our floor plan. By comparison in multiple rooms and hallways, the generated trace by CRISP is closer to the ground truth than the trajectory made by dead reckoning. As depicted in Figure 5.17, the dash lines refer to the time points when the user changes their room. For example, at the 50th second, a user leaves a room and enters the hallway. The localization results of the

combination approach are more accurate than the other two approaches, especially for the dead reckoning. The errors of localization using CRISP are still within 1 meter.

5.4.3.3 Counting Steps by CRISP

We also focus on step counting in this experiment. Fig. 5.18 illustrates the step counting and two pedometers on the smartphones. The stems refer to the accumulative error of the number of walking steps computed by the combination approach in CRISP. The remaining two lines refer to the accumulative error of number of walking steps caused by two popular pedometer applications (Noom Walk, ACCUPEDO) from Google Play [50, 48]. CRISP maintains less errors than the other two pedometers in the whole procedure.

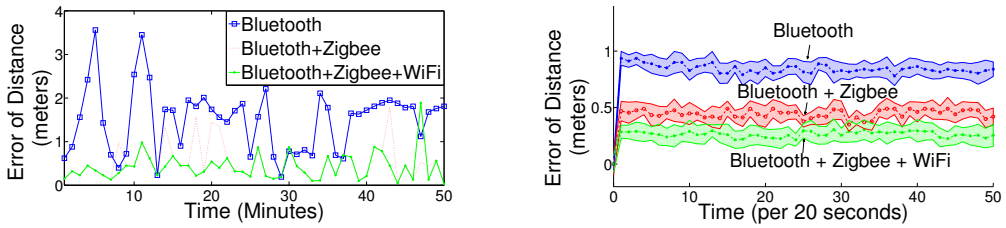
5.4.3.4 Average Speed Prediction Measurement

The above positioning results adopt the Average Speed Prediction (ASP). Fig. 5.19 compares the localization errors of two groups. One uses the ASP, the other does not include AS prediction. The group with ASP outperforms the other group at different time slots. Therefore, Average Speed prediction boosts the proposed approach effectively.

5.5 Discussion

5.5.1 Compare different types of signals

CRISP integrates Bluetooth, ZigBee and WiFi filter to implement triangle calculation for measuring users' locations. To analyze the effectiveness of each technology, we conduct the following evaluation: while maintaining the same experimental environment, as given in Figure 5.20(a), first, we use Bluetooth RSSI without the ZigBee sensor and the WiFi filter to collect RSSI. Second, by adding the ZigBee approach to the Bluetooth approach, we observe the indoor localization results. Third, the green line on the figure denotes the experimental



(a) Comparison of different types of signals

(b) Repeated comparisons of different types of signals

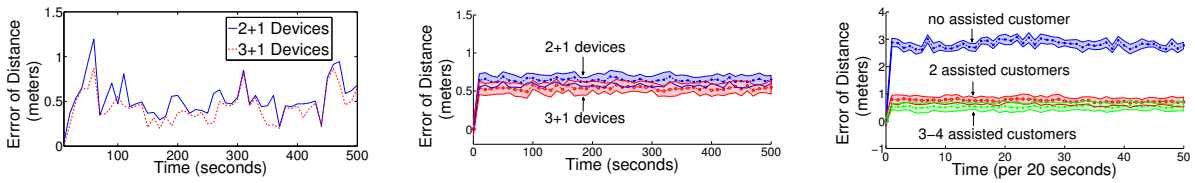
Figure 5.20 Shows Bluetooth RSSI, Zigbee RSSI, and WiFi filter to improve the experiment results, respectively.

results by combining all the three technologies. Then, we simulate our experiments 100 times. The simulation is generated as in the single room evaluation section. The time period is reduced from 45 minutes to 1000 seconds. Figure 5.20(b) presents the localization results by the repeated simulations. As depicted in Figure 5.20, we believe that 1) adding more RSSI samples from ZigBee model and 2) filtering the interferences in the Noise Period by the WiFi filter are helpful for improving the indoor positioning results.

5.5.2 The number of mobile devices encountered influence the localization accuracies

In this section, we discuss whether the number of mobile devices a user encounters can influence the localization accuracy. First, we assume that Alice carries a smartphone and walks freely in one room within 500 seconds. Then, other people help Alice to locate herself by CRISP. When we start this experiment, we set two control groups: 1) the group includes Bob and Carson, who will help Alice to apply triangle computation, and 2) the group contains Bob, Carson, and David to do triangle computation after using "polygon decomposition." According to our observation in Figure 5.21(a), the "3+1" control group has less errors than the "2+1" group. Then, we repeat our experiments 500 times of simulation, shown in Figure 5.21(b). We draw the same conclusion as what we had in the physical experiments.

To further support the above conclusion, a more complex experiment is conducted: a user of CRISP walks in an indoor building. Three traces are generated: 1) a user does not



(a) Observation of different number of devices encountered

(b) Repeated observation of different number of devices encountered

(c) Control groups comparison

Figure 5.21 Number of devices encountered to differentiate localization accuracies.

meet any other users (a user's location is computed by dead reckoning), 2) a user always has other two users assisting him to locate himself by the combination approach 3) a user always has three to four users to help him to locate himself. The trace continues 1000 seconds and we simulate such traces 100 times. In Figure 5.21(c), trace 1) only uses dead reckoning and often has a serious deviation (around 3 meters) from the ground truth. The trace 3) uses a combination approach and encounters more people to achieve the best performance.

Based on the above evaluations, if all users run CRISP on their mobile devices, the localization accuracy of each user can be improved.

5.5.3 AP enhances the accuracy of triangulation model

To validate the function of fixed Access Points (AP), in this subsection, we did the comparison: based upon the existing evaluation in the indoor building, we set three other control groups: 1) deploy 3 APs (2 Bluetooth beacons, 1 WiFi router), 2) deploy 10 APs (7 Bluetooth beacons, 3 WiFi routers), and 3) deploy 15 APs (8 Bluetooth beacons, 4 WiFi routers, 3 Zigbee senders). As shown in Figure 5.22, we can conclude that by using the accurate positions provided by fixed AP, the deviations of CRIPS are further reduced. Besides, more AP can improve the localization accuracies better via providing more correct positions samples.

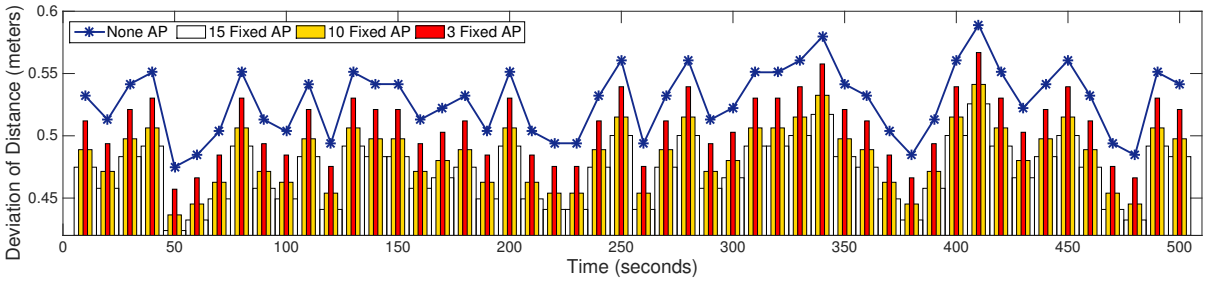


Figure 5.22 By employing more fixed AP, the localization accuracies of CRISP increase gradually.

Table 5.2 WiFi filter detection under different thresholds

	Real Interference	Detected Interference	Misjudged
3dBm	8 times	8 times	23 times
5dBm	8 times	7 times	2 times
10dBm	8 times	2 times	0 times
$\geq$15dBm	8 times	0 times	0 times

5.5.4 Thresholds in CRISP

5.5.4.1 Thresholds of WiFi Filter

In our previous experiments and simulations, we set the threshold of WiFi filter at 5 dBm. In fact, if we set P as the threshold for filtering, if the interference that causes the value of RSSI change is less than P, the interference will be neglected. If we set the threshold of the WiFi filter too low, the normal variation of RSSI values will be judged as the interference, and hence, an optimal threshold of WiFi filter is a key factor. Based on the Samsung Galaxy S5 and Google Nexus 7 devices, we use our approach in a room in the Engineering Building and conduct the experiment as described in the previous section. The only difference is that we test different thresholds of the WiFi filter: 3dBm, 5dBm, 10dBm, 15dBm. After 5 minutes' of experiments, the successful rate of WiFi filter detection is given in Table 5.2.

Real Interference refers to the number of times interference was generated in the experiment; *Detected Interference* refers to the number of times interference was detected by the WiFi filter; *Misjudge Interference* is the number of false positives of interference occured. From the table, we observe when the value of the threshold equals 5, CRISP performs best.

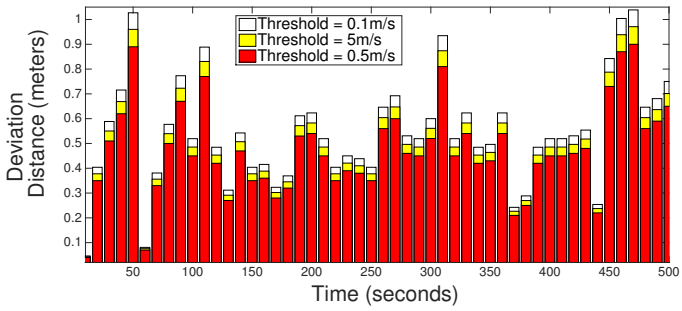


Figure 5.23 Comparison of Thresholds in Average Speed Prediction.

5.5.4.2 Thresholds of Average Speed Prediction

CRISP adopts Average Speed Prediction (ASP) to improve the original dead reckoning. The default value of threshold Δb is $0.5m/s$. Because Δb determines the error range of estimated average speeds, if the Δb is too large, ASP may allow some noises as useful data. On the contrary, if we set the value of Δb too small, some correct average speeds will be disposed. We compare three groups of Δb values, Figure 5.23 plots the default value of Δb ($0.5m/s$) outperforms other two values.

5.5.5 Complexity of our approach

First, we focus on the computational complexity for the mobile devices. To analyze the complexity our approach, we focus on the worst and the best case of CRISP, respectively. For the worst case, we assume all devices are in a reachable range of the Bluetooth adapter (or ZigBee base station), namely, the connection should be created between each of the mobile devices. If the system contains n devices, the complexity of the system is $O(n^2)$. For the best case, a user only accesses other people in different ranges of Bluetooth adapter (or ZigBee base station), the complexity of the system is $O(n)$. Therefore, we confirm the complexity of our approach is acceptable, we can apply it in large scenarios even if the number of devices is not small. Additionally, although the tasks deployed on the server may be more complex, it still can be processed easily because of the strong computation capabilities of

the cloud server. Therefore, we believe CRISP is a low-complexity and light-weighted indoor localization approach.

5.6 Conclusion

We present a RSSI based indoor localization system called CRISP. Different from traditional indoor localization systems, a user walks in an indoor environment and opens the Bluetooth scanning option on a smartphone. The smartphone interacts with other smartphones and exchanges RSSI values. CRISP not only improves the devices' localization accuracies, but also provides the extra benefits - the number of walking steps for the user who holds a smartphone.

In CRISP, we build relations between RSSI and distances for different mobile devices. CRISP uses geometry computation to reduce the errors caused by dead reckoning. By our experiments and evaluation in the Engineering Building of Michigan State University, we show that if a walking user who carries a mobile device and uses CRISP in a building, and if he/she encounters other people using CRISP, the localization results will be more accurate. The range of error is within 1 meter. We combine the ZigBee RSSI values to the RSSI obtained from Bluetooth to collect more samples, and use a filter that is based on WiFi. Both of the two technologies improve the localization accuracy in our evaluation. With known location information, CRISP can count a walker's steps and reduce the common errors caused by smartphone based pedometers.

CHAPTER 6

EFFECTIVE INDOOR POSITIONING WITH ASSISTANCE FROM ACOUSTIC SENSING ON SMARTPHONES

6.1 Introduction

By using crowd sourcing and opportunistic sensing approaches, indoor localization systems do not rely on the pre-deployed scenarios. However, once the mobile devices turn on their WiFi or Bluetooth adapters or other hardware to communicate, the non-reliable information might be shared between devices by the building connections. If these information including deviated location, the localization accuracy will be reduced. In addition, the time of scanning and building connection is time consuming. Sometimes, when samples are collected by mobile devices, the users of devices move to other locations. Therefore, this leads to our conjecture: *Can we provide a passive indoor localization system with real time and fault tolerance features based on the daily uses of smartphones?*

In this chapter we propose SilentWhistle [59], a light-weight indoor positioning approach that leverages acoustic sensing and dead reckoning to obtain the users' locations. SilentWhistle improves its performance in terms of accuracy, pervasiveness, and deployment without any hardware modifications of smartphones. We use the accelerometers on smart devices to generate the motion traces of users initially. Considering the accumulative errors of motion traces, our goal is to employ acoustic sensors on smartphones to enhance the positioning accuracy. In fact, translating this idea-sketch into a prototype still faces a variety of challenges: (1) even if acoustic sensors can detect the sounds in an indoor environment, how do we use the acoustic information that humans cannot hear to locate the users' positions accurately? (2) how do we deploy the system just using smartphones without extra devices? (3) how does the system filter incorrect location messages when they conduct Machine-to-Machine (M2M)

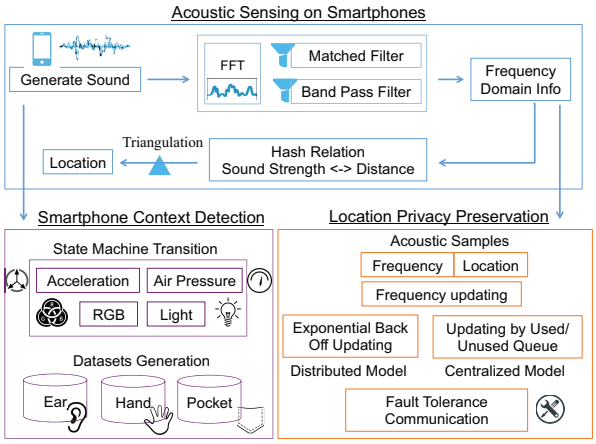


Figure 6.1 The framework of SilentWhistle.

communications? (4) SilentWhistle needs to reduce the noise caused by Doppler effects, background noise, and some exceptional samples from accelerometers. This paper addresses these problems step by step and evaluates the system by real scenario studies. Our extensive evaluation results indicate the accuracy range of SilentWhistle can achieve 0.88-1.24m.

The main contributions of SilentWhistle are as follows:

(1) SilentWhistle leverages the off-the-shelf acoustic sensing information on smartphones to improve the localization results computed by dead reckoning.

(2) We identify an opportunity to avoid incorrect location information spreading among M2M communications.

(3) By applying position context detection, the proposed system is light-weight and passive for users.

6.2 Preliminaries

In this chapter, we generate user's motion traces by dead reckoning. Then, we improve the dead reckoning by analyzing the average speed of user's motions as what we mentioned in previous chapters.

Sound intensity is defined as the sound power per unit area. The sound intensity from a

point source of sound will obey the inverse square law as formula (6.1). P refers to the source power. $4\pi r^2$ is the sphere area. I is the sound intensity. Therefore, the sound intensity is highly related to the distance between source of sound and the sound receiver.

$$I = \frac{P}{4\pi r^2} \tag{6.1}$$

The most common approach to sound intensity measurement is to use the decibel scale.

6.3 Design of SilentWhistle

6.3.1 Overview of SilentWhistle

As shown in Fig. 6.1, our system contains the following main parts:

(1) The sender smartphone generates the sound at approximately 20KHZ (people cannot hear). The receiver smartphone can obtain the sound strength with corresponding sound frequencies. We leverage FFT to transfer the time-domain signal to sequential-domain signal and filter the noises by band-pass filter and matched filter.

(2) A smartphone can generate its own trace by dead reckoning. However, there still exists some distance deviations. Based on the obtained sound strength and pre-trained relation, we transfer the sound strength to distance and use triangulation to reduce the errors caused by dead reckoning.

(3) Our approach proposes two types of frequencies assignment methods: 1) a centralized model exchanges the frequencies by a cloud server. 2) a distributed model exchanges the frequencies by device to device and use an exponential back off algorithm. By broadcasting acoustic information, SilentWhistle has the property of fault tolerance by to avoid accepting some incorrect location messages.

(4) In addition, we use the smartphone position context to distinguish the relative positions of smartphones to further improve the localization accuracies. A coloring strategy extends

Figure 6.2 Sender and Receiver of sound in SilentWhistle.

Silent Whistle to the large room.

6.3.2 Sound Generation and Detection

M2M communication is a common mechanism for crowd sensing. However, common communications methods are not satisfactory. WiFi and Bluetooth need a long time to broadcast and build connections between discovered devices. When mobile devices broadcast their information, the identifiers are exposed to other devices. Moreover, since the information shared between devices are often not reliable, it may cause serious errors for the computational results of the whole communication system. Therefore, a new communication channel on a smartphone for crowd sensing is required.

Most modern smartphones are able to generate sound from 20HZ to 22KHZ. The sound between 18KHZ and 22KHZ can not be heard by most adults. Therefore, the sound generators in SilentWhistle produce the source of sounds with frequencies from 18KHZ to 22KHZ. Namely, the users of SilentWhistle can leverage such sound to communicate and not suffer from the hearable noises.

If one smartphone generates the sound with a certain frequency (18KHZ and 22KHZ), another smartphone can detect such sound by its microphone. The Fast Fourier Transform (FFT) resolves a time waveform into its sinusoidal components. The FFT takes a block of time-domain data and returns the frequency spectrum of the data.

6.3.3 Filter of Acoustic Information

Considering that we choose the frequency from 18KHZ to 22KHZ on a smartphone, we need to avoid the following types noises: 1) background noises, 2) noises caused by Doppler effect, and 3) other potential noises.

For reducing the background noises, we choose a band-pass filter. A band-pass filter [28] passes frequencies within a certain range and rejects frequencies out of that range. In our design, we filter the sound that are out of the range between 18KHZ and 22KHZ.

In addition, we further filter the background noises and other potential noises by a matched filter. A matched filter [23] is obtained by correlating a known signal, or template, with an unknown signal to detect the presence of the template in the unknown signal. In SilentWhistle, we can detect the sound source from other devices in the range between 18KHZ to 22KHZ. Once smartphone can detect the sound's strength is greater than a threshold and the frequency range is between 18KHZ to 22KHZ, we will set the detected frequency as the "matched" value, and filter the values that are not matched.

6.3.4 Filter of Doppler effect

The Doppler effect (or Doppler shift) [12] is the change in frequency of a wave (or other periodic event) for an observer moving relative to its source. Once a user holds a smartphone and moves fast, it may cause the Doppler effect and change in frequency of a wave. Since our approach is based on the frequency of sound, we need to avoid the influence of Doppler effect.

As shown in formula (6.2) and (6.3), f is observed frequency, f_o is emitted frequency, v_w refers to the velocity of the wave in the medium. V_r is the velocity of the receiver relative to the medium; positive if the receiver is moving towards the source. V_s is the velocity of the source relative to the medium; positive if the source is moving away from the receiver. In our design, when Δv (Δv=V_s - V_r) is greater than $3m/s$, we will focus on the change in frequency, which may cause the incorrect detection. Since the velocity of each user can be

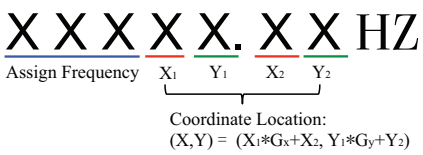


Figure 6.3 The format of frequency message.

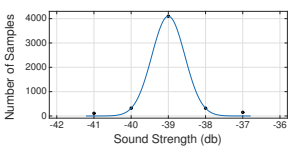
computed by obtained acceleration on smartphones, the value of Δf also can be estimated by formula (6.2) and (6.3). When user's smartphone computes the detected frequencies, we will consider the Δf. We will adopt the frequency by deducting the change in frequency caused by Doppler effect.

$$f = (1 + \Delta v/c) \times f_o \tag{6.2}$$

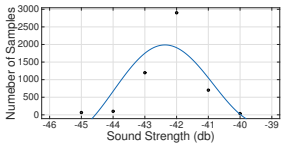
$$\Delta f = \frac{\Delta v}{c} f_o \tag{6.3}$$

6.3.5 Leveraging Triangulation for Improving Localization

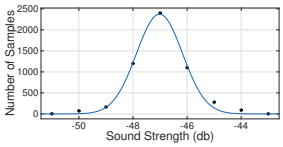
Since dead reckoning is not enough to provide satisfactory location information, we adopt triangulation to further locate a user's position by knowing other detected devices' locations and sound strength: each smartphone computes its own position by dead reckoning initially, and then shares it with nearby smartphones by M2M communication. Furthermore, the signal strengths of multiple radios are used on smartphones to estimate distances between the devices. When individual smartphones may provide some positioning (possibly inaccurate) information, opportunities of accuracy improvement occur when several smartphones cooperate and share position information. Accuracy may improve as multiple iterations of information sharing and computations are made.



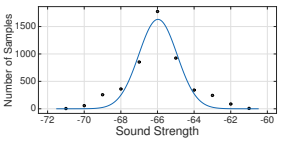
(a) Half meter observation



(b) One meter observation



(c) Two meters observation



(d) Four meters observation

Figure 6.4 Relation between sound strength and distance.

6.3.6 Format of SilentWhistle Message

As shown in Fig. 6.3, the last four digits present the location of a mobile device. The three leading digits are assigned frequencies by our system. Once an other user's smartphone detects the sender's frequencies by FFT, the sender's location information can also be obtained by the receiver. This procedure is similar to encoding and decoding: for a certain indoor room, we can transfer the map into a matrix. First, we cut the map into $X_1 \times Y_1$ grids. The number of grids on length side is X_1, the number of grids on width is Y_1. For each grid, we further cut them into sub-grids. X_2 refers the number of sub-grids that is on the length side of a defined grid. Y_2 refers the number of sub-grids that is on the width side of a defined grid. When we adopt a decimal system, G_x and G_y refer to 10.

6.3.7 How the acoustic information improve localization accuracies

Initially, each smartphone can use dead reckoning to compute its own locations. Even if there exist some deviations on the smartphone, by analyzing the last four digits of a received acoustic message, the user can obtain the location of encountered devices. In the previous section, we introduced how users improve localization accuracy by sharing the location information and do triangulation computation. Since the strength of sound can be detected by each device, we transfer the strength to distance by a hash map relation (distance - sound

strength). We pre-trained these relations as shown in Fig. 6.4. In the experimental groups, we record the samples including the distance and sound strength information. Then, we use the average value in the normal distribution as the sound strength for a certain distance. All of these relations are stored in hash maps. Based on the estimated distance from sound strength, after running the triangulation approach many times, the errors of distance are eliminated effectively. The further results are demonstrated in the evaluation section.

6.3.8 Frequency Assignment

6.3.8.1 Centralized Architecture

In this section, we introduce two architectures to assign frequency ranges for smartphones. The central server of SilentWhistle stores two types of information. The first type is the real-time location of each smartphone. The smartphone will update its own location information and uploaded them to the central server continuously. The second type of information is the sound frequencies for each smartphone. As shown in Fig. 6.5, the initial frequency queue is stored on the cloud server. When a user enters a room, the last four digits can indicate the user is close to a entrance of a room/hallway, the server will assign a frequency to him/her immediately. The assigned sound frequency will be stored on the used queue. The remaining unused frequencies are stored in the unused list. Once the user leaves the room or turns off the acoustic sensing option, the assigned frequency will be released and go back to the unused queue.

6.3.8.2 Distributed Architecture

In practice, some indoor buildings do not include remote servers to support smartphones' location based services. We provide a mechanism to distribute localization tasks on the remote servers to users' smartphones. When a users enters a room, he/she can obtain initial position from a fixed beacon rather than by a cloud server deployed at the entrance. Then,

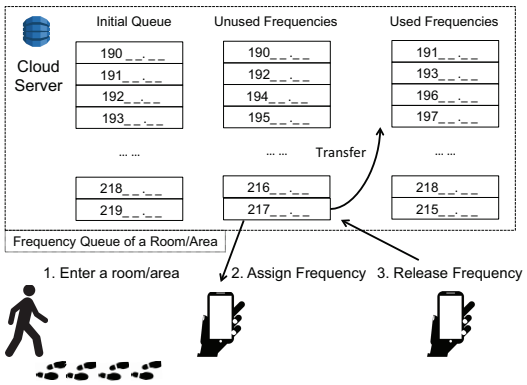


Figure 6.5 The Frequency assignment and releasing.

the users of a distributed model have two functions: 1) computing its location by dead reckoning and 2) using triangulation to calibrate the localization errors. The users update the frequencies periodically. In each area/room, different users choose their frequencies from the previous frequencies tables that are stored in the smartphone rather than on the server. Sometimes, a user needs to change his/her frequencies when he leaves or enters a room. How to set the priority of items in the waiting table? We adopt an exponential back-off method to achieve our goal. Exponential back-off is an algorithm that uses feedback to multiplicatively decrease the rate of some process in order to gradually find an acceptable rate. In our design, after k times of encountering, a random number of slot times between 0 and 2^k- 1 is chosen. For the first encountering, each sender will wait 0 or 1 slot times. After the second encountering, the senders will wait anywhere from 0 to 3 slot times (inclusive). After the third encountering, the senders will wait anywhere from 0 to 7 slot times (inclusive), and so forth. As the number of retransmission attempts increases, the number of possibilities for delay increases exponentially. At the same time, the waiting list will be updated (sorted) based on the length of waiting time. Once a user hopes to change his frequency, he/she will choose the item with shortest waiting time on the table.

6.3.9 None Connection Model

Different from traditional M2M communication by Bluetooth or WiFi, SilentWhistle provides a method to enhance the localization accuracy by acoustic communication. As shown in Fig. 6.6, for tradition Bluetooth or WiFi communication, different devices have to build the connection to share the location information and RSSI. Namely, for some scenarios, when a smartphone of a normal user with obvious location deviation, once the device sends an incorrect location message to another user, it is difficult to distinguish it from the side of the other device when the location is in RSSI range. Since we use triangulation to localize users, the group of users' location accuracies will be reduced. In triangulation, users can compute its own position by the other two devices. Since one user (Alice) can receive the other two users' positions and RSSI (transferred to distance), the positions and distances can be abstracted as the centers and radiums of two circles. The coordinates of the two centers are (x_1, y_1) and (x_2, y_2). The radiums are r_1 and r_2. We assume the intersection is (x, y). Then, we assume $x = r_1 cos\Theta + x_1$ and $y = r_1 sin\Theta + y_1$. In order to simplify the expressions of intersection, let $a = 2r_1(x_1 - x_2)$, $b = 2r_1(y_1 - y_2)$, $c = {r_2}^2 - {r_1}^2 - (x_1 - x_2)^2 - (y_1 - y_2)^2$. After solving the equations, we can obtain:

$$\Theta = ((q^2 - 4pr)^{1/2} - q)/2p \tag{6.4}$$

$$p = a^2 + b^2, q = -2ac, r = c^2 - b^2 \tag{6.5}$$

Therefore, if the value of x_1 and y_1 include some deviations, the final result of (x,y) will be changed.

Then, we need to compute the intersection as the estimated position of Alice. One user sends the location information that includes some errors. Such errors will be computed when we compute Alice's position (intersection of two circles). Therefore, when Alice uses the computation position by triangulation, it also includes certain errors.

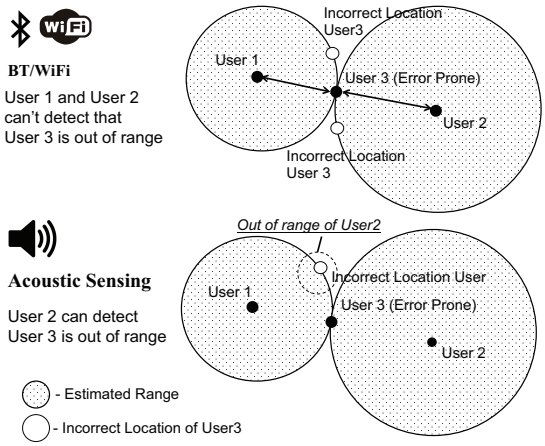
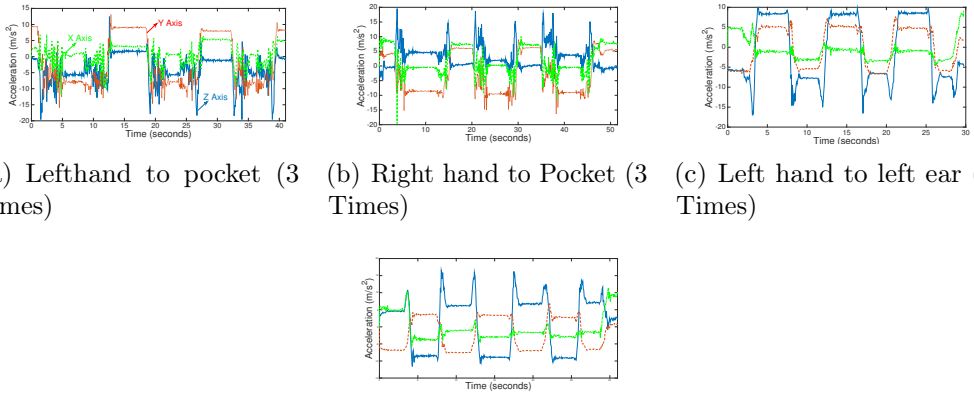


Figure 6.6 The comparison between different connection models.

In SilentWhistle, each user broadcasts its location information to other users. Once a user broadcasts incorrect location information, the other two users will share the information. Both of the users will check whether the received location message is in the radio range. Once a user confirms the location message is out of range, he/she will report this situation to the remote server. The remote server will forbid the suspicious device to broadcast its information. In the distributed model, he/she will dispose the location information broadcasting by the suspicious device and forward the ID of the suspicious device to the other users by the acoustic information. Therefore, our approach can further enhance the localization accuracy by using the connectionless communication model.

6.3.10 Adopt Coloring Strategy to Cover Large Room

For a certain room, the number of smartphones may exceed the maximal number of devices that a channel list could contain. Considering the range of acoustic sensing is 5 meters, we need to partition the larger room into some smaller areas. The size of the subarea is 5m×5m. SilentWhistle chooses a greedy strategy of a coloring problem. First, we abstract each area (5m×5m) as a vertex on a graph. The adjacent vertices are connected by an edge. Then, we color the first vertex with the first color. In SilentWhistle, a color represents a group of



(a) Lefthand to pocket (3 times)

(b) Right hand to Pocket (3 Times)

(c) Left hand to left ear (3 Times)

(d) Right hand to right ear (3 Times)

Figure 6.7 Acceleration for different context of the smartphone.

frequencies. We do the following for the remaining vertices: consider the currently picked vertex v and color it with the lowest numbered color that has not been used on any previously colored vertices adjacent to it. If all previously used colors appear on vertices adjacent to v, assign a new color to it. Based on the coloring strategies, we reuse some groups of frequencies to cover a larger area.

6.3.11 Smart Phone Context Detection

Considering that acoustic sensing information is used to enhance localization accuracies, we need to hold the smartphone in the hand or in an open area. In SilentWhistle, we propose an approach to distinguish three common states 1) close to ear, 2) in the pocket/bag, and 3) in the users' hands.

We build a state machine to describe the states on the smartphone. In the state machine as Fig. 6.9 and Table 6.1, four types of information are leveraged: 1) the RGB color of picture captured from camera sensor; 2) the lumen from camera sensor; 3) the acceleration obtained on a smartphone; 4) air pressure recorded by a barometer. As Fig. 6.7, the accelerations are different when it is on different sides or places (ear, hand, and pocket).

When a users of SilentWhistle change the relative position of smartphone, namely, if the

(a) Air pressure values (b) Smartphone in pocket (c) Smartphone in hand (d) Smartphone close to ear

Figure 6.8 The position of smartphones.

Table 6.1 The state machine of smart phone context

	Lumen	Color (RGB)	Acceleration	Barometer
P-H	+	+	Z+	N/A
H-E	N/A	N/A	Z-	+
E-H	N/A	N/A	Z-	-
H-P	-	-	Z+	N/A

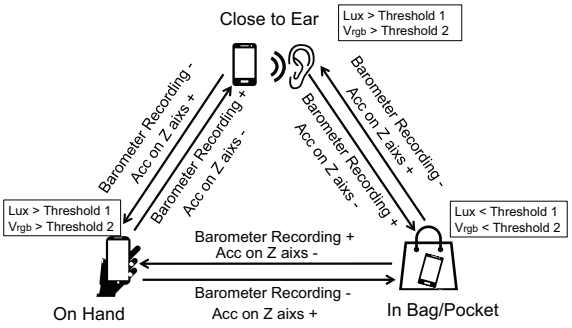


Figure 6.9 The contexts transfermation.

states will be transferred, there are two types of conditions that should be satisfied: 1) the new states's RGB values and Lumen values should be greater than certain values. The values of RGB and lumen will be discussed in the evaluation sections, 2) before the state transition, the average change of acceleration on Z axis should be increased or decreased based on Fig. 6.7; and 3) the air pressure detected by barometers should be varied or decreased based on Fig. 6.8.

After distinguishing the three states of smartphones, our system builds three separate datasets for three different states. Since our system uses the distance and sound strength

relation to compute locations, the data from dataset generated in the pocket/bag will not be used for triangulation because of interferences. Additionally, considering the accelerations are obvious different when smartphone are in user's hand and pocket, the dataset of on the hand will not be applied for dead reckoning.

6.3.12 Energy Saving Strategy

Considering the acoustic sensing on smartphones is energy consuming, when we select centralized model, the acoustic sensing function is only turning on when a user of SilentWhistle can detect two nearby users based on the location information computed by dead reckoning and stored on the remote server. When the user cannot detect any nearby user, the acoustic sensing function will turn off for energy saving.

6.4 Case Study and Evaluation

In this section, we attempt to answer two main questions:

- By applying SilentWhistle, what is the accuracy of indoor localization?
- How acoustic information, phone context detection, and coloring strategy help dead reckoning based indoor localization?

6.4.1 Experimental Setting

We build a prototype of SilentWhistle on the Android platform (version 5.0) and evaluate its performance on two types of smartphones (Samsung Galaxy S7 and Google Nexus 5). The smartphones are equipped with standard sensors that include accelerometers. The user's motion trace is computed by 5HZ. Initially, we focus on the case study in a single room. The size of the room is 20m $\times$ 6m. There are 4 users who walk freely in the room and turn on the

Table 6.2 The detections of in/out positions of smartphones

Truth / Detected	Pocket	Hand	Ear
Pocket	291	8	1
Hand	7	287	6
Ear	0	5	295

sound generators on the smartphone. We set 4 initial frequencies: 19000.00HZ, 19500.00HZ, 20000.00HZ, 20500.00HZ. All four frequencies are stored in the unused queue.

In this section, we introduce the following metrics to measure our system:

1) Euclidean distance: measure the distance between the ground truth and estimated location;

2) p_{ij} value: as shown in formula (6.6), the parameter p_{ij} refers to the probability of obtaining the user i's location on the side of user j via Bluetooth, WiFi or acoustic communications.

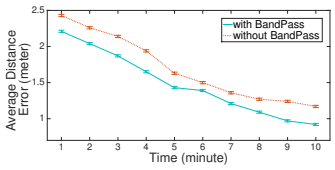
$$p_{ij} = \frac{\text{received samples including i's location on j side}}{\text{i's broadcasting samples}} \tag{6.6}$$

6.4.2 Evaluation of Context Detection

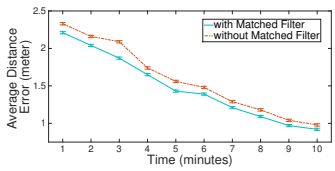
Before we evaluate the localization performance, we focus on the Context Detection. Through applying our proposed approach, we test 300 cases on three volunteers. The results are shown in Table II. Based upon the detection results, we verify our proposed approach is able to distinguish the relative positions of smartphones effectively. Less than 10 samples are misanalyzed.

6.4.3 Evaluation of Noise Filters

In SilentWhistle, we adopt bandpass filters and matched filters. As shown in Fig. 6.10, both of the two filters are able to improve the localization accuracies.

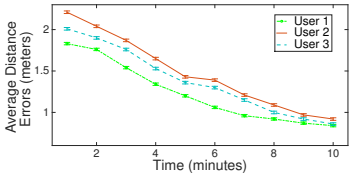


(a) Localization results comparison of bandpass filter.

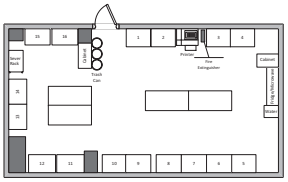


(b) Localization results comparison of macthed filter.

Figure 6.10 The evaluation results of filtering methods.



(a) The average errors within 10 minutes



(b) The layouts of eLANS lab

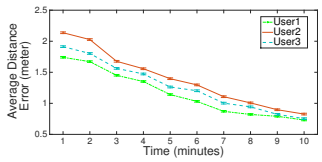
Figure 6.11 The case study in single room.

6.4.4 One room measurement by centralized model

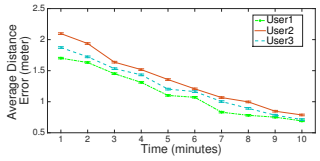
In the measurements in a certain room, 3 users walked and stopped freely for 10 minutes. Then, we repeated the case study 10 times. As shown in the Fig. 11, we conclude that 1) the accuracies of our approach can be achieved within 0.93 meters and 2) the accuracies are improved with the time increasing. Based on the initial step, we add four beacons that can communicate acoustic information. We repeat the previous experiment. After 10 minutes observation, Fig. 6.12 indicates the localization accuracy is further improved. Therefore, in our approach, more beacons in the proposed system can provide more opportunities to improve indoor localization accuracies.

6.4.5 Fault tolerance analysis in one room (centralized)

When we analyze the fault tolerance feature of the SilentWhistle, we specify that one user will send incorrect location information to the other two users. The distance between ground



(a) The average errors with two beacons.



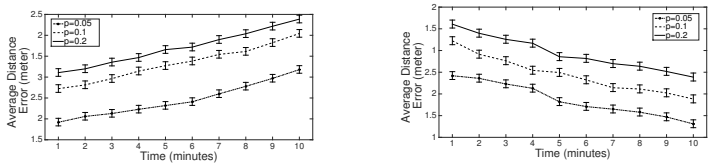
(b) The average errors with four beacons.

Figure 6.12 The case study in single room with beacons.

truth and sending incorrect location is 1 meter. Based on sensing the location information by Bluetooth or WiFi information, the average location accuracies of each users (user1, user2, and user3) are estimated as Fig. 6.13(a). In Fig. 6.13(a), we set the value of P as 0.05, 0.1, 0.2. The errors are difficult to reduce because of the connections. Then, by keeping the same experimental conditions, by using our approach (leveraging the acoustic information and turn off Bluetooth and WiFi adapters), as shown in Fig. 6.13(b), the broadcasting model without the connections outperforms. The sampling frequency of acoustic broadcasting is 10HZ. Namely, by cooperating with different smartphones' users, the incorrect location information can be reduced effectively in SilentWhistle.

6.4.6 Fault tolerance analysis in one room (distributed)

The above analysis and evaluation are based on a centralized model. In our room observation, we use the exponential back-off algorithm for arranging the used list of frequencies. The length of each waiting slot is 10 seconds. When users enter the room, the initial position will be provided by an audio generator that is near the door. The relations between distance and sound strength are stored on each smartphone as hash tables. We keep the same experimental conditions as in the centralized evaluation. As shown in Fig. 6.14, the localization results by the distributed model are close to the centralized model by using SilentWhistle. Therefore, the fault tolerance features in both centralized and distributed models are able to assist triangulation localization.



(a) Localization results by using connection model.

(b) Localization results by using unconnection model.

Figure 6.13 The case study by acoustic sensing channel.

Table 6.3 Location Preservation Results in a Large Single Room

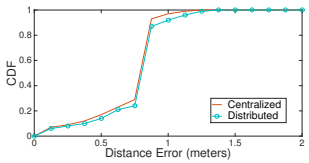
Running Time	**5min**	**10min**	**15min**	**20min**
Centralized	1.21m	1.12m	1.02m	0.98m
Distributed	1.33m	1.23m	1.09m	1.01m

6.4.7 Large Single Room Simulation

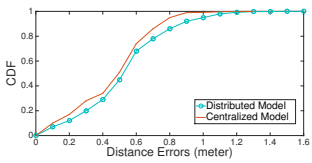
Based on the single room measurement, we apply the coloring strategy in a large room (50m $\times$ 50m) that can be divided into 100 subareas (5m $\times$ 5m). The indoor map can be transferred to a graph with 100 vertices. We assume that there are 150 people to walk and stop freely in the map. Considering that our approach needs 100 different groups of frequencies, the length of each group may be close to each other. We reuse some groups by coloring. Five users send incorrect location messages continuously. The time length of the simulations are 20 minutes.

By adopting the distributed model mode, we observe the localization errors in Table 6.3. We believe the 1) users in both the distributed model and the centralized models can localize themselves successfully even if 5 users broadcast incorrect location information and 2) the centralized model has better performance than the distributed model.

We compare the system by using a coloring strategy and without the coloring strategy. As shown in Fig. 6.15 (a) and (b), we can draw the conclusion that the coloring strategy can enhance the location accuracies by reuse of the frequencies in a large area.

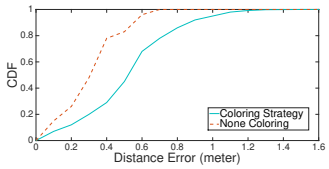


(a) Average localization results by different models (10 minutes).

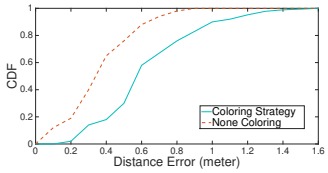


(b) Average localization results by different models (30 minutes).

Figure 6.14 The comparison between distributed model and centralized model.



(a) Average localization accuracy by centralized model.



(b) Average localization accuracy by distibuted model.

Figure 6.15 The extended simulation for multi-room .

6.4.8 Multi-room Scenario Measurement

We extend our evaluation from a single room to a multi-room scenario. We conduct the experiment as follows: as shown in Fig. 6.16, four users walk around the 2rd floor of Engineering Building in Michigan State University. The scenario contains four rooms and three related hallways. We deployed beacons at each entrance/exit on the floor plan. We keep the same experimental conditions as single room case, one user played role of broadcasting the location information within 1 meter deviation in the whole procedure. As the control group illustrated in Fig. 6.17, by applying SilentWhistle for 30 minutes, we draw the conclusion: 1) the average indoor localization accuracy of SilentWhistle users can achieve 0.88 meter; 2) on the group using Bluetooth/WiFi connection approach, the distance deviations increase gradually. The average obtained location results are not satisfactory.

(a) Room Measurement (b) Hallway Measurement

Figure 6.16 Measurement in a building.

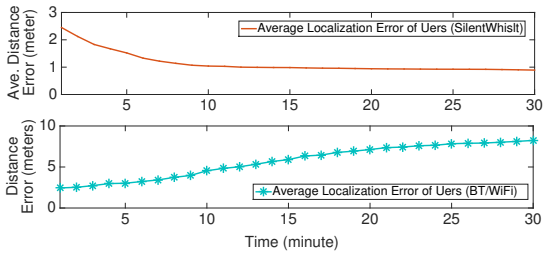


Figure 6.17 Localization results in the multi-room scenario.

Table 6.4 Localization results under different number of users

	1 User	**2 Users**	**3 Users**
Ave. Localization Accuracy (5min)	1.23m	1.10m	0.92m
Ave. Localization Accuracy (20min)	1.03m	1.00m	0.89m

6.5 Deep Dive into SilentWhistle

In this section, we will discuss some important factors that influencing the performance of the proposed system. To analyze these factors, we still use the multi-room scenario.

6.5.1 Number of users

For the real scenario, we change the number of users of SilentWhistle. Based on evaluation on previous sections, we add two users and repeat the experiment. Then, we analyze the data received from 1 users, 2 users, and 3 users separately (not including the user broadcasting fault locations). From Table. 6.4, we believe the more normal users are able to enhance the localization accuracies on the user's side by cooperative confirmations among them.

Table 6.5 Length of different time slot for exponential back-off

Length of Time Slot	**1s**	**5s**	**10s**	**30s**	**1min**	**5min**
Est. Localization Err.	1.01m	0.92m	0.90m	0.93m	0.98m	1.03m

6.5.2 Frequency of changing acoustic frequencies

In the evaluation section, the used and unused waiting queues store the acoustic frequencies for all the smartphones. In the centralized model, when a user does not change his position within 3 minutes, the system will re-assign a new frequency to him/her. In this section, we observe the average localization results by using a different waiting time.

When we set the waiting time as 0.25, 0.5, 1, and 4 minutes, the average localization results based on the experiment in section 6.4, after running SilentWhistle for 30 minutes (p=0.2), the estimated average error of the system is: 0.89m, 0.91m, 0.93m, and 0.97m. Therefore, the higher frequency of acoustic frequencies changing can reduce the accuracy the attacker's computation. In the distributed model, we use the exponential back-off algorithm to update the waiting queue. In the previous evaluation, we set the time slot for back-off exponential to 10 seconds. As shown in Table 6.5, we test six different values of time slots 1s, 5s, 10s, 30s, 1m, and 5m. When we choose 10s as the time length, we can achieve the best results. Actually, if the time slot is too large, it cannot update the waiting queue in time. Once the time slot is too small, the function of exponential back-off will be reduced.

6.5.3 Multiple Incorrect Senders

In our previous experiment, the whole system includes one attacker. For some situations, there might be more than one attacker, which intentionally sends incorrect information. Moreover, the attackers can collude and share the obtained location information from other users. Therefore, we also observe the performance of SilentWhistle containing multiple users sending incorrect location messages. Based on the evaluation for the multi-room study, we

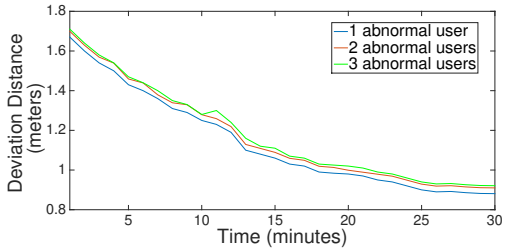


Figure 6.18 Results of different number of users sending fault locations.

add 1 or 2 more attackers in the system. All the attackers can collude. We run the evaluation as the same conditions. As in Fig. 6.18, we conclude that even if more users broadcast false locations that can reduce the estimated location accuracies, but the results are still acceptable.

To further analyze the number of upper bound of abnormal users in our system, these problems can be abstracted as a Byzantine General problem [35]. Based on the classical solution: it can be shown that if n is the number of devices in total, and t is the number of abnormal users in that n, then there are solutions to the problem only when n>3t and the communication in the systems are reliable.

6.5.4 Device Diversity

Although our experiments are applied on Samsung Galaxy smartphones, all the smartphones that can generate the 20KHZ frequency sounds can be suitable for SilentWhistle. By pre-training the relation between sound strength and distance, the hash tables are stored in cloud server/smartphone. Considering the differences for audio generator and microphones for different types of smartphone, when we add a new brand of smartphone into our system, the mapping relation should be pre-deployed.

6.5.5 Difficult to Track

The communication model is based on acoustic sensing and hides the real physical ID of a user (such as MAC address, IP address). Therefore, the traditional tracking approach cannot be applied. Existing tracking solutions can detect the physical IDs of devices even if the IDs can be changed at the software level for a certain period of time. They can still be detected since the ID on the hardware cannot be modified. Moreover, when one smartphone broadcasts via Bluetooth/WiFi by a connectionless model, even if some users can hear Bluetooth positions without needing to expose their own identities, the smartphone that broadcasts information has to expose its unique ID. For SilentWhistle, only the attacker can remember the user by his/her features such as face, clothes. The users of SilentWhistle cannot be tracked by IP and MAC addresses.

6.5.6 Overcome the long-time device discovering

SilentWhistle receives RSSI values without the long time scanning by WiFi and BLE adapters. For the smartphones we used in our evaluations, the scanning time of BLE and WiFi beacons are from 3 to 15 seconds. If the approach uses WiFi and BLE rather than acoustic sensing, the walking users of smartphones may miss each other and did not discover the device. Namely, triangulation cannot be performed by BLE and WiFi sometimes. Considering using acoustic sensing, SilentWhistle does not need long time to scan and detect device.

6.6 Conclusion

This section proposed SilentWhistle, an accurate indoor localization system leveraging acoustic information and preserving the privacies of users effectively. Each use of SilentWhistle generates a sound that cannot be heard by people but can be detected by a smartphone. By using the relation between distance and sound strength, SilentWhistle uses the triangulation to improve indoor localization accuracies from initial dead reckoning. Considering protecting

the location privacy of user's devices, we exchange the sound frequencies of smartphone periodically. Based on centralized model and distributed model, the senders of obvious incorrect locations will be detected and disposed in our system.

CHAPTER 7

MOBILE ROBOTS ASSISTED INDOOR LOCALIZATION

7.1 Introduction

In the previous chapters, we have introduced smartphone based indoor positioning approaches [3, 32, 75, 85, 42, 39, 57]. By leveraging the cooperation among different smartphones, we improve the indoor localization results computed by dead reckoning effectively. Some indoor environments, such as convention centers, museums, and hospitals provide various location services. Mobile robots may be able to supply certain services to users. These robots have the abilities to establish their own positions and orientations within the frame of reference. By using a camera or an infrared sensor, the robot can find its route in an indoor environment and avoid dangerous situations by SLAM approaches. Besides, the robots can calibrate their positions computed by SLAM to ensure their high localization accuracies [13, 73, 44].

Motivated by the increasing availabilities of smartphones and mobile robots, we present AirLoc (Adopting mobile robots to assist indoor Localization of smartphones), a low-cost, highly-accurate and large-scale indoor localization approach that integrates the off-the-shell smartphones with the mobile robots. By installing a tablet, the proposed mobile app and a known map on a mobile robot, the mobile robot can improve a smartphone's localization accuracy. The robot is inexpensive, and its movement can be calibrated to have accurate position information. The robot leverages Bluetooth broadcast to send its correct location to the users' smartphones. AirLoc provides a path for a robot so that smartphones might minimize the deviations between the ground truth and estimated positions by interaction with the robot: the robot collects Bluetooth RSSI from smartphones in different rooms. We classify different rooms into different crowd density levels by the RSSI values. Higher crowd density rooms should be served more often. By utilizing different crowd density levels, we

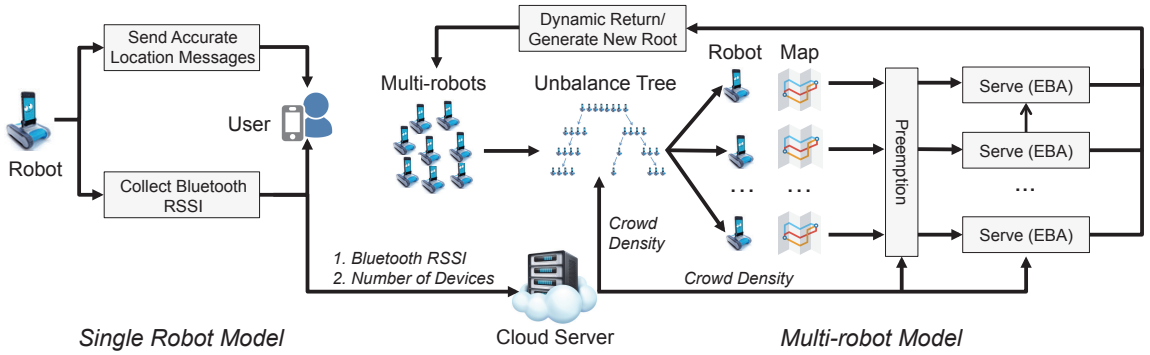


Figure 7.1 Overview of AirLoc

design the Edge-Based Algorithm (EBA) and Node-Based Algorithm (NBA) to generate a robot's moving route.

Since one robot takes a long time to travel all the rooms and the crowd density may change fast, the collected crowd density information is inaccurate. To further strengthen the performance and applicability of AirLoc, we extend the single robot approach to the multi-robot model. AirLoc uses a cloud server to store the Bluetooth RSSI submitted by multi-robots and update the crowd density levels continuously. When the mobile robots compute their serving paths, the crowd density levels can be accessed by WiFi. AirLoc organizes multi-robots by an unbalanced tree, which is dynamic and low complexity. In each layer of the unbalanced tree, the robots are divided into two sub-groups. The sub-group containing more robots serves the area with higher crowd density. In the base case of a serving tree, each robot obtain its serving area by the Distance/Density First Algorithm (DDFA) and serve it via EBA.

In our field study and large scale simulation, AirLoc improves the smartphones' localization results successfully with low cost and acceptable complexity. This approach can be potentially applied in more indoor buildings to provide practical applications, such as in-store navigation, location-based healthcare.

The main contributions of this paper include the following aspects:

1. To the best of our knowledge, AirLoc is the first of it kind to i) proposed mobile robots to interact with smartphones to help indoor localization; ii) design and evaluate a system to organize multi-robots for improving the smartphones' positioning information in real indoor environments.

2. To apply AirLoc on large scale indoor environments, based on the crowd density distribution, we design an algorithm to generate the optimized serving route for a single robot. AirLoc exploits an unbalanced tree and Distance/Density First Algorithm (DDFA) for low complexity and costs.

3. AirLoc updates the crowd density levels continuously to further increase the smartphones' localization accuracies.

7.2 Preliminaries

7.2.1 Employ mobile robots to improve indoor localization

The accuracy is a main criterion for evaluating indoor localization approaches. Although RFID readers are expensive, LANDMARC [51] (Location Identification based on Dynamic Active RFID Calibration) approach uses extra fixed location reference tags to calibrate the localization results computed by the tracking tag. LANDMARC improved the localization accuracy successfully, but it has limitations: 1) users have to carry the RFID tags when they move in indoor environments 2) fixed location reference tags are needed in indoor environments where they provide Location Based Services.

Users of smartphones can obtain location services without any extra devices. They employ sensors (accelerometer, gyroscope) on the smartphone to compute motion traces in certain environments. However, for some sensors used for localization, such as UM6, UM6-LT [63], small errors in the orientation estimation will cause serious deviations in the measured acceleration, velocity and position. The accuracy of the UM6 and the UM6-LT Orientation

Sensors are expected to be within about 2 degrees. Taking 2 degrees as an upper bound, when UM6 or the UM6-LT are leveraged for velocity and position estimation, 10 seconds will accumulate $3.4m/s$ of velocity error and $34.2m$ of positioning error.

Mobile robots have been used in some research or commercial areas and can interact with smartphones via wireless communication. By sending correct location information to smartphones, mobile robots boost the positioning accuracies of users' smartphones. Namely, the location information on smartphones are calibrated while users are communicating with the mobile robots. Different from LANDMARC, mobile robots do not need setup an fixed array of RFID tags, also, the users do not need carry tracking tag.

7.2.2 Preliminary Observation

7.2.2.1 Experimental Setup

To illustrate this perspective, we conduct preliminary evaluation in a small and empty environment, the size of the environment is $4{\times}4m^2$. We simulate a single smartphone's motion and its interaction with a robot. The smartphone moves randomly and the robot moves on the diagonal line of the environment.

On the head of the robot, an Android tablet installs an application and a known map. The robot gets accurate location information by the map and calibrates its own position. For users of smartphones, a third-party application on the smartphone draws the user's trajectory by acceleration and direction. The acceleration and direction are obtained from the accelerometers and magnetic sensors. For most motions of people who carry a smartphone, their accelerations are between 0-$20m/s^2$ [9]. We consider the initial acceleration a_x (on x axis) to be $0.04m/s^2$, the initial acceleration a_y (on y axis) to be $0.10m/s^2$, and the observing time period from $1s$ to $40s$. In the beginning, the acceleration values obtained from the accelerometer are incorrect. The margin of acceleration error on the x axis and the y axis is 50%. The trajectory generated on a smartphone deviates from the ground truth.

7.2.2.2 Communication between single robot and user

Common smartphones and tablets have Bluetooth adapters. As a low energy technology, Bluetooth works within a short range. Received Signal Strength Indicator (RSSI) is a measurement of the power present in a received radio signal. Different from Channel State Information, RSSI values can be accessed on most modern mobile devices. RSSI is in units of "dBm" (dB per milliwatt). The smaller magnitude negative numbers denote to the higher signal strength. For the percentage of RSSI, the expression to convert is: rssiPercentage = (currentRSSI/ RSSIMAX)×100. The RSSI value is influenced by the distance between the sender and receiver of the radio frequency signal. Shorter distance represents stronger RSSI [70].

The relationship between distance and Bluetooth RSSI values is used to calibrate the deviations: the robot sends its accurate position to smartphones when they are close (distance between them is less than 1.5 meters, the Bluetooth RSSI value obtained from smartphone is no less than 80%). After we started our simulation, the robot sends the accurate location message to the smartphone after 20 seconds, then, for the user's smartphone, the "new" received position from the robot replaces the "old" position. The sending position message lessens the deviation of the computed position on the smartphone. Then, we repeat the simulation 50 times. The results are shown in Figure 7.2. The blue line refers to the distance from the computed position to the ground truth without any calibration. The red line is the distance after calibration. The shadow areas denote the confidence interval for each case. Less distance means more accurate localization. It indicates: 1) as time increases, the deviation also increases; 2) the error range after position replacement is reduced.

Then, by employing the same experimental environment and devices, we record the accumulative deviation distance (Euclidean Distance) from the ground truth for each second. We control whether and when the robot sends the location information. Sending frequency is defined as the number of the location messages sent to the smartphone per hour. As Figure

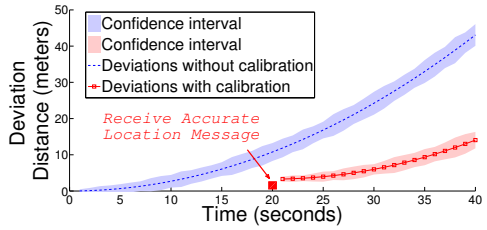


Figure 7.2 Impact of the calibration on the deviation distance

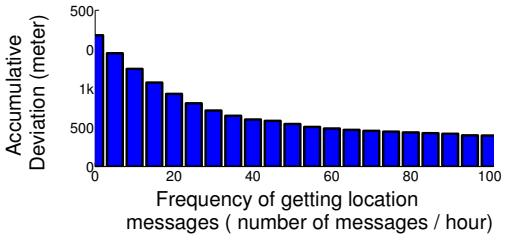


Figure 7.3 Distance errors under different frequencies

7.3, we test the sending frequency from 0 to 100. The y axis indicates the smartphone's accumulative deviations after our simulation.

We draw the conclusion: 1) by sending accurate location messages to the smartphone, the robot can reduce the smartphone's localization error, and 2) if the smartphone receives accurate location messages more frequently, the accumulative deviation decreases more. Therefore, we need to design an approach that the robot may frequently send its accurate location messages to the smartphone.

7.3 Single Robot Assisted Indoor Localization

7.3.1 Overview of Single Robot Model

Figure 7.4 shows an overview of the single robot assisted indoor localization. By installing a tablet computer on the mobile robot, the robot contains the proposed application and a known map. The moving robot in our approach has two parallel functions: 1) send location messages to a smartphone, and 2) collect crowd density information in rooms. The robot

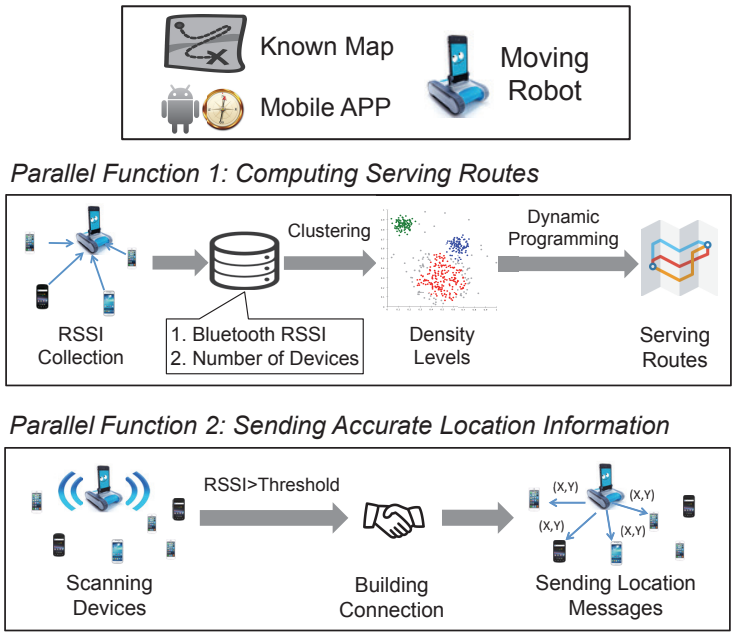
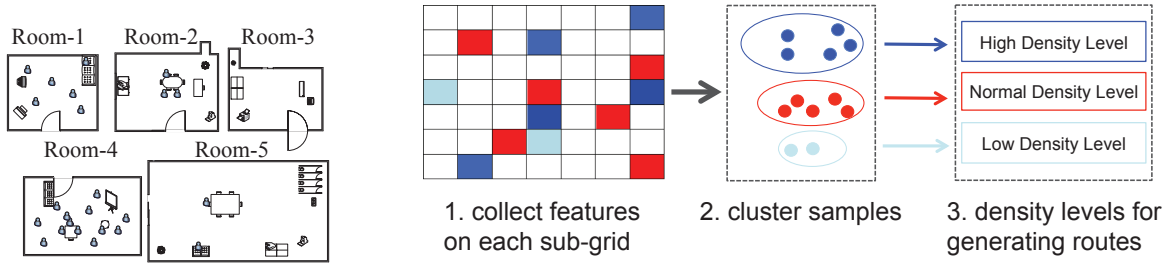


Figure 7.4 Framework of single robot assisted indoor localization

employs Bluetooth broadcast to communicate with the smartphones.

Since the Bluetooth signal strength depends on the distances between the adapter and the receiver, the stronger signal means a shorter distance. If the detected signal strength on a robot is above a certain threshold, distances between the Bluetooth adapter (on a robot) and receiver (on a smartphone) can be ignored. The robot can send its accurate position from an installed map to a smartphone. The location from robot replaces the location calculated by the smartphone. When the installed map and routes generated from the robot are accurate, the localization accuracy on the smartphone will increase.

Next, we attempt to provide a serving route for the robot. The objective of design is to minimize the deviations of customers' smartphones. As analyzed in our preliminaries, if the robot has more chances to send the accurate location messages to smartphones, the deviations of smartphones can be reduced more often. Hence, the robot should distinguish which rooms have more users.



(a) Crowd density distribution in different rooms

(b) The procedure of crowd density estimation

Figure 7.5 Crowd density estimation

7.3.2 Crowd Density Estimation

Crowd density is introduced to measure the number of people in a certain area. In addition to broadcasting position information to a smartphone, a robot can collect such data in a certain room: 1) the number of discovered smartphones, 2) the mean value of Bluetooth RSSI, the two types of data can reflect the crowd density for the room [82, 92]. AirLoc adopts such two types of data as features for clustering. As shown in Figure 7.5, after collecting these data in each room, k-means clustering algorithm [19] divides crowd densities in different rooms into different ranks. Based upon crowd densities, we will assign corresponding serving time periods for different rooms. In higher crowd density rooms, the robot will spend more time to broadcast location messages. In lower density rooms, the robot will serve less time.

Table 7.1 Main notions in AirLoc

Terms	Definition
$\mathbb{N}$, n, P,	Number of serving rounds, rooms, robots
i, j	Current position (room), Next position (room)
$V, \bar{V}$	Set of rooms, Set of visited rooms
$G[V-\bar{V}]$	Sequence of rooms will serve;
$d_{i,j}$	Time from moving room i to j
$Dis(i,j)$	Euclidean distance from room i to j
S_j	Possible serving time for room j
$T_k(\bar{V},j)$	Time cost for serving round k
$Den(i)$	Crowd density level for room i
α, β	Two separate serving areas
$\mathbb{I}_\alpha$,$\mathbb{I}_\beta$	Two initial position of α and β
T_1, T_2	Thresholds for constraining merging samples
$\mathbb{R}$	Radius of increasing serving area
HDA, LDA	High Density Area, Low Density Area
$nHDA, nLDA$	Number of mobile devices in HDA, LDA
ω	Parameter for evaluating room's crowd density
θ	Parameter for assign robots
nd_i	Number of scanned devices in room i

Algorithm 9 Edge-Based Algorithm (EBA)

Input:
$\mathbb{N}$, i, j, V,$\bar{V}$, $d_{i,j}$, $G[V - \bar{V}]$;
Output:
$T_k(V, j)$; $G[V - \bar{V}]$;
for each $\mathbb{N} > 0$ **do**
  robot starts serving for $G[V - \bar{V}]$;
  traveling & scanning & sending message;
  $V_l \leftarrow$ lowest density rooms by k-means(V);
  // delete nodes and joint edges with lowest density level;
  $G[V - \bar{V}] \leftarrow G[V - \bar{V}]$-$V_l$;
  // Recursive relation for computing time cost
  $T_k(\bar{V}, j) = \min\{T_{k-1}(\bar{V} - j, i) + d_{i,j}\}$;
  $\bar{V} = i \cup \bar{V}, K = K - 1$;
  for $\gamma = \text{num}(G[V - \bar{V}])$; $\gamma > 0$; γ - - **do**
    assigning the same serving time for $v \in G[V - \bar{V}]$;
  end for
  $\mathbb{N} - 1$;
end for

7.3.3 Generate Optimal Serving Routes

Before generating the serving paths for a mobile robot, we assume the robot moves between different rooms at uniform speed so the time to travel between different rooms can be calculated by distance (time cost on path is proportional to distance). The real map is converted to an abstract graph: rooms can be abstracted as nodes; paths between different rooms can be seen as edges. A robot's optimal serving route is the route that a robot can minimize the smartphones' deviations from the ground truth in a certain time period. Because the movement of crowd is difficult to predict, it is difficult for a robot to obtain the optimal route. Nevertheless, we design an algorithm to generate serving routes with close serving results as optimal routes.

7.3.3.1 Edge-Based Algorithm (EBA)

Edge-Based Algorithm aims to find an optimal route covering each room and reduce time costs. It is derived from the *TSP* (*Traveling Salesman Problem*) [36, 53, 64]. *TSP* does not

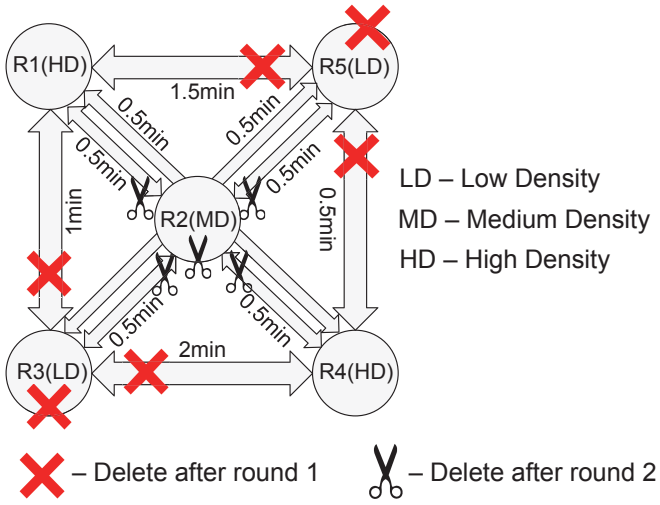


Figure 7.6 Edge Based Algorithm

consider the time cost in each node. However, in AirLoc, the time length of serving in each room is crucial. For EBA, each room will be assigned the same serving time in each serving round. The rooms with higher crowd density are served more often than lower density rooms. As Figure 7.6, Room 1 (R1 for short), R4 are high density rooms, R2 is medium density room, R3 and R5 are low density rooms. We specify the robot to go around the whole map 3 times. High density rooms will be visited 3 times, medium density room will be visited 2 times, low density room will be visited just once. In the first serving round, all the rooms will be visited. For the second serving round, R1, R2 and R4 will be served. R3, R5 and their jointed edges will be deleted. For the third serving round, R2 and related edges will be deleted. Only R1 and R4 will be served. The main task in each serving round is converted to find a optimal tour in the remaining map. It is similar to *TSP* problem, which could also be solved by dynamic programming in Algorithm 9. The total time complexity of EBA is $\Theta(n^3 2^n)$.

Algorithm 10 Node-Based Algorithm (NBA)

Input:
N; i; j; V; $d_{i,j}$; S_j;
Output:
$T_k(V, j)$; Sequence of visited rooms (Generated Route);
1: **for** each $N > 0$ **do**
2: $Find\_Route\_NBA()$;
3: Traveling & Scanning & Sending Message;
4: $N = N - 1$;
5: **end for**
6: $Find\_Route\_NBA()$:
7: $T_k(V, j) = \min\{T_{k-1}(V - j, i) + d_{i,j} + S_j\}$;
8: $V = i \cup V, K = K - 1$;
9: Assigning longer serving time for high density rooms;

7.3.3.2 Node-Based Algorithm (NBA)

The main idea of Edge Based Algorithm (EBA) is that allocating more serving rounds for the rooms with higher crowd density. Apart from EBA, we propose another algorithm: in one serving round, mobile robots visit each room, but we assign longer serving time to the the rooms with higher crowd density. Considering this strategy is based upon the crowd density of each room, it is named Node Based Algorithm (NBA). The description of NBA is in Algorithm 10. The time complexity of NBA is $\Theta(n^2 2^n)$. The differences between EBA and NBA will be discussed in evaluation section.

7.4 Multi-robot Assisted Indoor Localization

7.4.1 Single is not enough

Although the single robot approach is effective for some indoor environments, the problem is more challenging if 1) robot assisted indoor localization approach can be applied in more types of indoor environments, especially for the environments containing more rooms; 2) achieves higher positioning accuracies for smartphones.

The complexity of the EBA is sensitive to the number of rooms. If we adopt some common

tablets (such as Google Nexus 7, iPAD, the memory is within 4G), the single robot approach can handle a map within 1-30 rooms. In order to extend the existing approach to a larger indoor environment, a new approach that can process the map with more rooms should be provided.

When a single robot employs crowd density to generate routes, how to guarantee the accuracy of crowd density estimation is intractable, because 1) people do not always keep stationary, the crowd density distribution changes at anytime, the samples collected in each room need to be updated in time. Unfortunately, one robot cannot cover many rooms in a short time; 2) the crowd density levels used in EBA reflect the crowd distribution of previous periods rather than the current period.

7.4.2 Two Robots Working Model

Based on the motivation in previous section, we design a multi-robots approach to replace the single robot approach. An important task is to organize the robots. To achieve this goal, we design our system step by step. Namely, we introduce the design and analysis of two robots working model first, then, we extend the two robots approach to the multi-robot model.

7.4.2.1 Brute-Force Algorithm

In this subsection, we provide a directive algorithm to organize two robots. An intuitive approach is to find all the connected sub-graphs, each robot would travel on it by NBA/EBA and get the time cost. The recursive relation of this approach is as follow:

$$T_k(V, j_A, j_B) = \min\{T_k(V - (j_A, j_B), i_A, i_B) + d_{i_A,j_A} + d_{i_B,j_B} + S_{j_A} + S_{j_B}\}$$

S_{i_A}, S_{i_B} refer to the possible serving time for room i by robot A and B. d_{i_A,j_A}, d_{i_B,j_B} denote the time of robot A and B moving from room i to j. The system can choose the serving routes covering two sub-graphs with least time costs as optional solution. Although this brute-force approach could find the optimal routes, it has to test each of the potential

solution, the complexity of this brute-force approach is $O(n^3 2^n)$. If we extend this approach to multi-robots, the complexity will be $O(n^{k+1} 2^n)$. k refers to the number of robots, n refers to the number of rooms. For most of tablets and smartphones, this approach cannot be applied on the environments including 30 or more rooms. Therefore, an approach that can replace the brute-force approach is required.

7.4.2.2 Graph Division Strategy

Before employing multi-robots to construct our system, we consider how two robots work together first. AirLoc tries to partition the whole graph to two components. Each robot will serve one of them by EBA. The strategy of graph partition should satisfy: 1) the robots allocate more time to serve higher density rooms as possible, and 2) limit the time costs on the edges.

We propose the partition strategy in Fig. 7.7(a)-(b). We abstract a two dimension plane. The samples on the plane represent the rooms on the map. The number of each sample is the crowd density level of the room. The density level 10 is the highest level and 0 is the lowest level. The samples' density levels are obtained by accessing the cloud server. The cloud server computes the crowd density levels for each room by the k-means algorithm [19]. Two rooms (0 and 8) are chosen as the initial center of the serving area. By increasing the radius $\mathbb{R}$ of the area, rooms that are close to the initial samples are merged into the two serving areas by iteration. This merging procedure is based on the Euclidean distance. It is named "Distance First Algorithm".

In an indoor building, some rooms close to each other might have different crowd densities. For example, in Fig. 7.7(a), one room with density level 1 is merged into the high density area by the Distance First Algorithm. To tackle the problem, in contrast to the Distance First Algorithm, Fig. 7.7(b) depicts that samples are merged into the two areas by closest density levels rather than distances, it is named "Density First Algorithm".

Table 7.2 Memory consumption by Brute-force algorithm (OM - Out of Memory)

	10 Rooms	20 Rooms	100 Rooms	500 Rooms
2 Robots	1 M	8 G	OM	OM
4 Robots	100 M	OM	OM	OM
5 Robots	OM	OM	OM	OM

Table 7.3 Memory consumption by DDFA (OM - Out of Memory)

	10 Rooms	20 Rooms	100 Rooms	500 Rooms
2 Robots	40 K	800 K	700 M	OM
4 Robots	300 M	OM	OM	OM
5 Robots	OM	OM	OM	OM

AirLoc takes advantage of the above two algorithms. We define the threshold T_1 and T_2 as the constraint to make a balance between "density first" and "distance first". T_1 refers to the difference between the crowd density level of initial room and the crowd density level of the room being merged. T_2 is the difference between the crowd density level of the room merged in previous iteration and the crowd density level of the the room being merged. If T_1 and T_2 are set as the larger numbers, it means even if the two samples' density levels have significant differences, they can be merged in one serving area when their positions are close to each other. If T_1 and T_2 are set as the smaller numbers, when the two samples' density levels are different, they have less chances of being merged. This strategy is named "Distance/Density First Algorithm (DDFA)".

The complexity of DDFA is $O(n^{2k}logn)$, k refers to the number of robots, n is the number of rooms. Whereas this algorithm is limited by the number of robots, it is not sensitive to the number of rooms. As shown in Table 7.2 and Table 7.3, for most common tablets, which have more than 1G memory, they can handle 100 rooms if they use DDFA. We can conclude: 1) DDFA can be used on the map more than 100 rooms; 2) DDFA is difficult to be implemented on the group containing more than 4 robots, thus, it is necessary to extend our approach from 2 robots to n robots (n is greater than 2); 3) after obtaining their serving areas by DDFA, the robots will adopt EBA to travel on the assigned serving area. Since each serving area does not include many rooms, the memory bottleneck will not occur.

Algorithm 11 Distance/Density First Algorithm

Input:

n-number of rooms on the map; $\mathbb{R}$-radius of serving area;

Output:

two separate serving areas α and β;

1: select two Initial Positions ($\mathbb{I}_\alpha$,$\mathbb{I}_\beta$) from n rooms;
2: **while** num(α)+num(β) $< n$ **do**
3: increase radius: $\mathbb{R} \leftarrow \mathbb{R}+\Delta\mathbb{R}$;
4: **if** α or β satisfies *constrains* **then**
5: // merge room m into nearby serving area α or β;
$\alpha \leftarrow \alpha \bigcup m$ or $\beta \leftarrow \beta \bigcup m$;
6: **end if**
7: $n \leftarrow n-1$;
8: **end while**

Constrains:

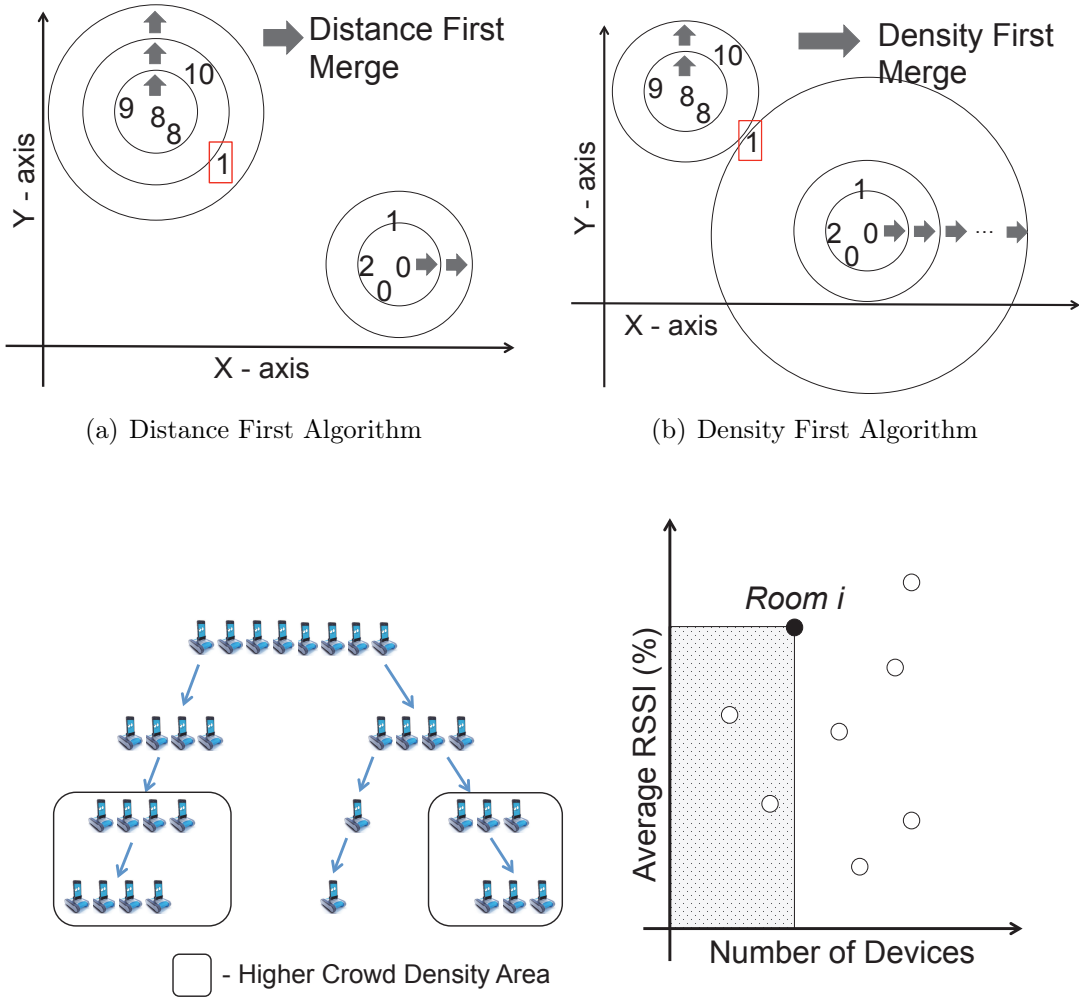
1). keep connectivity (complexity O(nlogn))

2). |Den($\mathbb{I}_\alpha$) - Den(m)| $< T_1$, |Den($\mathbb{I}_\beta$) - Den(m)| $< T_1$

3). |Den(m+1) - Den(m)| $< T_2$

7.4.2.3 Preemption

By applying DDFA, the two robots have obtained their serving areas respectively. The area containing more higher density rooms is High Density Area (HDA), the other with lower density rooms is named Low Density Area (LDA). Since the robot travels on LDA is not as efficient as HDA, if the number of scanned smartphones in LDA is less than 10% of scanned smartphones in HDA, the robot in LDA will go to HDA to serve. This mechanism can be seen as the HDA preempting the serving period of LDA. The time of initial preemption period is 2^1%. When the LDA robot moves back to LDA, if the number of scanned smartphones in LDA is still less than 10% of scanned smartphones in HDA, the time of the second preemption period next will increase to 2^2%. Therefore, the time of nth preemption period is 2^n%. When the ratio is greater than 10%, the preemption will end and the exponent will decrease to 1.



(a) Distance First Algorithm

(b) Density First Algorithm

(c) Unbalance Tree

(d) Definition of ω

Figure 7.7 Key technologies in multi-robot system

7.4.3 Extension to Multi-robot

In order to extend the existing approach to the multi-robot model, AirLoc assigns more than 2 robots to different serving areas. If we generate serving areas for multi-robots by DDFA directly, the computational complexity might become an obstruction for common tablets. Hence, we partition the robots to serving groups layer by layer as a tree. Before reaching the bottom layer, for each layer, we focus on how to divide the robots into sub-working groups. One group is separated into two sub-groups. Each sub-group has the same number of robots. It can be seen as two robots and using the DDFA to generate two serving areas, with each

Algorithm 12 Preemption Algorithm

1: **Input:** a pair of robots; $nLDA$, $nHDA$ - number of smartphones in LDA, HDA; **Output:** serving areas for the 2 robots;
2: // initializing length of preemption period, serving period number; $\lambda \leftarrow 1, i \leftarrow 1$;
3: **for** in each serving period i **do**
4: **if**

$$\frac{nLDA}{nHDA} \leq 10\%$$

then
5: the robot moves to the HDA for serving;
6: serving for $2^{\lambda}\%$ of serving period in HDA;
7: the robot moves back to LDA & $\lambda \leftarrow \lambda+1$;
8: call Preemption(i+1);
9: **else**
10: the robot travels and serves on LDA;
11: $i \leftarrow i+1$, $\lambda \leftarrow 1$;
12: **end if**
13: **end for**

subgroup responsible for one of them. Then, each sub-group is divided again until it reaches the base case (bottom) of the tree. This tree is a balanced tree.

Nevertheless, this approach has a shortcoming: even though one serving area has low density distribution and the other's density distribution is high, the number of robots serving them is the same. It contradicts our goal to allocate more robots to serve high density areas.

AirLoc proposes an unbalanced tree model to address the problem. First, an alternative method to measure the crowd density of serving area is introduced: as Fig. 7.7(d), by building an x-y plane, the x-axis refers to the number of mobile devices and the y-axis refers to the average RSSI value. We define the parameter ω to depict the crowd density for each room, the value of ω equals to the product of number of mobile devices and the average RSSI value. The shadow area on the plane represents the value of ω for a room. Lager size of the shadow area means higher crowd density. As equation (7.1), i is the ith room, nd_i indicates the number of devices, m refers to the number of smartphones in room i, $RSSI_j$ is the signal strength received from smartphone j. ω_i is the ω value of room i.

Algorithm 13 Generating Unbalance Tree (GUT)

```
Input: P mobile robots, original graph; Output: unbalance serving tree;
if not reach the base case then
    split the graph into two sub-graphs by DDFA;
    find HDA, LDA by computing ω by equation (1);
    allocate (P × θ)/(θ + 1) robots to HDA;
    allocate P − (P × θ)/(θ + 1) robots to LDA;
    for in each split sub-graph g do
        call GUT(g);
    end for
else
    call Edge Based Algorithm;
end if
```

$$\omega_i = (nd_i \times \sum_{j=0}^{m} RSSI_j)/m, \theta = (\sum_{i=1}^{H} \omega_i)/(\sum_{j=1}^{L} \omega_j) \tag{7.1}$$

where H represents the number of rooms in higher crowd density area, L is the number of rooms in the lower crowd density area. If θ is a large number, it means the higher crowd density area needs more robots to serve. Based on θ, when dividing robots into sub-groups, we allocate the number of robots as follows: let P denote the number of robots, the higher crowd density area will be assigned $(P \times \theta)/(\theta + 1)$ robots, the rest of robots will be sent to the lower crowd density area. If $(P \times \theta)/(\theta + 1)$ is not an integer, it can be processed as the ceiling of $(P \times \theta)/(\theta + 1)$. Therefore, if there exists different crowd density distributions in the two serving areas, the serving tree will be formed as an unbalanced one.

7.4.4 Dynamic Return

We explained the mechanisms for sending robots to a serving area, but we did not mention how the robots go back to the initial position. For a multi-robot system, the time costs of returning might be large. Thus, the paths for returning are important. A practical and concise method is designed for generating the returning paths: find a new tree root that is relatively close to each leaf node. Namely, we find the node k, which has the smallest sum of distances between k and each other room i, then, we arrange k as the "new" root. After

Algorithm 14 Dynamic Return Algorithm

Input:
i;p;$Dist(k,i)$;
Output:
The new initial position (new root of the serving tree);

if All the robots finish the assigned serving task **then**
  for each room i & i is not greater than n & i=i+1 **do**
    if i satisfies that the value of $\sum_{k=1}^{p} Dist(k,i)$ is minimum **then**
      Return the value of $\sum_{i=1}^{n} Dist(k,i)$;
      Set the room i as the new root of serving tree;
    end if
  end for
else
  wait for robots finishing the serving assigned serving task
end if

finishing the tasks in one round, all the robots will move to the new root and restart the next serving round.

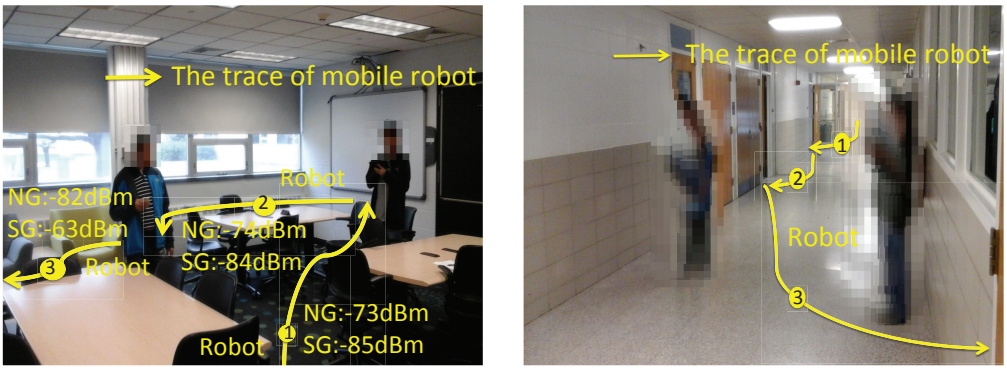
7.5 Evaluation

In evaluation phase, we seek to answer four questions:

1. Whether AirLoc can increase smartphones' localization accuracies?
2. How well is the updated crowd density levels of AirLoc?
3. How some features such as number of robots can influence the performance of system?
4. Whether proposed techniques such as unbalanced tree, dynamic return make contribution to AirLoc, how they enhance the performance of AirLoc?

7.5.1 Experimental Setup

As illustrated in Figures 7.8-7.9, we performed our evaluation on the first floor of Engineering Building at Michigan State University, which is an indoor environment containing more than



(a) Samples collected in a room (b) Samples collected in a hallway

Figure 7.8 Implement AirLoc in a real indoor environment

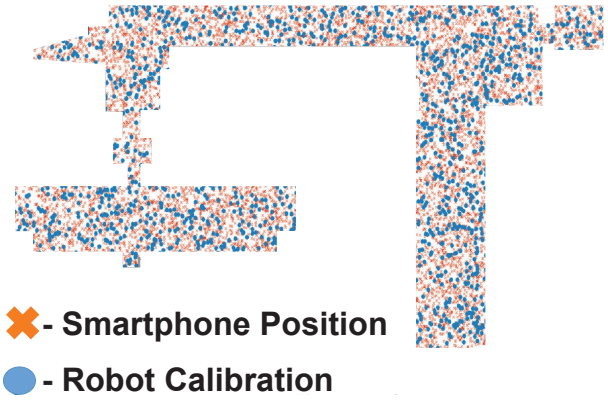


Figure 7.9 Data samples on the map

130 rooms. Rooms are abstracted as nodes, corridors or paths that are narrower than 3 meters are abstracted as edges. The corridors or paths whose width are greater than 3 meters also processed as nodes.

In our indoor experiment, we consider the TurtleBot [16] as the initial mobile platform, it is a common type of robot with multifunction and decent price. The height of the tablet (to be installed on a robot) is 1 meter, which is similar to the height of user's pocket. The speed of is $0.3m/s$. There are 0-6 volunteers in each room or hallway, each volunteer carries Samsung Galaxy 4 smartphone or Google Nexus Tablet and turn on the Bluetooth. Volunteers in the experimental environment walk freely.

By running proposed program on a tablet and smartphones, for each 10 seconds, the tablet

scans other Bluetooth devices and collects the RSSI values and the number of discovered devices periodically. Fig. 7.8 shows the scenario that we conduct the experiment. Based on the experiment, for each smartphone, it receives the location messages from the robot via Bluetooth communication. In Fig. 7.8(a) and Fig. 7.8(b), we assume a robot moves and collects data in a room or hallway. GN refers to the RSSI obtained from the user's Google Nexus tablet; SG refers to the RSSI obtained from the user's Samsung Galaxy smartphone.

On the completion of collecting data in one experiment, we extend our indoor experiment results to a large scale simulation. By using the same floor plan, our simulation includes 16 mobile robots that carry Google Nexus 7 tablets to visit different places in the building. Each robot travels and works by AirLoc. We assume 350 volunteers are distributed in each room or hallway rather than the rooms where the single robot collected Bluetooth information. After the assumed multi-robots collect samples of Bluetooth RSSI values in the first round, we use MATLAB and WEKA [18] on the cloud server to analyze the samples. The WEKA software supports k-means algorithm that can divide different rooms into different crowd density levels. There are three parameters in the dataset built by collected samples: *number of devices*, *mean value of RSSI*, and *identifier (ID) of each room* (only number of devices and mean value of RSSI are used to cluster). As shown in Fig. 7.11(d), collected samples are distributed on the two-dimensional surface. It is formed by two features: average value of RSSI and the number of smartphones. After executing the k-means clustering algorithm, each room is categorized to the corresponding crowd density level. T_1 and T_2, the parameters in DDFA, are set as 2.

7.5.2 Metric of Evaluation

Besides Euclidean distance, to measure the localization accuracies of AirLoc, we introduce other metrics:

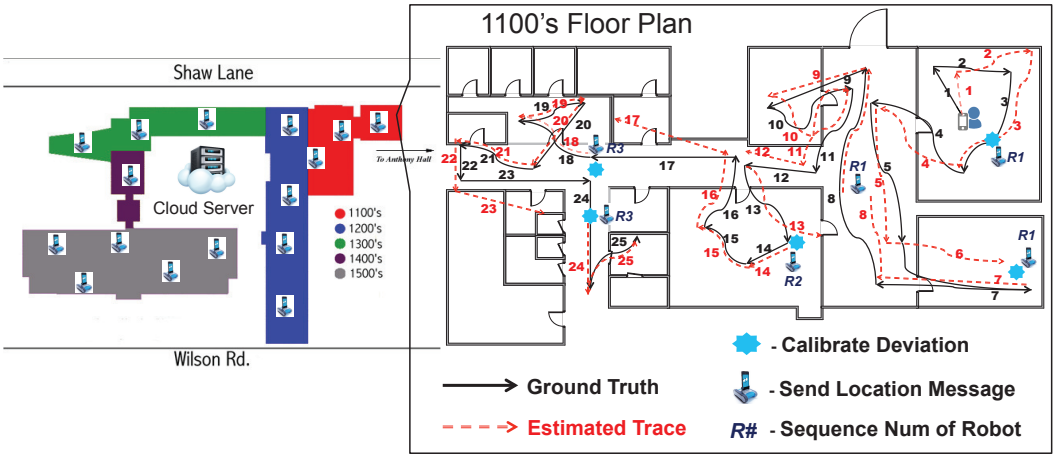


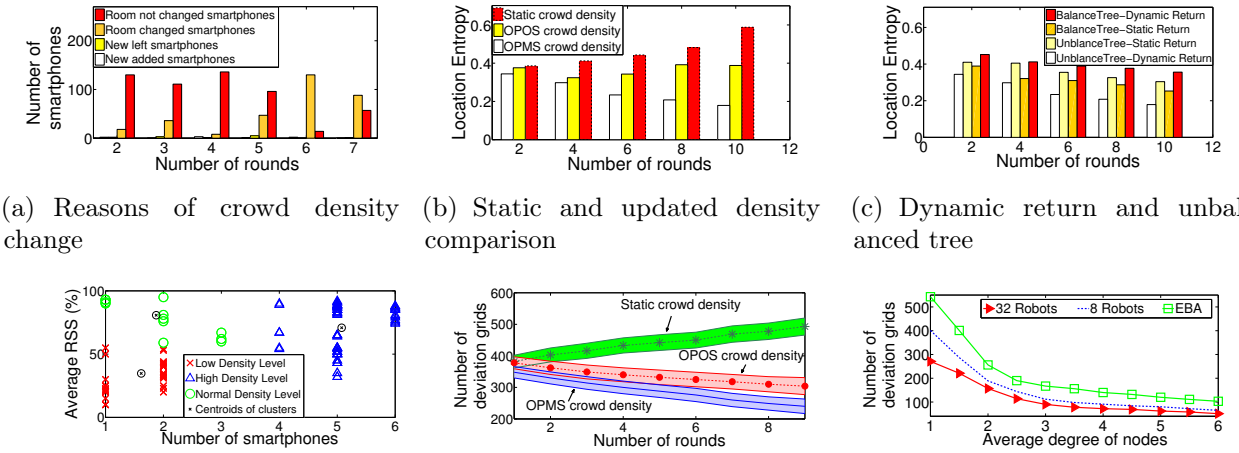
Figure 7.10 Experiment floor plan and single trace study

7.5.2.1 Number of Deviation Grids

One metric is $E = \sum_{i=1}^{N} D_i$, let N denote the number of smartphones; D_i refers to the deviations grids for each smartphone; E refers to the remaining errors after carrying out the proposed approaches; The size of each grid is $0.4m \times 0.4m$.

7.5.2.2 Location Entropy

The other metric is Location Entropy. The expression of Location Entropy is: $L(x) = -\sum_{i=1}^{m} P(x_i)\log_2(P(x_i))$. Higher value of the location entropy represents the localization results of all the smartphones deviate more from the ground truth. m is the number of grids; for each grid $P(x_i)$ represents the probability that the smartphone's estimated position is the ground truth. $P(x_i)$ equals to the ratio of (times of estimate successfully)/(estimated times). In our simulation, we record the $P(x_i)$ of each grid every 5 seconds.



(a) Reasons of crowd density change

(b) Static and updated density comparison

(c) Dynamic return and unbalanced tree

(d) Result of data clustering by k-means algorithm

(e) Static and updated density comparison (100 times)

(f) Nodes' degree measurement for comparing the number of robots

Figure 7.11 Field study and simulation results

7.5.3 Evaluation of Crowd Density Updating

7.5.3.1 Reasons of Crowd Density Variations

In indoor environments, such as convention center, hospital, and hotel, the crowd density distribution is always changing. For the single robot approach, the robot computes and updates the crowd density levels after each serving round. As the reasons explained in the previous section, it is difficult to guarantee the correctness of crowd density levels. Fig. 7.13(a) illustrates the reasons for crowd density variations.

7.5.3.2 Duty Cycles Analysis

For each serving period in multi-robot system, we define three serving slots as Fig. 7.12: 1) T Slot (Tree generating Slot): the robots build the unbalanced tree until they are assigned to the final serving areas; 2) S Slot (Serving Slot): when the tree reaches bottom (base case), they will conduct EBA that are also relied on the latest crowd density; 3) R Slot (Return Slot): after each period of serving, all the robot return to the "new" robot.

The multi-robot system can provide more accurate crowd density: 1) using global and

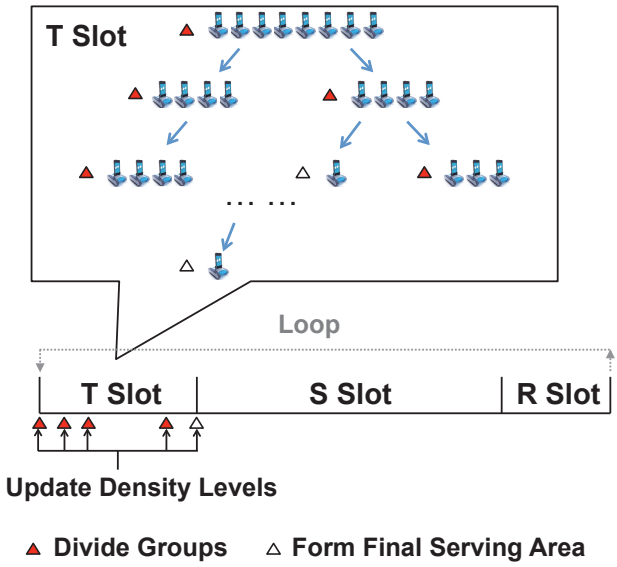


Figure 7.12 Three serving slots: 1) T slot; 2) S slot; 3) R slot

concurrent Bluetooth information to replace the Bluetooth information collected serially by the single robot, the computed crowd density levels will be closer to the ground truth; 2) before each robot traveling on serving area, it can access the latest crowd density levels to help them make correct decision by proposed algorithms.

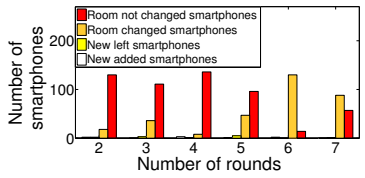
Then, we compare the location entropies of AirLoc when it adopts static or updated crowd density information. Fig. 7.13(b) plots the location entropy values on the y axis, that are the average location entropy values when each round ends. Static crowd density refers to robots that use the crowd density obtained at the end of first round to make a decision; OPOS (One Period One Sample) represents using the crowd density levels obtained at the first splitting group in each round when generating the serving tree. Namely, for one period, as T Slot, all robots do not update the crowd density levels when they execute the proposed algorithm until the next period. OPMS (One Period Multi-Sample) refers to the crowd density levels obtained each 10 seconds when the system generates the serving tree, forming room clusters and using EBA. By employing updated crowd density, AirLoc reduces the deviations and increases the localization accuracies gradually. Then, we repeat this simulation 100 times,

shown in Fig. 7.13(e), we leverage the number of deviation grids to illustrate the advantages of using the crowd density information that is updated continuously by multi-robots. It proves that using concurrent crowd density information collected by multi-robots can reduce the deviations effectively. The shadow areas in Fig. 7.13(e) represent the confidence intervals of the simulation.

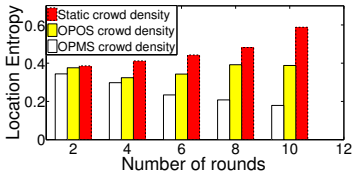
7.5.4 Evaluation of Reducing Deviation

Keeping the same conditions as the previous experiment, as Fig. 4.10, we study the case of each user. Even if one user's estimated trace deviates sometimes, after receiving the accurate location messages from mobile robots, the user's estimated trace is close to the actual one. Next, we compare the 1) the robots using dynamic return versus static return, 2) Balanced Tree and the Unbalanced Tree mechanisms. Fig. 7.13(c) shows that: 1) Unbalanced Tree has better performance than Balanced Tree; 2) Dynamic Return has less deviations than returning to a fixed root. Both Unbalanced Tree and Dynamic Return technologies can help the smartphones' localization.

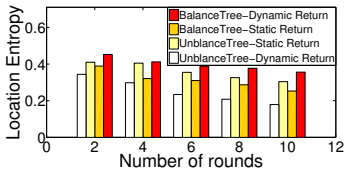
Fig. 7.13(f) illustrates the two relationships: one is the relationship between localization errors and average degree of nodes, the other is the relationship between deviations and number of robots. The degree of a node represents the number of edges connected to the node. Higher degree means more edges and better connectivity. For the single robot model, it employs EBA to serve all the rooms. For multi-robot approach, it uses OPMS and EBA. The location entropy is obtained after 10 serving rounds. We draw the conclude: 1) proposed approaches work well, and 2) when the average degree increases, the remaining deviations from the ground truth will decrease, because the better connectivity gives the robots more opportunities to choose better optimized serving routes, 3) more robots in AirLoc further enhance smartphones' localization. In general, when we deploy no less than 8 robots and run AirLoc over 8 rounds, the average localization error of each smartphone is not beyond $0.81m$.



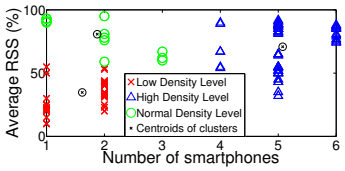
(a) Reasons of crowd density change



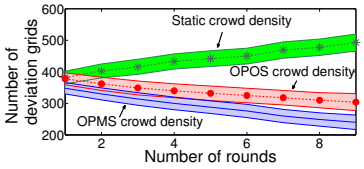
(b) Static and updated density comparison



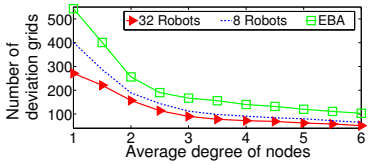
(c) Dynamic return and unbalanced tree



(d) Result of data clustering by k-means algorithm



(e) Static and updated density comparison (100 times)



(f) Nodes' degree measurement for comparing the number of robots

Figure 7.13 Field study and simulation results

7.5.5 EBA and NBA

In design section, we introduced two algorithms (Edge Based Algorithm and Node Based Algorithm) for single robot to select its serving routes. In reality, these two algorithms have difference performances for different types of indoor maps. NBA outperforms EBA when the indoor map has higher average degree of nodes. As shown in Figure 7.14, we simulate the map with different average degrees. By assuming using 32 mobile robots and AirLoc approach, we confirm that the EBA is suitable for the map with lower average degree and NBA is suitable for the map with higher average degree.

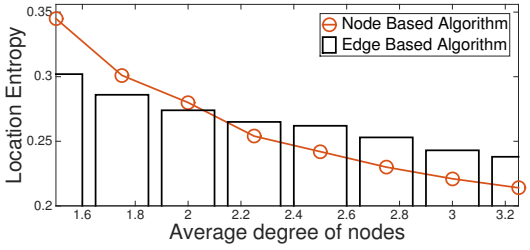


Figure 7.14 Comparison between EBA and NBA

7.5.6 Performance Comparison

For AirLoc, Table 7.4 shows a comparison with the state-of-the-art indoor localization solutions. AirLoc does not request smartphones' users carrying extra devices. Different from most of other approaches, AirLoc can be deployed in the large environment with low cost. Besides, existing smartphone-based approaches are room levels or have serious error accumulation, even if Bluetooth communication has its range limitation (10 meters) and the people's crowd distribution is dynamic, after conducting serving rounds iteratively, AirLoc promotes these approaches to achieve better localization results.

Table 7.4 Comparison AirLoc with other indoor localization systems

Approach	Signal Type	Accuracy	Device-Based	Scalability	Pros	Cons
AirLoc	Bluetooth, WiFi	within $1m$	Low	High	High scalability and accuracy, No extra device, Low cost	Need known map
Ultrasound assistant	Ultrasound	$10cm$ to $1m$	High	Limited	High accuracy	Need expensive devices for each user
Finger Printing	WiFi, cellular and RF signal, Bluetooth, FM	Room Level	Low	Medium	Device-Free, Low cost	Complex training, Low accuracy
SLAM	Laser, WiFi	within $5m$	High	Medium	Accuracy is fine, Self-calibrate position	Just serving the robot, Need special sensors
Smartphone Based	WiFi	Low	Low	Medium	No extra device for customers, Low cost	Error accumulation, Low accuracy

7.6 Conclusion

Most of previous indoor localization approaches cannot avoid accumulative errors. AirLoc boosts smartphones' localization accuracies on large indoor environments. We build an unbalanced tree to organize the multi-robot. From the root to bottom, the robots are divided into sub-groups according to the distribution of people's crowd density. At the bottom, all the robots are assigned their working areas by DDFA and serve these areas by EBA. Evaluation results indicate 1) AirLoc can be applied on the map containing more than 100 rooms; 2) localization errors of smartphones are reduced effectively via AirLoc. The average deviation of each smartphone belows $0.9m$; 3) the time complexity of introduced algorithms are acceptable for common tablets.

CHAPTER 8

SUMMARY AND PROPOSED FUTURE RESEARCH WORK

8.1 Summary

As mobile and smart devices have been increasingly popular in the last decade, location services have become an important issue. In this dissertation, we introduce approaches to provide location-based services in indoor environments where GPS cannot be used directly. In general, we answer two questions: 1) how users of mobile devices obtain indoor maps and 2) how users of mobile devices obtain indoor location information.

For reconstructing the indoor floor plan, we propose iFrame, a dynamic and light-weight approach that leverages mobile sensing capabilities for constructing indoor maps. iFrame uses dead reckoning, Bluetooth and WiFi detections initially to describe the state of each grid on the 2-dimensional map. iFrame fuses the sensing data from the three approaches to generate the shadow map of indoor scenarios. Our experiments and simulations verify that by merely carrying smartphones or other mobile devices, if users allow their mobile devices to share information with other users' devices, iFrame is able to reconstruct the dynamic shadow map of indoor environments within 5 minutes.

Based on the known map installed on mobile devices, we propose four indoor localization approaches. In the beginning, we attempt to improve the users' indoor localization accuracies without the requirements of cooperating with other mobile devices. We introduce iLoom, an indoor positioning approach that leverages the users' outdoor walking behaviors. By adopting dead reckoning as basic method, we generate people's motion traces. We propose use an Acceleration Range Box to optimize the user's received accelerations. To obtain the ranges of Acceleration Range Box, we combine the data describing the outdoor motion by accurate GPS and the data of the indoor movements via transfer boosting. Our case studies

including 15 users and 3 indoor buildings indicate iLoom improve dead reckoning and achieve accuracy to 0.35 meter. This approach does not need extra devices and data training for certain environments.

Since iLoom is a data-driven approach and does not leverage the cooperation of different mobile devices, we also propose three indoor localization methods based on the wireless communications between multiple mobile devices.

First, we present CRISP - CoopeRating to Improve Smartphone Positioning, which uses the accelerometers on smart devices to generate the motion traces of users initially. CRISP leverages opportunities of the interaction of multiple smartphones to enhance accuracy. When individual smartphones may provide some positioning (possibly inaccurate) information via Bluetooth and WiFi communication, by applying triangular calibration, opportunities of accuracy improvement occur when several smartphones cooperate and share position information. Through extensive indoor experiment and simulation, CRISP is able to localize users within 1 meter distance deviation. In addition, based upon existing location information, CRISP is able to profile users' steps without accumulative errors.

Second, we design and implement SilentWhistle, a light-weight indoor positioning approach that leverages acoustic sensing and dead reckoning to compute the users' locations. We assume that dead reckoning has inaccuracies from received accelerations. Each use of SilentWhistle generates a sound that cannot be heard by people but can be detected by a smartphone. The sound frequencies are between 18KHZ and 22KHZ. By leveraging the relation between distance and sound strength, SilentWhistle uses triangulation to improve indoor localization accuracies from initial dead reckoning. By applying a centralized model and distributed model, the users who send obvious incorrect locations will be detected and disposed. Our extensive experimental results indicate the accuracy of SilentWhistle can achieve 0.88-1.24m.

Third, motivated by the increasing availabilities of mobile robots, we propose AirLoc, an indoor localization approach that integrates the off-the-shell smartphones with the mobile robots. The robot and tablet are inexpensive and including the tablet. By installing the

known map on a tablet, the robot uses Bluetooth to send its correct location to the users' smartphones. Based on crowd density of smartphone users, AirLoc provides the paths for single and multi-robots so that smartphones can minimize the deviations between the estimated positions by interaction with the robot and the ground truth.

	Device	Infrastructure	Accuracy	Data Training
iLoom	Smartphone Smartwatch	no beacon	0.53m	off-line Transfer Learning
CRISP	Smartphone Smartwatch Tablet	no beacon	0.91m	none
Silent Whistle	Smartphone Smartwatch	no beacon	0.88m~1.2m	none
AirLoc	Smartphone Smartwatch Tablet	Mobile Robot	0.90m	on-line clustering
	no extra devices	*low costs*	*within 1m*	*acceptable data training*

Figure 8.1 Comparison among four proposed indoor localization approaches

As shown in Figure 8.1, we analyze introduced indoor localization by the following aspect: 1) all the proposed indoor localization approaches do not require users to carry extra mobile devices except for smartphones and tablets, 2) AirLoc needs the environment to include mobile robots, while the other approaches do not require extra infrastructure to support the indoor positioning solution, 3) the average localization accuracy of iLoom (0.53m) is better than the other three approaches because the results of iLoom depend on the size of pre-training data. The average localization accuracies of all of the approaches could achieve 1m, 4) from the perspective of computational complexity, iLoom's is more complex than other methods because of the transfer learning procedure. Although AirLoc needs online clustering, considering the number of limited data samples, the computation can be processed with acceptable complexity.

Based upon the proposed indoor map construction solution and indoor localization approaches, users of mobile devices can obtain indoor location based services with acceptable hardware costs, computational complexities, and accuracies.

8.2 Proposed Future Research Work

8.2.1 Indoor Localization Future Work

In the previous chapters, by using the inertial sensors on smartphone, we have introduced three indoor localization approaches: iLoom, CRISP, SilentWhistle, and AirLoc. Nevertheless, some sensors on smartphone have not been explored in our research, such as camera sensor.

8.2.1.1 Visible Light Communication

Considering smart LED bulbs are installed in some indoor environments (such as smart home and smart office), some research adopts the visible lights that are sensed by the camera on smartphones to locate people [90, 33]. However, most of the existing approaches need complex image processing for the photos obtained by the camera. As represented in Fig. 8.1, we plan to use the partial information obtained from the camera sensor (such as the color of the center in a photo) to reduce the computation burden and also achieve accurate localization results.

8.2.1.2 Multi-radio Indoor Localization

In addition, we will use more types of radios to enhance the existing indoor localization mechanisms. Mobile devices may support Channel State Information (CSI), Zigbee, and WiFi backscatter in the future. Based on the features in these radios, we will obtain more channels to analyze the signal strength, interferences, and Euclidean distances. These information might enhance the localization accuracies in different perspectives. For example, CSI over multi-subcarrier can travel along different fading or scattering paths on account of the multi-path effects. This phenomenon is able to enhance the accuracies of traditional fingerprinting indoor localization.

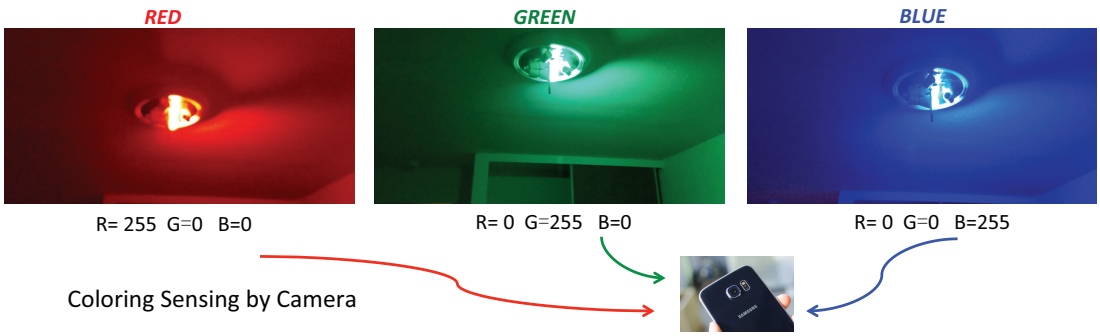


Figure 8.2 Use camera sensor to detect the color of captured images.

8.2.1.3 Leverage Machine Learning to Enhance Indoor Localization

In iLoom, we have leveraged transfer learning mechanism to enhance indoor localization accuracies by profiling the walking behaviors outdoors. In the future, we will further utilize other machine learning algorithms (decision tree classification) to detect different postures including sitting, standing, lying down, walking, and climbing stairs. In addition, we will use spatio-temporal structural context learning to further analyze the generated motion trace. For radio-based indoor localization, deep learning approach will be applied to process the CSI data. In order to reduce the computational complexity, a greedy learning method will be leveraged to train the weights layer-by-layer.

8.2.2 Indoor Location Based Services

8.2.2.1 3-dimensional Indoor Map Construction

In our existing indoor map reconstruction approach, iFrame, we have rebuilt the indoor map based on the mobile sensing techniques. However, the generated map is a 2-dimensional version. For applications such as smart guide in a museum or shopping mall, a 3-dimensional version of indoor map is needed sometimes. We plan to deploy some radio beacons (such as Bluetooth Low Energy Beacons) on the ceiling of the indoor environments. By analyzing the communications between different beacons and mobile devices, we are able to extend

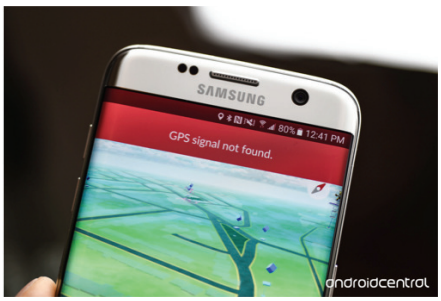


Figure 8.3 Pokemon game cannot obtain indoor location because GPS does not work indoors.

the 2-dimensional map construction approach to a 3-dimensional method. In addition, we will consider using Canny edge detector to build the edges of objects and combine with our 3-dimensional indoor map.

8.2.2.2 Entrance of Virtual Reality

In the virtual reality area, indoor positions has been the entrance of virtual services. For example, one the of most popular video game in 2016, Pokemon Go [47] has provided lots of location based services in outdoor environments via GPS. Unfortunately, as shown in Fig. 8.2, these services cannot be available indoors because of various interferences. To provide location capacities to the virtual reality services, we hope to leverage dead reckoning technique to draw the motion trace in indoor environments. The latest GPS samples can be defined as the anchor point of the trace. If there exist hand-off beacons in indoor environments (such as Bluetooth printer and WiFi access points), these beacons can be used to calibrate the deviation caused by dead reckoning.

8.2.2.3 Applications of Smart Building and IoT

Based on known indoor location, some valuable enterprise-level and commercial-level applications can be delivered, such as indoor navigation, indoor activity detection, and smart office control. For example, a user of smartphone walked in a shopping mall, when she wants to drink coffee, the indoor location-based application will navigate the user to Starbuck and

send the corresponding coupon or advertisement once she is close to the coffee shop. In some smart buildings equipped with Internet of Things, indoor locations information can be leveraged pervasively. When an employee needs to meet her colleagues at the company site, the indoor location-based application can reserve a nearby conference room and navigate her to the conference site.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Alismail, Hatem, L Douglas Baker & Brett Browning. 2014. Continuous trajectory estimation for 3d slam from actuated lidar. In *Robotics and automation (icra), 2014 ieee international conference on*, 6096–6101. IEEE.

[2] Alzantot, Moustafa & Moustafa Youssef. 2012. Crowdinside: automatic construction of indoor floorplans. In *Proceedings of the 20th international conference on advances in geographic information systems*, 99–108. ACM.

[3] Azizyan, Martin & Romit Roy Choudhury. 2009. Surroundsense: mobile phone localization using ambient sound and light. *ACM SIGMOBILE Mobile Computing and Communications Review* 13(1). 69–72.

[4] Bahl, Paramvir & Venkata N Padmanabhan. 2000. Radar: An in-building rf-based user location and tracking system. In *Infocom 2000. nineteenth annual joint conference of the ieee computer and communications societies. proceedings. ieee*, vol. 2, 775–784. Ieee.

[5] Baronti, Paolo, Prashant Pillai, Vince WC Chook, Stefano Chessa, Alberto Gotta & Y Fun Hu. 2007. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications* 30(7). 1655–1695.

[6] Chen, Po-Wei, Kuang-Shun Ou & Kuo-Shen Chen. 2010. Ir indoor localization and wireless transmission for motion control in smart building applications based on wiimote technology. In *Sice annual conference 2010, proceedings of*, 1781–1785. IEEE.

[7] Chen, Yin, Dimitrios Lymberopoulos, Jie Liu & Bodhi Priyantha. 2012. Fm-based indoor localization. In *Proceedings of the 10th international conference on mobile systems, applications, and services*, 169–182. ACM.

[8] Chintalapudi, Krishna, Anand Padmanabha Iyer & Venkata N Padmanabhan. 2010. Indoor localization without the pain. In *Proceedings of the sixteenth annual international conference on mobile computing and networking*, 173–184. ACM.

[9] Chon, John & Hojung Cha. 2011. Lifemap: A smartphone-based context provider for location-based services. *IEEE Pervasive Computing* (2). 58–67.

[10] Dai, Wenyuan, Qiang Yang, Gui-Rong Xue & Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th international conference on machine learning*, 193–200. ACM.

[11] Device, TelosB. 2012. Crossbow.

[12] Dicke, RH. 1953. The effect of collisions upon the doppler width of spectral lines. *Physical Review* 89(2). 472.

[13] Dissanayake, MWMG, Paul Newman, Steven Clark, Hugh F Durrant-Whyte & Michael Csorba. 2001. A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on* 17(3). 229–241.

[14] Fukuju, Yasuhiro, Masateru Minami, Hiroyuki Morikawa & Tomonori Aoyama. 2003. Dolphin: An autonomous indoor positioning system in ubiquitous computing environment. In *Wstfeus*, 53–56.

[15] Gao, Ruipeng, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, Kaigui Bian, Tao Wang & Xiaoming Li. 2014. Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing. In *Proceedings of the 20th annual international conference on mobile computing and networking*, 249–260. ACM.

[16] Garage, Willow. 2011. Turtlebot. *Website: http://turtlebot. com/last visited* 11–25.

[17] Gardiner, Crispin W et al. 1985. *Handbook of stochastic methods*, vol. 4. Springer Berlin.

[18] Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann & Ian H Witten. 2009. The weka data mining software: an update. *ACM SIGKDD explorations newsletter* 11(1). 10–18.

[19] Hartigan, John A & Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Applied statistics* 100–108.

[20] Hightower, Jeffrey, Roy Want & Gaetano Borriello. 2000. Spoton: An indoor 3d location sensing technology based on rf signal strength. *UW CSE 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA* 1.

[21] Hilsenbeck, Sebastian, Dmytro Bobkov, Georg Schroth, Robert Huitl & Eckehard Steinbach. 2014. Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning. In *Proceedings of the 2014 acm international joint conference on pervasive and ubiquitous computing*, 147–158. ACM.

[22] Jafri, Rabia, Rodrigo Louzada Campos, Syed Abid Ali & Hamid R Arabnia. 2016. Utilizing the google project tango tablet development kit and the unity engine for image and infrared data-based obstacle detection for the visually impaired. In *Proceedings of the 2016 international conference on health informatics and medical systems (hims'15), las vegas, nevada*, .

[23] Jan, Ea-Ee & James Flanagan. 1996. Sound capture from spatial volumes: Matched-filter processing of microphone arrays having randomly-distributed sensors. In *Icassp*, vol. 2, 917–920. IEEE.

[24] Jiang, Yifei, Yun Xiang, Xin Pan, Kun Li, Qin Lv, Robert P Dick, Li Shang & Michael Hannigan. 2013. Hallway based automatic indoor floorplan construction using room fingerprints. In *Proceedings of the 2013 acm international joint conference on pervasive and ubiquitous computing*, 315–324. ACM.

[25] Jin, Guang-yao, Xiao-yi Lu & Myong-Soon Park. 2006. An indoor localization mechanism using active rfid tag. In *Sensor networks, ubiquitous, and trustworthy computing, 2006. ieee international conference on*, vol. 1, 4–pp. IEEE.

[26] Joachims, Thorsten. 1999. Transductive inference for text classification using support vector machines. In *Icml*, vol. 99, 200–209.

[27] Kemper, Jürgen & Holger Linde. 2008. Challenges of passive infrared indoor localization. In *Positioning, navigation and communication, 2008. wpnc 2008. 5th workshop on*, 63–70. IEEE.

[28] Kobayashi, Fumihiko & Isamu Umino. 1991. Band pass filter. US Patent 5,021,757.

[29] Košecká, Jana, Liang Zhou, Philip Barber & Zoran Duric. 2003. Qualitative image based localization in indoors environments. In *Computer vision and pattern recognition, 2003. proceedings. 2003 ieee computer society conference on*, vol. 2, II–3. IEEE.

[30] Koyuncu, Hakan & Shuang Hua Yang. 2010. A survey of indoor positioning and object locating systems. *IJCSNS International Journal of Computer Science and Network Security* 10(5). 121–128.

[31] Krebs, Paul & Dustin T Duncan. 2015. Health app use among us mobile phone owners: a national survey. *JMIR mHealth and uHealth* 3(4). e101.

[32] Kumar, Swarun, Stephanie Gil, Dina Katabi & Daniela Rus. 2014. Accurate indoor localization with zero start-up cost. In *Proceedings of the 20th annual international conference on mobile computing and networking*, 483–494. ACM.

[33] Kuo, et al., Ye-Sheng. 2014. Luxapose: Indoor positioning with mobile phones and visible light. In *Proceedings of the 20th annual international conference on mobile computing and networking*, 447–458. ACM.

[34] Küpper, Axel. 2005. *Location-based services: fundamentals and operation*. John Wiley & Sons.

[35] Lamport, Leslie, Robert Shostak & Marshall Pease. 1982. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4(3). 382–401.

[36] Lawler, Eugene L. 1985. The traveling salesman problem: a guided tour of combinatorial optimization. *WILEY-INTERSCIENCE SERIES IN DISCRETE MATHEMATICS* .

[37] Lee, Jung-Min. 2013. Validity of consumer-based physical activity monitors and calibration of smartphone for prediction of physical activity energy expenditure .

[38] Levis, Philip, Sam Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer et al. 2005. Tinyos: An operating system for sensor networks. In *Ambient intelligence*, 115–148. Springer.

[39] Li, Fan, Chunshui Zhao, Guanzhong Ding, Jian Gong, Chenxing Liu & Feng Zhao. 2012. A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 acm conference on ubiquitous computing*, 421–430. ACM.

[40] Liu, Hongbo, Yu Gan, Jie Yang, Simon Sidhom, Yan Wang, Yingying Chen & Fan Ye. 2012. Push the limit of wifi based localization for smartphones. In *Proceedings of the 18th annual international conference on mobile computing and networking*, 305–316. ACM.

[41] Liu, Hui, Houshang Darabi, Pat Banerjee & Jing Liu. 2007. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37(6). 1067–1080.

[42] Liu, Kaikai, Xinxin Liu & Xiaolin Li. 2013. Guoguo: Enabling fine-grained indoor localization via smartphone. In *Proceeding of the 11th annual international conference on mobile systems, applications, and services*, 235–248. ACM.

[43] Liu, Tianci, Lei Yang, Qiongzheng Lin, Yi Guo & Yunhao Liu. 2014. Anchor-free backscatter positioning for rfid tags with high accuracy. In *Infocom, 2014 proceedings ieee*, 379–387. IEEE.

[44] Lui, Vincent & Tom Drummond. 2015. Image based optimisation without global consistency for constant time monocular visual slam. In *Robotics and automation (icra), 2015 ieee international conference on*, 5799–5806. IEEE.

[45] Mandal, Atri, Cristina V Lopes, Tony Givargis, Amir Haghighat, Raja Jurdak & Pierre Baldi. 2005. Beep: 3d indoor positioning using audible sound. In *Second ieee consumer communications and networking conference, 2005. ccnc. 2005*, 348–353. IEEE.

[46] Mariakakis, Alex T, Souvik Sen, Jeongkeun Lee & Kyu-Han Kim. 2014. Sail: Single access point-based indoor localization. In *Proceedings of the 12th annual international conference on mobile systems, applications, and services*, 315–328. ACM.

[47] McCartney, Margaret. 2016. Margaret mccartney: Game on for pokémon go. *BMJ: British Medical Journal (Online)* 354.

[48] Milanović, Zoran, Nenad Stojiljković, Ljubomir Pavlović, Vladimir Antić & Nemanja Stanković. 2016. Accupedo pedometer: daily walking step counter. *British journal of sports medicine* 50(22). 1417–1418.

[49] Misra, Pratap & Per Enge. 2006. *Global positioning system: Signals, measurements and performance second edition*. Lincoln, MA: Ganga-Jamuna Press.

[50] Murnane, Elizabeth L, David Huffaker & Gueorgi Kossinets. 2015. Mobile health apps: adoption, adherence, and abandonment. In *Adjunct proceedings of the 2015 acm international joint conference on pervasive and ubiquitous computing and proceedings of the 2015 acm international symposium on wearable computers*, 261–264. ACM.

[51] Ni, Lionel M, Yunhao Liu, Yiu Cho Lau & Abhishek P Patil. 2004. Landmarc: indoor location sensing using active rfid. *Wireless networks* 10(6). 701–710.

[52] Pan, Sinno Jialin & Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on* 22(10). 1345–1359.

[53] Pei, Yuanteng & Matt W Mutka. 2012. Steiner traveler: Relay deployment for remote sensing in heterogeneous multi-robot exploration. In *Robotics and automation (icra), 2012 ieee international conference on*, 1551–1556.

[54] Philipp, Damian, Patrick Baier, Christoph Dibak, Frank Durr, Kurt Rothermel, Steffen Becker, Minin Peter & Dieter Fritsch. 2014. Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data. In *Pervasive computing and communications (percom), 2014 ieee international conference on*, 139–147. IEEE.

[55] Priyantha, Nissanka Bodhi. 2005. *The cricket indoor location system*: Massachusetts Institute of Technology dissertation.

[56] Qiu, Chen & Matt W Mutka. 2015. Aicloc: mobile robots assisted indoor localization. In *Mobile ad-hoc and sensor systems (mass), 2015 ieee 12th international conference on*, IEEE.

[57] Qiu, Chen & Matt W Mutka. 2015. Cooperation among smartphones to improve indoor position information. In *World of wireless, mobile and multimedia networks (wowmom), 2015 ieee 16th international symposium on a*, 1–9. IEEE.

[58] Qiu, Chen & Matt W Mutka. 2016. iframe: Dynamic indoor map construction through automatic mobile sensing. In *Pervasive computing and communications (percom), 2016 ieee international conference on*, IEEE.

[59] Qiu, Chen & Matt W Mutka. 2017. Silent whistle: Effective indoor positioning with assistance from acoustic sensing on smartphones. In *World of wireless, mobile and multimedia networks (wowmom), 2017 ieee 18th international symposium on a*, IEEE.

[60] Qiu, Chen & Matt W Mutka. 2017. Walk and learn: Self-improving indoor localization by profiling outdoor movement on smartphones. In *World of wireless, mobile and multimedia networks (wowmom), 2017 ieee 18th international symposium on a*, IEEE.

[61] Radu, Valentin, Panagiota Katsikouli, Rik Sarkar & Mahesh K Marina. 2014. A semi-supervised learning approach for robust indoor-outdoor detection with smartphones. In *Proceedings of the 12th acm conference on embedded network sensor systems*, 280–294. ACM.

[62] Rai, Anshul, Krishna Kant Chintalapudi, Venkata N Padmanabhan & Rijurekha Sen. 2012. Zee: zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th annual international conference on mobile computing and networking*, 293–304. ACM.

[63] Robotics, CH. 2013. Um6 ultra-miniature orientation sensor datasheet. *CH Robotics* .

[64] Rogers III, John G, Carlos Nieto-Granda & Henrik I Christensen. 2013. Coordination strategies for multi-robot exploration and mapping. In *Experimental robotics*, 231–243. Springer.

[65] Samson, Monique M, IB Meeuwsen, Alan Crowe, JA Dessens, Sijmen A Duursma & HJ Verhaar. 2000. Relationships between physical performance measures, age, height and body weight in healthy adults. *Age and ageing* 29(3). 235–242.

[66] Sattler, Torsten, Bastian Leibe & Leif Kobbelt. 2011. Fast image-based localization using direct 2d-to-3d matching. In *Computer vision (iccv), 2011 ieee international conference on*, 667–674. IEEE.

[67] Seifeldin, Moustafa, Ahmed Saeed, Ahmed E Kosba, Amr El-Keyi & Moustafa Youssef. 2013. Nuzzer: A large-scale device-free passive localization system for wireless environments. *Mobile Computing, IEEE Transactions on* 12(7). 1321–1334.

[68] Sen, Souvik, Jeongkeun Lee, Kyu-Han Kim & Paul Congdon. 2013. Avoiding multipath to revive inbuilding wifi localization. In *Proceeding of the 11th annual international conference on mobile systems, applications, and services*, 249–262. ACM.

[69] Shen, Guobin, Zhuo Chen, Peichao Zhang, Thomas Moscibroda & Yongguang Zhang. 2013. Walkie-markie: Indoor pathway mapping made easy. In *Proceedings of the 10th usenix conference on networked systems design and implementation*, 85–98. USENIX Association.

[70] Soh, Wee-Seng et al. 2007. A comprehensive study of bluetooth signal parameters for localization. In *Personal, indoor and mobile radio communications, 2007. pimrc 2007. ieee 18th international symposium on*, 1–5. IEEE.

[71] Steinhoff, Ulrich & Bernt Schiele. 2010. Dead reckoning from the pocket-an experimental study. In *Pervasive computing and communications (percom), 2010 ieee international conference on*, 162–170. IEEE.

[72] Sturm, Jürgen, Nikolas Engelhard, Felix Endres, Wolfram Burgard & Daniel Cremers. 2012. A benchmark for the evaluation of rgb-d slam systems. In *Intelligent robots and systems (iros), 2012 ieee/rsj international conference on*, 573–580. IEEE.

[73] Suger, Benjamin, Gian Diego Tipaldi, Luciano Spinello & Wolfram Burgard. 2014. An approach to solving large-scale slam problems with a small memory footprint. In *Robotics and automation (icra), 2014 ieee international conference on*, 3632–3637. IEEE.

[74] Tian, Yang, Ruipeng Gao, Kaigui Bian, Fan Ye, Tao Wang, Yizhou Wang & Xiaoming Li. 2014. Towards ubiquitous indoor localization service leveraging environmental physical features. In *Infocom, 2014 proceedings ieee*, 55–63. IEEE.

[75] Tung, Yu-Chih & Kang G Shin. 2015. Echotag: Accurate infrastructure-free indoor location tagging with smartphones. In *Proceedings of the 21st annual international conference on mobile computing and networking*, 525–536. ACM.

[76] Tversky, Amos & Itamar Gati. 1982. Similarity, separability, and the triangle inequality. *Psychological review* 89(2). 123.

[77] Wang, He, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef & Romit Roy Choudhury. 2012. No need to war-drive: unsupervised indoor localization. In *Proceedings of the 10th international conference on mobile systems, applications, and services*, 197–210. ACM.

[78] Wang, Ju, Hongbo Jiang, Jie Xiong, Kyle Jamieson, Xiaojiang Chen, Dingyi Feng & Binbin Xie. 2016. Robust indoor localization using unmodified light fixtures. In *Proceedings of the 22th annual international conference on mobile computing and networking*, .

[79] Ward, Andy, Alan Jones & Andy Hopper. 1997. A new location technique for the active office. *Personal Communications, IEEE* 4(5). 42–47.

[80] Werner, Martin, Moritz Kessel & Chadly Marouane. 2011. Indoor positioning using smartphone camera. In *Indoor positioning and indoor navigation (ipin), 2011 international conference on*, 1–6. IEEE.

[81] Wu, Chenshu, Zheng Yang, Yunhao Liu & Wei Xi. 2013. Will: Wireless indoor localization without site survey. *Parallel and Distributed Systems, IEEE Transactions on* 24(4). 839–848.

[82] Xi, Wei, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu & Zhiping Jiang. 2014. Electronic frog eye: Counting crowd using wifi. In *Infocom, 2014 proceedings ieee*, 361–369. IEEE.

[83] Xiao, Jiang, Kaishun Wu, Youwen Yi, Lu Wang & Lionel M Ni. 2013. Pilot: Passive device-free indoor localization using channel state information. In *Distributed computing systems (icdcs), 2013 ieee 33rd international conference on*, 236–245. IEEE.

[84] Xu, Chenren, Bernhard Firner, Robert S Moore, Yanyong Zhang, Wade Trappe, Richard Howard, Feixiong Zhang & Ning An. 2013. Scpl: indoor device-free multi-subject counting and localization using radio signal strength. In *Information processing in sensor networks (ipsn), 2013 acm/ieee international conference on*, 79–90. IEEE.

[85] Xu, Qiang, Rong Zheng & Steve Hranilovic. 2015. Idyll: indoor localization using inertial and light sensors on smartphones. In *Proceedings of the 2015 acm international joint conference on pervasive and ubiquitous computing*, 307–318. ACM.

[86] Yang, Jun, Emmanuel Munguia-Tapia & Simon Gibbs. 2013. Efficient in-pocket detection with mobile phones. In *Proceedings of the 2013 acm conference on pervasive and ubiquitous computing adjunct publication*, 31–34. ACM.

[87] Yang, Lei, Yekui Chen, Xiang-Yang Li, Chaowei Xiao, Mo Li & Yunhao Liu. 2014. Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices. In *Proceedings of the 20th annual international conference on mobile computing and networking*, 237–248. ACM.

[88] Yang, Zheng, Chenshu Wu & Yunhao Liu. 2012. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proceedings of the 18th annual international conference on mobile computing and networking*, 269–280. ACM.

[89] Yang, Zheng, Zimu Zhou & Yunhao Liu. 2013. From rssi to csi: Indoor localization via channel response. *ACM Computing Surveys (CSUR)* 46(2). 25.

[90] Yang, Zhice, Zeyu Wang, Jiansong Zhang, Chenyu Huang & Qian Zhang. 2015. Wearables can afford: Light-weight indoor positioning with visible light. In *Proceedings of the 13th annual international conference on mobile systems, applications, and services*, 317–330. ACM.

[91] Youssef, Moustafa, Matthew Mah & Ashok Agrawala. 2007. Challenges: device-free passive localization for wireless environments. In *Proceedings of the 13th annual acm international conference on mobile computing and networking*, 222–229. ACM.

[92] Yuan, Yaoxuan, Chen Qiu, Wei Xi & Jizhong Zhao. 2011. Crowd density estimation using wireless sensor networks. In *Mobile ad-hoc and sensor networks (msn), 2011 seventh international conference on*, 138–145. IEEE.

[93] Zhang, Chi & Xinyu Zhang. 2016. Robust indoor localization using unmodified light fixtures. In *Proceedings of the 22th annual international conference on mobile computing and networking*, .

[94] Zheng, Xin, Guanqun Bao, Ruijun Fu & Kaveh Pahlavan. 2012. The performance of simulated annealing algorithms for wi-fi localization using google indoor map. In *Vehicular technology conference (vtc fall), 2012 ieee*, 1–5. IEEE.

[95] Zhou, Pengfei, Yuanqing Zheng, Zhenjiang Li, Mo Li & Guobin Shen. 2012. Iodetector: A generic service for indoor outdoor detection. In *Proceedings of the 10th acm conference on embedded network sensor systems*, 113–126. ACM.