Smartphone-based indoor positioning using Wi-Fi, inertial sensors and Bluetooth

Viet-Cuong Ta

Communauté

UNIVERSITE Grenoble Alpes
Hanoi University of Science & Technology

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

préparée dans le cadre d'une cotutelle entre **la Communauté Université Grenoble Alpes** et **Hanoi University of Science & Technology**

Spécialité : **Informatique**

Arrêté ministériel : 25 mai 2016

Présentée par :

Viet-Cuong TA

Thèse dirigée par : **Éric CASTELLI**

codirigée par : **Trung-Kien DAO**

codirigée par : **Dominique VAUFREYDAZ**

préparée au sein de l'**Institut de Recherche International MICA**

et du **Laboratoire d'Informatique de Grenoble/Inria équipe *Pervasive Interaction***

dans **l'Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique**

SMARTPHONE-BASED INDOOR POSITIONING USING WIFI, INERTIAL SENSORS AND BLUETOOTH

Thèse soutenue publiquement le **15 décembre 2017** devant le jury composé de :

Fawzi Nashashibi,
Directeur de Recherche, Inria (Paris), Président
Yacine Amirat,
Professeur Univ. Paris-Est-Creteil-Val-De-Marne (Paris), Rapporteur
François Charpillet,
Directeur de Recherche, Loria (Nancy), Rapporteur
Eric Castelli,
Chargé de Recherche, CNRS, UMR MICA (Hanoï, Vietnam), Directeur
Dominique Vaufreydaz,
Maître de Conférences, Université Grenoble Alpes/Inria, Co-directeur
Trung-Kien Dao,
Maître de Conférences, HUST (Hanoï, Vietnam), Co-directeur

Acknowledgements

I would like to express my special appreciation for my advisors, Professor Eric Castelli, Dominique Vaufreydaz and Dao Trung Kien. They have provided me countless supports from the beginning of my study to this point. I also would like to thank all the members of the PSI team in MICA and the Pervasive Interaction team in INRIA-Grenoble for all the research advices and also the friendly, research-oriented environment. I also want to thank the 911 Program, MICA Institute and INRIA-Grenoble Institute for the financial and technical supports. A special thanks to my family who provide me the vital support for chasing my research career. I would like to thank all my friends for their cheers and advices through the study.

Abstract

Position information provides useful data for a wide range of applications, including tracking, assisting, automation, energy saving, etc. There are a number of technologies which can be used for positioning task. With the popularity of smartphones and tablets in daily life, the task of finding users' position through their phones gains much attention from both the research and industry communities. Technologies integrated in smartphones such as GPS, Wi-Fi, Bluetooth and camera are all capable for building a positioning system. Among those technologies, GPS approaches have become a standard and achieved much success for the outdoor environment. Meanwhile, Wi-Fi, inertial sensors and Bluetooth are more preferred for positioning task in indoor environment. In our work, the performance of the three above technologies for positioning users by their smartphones in indoor environment is studied.

For smartphone positioning, Wi-Fi fingerprinting based approaches are well established within the field. Generally speaking, the approaches attempt to learn the mapping function from Wi-Fi signal characteristics to the real world position. They usually require a large amount of data for finding a good mapping. When the available training data is limited, the fingerprinting-based approach has high errors and becomes less stable. In our work, we want to explore different approaches of Wi-Fi fingerprinting methods for dealing with a lack of training data. Based on the performance of the individual approaches, several ensemble strategies are proposed to improve the overall positioning performance. All the proposed methods are tested against a published dataset, which is used as the data for the competition at the IPIN 2016 Conference with offsite track (Track 3). For detecting the user's floor, the accuracy can reach 93% of accuracy on the testing data. For computing the user's position, standard learning models could reach around 6m in average distance error. By combining these models, the mean distance error can be reduced to around 5.12m.

Besides the positioning system based on Wi-Fi technology, the smartphone's inertial sensors are also useful for the tracking task. The three types of sensors, which are acceleration, gyroscope and magnetic ones, can be employed to create a Step-And-Heading (SHS) system. Several methods are tested in our approaches. The number of steps and user's moving distance are calculated from the accelerometer data. The user's heading is calculated from the three types of data with three methods, including rotation matrix, Complimentary Filter and Madgwick Filter. Between the step and heading component of the user's movement, the heading is usually more affected by noisy data and has more contribution to the tracking errors overtime. Practically, it is difficult to have an error-free heading estimation method. Therefore, it is reasonable to combine SHS outputs with the outputs from Wi-Fi because both technologies are generally present in smartphones. Two combination approaches are tested. The first approach is to use directly the Wi-Fi outputs as pivot points for fixing the SHS tracking part. In the second approach, we rely on the Wi-Fi signal to build an observation model, which is then integrated into the particle filter approximation step. The combining paths have a significant improvement from the SHS tracking only and the Wi-Fi only. Although, SHS tracking with Wi-Fi fingerprinting improvement achieves promising results, it has a number of limitations such as dependence on the user's walking styles and restriction on smartphone handling position and orientation.

In the context of multiple users, Bluetooth technology on smartphones could provide the approximated distance between users. The relative distance is calculated from the Bluetooth inquiry process. It is then used to improve the output from Wi-Fi positioning models. We study two different combination methods. The first method aims to build an error function which is possible to model the noise in the Wi-Fi output and Bluetooth approximated distance for each specific time interval. It ignores the temporal relationship between successive Wi-Fi outputs. Position adjustments are then computed by minimizing the error function. The second method considers the temporal relationship and the movement constraint when the user moves around the area. The tracking step are carried out by using particle filter. The observation model of the particle filter is a combination between the Wi-Fi data and Bluetooth data. Both approaches are tested against real data, which include up to four different users moving

in an office environment. While the first approach is only applicable in some specific scenarios, the second approach has a significant improvement from the position output based on Wi-Fi fingerprinting model only.

Key words: indoor positioning, smartphone-based tracking, Wi-Fi, fingerprinting, inertial sensors, Bluetooth, collaborative positioning.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Name
AGPS	Assistant-GPS
AOA	Angle Of Arrival
BLE	Bluetooth Low Energy
CDMA	Code Division Multiple Access
CNN	Convolution Neural Network
GPS	Global Position System
GSM	Global System for Mobile
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IPIN	Indoor Positioning and Indoor Navigation
KNN	K-Nearest Neighbors
LDPL	Log-Distance Path Loss
LTE	Long Term Evolution
MEMS	Micro Electro-Mechanical Systems
ORB	Oriented Fast and rotated Binary robust independent elementary features
PDR	Pedestrian Dead Reckoning
PTAM	Parallel Tracking and Mapping
RF	Random Forest
RFID	Radio Frequency Identification
RSS	Received Signal Strength
RSSI	Received Signal Strength Index
RTOF	Round Time Of Flight
SHS	Step-Heading-System
SLAM	Simultaneous Localization And Mapping
std	Standard Deviation
TDOA	Time Different Of Arrival
TOA	Time Of Arrival
UWB	Ultra-Wide Band
WAF	Wall Attenuation Factor
WLAN	Wireless Local Area Network
XGB	Extreme Gradient Boost

Chapter 1 Introduction

Position information is useful data for a wide range of real applications, including tracking, assisting, automation, energy saving, etc. Basically, the task of a positioning system is to identify the position of the users or moving objects within an area. There are a number of technologies which can provide useful information for positioning. Depending on the technologies in use, the accuracy of the positioning estimation can vary from sub-meter to several meters or room sized accuracy. Other aspects like robustness, deployment and management cost, and user friendliness may also play an important role in designing such systems.

With the popularity of smartphones and tablets, the task of finding user's position through their phone gains much attention from both the research and industry communities. There are more than ten kinds of sensors which are present in most of smartphones nowadays. Based on the localization purpose, the sensors can be classified into three separate groups, including wireless-based technology, optical-based technology and sensors-based technology. The wireless-based technology includes GPS, cellular, Wi-Fi and Bluetooth. The optical-based technology is the smartphone's camera(s). The inertial-sensors-based technology involves the data from accelerometer sensor, gyroscope sensor and magnetic sensor. There are also other sensors such as light sensor and microphone, which could be employed for determining the user's positioning. However, they are not as popular as those in the above three groups.

Within the wireless technology based methods for user positioning task, the GPS has already become an industrial solution for smartphones. Its advantages include the global coverage with meter-accuracy in outdoor environment. Meanwhile, the GPS-solution is not stable enough for indoor environment because the quality of GPS signal decreases by the building. For the remaining technologies, the cellular-based positioning also does not work well in indoor environment for the same reasons. The usage of smartphone's camera requires heavy

computing efforts and is not user-friendly. Therefore, Wi-Fi, Bluetooth and inertial sensors are considered as alternative solutions for indoor environment.

Wi-Fi and Bluetooth work in a similar way. However, the range of Wi-Fi is about two to three times to that of Bluetooth. Wi-Fi is also much more popular than Bluetooth because it is the main technology for high-speed and low-cost wireless network access. Wi-Fi access points nowadays appear in almost every public area. Therefore, Wi-Fi is one of the most interesting technology for smartphone-based positioning. To date, the accuracy of Wi-Fi is expected to be around 5m in terms of mean distance error. The fingerprinting-based approach is among the mostly used approaches for Wi-Fi indoor positioning thanks to the full exploitation of deployment Wi-Fi access points. The main drawback of fingerprinting methods is that the fingerprinting model is required to collect a Wi-Fi signal map for the entire area. This task is usually a time consuming step, which should be done by an expert. However, to make the approach flexible for a large public area, we explore a more user-friendly context of data collecting process. The data is published by the IPIN 2016 Conference with offsite track competition (Track 3) [Torres-Sospedra et al. 2017]. The published data could be collected in an easy way and does not require much labor cost. As a trade-off, it results in less training data than the traditional Wi-Fi fingerprinting database. The limited training data raises many difficulties for learning the mapping from the noisy Wi-Fi signal to the real position. To reduce the effect of noise in the learning phase, we employ several learning methods and feature sets. A further layer of combinations between various models could be added for improving the fingerprinting model.

Besides the data for Wi-Fi technology, smartphone's inertial sensors provide an alternative way to localize the user's phone. The inertial sensors include the accelerometer sensor, gyroscope sensor, and magnetic sensor, that could be grouped for building Step-Heading-System (SHS) tracking. The smartphone is usually restricted to be handled and pointed forward when the user is walking. Due to this restriction, the phone's direction has high correlation to the user's moving direction. The SHS system attempts to compute the step and heading of the user's movement. It then tracks the movement of the user from a given starting point. However, the smartphone's sensors come at a low cost, which results in some degree of noise in the data. Other sources of noise could be the non-calibrated gyroscope and magnetic sensors.

In addition to that, the speed and heading error from each computing step are added up by times. This result leads to a huge drifting in the output position. Therefore, it is mandatory to have an alternative positioning model for reducing the drifting effects. In our work, we choose the output from Wi-Fi fingerprinting model as an additional source of information for making the SHS more reliable. The SHS system is built from the standard step counter and heading estimation. In order to reduce the effect of noise, the two computed values are combined by a particle filter approximation. Several ways to fuse the Wi-Fi and SHS position information are introduced and tested. The first fusing method uses the position output from Wi-Fi scans and adjusts the SHS tracking paths directly. In the second fusion approach, the Wi-Fi output is integrated into an observation model for the SHS's particle filter step. The two proposed fusion approaches could reduce the noise from both the Wi-Fi data and inertial-sensors data significantly. Although the fusion-based approaches have prominent results, several weaknesses of the SHS approaches are difficult to improve, such as the restriction on the phone holding position.

Beside the inertial-sensors data, the Bluetooth technology is considered as another source of data for positioning. Among the available communication on the smartphone, Bluetooth is able to provide information about the neighborhood devices. In a multiple-user context, where multiple smartphones could be simultaneously present, we could employ the information for improving the individual positioning. From the Bluetooth inquiry process, the signal strength from one smartphone to another could be acquired and then used to compute the relative distance between the two devices. Alternatively, the relative distance could be estimated by knowing the two devices' position. We rely on the Wi-Fi data with fingerprinting model for deriving such distance. In an ideal situation, the two computed distance values should be the same. However, because both the Bluetooth estimated distance and Wi-Fi estimated positions are affected by noise, they would be different in general. In our approach, error functions based on the mismatch between the two distances are built. The user's position is then updated based on the minimal values of the functions. Two types of functions are proposed. The first one neglects the temporal relationship between successive user's estimated positions. It only focuses on the errors of individual technologies. Two separate Gaussians error distribution models are employed for the Bluetooth data and Wi-Fi data. In the

second approach, the movement of the users are included into the error function. We employ particle-filter-based tracking for minimizing the error function. The particle filter tracks the user by a simple motion model. Additional map-based information is added for removing bad particles. The observation model of the particle filter is combined from the Bluetooth data and Wi-Fi data. Both approaches are tested with real scenarios which include multiple devices. The testing results show that Bluetooth-based relative distance could provide sufficient information for improving the user's localization results in a multiple-user context.

Chapter 2 Literature Review

Nowadays, different sources of information, which come from various underlying technologies, can be used for localization purpose. These technologies are present in different types of devices, such as cameras, motion sensors, identity cards or smartphones. In general, each technology comes with a set of parameters such as type of data, update rate, cover area and the amount of noise. Therefore, the methods to extract the user's position from one specific technology will mostly differ to another technology. At the highest abstract level, these methods can be roughly divided into three main categories: optical-based methods, wireless-based methods and inertial-sensor-based methods. In this chapter, we first start to review some positioning methods for localization purpose. In the second part, user localization by smartphone are presented.

2.1 Optical-based Methods

With the recent advances of technology in image processing area, optical-based positioning has become one of the dominating technique for indoor positioning, which can achieve a high level of accuracy [Mautz and Tilch, 2011]. These systems can be divided into two separate categories, including mobile cameras and fixed cameras. With fixed-camera approach, one or more cameras are placed at several locations in the environment. The cameras then detect and track moving objects through the scene. In the mobile-camera approach, the camera captures the scene while it is moved around.

For both categories, various image-processing techniques are employed to detect specific patterns within the captured imaged. The patterns represent moving objects or specific landmarks. The transformation between the image coordinates and the world coordinates could be established by standard collinearity model:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X^c \\ Y^c \\ Z^c \end{pmatrix} + \lambda R \begin{pmatrix} x - x_p \\ y - y_p \\ -c \end{pmatrix} \quad (1)$$

Equation (1) is introduced in photogrammetric reference books such as Luhmann et al. [2006]. The two points (X, Y, Z) and (X^c, Y^c, Z^c) are the positions of the interested patterns and the camera center in world coordinates respectively. Constant λ is the distance factor and R is the rotation matrix. The point (x, y) is given in the image coordinates with (x_p, y_p) as the principal point and c is the principal distance. The values of R, (x_p, y_p) and c are involved in the intrinsic and extrinsic parameters of the camera. Parameters which describe the camera distortion could be added to Equation (1) for a more robust transformation.

2.1.1 **Fixed Cameras**

The works for fixed-camera tracking are usually referred as visual tracking. All the necessary information, the camera's intrinsic and extrinsic parameters and camera's position are assumed to be known. Therefore, it is possible to establish the coordinate transformation from the image coordinates to the real world coordinates by Equation (1). This task is then evolved to find the tracking objects within an image or a set of images. The area of object is usually named as the region of interest (ROI) or target region. Novel approaches include environment modeling, motion segmentation and object classification. More details for each approach could be found in [Hu et al. 2004]. Recently, deep convolution neural networks (CNN) are applied for visual tracking purpose, such as in Nam and Han [2016] and Wang et al. [2015].

For the localization purpose, the visual tracker also needs to identify the moving objects across a sequence of images. Therefore, it should be able to combine the results from different frames into stable object trajectories. When more than one objects are tracked, the tracker is also required to differ between each of the objects' region. In early works, Trucco and Plakas [2006] discussed in depth these methods, which include window tracking, feature tracking, planar rigid shapes, solid rigid shapes, contours tracking and visual learning. In more recent works, the trackers are divided by several characteristics such as tracking region, appearance model, motion model and update methods. A novel model for tracker is presented in Smeulders et al. [2014] (Figure 1).

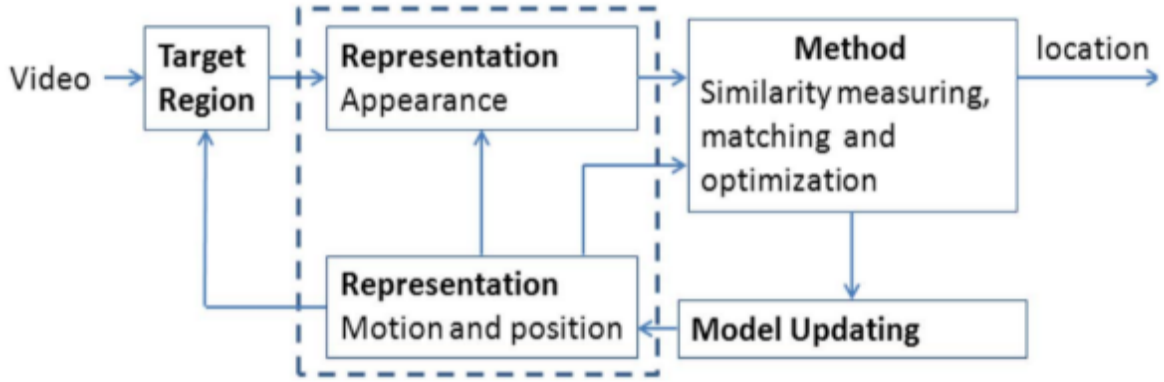


Figure 1. Model for objects tracking in a sequence of images [Smeulders et al., 2014]

The model involves a representation model and then a matching step for two consecutive frames in a sequence of images. The representation of object could be a rectangle of image-like data, histogram based or feature-vector based. The matching step usually depends on the object representation. Methods based on gradient ascent, subspace matching or discriminative supervised classifier are available. The performance of the novel tracking methods is compared in Smeulders et al. [2014]. Recently, in the Visual Object Tracking challenge 2015, Kristan et al. [2015] presents the results of 62 trackers. The challenge focuses on the tracking context of single-camera, single-target, model-free, casual-trackers and short-term. The dataset contains several visual attributes such as occlusion, illumination change, motion change, size change and camera motion. Deep CNN based approaches are among the top performance trackers.

For tracking users at large scale, multiple cameras could be involved. The use of multiple cameras is needed as the point of view from one single camera has issues for covering a large area, especially in indoor environment. The existence of walls and objects in indoor environment would block the line-of-sight between the camera and the tracking targets. A general framework for multiple cameras surveillance is proposed by Hu et al. [2004] (Figure 2).

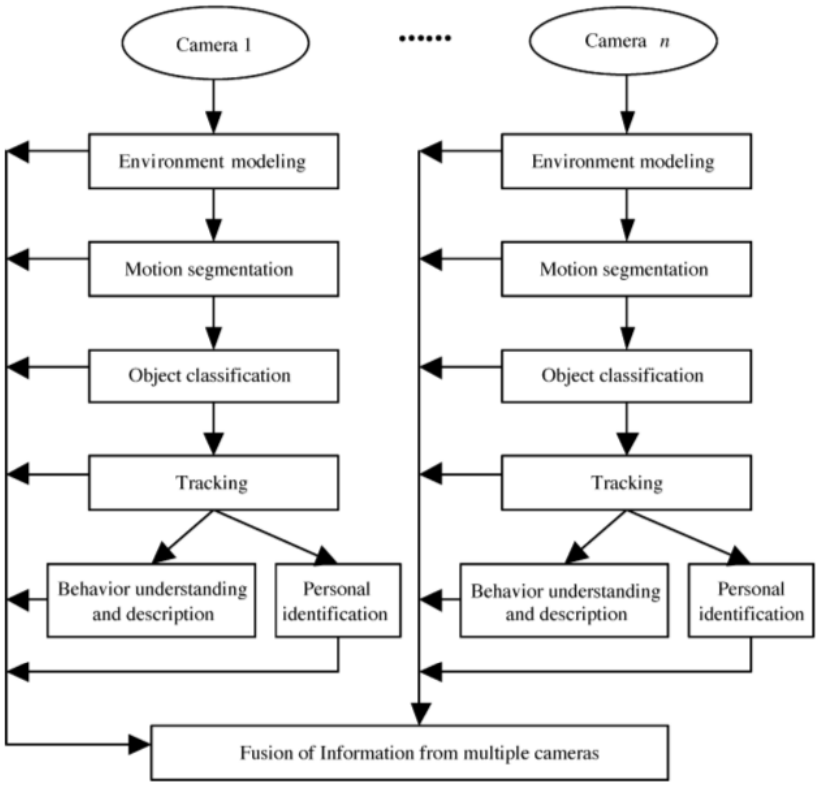


Figure 2. General framework for video surveillance across multiple cameras [Hu et al., 2004]

In general, when the tracking task is expanded from one camera to multiple cameras, the main challenge is the association between target objects appearing in the fields of view of different cameras. The camera installation and location also play an important part in such systems. The installation is usually required to have overlapping regions between cameras' field of view. It is also need a calibration process for transforming image-coordinates from one camera to other cameras. For example, Zhang et al. [2009] propose a calibration approach, which maps the point in each camera's field of view to a global map by a semi-automatic process. The approach is able to fuse the tracking results up to 50 individual cameras.

In a non-overlap multiple camera setup, the tracking system should be able to identify the user, who disappears from one camera and reappears under the field of view of another camera. This issue usually arises for the indoor environment where it is difficult to get the overlapping field of view from different corridors, doors and rooms. For instance, one setup for indoor environment is presented by Thi-Thanh-Thuy et al. [2016] (Figure 3). There are four cameras which are placed at different places in the office environment. The setup is mixed

between sharing fields of view and non-sharing fields of view, thus, makes the tracking task become more challenging. For tracking people in the entire area, it is required a matching step through the images of different cameras. The work discusses in-depth several challenges of tracking across multiple cameras, such as user's pose, scale in variation and user's occlusion. In order to solve the re-identification tasks, they propose a support vector machine ranking method using Kernel Descriptors as the image-based features for the person [Bo et al., 2010]. The proposed work has achieved prominent results, though, it is still necessary to have an additional wireless-based position system to stabilize the person matching performance. The other challenges for multiple camera tracking include calibration of cameras, camera's setup topology and information fusion [Wang and Lu, 2017].

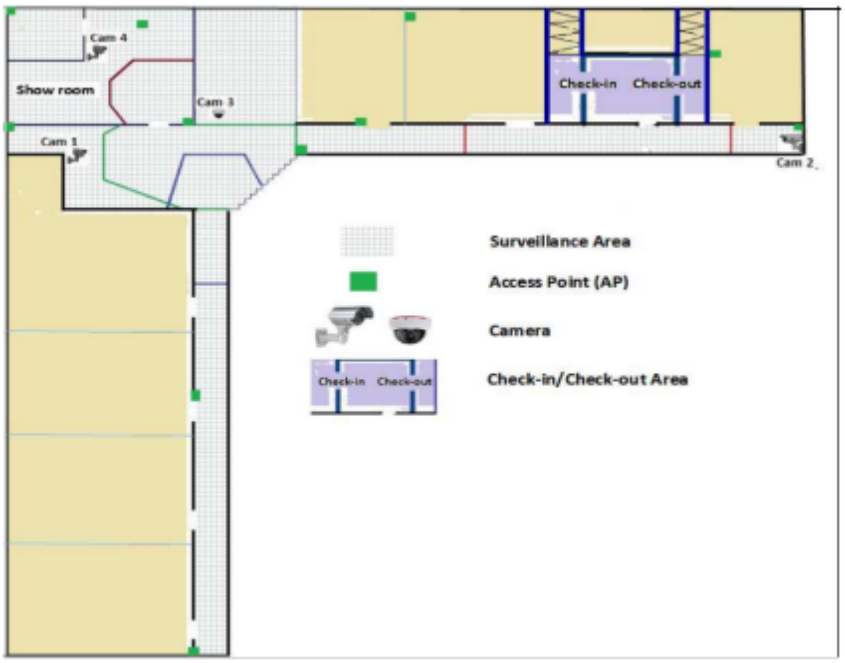


Figure 3. A setup for tracking with non-overlap area between multiple cameras [Thi-Thanh-Thuy et al., 2016]

2.1.2 **Mobile Camera**

The task of tracking when the camera moves is also known as visual navigation. The position of the camera is calculated based on the received image sequence. It is usually needed an additional reference database to be able to calculate the camera position. The standard approach is based on image reference database. For example, Figure 4 illustrates this approach.

Computer vision algorithm is used to compare the similarity between different images. However, as the camera is moving, the matching computation needs to be done in real time, which is one of the main challenges of this approach. The other challenges include the error accumulated over time and environmental changing. The work proposed by Ido et al. [2009] uses a camera mounted on the head of a humanoid robot for indoor navigation purpose in the indoor environment. In Deretey et al. [2015], the authors collect a large amount of images of a specific scene to build the entire structured of the scene. The image reference database is converted into a feature database. To positioning task is reduced from the standard image matching to feature matching, which make the computation less heavy. The results are reported at less than 10mm for an office and a laboratory environment.

Figure 4. Reference image database (left) is used for calculating the camera position (right image) [Mautz and Tilch, 2011]

In order to reduce the computational complexity of the image reference database approach, specific type coded targets could be deployed as landmarks. For instance, Atiya and Hager [1993] propose a method of using stereo images for robot positioning. The landmarks' role is to provide a robust matching between image-coordinate system and real-world-coordinate system. The proposed approach first detects the landmarks through various robot poses and tries to match it with known landmarks' coordinates in the database. Then, a geometric-based solution is developed to find the robot's pose and position. Figure 5 illustrates their experiment setups. An error of less than 1 centimeter is reported. Camera distortion effects and computation complexity are also discussed in the article. In some recent works, the localization task could be considered as a sub-task in the field Simultaneous Localization And Mapping (SLAM). SLAM is the task of building the environment map and then using this map to

deduce the object location at the same time [Aulinas et al., 2008]. When the cameras are used as the main sensors, the task is referred as visual SLAM [Se et al., 2005]. In visual SLAM, the landmarks are discovered and registered their positions automatically when the camera is moved around the scene. There are several novel approaches for visual SLAM such as PTAM [Strasdat et al., 2011], CD-SLAM [Pirker et al., 2011] and ORB-SLAM [Mur-Artal et al., 2015]. Although visual SLAM could provide a good solution for robot localization, the system's performance still varies on the specific type of environment. Complex, dynamic or visual-repetitive environments are among the challenged environments for visual SLAM approaches [Fuentes-Pacheco et al., 2015].

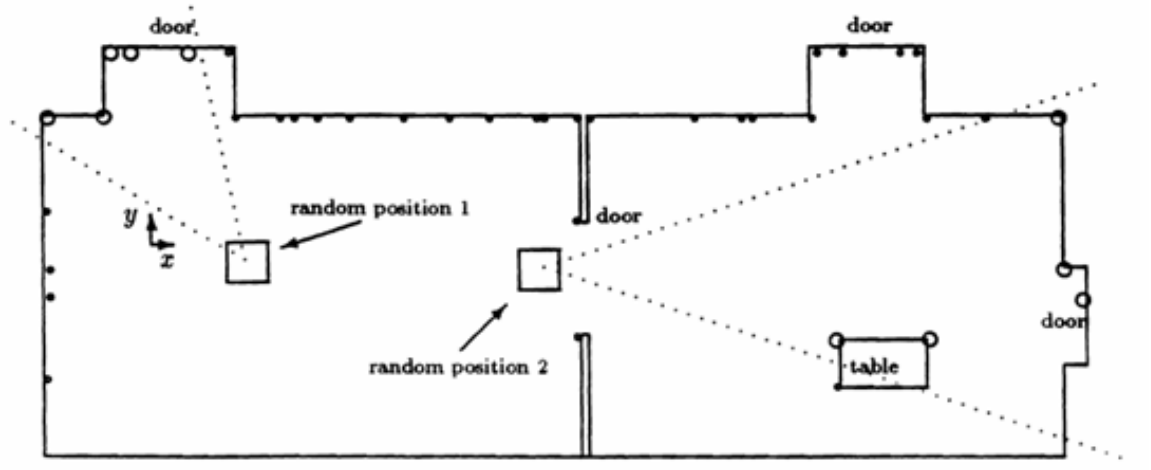


Figure 5. Using installed landmarks for robot localization. The landmarks are drawn as circles in the floor plan and are used to differ between two random positions [Atiya and Hager, 1993].

The laser scanning based positioning could also be grouped into this category. For example, Light Detection and Ranging (LiDAR) based system is a preferred approach for the GPS-denied environments [Miller et al., 2010]. The LiDAR system provides the distance to the nearest obstacle within a large angle. The information then is used to build a point cloud, which is then matched to the stored point cloud database. By combining with the inertial measurements, Soloviev et al. [2007] propose a laser scanner based approach which could reach submeter accuracy.

Because of the requirements for computational capacity and the camera's position, most of the proposed approaches with the mobile cameras only work on moving robots. For extending the work to user localization, it requires the camera to be carried by the user at a specific position. Marouane et al. [2016] proposes a method for step detector from a user carrying camera. The camera setup is illustrated in Figure 6.

Figure 6. User-carrying camera setup for step detection [Marouane et al., 2016]

The results show that the proposed system is able to compete against other step counting methods, which are produced by Apple iPhone 6 and Apple iPod Nano (6^{th} generation). In Mulloni et al. [2009], a landmark-based method is proposed for localizing smartphones in the indoor environment. The approach, first, requires the markers to be deployed at some specific locations within the environment. The user is required to scan the available markers in the area. The Figure 7 gives an illustration of the scan step with a phone's camera. The images are then processed to compute the phone's position. The works have developed and tested in real world deployment.

Figure 7. A user scans the marker with the smartphone [Mulloni et al.,2009]

To summarize, depending on the application of the positioning task and the type of camera in use, the accuracy of optical-based systems could range from less than 1cm to 0.5m [Mautz and Tilch, 2011].

2.2 Wireless-based Methods

Compare to optical-based methods, wireless based approach brings more flexible options for the positioning tasks. It covers a great range in terms of real world applications such as self-organizing sensor networks, location sensitive billing, ubiquitous computing, context-dependent information services, tracking and guiding [Liu et al., 2007]. The hardware implementation usually includes a set of base stations and tracked devices. The base stations can be called the landmarks, which positions could be available or unavailable based on specific localization approaches. The tracked devices are the mobile devices. The mobile devices have a certain form of communication with the base stations. Their locations are then calculated based on the data of this communication. In most of the systems, the base stations play the role of the signal transmitter and the mobile devices are the signal receivers. The type of signal could be radio wave or sound wave. The propagation properties depend on the wave length and the environment. Popular technologies are GPS-based, mobile cellular network, wireless local area network (WLAN), radio frequency identification (RFID), Bluetooth and Ultra-Wide Band (UWB). The techniques also vary based upon the specific technologies. Generally, the popular

approaches for wireless-based positioning can be grouped into geometry-based approaches, fingerprinting-based approach and proximity-based approach.

2.2.1 **Techniques**

2.2.1.1 *Geometry-based*

The geometry based measures the object position by the related geometry characteristic between the tracked device and the base stations. The most widely used properties are distance between the two. It is often required that the position of the base stations is available. By knowing the distance between the mobile device and one of the base stations, we could deduce that the object position would be on a sphere. The exact position then can be calculated by intersections of those spheres.

Formally, let the object location be (X, Y, Z), which need to be calculated. The base stations are located at (x_i, y_i, z_i) and have the distances d_i to the object. In case of three stations only, the positioning problem can be converted to solve the a system of equations with the unknowns are (x, y, z):

$$\begin{cases} \sqrt{(x-x_1)^2+(y-y_1)^2+(z-z_1)^2} = d_1 \\ \sqrt{(x-x_2)^2+(y-y_2)^2+(z-z_2)^2} = d_2 \\ \sqrt{(x-x_3)^2+(y-y_3)^2+(z-z_3)^2} = d_3 \end{cases} \quad (2)$$

Figure 8 illustrates the way to find the object location in a geometric way. Equation (2) could results in two possible solutions. One of the two possible solutions could be excluded by adding constraints. The main obstacle of this approach is to find the distances d_i accurately. There are several techniques, such as Time of Arrival (TOA), Time Difference of Arrival (TDOA), Round Time of Flight (RTOF) and Received Signal Strength (RSS) based.

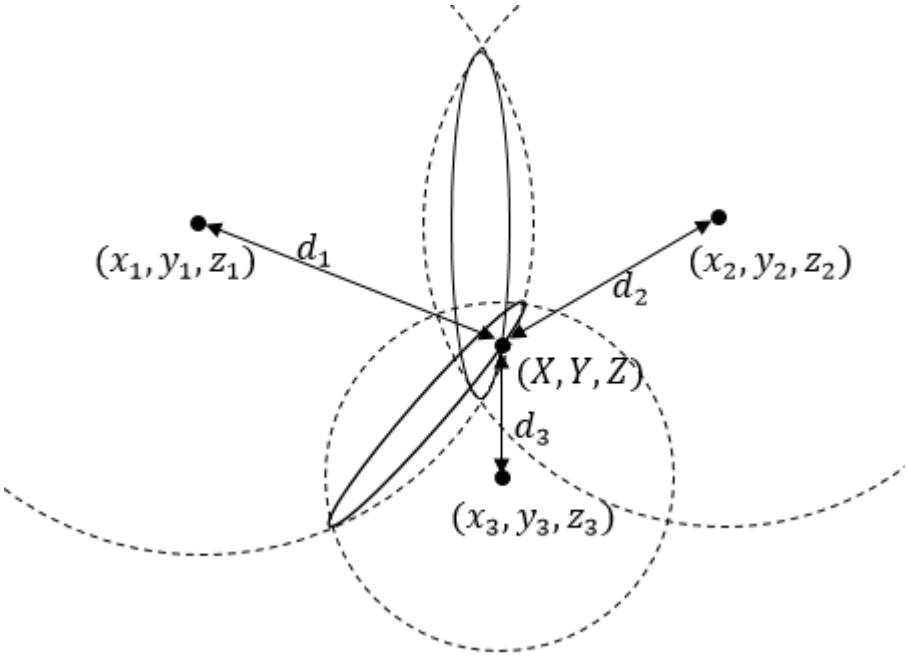


Figure 8. An object location (X, Y, Z) could be identified by knowing its distance to three base stations

The TOA approach uses the straightforward approach, which calculates the distances d_i from the signal's velocity and the signal's travel time. The signal between the tracked device and each base station is transmitted with the timestamp of transmission. The travel time then can be derived from the timestamp. This approach is usually required both the base stations and the receiver are synchronized properly. There are several technologies using this technique. They include Global Position System (GPS) [BH-Wellenhof and Collins, 1997; Misra and Enge, 2006], Ultra-Wide Band [Fontana and Gunderson, 2002] and Wireless Local Area Networks [Ali and Omar, 2005; Golden and Bateman, 2007].

In Priyantha et al. [2000], TDOA is used for measuring the distance between two devices. The authors implement a system which transmits simultaneously two types of signal, a radio-based signal and a sound-based signal. The received time difference at the receivers is then used to calculate the distance from the transmitter to the receivers. Gentner and Jost [2013] use the multipath effects for calculating the TDOA.

With the RTOF approaches, one of the two ends acts as a reflect unit. The signal is reflected to the transmitter and the time is calculated based on the time of a round trip flight [Günther

and Hoene, 2005]. Therefore, it is not required to synchronize time between the tracked device and the base stations like in the TOA-based approach.

In RSS based approach, the distance is calculated from the signal strength at the receiver. When the signal is transmitted from the base station to the mobile device, its propagation properties could be modeled as a function of frequency, distance and environment's factors. Generally, it is difficult to create a general model for covering all the possible situations. For the positioning purpose, the log-distance path loss model (LDPL) model and its variation are widely used. It provides a simple estimation of the signal power based on its distance. The signal power $P(d)$ at distance d from the base station is computed as:

$$P(d) = P(d_0) - 10n\log\frac{d}{d_0} \quad (3)$$

where $P(d_0)$ is the value at a reference point with the distance d_0 and n is the path loss exponent. While the value of P_0 needs to be measured directly for each specific system setup, the path loss exponent n mostly depends on the type of environment. Some values of n are presented in the Table 1 [Rappaport et al., 1996]. If the signal strength $P(d)$ can be measured at the mobile device, it is able to compute the distance between the device to the base station by using the formula in Equation (3).

Table 1. The path loss exponent values in some popular environments [Rappaport et al., 1996]

Environment	**n**
In building line of sight	1.6 to 1.8
Free space	2
Urban area	2.7 to 3.5
Shadowed urban	3 to 5
Obstructed in building	4 to 6

With the presence of many obstacles in the environment, the Wall Attenuation Factor (WAF) is proposed by Bahl and Padmanabhan [2000] as an alternative version of LDPL model. It describes the relationship between the signal strength loss and the obstacles:

$$P(d) = P(d_0) - 10n\log\frac{d}{d_0} - \begin{cases} w * WAF & \text{if } w < C \\ C * WAF & \text{if } w \geq C \end{cases} \quad (4)$$

where C is the maximum number of obstructions which can makes a difference in signal power, w is the number of obstructions which the signal has to pass. The path loss model and others derived formulas are popular among the WLAN technologies and Bluetooth because the technology infrastructure are well supported for the RSS reading. Using the model in Equation (4) a resolution of 4.3m can be achieved with 50^{th} percentile. To get a better performance, more complex models are proposed to simulate the propagation properties of radio signal as proposed by Chintalapudi et al. [2010]; Della Rosa et al. [2010] and Bose and Foh [2007]. These models include a number of parameters and require calibration processes to find an optimization set of parameters.

It is likely that approaches like TOA and RSS-based results in inaccurate distance. Therefore, more base stations could be added to the Equation (2) for improving the system performance. The task of finding the value (X, Y, Z) is then transformed into a constrained minimization problem. More specifically, each known distance d_i between the mobile device and the base station i^{th} is used to form a cost function f_i:

$$f_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} - d_i \qquad (5)$$

where (x_i, y_i, z_i) is the position of the i^{th} base station. It is needed to find (X, Y, Z) which minimizes the sum of cost functions [Liu et al., 2007]:

$$F = \sum_{i=1}^{N} \alpha_i^2 f_i^2 \qquad (6)$$

where N is the number of base stations and α_i reflects the reliability of the distance estimation at base station i. The least-square algorithm could be employed to find the appropriate value of (X, Y, Z) [Blewitt, 1997; Cheung et al., 2004]. In practice, it usually revolves more than three base stations to make the above method applicable. When there are less than three base stations, the function *F* in Equation (6) is unusable for finding the object position (X, Y, Z). For example, in case of GPS-based technologies, it is difficult to measure the GPS receiver position when the signals from satellites to the receiver are blocked by buildings in indoor environment.

Apart from the distance based approach, the angle is also a geometric property could be employed for localization task. In Niculescu and Nath [2003], the authors propose the Angle Of

Arrival (AOA) positioning method. They design a sensor networks which include AOA-capable nodes. The main principal is explained in Figure 9. Four points A, B, C and D are AOA-capable nodes. The three points A, B, C plays the role of landmarks which positions are known. For computing the position (x, y) of D, it is required to know the AOA of D, which includes $A\hat{D}B$, $B\hat{D}C$ and $C\hat{D}A$. The position (x, y) is the intersection of the three respective circles each of AB, BC and CA side.

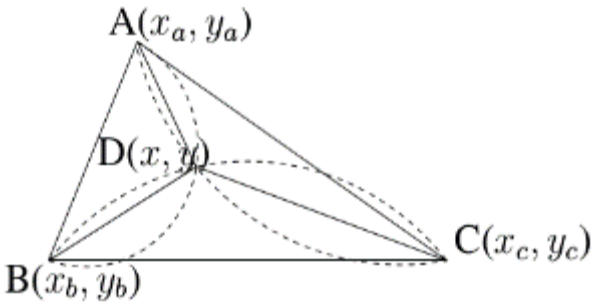


Figure 9. AOA approach for positioning, proposed by Niculescu and Nath [2003]

2.2.1.2 *Fingerprinting-based*

The fingerprinting based techniques focuses on the signal pattern at each specific location. It bases on the assumption that for each specific position has a unique signal pattern. Normally, a fingerprinting based technique includes two separate phases. The first one is site surveying, in which the signal patterns around the site are collected to build a database. Because the signal can vary over time at a specific place, multiple signals from base stations are recorded in a period of time. Assume that we have N base stations, namely $B_1, B_2, \ldots, B_N$. At a specific point $P(x, y, z)$, the signal characteristics are represented by r_i. Mostly, the r_i is the received signal strength (RSS) or the received signal strength index (RSSI). It can be explained by the highly correlation between the position P with the distance to the base station, which then results in the value of RSS. Beside the RSS/RSSI values, other signal characteristics such as direction of arrival could be used for representing the signal features [Tayebi et al., 2009]. Even in more simple cases, the values of r_i could be the presence signal from the base stations i^{th}, which can have 0 or 1 value. The whole signal characteristics of the point P at is created by joining each value of r_i into $r^P = \{r_1, r_2, \ldots, r_N\}$. It is not required that all the base stations

value r_i are present at the position P. For example, if the mobile device and the base station are too far from each other, the RSS value is not available. The results of the first phase are described as a signal map. The map would contain a list of elements. Each element is a pair of the position P and the signal characteristics vector r^P. There are usually multiple signal characteristics vectors r^P for each position.

In the second phase, the tracked device scans the signal and matches the scan results with the signal map to find its location. Generally, this step can be considered as a standard regression task if the positions are continuous values, or a classification task if the positions are treated as discrete values. One of the pioneer works in the field is described in Bahl and Padmanabhan [2000]. This paper proposes the framework of Wi-Fi fingerprinting approach, which introduces several methods for matching RSS-based signal vectors. For comparing the similarity between signal characteristics vector, they employ random selection, strongest base station selection and multiple nearest neighbors. Other issues of fingerprinting approach such as number of points, number of samples per point and device orientations are also discussed in the paper. In the experiments of tracking a mobile user, a median error of 3.5m is reported. Following works in the field mainly focuses on the ways for finding the most appropriate matching methods. There are many possible approaches such as Probabilistic method [Kontkanen et al., 2004], K-Nearest Neighbors (KNN) [Bahl and Padmanabhan, 2000; Saha et al., 2003; Li et al., 2006], Artificial Neural Network [Fang and Lin, 2008; Zhou et al., 2010, Dinh-Van et al., 2017], Support Vector Machine [Wu et al., 2004; Brunato and Battiti, 2005] and Random Forest [Jedari et al., 2015].

In general, the fingerprinting approach suffers the noisy propagation of signal in the environment, especially in the indoor environment. Potential factors that affect the signal characteristics are various such as the presence of wall, door, human, hardware and software dependent [Chen et al., 2005; King et al., 2006]. Thus, it is impossible to find an exact matching signal characteristics vectors from the collected database.

2.2.1.3 *Proximity-based*

In the proximity-based approach, the positioning areas are split into separated regions. The system then identifies which region the mobile device is belong to. The regions can be selected in different principles regarding the environment, the underlying wireless technologies and the signal characteristics. For example, mobile network based techniques are able to use the cell identification (Cell-id) approaches to get a rough estimation of the mobile device [Trevisani and Vitaletti, 2004]. In the proposed system, whenever the device connects to a cell tower of network, its will be positioned at the tower's location. In Wang et al. [2012], the authors suggest a way to cluster Wi-Fi signal into landmarks of $4m^2$ area approximately. The landmarks then are combined with other technologies to provide a more accurate tracking results. The proximity-based approach can be used together with the Bluetooth technology to locate stationary mobile users at room level [Bargh and de Groote, 2008]. The advantage of the proximity-based approach is that it would be easier to find the region of the object's position than to calculate the object's exact position. In practice, some applications such as home automation, can work on a room level accuracy efficiently.

2.2.2 **Technologies**

There are various technologies used in wireless positioning. Most of them are radio signal, which comes with a specific wave length and communication protocol. Technologies like GPS or Bluetooth iBeacon are designed for the positioning purpose from the beginning. Meanwhile, technologies like Cellular Network or Wi-Fi are designed for the communication purpose. In general, the later type of technologies comes up with more challenges to build a good positioning system. However, the effort to build a system based on those communication purpose ones could be justified as a way to exploit the available infrastructure. Beside radio-based technologies, sound-based technologies are also capable for positioning purpose. In term of performance, the GPS-based technology is a dominant approach for outdoor positioning. It provides a global coverage with meter-level accuracy. Nevertheless, there is an obvious significant decrease of accuracy for the indoor positioning with GPS-based systems. The other approaches could get to a more reliable performance for the indoor environment. Several popular technologies are discussed in the next section.

2.2.2.1 *GPS*

GPS-based approach is a prominent solution for localization purpose. The most important characteristics is its world-wide coverage. Basically, the underlying mathematical model is the distance based on Equation (2) with TOA approach. In GPS system, the base stations are satellites. They orbit the Earth and send signals to GPS receivers, which act as tracked devices. Both satellites and GPS receivers are time-synchronized precisely. Therefore, when the signal with timestamp is received at the GPS receivers, the travel time can be calculated and the distance is calculated by multiplying the time with the signal's speed.

Let $[x_i, y_i, z_i, t_i]$ is the received information of the GPS receiver from the GPS satellite i^{th}. The first three values are the satellite's coordinates and t_i is the time stamp when the message is sent. At the receiver, the time of message reception is t_r. Although the clocks between each endpoint are synchronized, GPS introduces the clock bias b at the GPS receiver, which results in the true reception time at the receiver is $t_r = t_r + b$. Then, the distance d_i between the satellite i^{th} and the receiver is:

$$d_i = (t_r + b - t_i)V \qquad (7)$$

where V is the speed of signal propagation, which is generally known.

The receiver is at the surface of the sphere which has the center at $[x_i, y_i, z_i]$ and the radius t_i. The relationship is presented in the form of a sphere equation:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = ([t_r + b - t_i]V)^2 \qquad (8)$$

There are four unknown parameters which are x, y, z and b in Equation (8). To solve the equations, at least 4 satellites are required to be visible by the GPS receiver. If the number of satellites is less than 4, it could result in an inaccurate position [Abel and Chaffee, 1991].

There are a number of issues with the geometry approach in GPS based localization. Firstly, the distances are calculated based on the speed of signal. Therefore, high precision in the synchronization step, message sending and receiving time at the both ends are required. In addition, considering the long way of travelling distance, signal inference in the Earth surface atmosphere can affect to the speed of signal in an unexpected way. Multipath and missing

sky visibility are other sources of error, especially in the indoor environment. Because of those errors, additional methods are employed to obtain a more stable performance, such as differential GPS [Moore, 2002] or pseudo-satellites [Wang, 2002]. Assisted GPS (AGNSS, AGPS) is a system which integrates GPS with other wireless communication means [Van Diggelen, 2009]. AGPS uses external data, which is mostly from cellular communication, to enhance the system's performance in poor satellite signal condition. While AGPS has an acceptable accuracy for outdoor environment, its performance in indoor environment is not reliable due to the obstruction of building. The High Sensitive GPS method for indoor environment employs a number of techniques for a more accuracy distance estimation. These are improving low signal to noise ratio and reducing the computing time of satellite messages. With various indoor environment, the study in Zhang et al. [2010] concludes that accuracy from 20m to 60m can be achieved. It is noted that GPS-based positioning has a long respond time and should be combined with other types for a finer tracking.

Besides GPS-based systems, there are other systems taking the same approach [Fernandez-Prades et al., 2011]. The most notable systems include the Russian Global Navigation Satellite System, Galileo Positioning System and China's BeiDou Nagivation Satellite. However, compared to the GPS, which is developed from the 1960s, all of the three systems are relatively new and still under development.

2.2.2.2 *Mobile Cellular Networks*

Cellular networks are the basic technologies for communication of mobile phones in many countries. The technologies for cellular networks are Global System for Mobile communication (GSM) or Code Division Multiple Access (CDMA). The cellular communication uses signal frequency at around 300 MHz to near 3 GHz. A mobile device is tracked based on the signal exchange between the mobile device and the cellular towers. The range between mobile device and tower can reach up to 35km. Because of its great range, the propagation signal is highly depending on the environment's condition such as weather and the building. Like GPS-based technology, cellular network positioning is also available in commercial positioning systems, for example Google Mobile Maps [Zandbergen, 2009].

Three main approaches for GSM-based localization are RSSI fingerprinting, distance based and angle based. RSSI fingerprinting method relies on the variation of GSM signal when the phone's position is changed. In Varshavsky et al. [2007], the authors show that the GSM signal is less affected by the condition of the outside environment. With a sufficiently-dense enough RSSI samples position, they conclude that using RSSI fingerprinting can reach the accuracy of 2m. The experiment is carried out in a multi-floor building and with 29 GSM channels. However, Popleteev [2011] reports that GSM signal can be less reliable due to the effect of weather, and construction. With distance based approach, it is difficult to find a proper model for GSM signal propagation taking into account its great travelled distance. Ikegami et al. [1984] propose a model which includes building heights, antenna heights and street widths. An accuracy of 100m is feasible with this type of approach according to Gustafsson and Gunnarsson [2005]. Similar accuracy can be achieved with AOA-based approach [Shen and Oda, 2010]. Driusso et al. [2016] employs the TOA approach with the new mobile communication standard, Long Term Evolution (LTE). It is able to achieve a root mean square distance error of 9.61m. However, the proposed system requires to have access to multiple base stations with different service providers. It is thus a limitation to daily usage.

2.2.2.3 *Wireless Local Area Networks*

Wireless local area network (WLAN) is a popular method for connecting two or more devices. WLANs are based on IEEE 802.11 standard and are often referred with the name of Wi-Fi. WLAN devices use the frequency of 2.4 GHz and 5 GHz bands for communicating. The underlying hardware includes a WLAN card which can send and received WLAN data. An infrastructure of a WLAN system consists of base stations, namely Wi-Fi access points, which are normally deployed within a building. A device supporting WLAN technology can scan the deployed access points and connect to the access points within a given range. The typical range between the station and device is 30-50m. However, with the presence of obstacles such as wall, the ranges could be reduced significantly. The advantage of WLAN-based for positioning purpose is the popularity of WLAN infrastructure in indoor environment. With the present of Internet, the WLAN technology is preferred in daily use because of its simple in installation and operation. Moreover, the mobile devices such as mobile phones, tablets and laptops, supports the capability to read the base station MAC address and the received signal strength.

These two parameters play an important role for the building of a positioning system. In general, it is feasible to implement a WLAN-based positioning system without additional hardware installation.

The device's reported RSS or RSSI values of WLAN-based communication have a great correlation with the distance between the Wi-Fi access point and the device. In the geometry based approach, the signal strength value is used to calculate the distance by some propagation models. Popular models include the LDPL model, given in the Equation (3), and its variants. For example, the work in Bose and Foh [2007] uses the Hata-Okumara model for calculating the distance. Compared to the standard LDPL model, the antenna gain parameters and signal wavelength are added [Hata, 1980]. Both cases of line-of-sight and non-line-of-sight are tested, and resulted in the mean errors of 2.3m and 2.9m, respectively. Several parameters for controlling the signal propagation behaviors are introduced with the model. To find the specific value of those parameters, the authors use a genetic algorithm which minimizes the cost function on collected Wi-Fi data. Apart from the usage of RSS and RSSI values, it is able to use TOA approach with WLAN technology. Ali and Omar [2005] propose the use of super resolution matrix pencil algorithm to estimate TOA. The matrix is built based on the frequency domain transfer function of radio signal propagation. This approach, however, is only applicable for a direct ray signal.

With the pervasive deployment of WLAN infrastructure in indoor environment, the fingerprinting based approach is likely the most benefit one [He and Chan, 2016]. Roughly speaking, the approach could use the installed access point stations as features to increase its discriminative power. RADAR [Bahl and Padmanabhan, 2000] is the earliest system using fingerprinting approach for localization purpose. The system performs a site surveys to build a Wi-Fi signal map of location database. Then, the location is determined by matching with the most matched signal strength in the database. The distance error is reported in the range of 2-3 meters in average. More calibration efforts can result in a robust system. An improvement of RADAR is introduced in Horus system [Youssef and Agrawala, 2005]. It uses probabilistic method to compute the most probable location by exploiting the structure of RSSI vector at a given point is used. The reported accuracy is 1.4 meters in 90% of the time. The study is

taken in a 68.2m by 25.9 m area with around 180 positions. The positions are randomly selected and distributed along the corridors and inside the rooms. Each position is covered by 6 access points in average and has 100 samples. However, the result is only reported in the scenario of static object positioning. Another study in King et al. [2006] reports an average distance of 1.6m. The map is created with a grid of 1m distance. Each point has about 110 RSSI measurements and calibration at 8 directions. In Dinh-Van et al. [2017], Wi-Fi fingerprinting-based approaches are employed for tracking indoor intelligent vehicle. The mean distance error is reported at 2.25m over 1000m moving distance, which is capable of replacing GPS for indoor environment. Other popular systems in this approach are Place Lab [Cheng et al., 2005] and Active Campus [Griswold et al., 2004]. There are some commercial systems for localization purpose. Gallagher et al. [2009] report the performance of those systems, including the Skyhook and Ekahau. The Skyhook system collects the RSSI of WLAN access points by car and builds a global database. They achieve the accuracy of 10m to 20m in outdoor environments and 30m to 50m in indoor environments The Ekahau Real-Time Location System uses the approach of RSSI fingerprinting to track at 1m to 3m accuracy. The infrastructure of the system and building RSSI map is generated by the system providers. The average accuracy in indoor environments is reported at 7m.

In general, the fingerprinting could handle the none-line-of-sight issue to some extent. However, there are several disadvantages. Firstly, the database building phase is a costly process and difficult to apply in a large scaled area. To overcome these disadvantages, robots can be used to collect the signal map automatically. Kothari et al [2012] employ a robot with laser range finder for building the RSSI database. In the test path with 120m length, the database results in a mean error of 15m with standard deviation of 10m. Park et al. [2010] take the approach of an organic system. Instead of using experts for building the signal map, the users are prompted their positions to construct the signal strength map throughout the time. The system then updates, refines and expands the map. This approach can reach 5m mean accuracy after 9 days of testing. Fully automatic approaches based on SLAM are introduced in Ferris et al. [2007]; Huang et al. [2011]. The SLAM approaches use the results from Wi-Fi scanning process as the sensor data for observing the environment.

Secondly, the RSS/RSSI values at the mobile device are highly affected by the noisy propagation of Wi-Fi radio signal, especially in indoor environments. The unstable characteristics of Wi-Fi signal have been mentioned in works of Kaemarungsi and Krishnamurthy [2004] or Haeberlen et al. [2004]. In these works, the authors present that giving the same Wi-Fi access points configuration, there are still variances of Wi-Fi patterns at the same places. In Chen et al. [2005], the authors suggest that the dynamic of environment has a large contribution the presence of those noise. For example, the moving of the indoor furniture, changes of access points or open/closed doors could make the RSS fluctuate. Other factors, which are able to contribute to the noisy RSS/RSSI values, are the device's orientation [Xiang et al., 2004] or the influence of user's body [King et al., 2006]. Apart from those environment factors, the device diversity also has a great effect on the RSS/RSSI values. In general, the RSS/RSSI values depend on the specific hardware of the receivers. When the mobile device, which is used in the database collection process, is differ to the tested one, the RSS/RSSI signal matching process likely results with unexpected errors. There are several works for stabilizing the RSS/RSSI fluctuation across different mobile devices. Wang et al. [2013] apply mean normalization to eliminating the RSS difference between devices. It is also able to use mapping function for RSS values. The mapping function transforms the RSS values from one device to another. In Bernardos et al. [2010], the authors rely on the assumption that the transform function is linear. They then proposed a least mean squares optimization to find the transform's parameter. Tsui et al. [2009] employ several approaches, including online linear regression algorithm, expectation maximization and neural network. The approaches, however, is needed additional data for calibrating between difference devices. There are also calibration free methods, such as Dong et al. [2009]; Kjaergaard and Munk [2008]. Although all the proposed calibrating methods are reported with some degree of success, the device diversity is still a practical challenge for Wi-Fi fingerprinting, especially in a large scale application [He and Chan, 2016].

A systematic study in Liu et al. [2012] shows that a reasonable accuracy of around 3 – 4m can be achieved. However, the larger errors (e.g., 6 − 8m) are always appeared. This degree of

error is unacceptable in many scenarios and makes the Wi-Fi-based approach become unreliable. The authors conclude these errors are mainly from an intrinsic phenomenon of the radio signal propagation and fundamental limit of Wi-Fi methods.

2.2.2.4 *Bluetooth*

The Bluetooth protocol is a wireless technology standard for data transferring. The Bluetooth standard is specified in IEEE 802.15. There are a number of major versions of Bluetooth ranging from 1.0 to 4.0. It operates in the band of 2.4 to 2.485 GHz. Its properties are similar to WLAN but has a shorter communicated range. The preferred range of Bluetooth is between 5m to 10m. The propagation of Bluetooth signal is also affected by non-line of sight and other indoor conditions. Compared to the popular WLAN technology, the Bluetooth's advantages are low-power consumption, low price and high portability. For the usage in positioning task, the standard Bluetooth protocol have some limitations, including a long-period of inquiry and a hardware/software dependency, as shown by Madhavapeddy and Tse [2005]. Besides that, missed detections can happen frequently when more devices participate in the network [Pei et al. 2010].

There are some works for indoor localization using Bluetooth technology. The Bluetooth RSSI can be used for find location with the average error of 2m within a small room [Bandara et al., 2004]. Pei et al. [2010] uses finger printing to reach an accuracy of 5.1m error in average in dynamic indoor positioning scenario. The high error value is explained by the low number of Bluetooth inquiry packets. The reported inquiry rate is around 20 samples for 2 minutes. To overcome this drawback with Bluetooth technology, Hay and Harle [2009] propose a method for inferring the distance from the Bluetooth pair-connection. The moving Bluetooth device is connected to the beacon stations within the environment. The pair-connection is then used to query the RSSI, Link Quality and echo response time. These values then can be used for positioning purpose. It is reported that the connection can be established within 1.28 seconds, in comparison to 10.24 seconds of a Bluetooth inquiry-based searching. In the work of Bargh and de Groote [2008], the author use inquiry respond rate from the inquiry process to locate the device. After obtaining the rate from the device, the device's position is computed by a relative entropy approach. With the room level, this approach has the accuracy of

98%. However, this approach requires the device to be stationary in a long period of time and cannot apply for real time scenarios.

More recently, Bluetooth Low Energy (BLE) is introduced [Gomez et al., 2012]. The technology allows to create cheap and small devices, which act as beacons. A BLE beacon has a low-power consumption and can be powered by coin cell battery for several years. In a positioning system, its role can be viewed as the base station. Therefore, positioning techniques like geometry-based and fingerprinting are feasible with BLE technology. Faragher and Harle [2014] study the performance of BLE system for indoor positioning. In a test-bed with sufficient density of beacons, the accuracy of tracking mobile devices is around 2.6m in 95% of time. In Zhuang et al. [2016], the authors propose a combination of polynomial regression model and fingerprinting for smartphone-based indoor localization with BLE devices. With a dense beacons setup, the system could reach 2.56m in 90% of time.

2.2.2.5 *Radio Frequency Identification*

Radio Frequency Identification (RFID) uses the radio waves to communicate between a RFID reader and a RFID tag. The communicated frequencies are various and divided into small categories. These categories are Low Frequencies (at 30 kHz to 500 kHz), High Frequencies (at 3 MHz to 30 MHz), Ultra High Frequencies (at 433 MHz and 868 MHz to 930 MHz) and microwave (at 2.4 GHz to 2.5 GHz and 5.8 GHz). The communication ranges between the reader and the tag is from several meters to around 30m, depending on the specific equipment and environment. The radio wave in a RFID system also suffers the effect of signal propagation through obstacles in the indoor environment as other types of wireless based system. The signal strength loss issue of higher frequencies is more serious than the lower ones.

In the market, there are two types of tag: active tags and passive tags. For the localization purpose, the active tags can work similarly as the other radio wave technology like WLAN or Bluetooth. The active tags carry their own battery power and can be detected within a long range. The exchange information is not only the tag's ID but also the RSSI or the time stamp. Hence, the active tags provide more flexible approaches for localization than the passive types. Their disadvantages are large size and high cost. The passive tags are generally smaller

because they do not require an additional battery. They operate by the energy transmitted from the reader.

The LANDMARC system proposed by Ni et al. [2004] employs the active RFID tags for indoor localization. The RFID readers are placed at specific locations and the RFID tag is carried for locating purpose. The detection ranges between the reader and the tag is around 40m and the exchanged data are signal strength values. After collecting the signal strength at a specific location, the system uses a K-nearest-neighbors approach to find the tag's location. With 4 fixed readers, the reported accuracy is 1m of error distance with 50% percent and the maximum error distances is less than 2m. However, these experiments are carried out with static objects. Seco et al. [2010] use a fingerprinting approach with the collected RSS values from active RFID tags. The active tags are fixed and continuously transmit their codes every time interval. Then, the readers are used to decode the messages for obtaining the signal strength values. Gaussian processes are used to find the reader's location. With 71 tags in an area of 1600m^2, the reported accuracy is 1.70m in average and 3.08m within 90 % of error.

In comparison to the active tags, passive tags operate without needing battery and receive the power directly from radio waves of the RFID reader, then transmit back its identification code to the scanner. The passive tag's advantage is its small size, low price and can be embedded fully into the environment. Using passive tags, one can design a proximity based localization, which is the Cell of Origin. Chon et al. [2004] proposes an approach of vehicle localization by using RFID technology. The tags are placed along the road, under the ground. Then, the reader is carried by a vehicle and scans the placed tags. The position of the vehicle is returned under a proximity based result. Hahnel et al. [2004] uses passive RFID tags along the corridor environment for robot localization. The tags have a detection range of approximately 6m. The accuracy of system consisting only of RFID technology is around 2m. A better accuracy can be achieved with a dense network. Park and Hashimoto [2009] places the tags on the floor in a grid with spacing of 34cm (Figure 10). Using this setup, a moving robot at speed of 12.2cm/s can be located around 10cm accuracy in average.

Figure 10. A grid of passive RFID tag for localization purpose [Park and Hashimoto, 2009]

2.2.2.6 *Sound-based*

Basically, sound signal has the capable of the radio signal in term of designing a positioning system. In additional to that, the sound velocity has a low velocity. Techniques like TOA thus suffer less from the distance measurement errors. Harter et al. [2002] use a small ultrasonic tracker, namely Bat. The Bat emits ultrasound signals to the receivers. The receivers then calculate the distance based on the TOA method. The proposed system is able to locate the Bat device within 9cm in 95% of time. It is also able to detect the orientation of the tracked device with a sufficiently dense receiver. The Cricket system in Priyantha et al. [2000] uses ultrasonic receivers with the radio frequency beacons to calculate the device's orientation. Hazas and Hopper [2006] use broadband ultrasound technology to create a fine-grained location sensing. Compared to the narrowband technologies, the broadband technologies provide a greater bandwidth and a more robust performance in the presence of noise. In a small office environment, distance errors of less than 2.3cm for 95% of time can be achieved.

2.2.2.7 *Others*

Other popular technologies include Ultra-Wide Band (UWB) and magnetic. UWB transmits signals over a large bandwidth (>500MHz) with ultra-short pulses [Gezici et al., 2005]. The wide bandwidth improves the chance of going through obstacles such as walls. The short pulses make the noise from multipath propagation easy to filter. Therefore, it comes with several advantages for the indoor positioning. UWB technology is applicable with most of the positioning technique include TOA/TDOA, AOA, signal strength based and proximity based.

Ubisense system [Cadman, 2003] is reported to reach less than 1m in accuracy. Though, UWB approach is usually considered as a commercial product for indoor localization because of its high cost.

The magnetic sensor is able to be used with the fingerprinting approach for positioning purpose [Gozick et al., 2011]. However, it is usually combined with other inertial sensors to create Step-and-Heading positioning systems.

2.3 Inertial-Sensors-Based Methods

The inertial sensors include accelerometer, gyroscope and magnetic sensors. They are usually combined into an inertial measurement unit (IMU). Figure 11a illustrates an Xsens MTi-G type sensor. The accelerometer and gyroscope sensors measure the device's acceleration and angular rate. Both values are relative to the device frame. The magnetic sensor measures the magnetic field. It is also referred as the compass component of the IMU. The magnetic vector is expected to point to the earth North magnetic pole in a noisy-free magnetic field. There are also other types of sensors that be integrated into IMU. For example, the Xsens MTi-G sensor includes a barometer and GPS. In this section, we only discuss the user's tracking based on accelerometer, gyroscope and magnetic sensors.

(a) Xsens MTi-G IMU (b) IMU with foot mounted position

Figure 11. Using IMU for PDR tracking [Feliz Alonso et al., 2009]

User tracking system by the IMU is known as Pedestrian Dead Reckoning (PDR) system. The user is required to carry the IMU. The data from inertial sensors then can be used to find the trajectory of the user. Figure 11b presents a foot mounted IMU for the PDR system. Because

the user's moving pattern is highly correlated to the swing of the foot, foot is a preferred mounting position for the IMU setup. The approach is proposed in several systems such as Foxlin [2005], Godha and Lachapelle [2008], Feliz Alonso et al. [2009] and Bird and Arden [2011]. In case of tracking through the smartphone's inertial sensors, the phone could be handled or put in the pocket [Steinhoff and Schiele, 2010].

The approaches for PDR can be divided into two categories, Inertial Navigation System (INS) and Step-Heading-System (SHS) [Harle, 2013]. The INS attempts to track the IMU's movement in 3D, which includes the movement distance and a 3D-direction vector at each step. Normally, the IMU's movement is not identical to the user's movement in terms of tracking purpose. For example, in foot mounted setup, the IMU's movement would reflect the foot's movement. A further step is needed to convert the foot's movement to the real user's moving path. The SHS takes a more specific approach which attempts to track the distance and the user's heading in 2D. Figure 12 expresses a comparison between the two approaches. For PDR applications, SHS approaches are more preferred ones thanks to its simplicity.

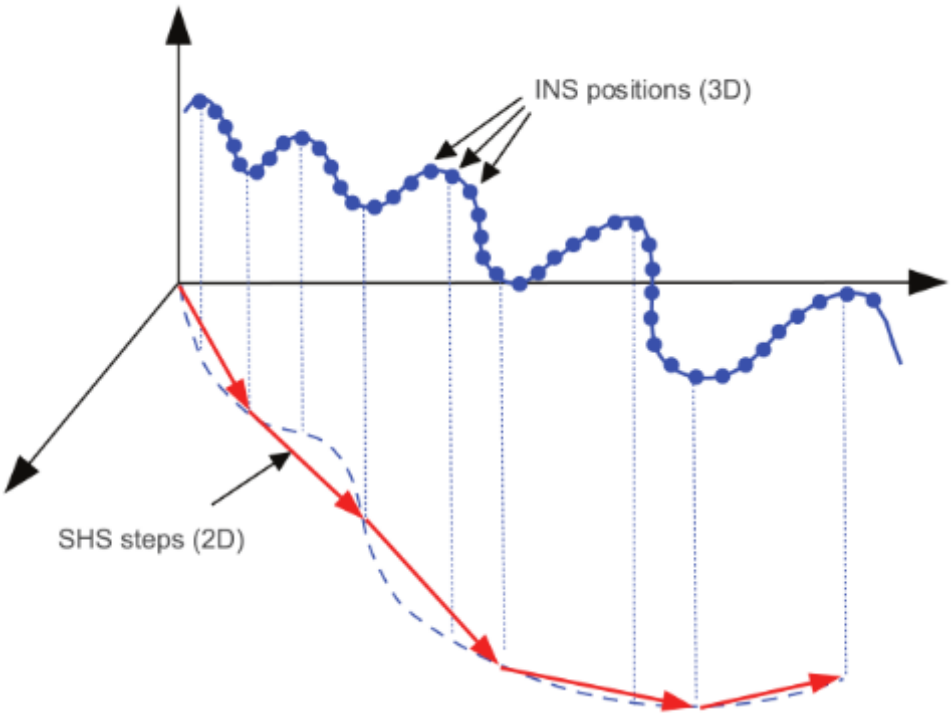


Figure 12. Illustration for INS and SHS approach [Harle, 2013]

IMU units are usually built on Micro Electro-Mechanical Systems (MEMS) technology. These MEMS sensors are lightweight, inexpensive and portable. However, the measured data contains high percentage of noise. The noise then results in the errors of derived velocity and direction values. Unlike the absolute positioning approaches as in wireless-based systems, the Dead Reckoning could make the errors accumulate overtime. This effect is usually known as the drifting errors. For example, a report in Woodman [2007] shows that integration of MEMS sensors could only be stable under around 1 minutes of tracking. Computing movement distance and heading of SHS should take into account the effect of noise overtime.

2.3.1 **Movement Distance**

In SHS approach, the movement distance can be computed from the number of user's step. The steps are usually represented as local maxima in the accelerometer sensor data. Therefore, peak detection is quite a popular technique for the counting purpose [Henk Muller, 2003; Dippold, 2006; Wang et al., 2010]. In addition, a threshold on the accelerometer magnitude could be sufficient [Krach and Roberston, 2008]. It is also possible to rely on the gyroscope data [Feliz Alonso et al., 2009] and magnetometer data [Jimenez et al., 2009]. In terms of accuracy, the system performance depends largely on the user's walking pattern. In Rai et al. [2012], the authors propose a Normalized Auto-correlation based Step Counting algorithm which can reach a false positive rate of 0% and false negative rate of 0.6%. The experiments were carried out on smartphone accelerometer's sensor. Recently, Edel and Köppe [2015] use recurrent neural network based approach for counting the steps. The network is trained with 10 hours of data collected by an IMU device. Several device's position are included in the data such as foot mounted, in pocket and handle. To be able to calculate the velocity from the number of steps, it is necessary to know the step's length. Most of the works in the field assume the length to be constant. However, in practice, the step length could vary in a large range because of changing in the walking pattern. In Qian et al. [2015], the authors introduce a set of conditions for detecting such patterns.

2.3.2 **Heading**

The data from IMU provide different ways for calculating the heading. The magnetic sensor or compass reading is the direction to the North magnetic sensors which could be used to calculate the device's heading. Moreover, the accelerometer values and compass values could be combined into a rotation matrix which represents the device's attitude. Integrating the gyroscope data are another way to calculate the heading. However, the IMU data reading would subject to noise. Magnetic data also subject to magnetic perturbation in the indoor environment. For example, works in Afzal et al. [2011] and Rai et al. [2012] illustrate the effect of the indoor environment to the compass direction.

In order to make the heading calculating more stable, popular approaches are to fuse different types of data in IMU. Three popular approaches are Complimentary Filter, Kalman Filter and Madgwick Filter. Complimentary Filter [Mahony et al., 2005] splits the direction into two components. The first component, which has a low update rate, is the heading from accelerometer and compass. The second component, which has a high update rate, is the heading from gyroscope integration calculation. The Kalman Filter approach uses covariance matrix to model the noise from IMU data. Choukroun et al. [2006] propose a linear Kalman Filter for heading estimation. The time update comes from gyroscope vector and the measurement update comes from the accelerometer and magnetic vectors. An extended Kalman Filter for PDR context is introduced by Renaudin and Combettes [2014]. Madgwick Filter [Madgwick, 2010] rewrites the heading calculation errors from accelerometer, magnetic and gyroscope as a minimization problem. He then uses a gradient descent approach to find the solution.

In terms of performance, it is difficult to measure the accuracy of heading calculation in general. For example, in case of Madgwick Filter, the root mean squared error is reported less than 1 degree in several testing configurations [Madgwick, 2010]. However, a specific calibration process is required to reach the reported performance. More general comparison between different attitude estimation methods are described in Mourcou et al. [2015], Michel et al. [2015] and Yadav and Bleakley [2014].

By combining moving distance output and heading output, it is able to track the user using IMU data. The distance errors, however, depends on the specific testing setup. For example,

in Beauregard et al. [2008], the tracking errors by using IMU alone is reported to be around 7.74m while it is reduced to 2.56m if there is a building plan as additional information.

2.4 Smartphone-based Indoor Positioning

Nowadays, smartphone has become a popular personal device in daily use. There are various positioning technologies from the above sub-sections which can be developed on the smart phone. Wi-Fi and inertial sensors have the most attention in the field. Bluetooth with recently BLE technology, GPS, cellular-based, camera-based, sound-based and other types of smartphone's sensors are also capable for using in localization purpose. In general, the specific approaches for the technologies on smartphone positioning do not vary much when they are developed in other platforms. It is important to note that the GPS-based positioning is only feasible in the outdoor environment [Link et al., 2011]. Table 2 gives some popular approaches and its accuracy for indoor positioning purpose with smartphones.

Among the available technologies on smartphones, Wi-Fi and inertial sensors are the most interesting for localization purpose in indoor environment. With Wi-Fi technology, the popular methods such as geometry or fingerprinting based are flexible enough for deploying in real world scenarios. The accuracy of Wi-Fi based systems is expected to be around 5-10m. Dao et al. [2013] employ the propagation model to build the distribution probability of the user. This approach is able to exploit the RSS values from multiple access points. A grid optimization process is used for speeding up the computation. For fingerprinting approach, the system performance depends on the size of the Wi-Fi database. In a large database setup, it could take some effort to collect sufficient training data. The work of Mathisen et al. [2016] requires three days for collecting data in a 160,000 m^2 building. The best accuracy is reported at 6.09m. Similar performance could be seen in Moreira et al. [2015]. For reaching the distance errors of 6.2m in average, they train several KNN-based models on a total of around 20,000 Wi-Fi fingerprinting samples. To reduce the cost of building database, a robot can be used for an autonomous collecting process as in Kothari et al. [2012].

Table 2. Several works on indoor positioning with smartphone

Author	Technology	Approach	Metric	Accuracy	Note
Zandbergen [2009]	Cellular-based	Cell identification	Median	>500m	-
Kothari et al. [2012]	Wi-Fi	Fingerprinting	Mean	10m	Wi-Fi data collected by robot
Dao et al. [2013]	Wi-Fi	Geometry	Mean	3-4m	-
Moreira et al. [2015]	Wi-Fi	Fingerprinting	Mean	6.2m	Large data size
Link et al. [2011]	Inertial Sensors	Step Counting + Phone's Compass	Mean	1.6m	Map matching
Zhou et al. [2014]	Inertial Sensors	Step Counting + Calibrating Heading	-	1.3m	Work on a specific gyroscope sensor
Faragher and Harle [2014]	BLE	Fingerprinting	Percentile 95%	2.6m	-
Luca and Alberto [2016]	BLE	Fingerprinting	Mean	1.7m	Dense sample size
Juri et al. [2016]	Image-based	3D map matching	Mean	0.6m	-
Xu et al. [2015]	Light sensors	Proximity Based	Precision	95%	Peak detection for lights on ceiling
Filonenko et al. [2010]	Ultrasound	Geometric based	-	-	-

Inertial-sensors-based approach could reach a better accuracy than Wi-Fi in general. Works of Link et al. [2011], Zhou et al. [2014] and Qian et al. [2015] have reported a distance error less than 2m. In order to control the natural drifting errors of the SHS, specific methods for calibrating the output are implemented. For example, Link et al. [2011] and Qian et al. [2015] employ map-matching method. The output path is adjusted to avoid crossing the wall in the environment. Zhou et al. [2014] rely on a specific calibrating method for reducing the heading errors. This method addresses the measurement error of the gyroscope sensor. The SHS tracking with different positions of smartphone such as in pocket or in hand could be done via a motion axis detection step, suggested by Steinhoff and Schiele [2010]. The unreliable output of smartphone's sensors also affects the SHS tracking approach. In Afzal et al. [2011],

the authors study the effect of magnetic perturbations in different indoor environments. The effects of linear acceleration and magnetic deviation to several attitude estimation approaches are discussed by Michel et al. [2015].

BLE technology offers a promising solution for indoor environment with smartphones. The BLE beacons have several advantages such as low cost and low power consumption. The BLE standard is also widely supported on smartphones. With the beacons installed in the environment, BLE provides the same infrastructure as Wi-Fi technology. In terms of accuracy, fingerprinting approach on BLE technology could reach 2m accuracy. The work in Faragher and Harle [2014] also suggests that BLE fingerprinting outperforms Wi-Fi fingerprinting in the same environment.

Other types of available technology on smartphone are still under development. The cellular-based approach is highly affected by indoor environment. Its results are unreliable. The camera based approach could achieve great accuracy. In Juri et al. [2016], the authors report an average distance error of 0.6m for image-based approach. There are several disadvantages. Firstly, it is required to build a 3D point cloud for the surrounding environment. Moreover, the image is captured by using the phone's camera, which makes the approach inconvenient for tracking over longtime. The *Perspective-n-Point* in the later matching process is also a heavy computing process. In the proposed approach, several improvements to decrease the computational time were proposed. The smartphone's light sensor can be used for a proximity-based positioning approach [Xu et al., 2015]. The light in the ceiling can be detected by the peak illumination of the sensors. By discovering successive lights, the walking distance can be calculated. The proposed approach, however, needs the phone to be handled facing up. Mobile phone speakers can also be used for positioning purpose with TOA methods as described in Filonenko et al. [2010].

Because a smartphone can have several feasible technologies for localization purpose, it is possible to combine several of them. For example, Dao et al. [2014] use a weighted probabilistic approach from several types of sensor such as GPS, Wi-Fi, RFID and inertial sensors. Instead of an absolute positioning, the output of each type of technology is modeled by a dis-

tribution functions. Moreover, the technologies are assigned with a precision value. The precision value is then used to combine different distribution functions into one. Deng et al. [2016] use two different Extended Kalman Filters to combine WiFi, PDR and landmarks for improving positioning results. The proposed approach achieved an error of 0.71m with standard deviation of 0.44m. Other works on combining multiple data sources for improving smartphone positioning could be found in Rai et al. [2012], Chen et al. [2014], Correa et al. [2016] and Wang et al. [2016].

2.5 Summary

In this chapter, popular technology and techniques for positioning have been mentioned. Each comes with advantages and disadvantages. While the GPS-based technologies have great success for positioning tasks in outdoor environment, it does not exist such similar approach in indoor environments. Optical-based methods come with low errors but have a high computational cost. The methods also have some restrictions on user localization tasks. Wireless-based technologies provide a wide range of choices for building positioning system. Inertial sensors also have comparative performances in the field. With the presence of camera, wireless sensors and inertial sensors in smartphones nowadays, smartphone-based positioning has become a promising direction for user's localization purpose.

Chapter 3 Using Wi-Fi Fingerprinting for Smartphone Indoor Positioning

3.1 Introduction

In a GPS-denied environment such as indoor environments, Wi-Fi based approaches are popular alternative choices for user positioning. It can benefit from the advantage of the underlying WLAN infrastructure, which is the existence of many Wi-Fi access points. Moreover, with the popularity of smartphones nowadays, the need of carrying a signal-receiving device can be easily solved. The smartphone could provide basic information of surrounding access points and their RSS/RSSI measurements approximately. Based on the values, several localization techniques can be developed such as geometry-based approaches or fingerprinting-based approaches. For a large public area, fingerprinting-based approaches are often more preferred thanks to the presence of a large number of Wi-Fi access points.

There are usually two separated phases for building a fingerprinting-based localization system. The first phase is collecting the Wi-Fi signal around the given area. Then, a learning model is trained using the collected data. The model could be considered as a mapping from Wi-Fi signals feature space to the real world positions. In the second phase, the trained model is used to localize the phone based on the smartphone's Wi-Fi scanned signals. In order to build a stable localization system, the data collection in the first phase has an important role. Because of the high variances of radio signal propagation from access points to the phone, a large amount of data is needed for modeling the relationship between the signals and positions. Most of the works in the field would assume that there are enough data to establish such relationship. For example, in [King et al. 2006], the authors use a point grid of 1m-wide cells for collecting the RSSI signals. Each point requires over 100 RSSI measurements. Bahl and Padmanabhan [2000] also suggest that more calibration effort is needed to improve the Wi-

Fi fingerprinting position accuracy. However, the need of a dense Wi-Fi signal map would create difficulties for deploying the fingerprinting system in a large area.

When the available training data is limited, novel Wi-Fi fingerprinting methods are able to produce the mapping from the RSS/RSSI signal to position. As a trade-off, the performance of the positioning system could be affected. The output positions likely have high distance error and variance. In our works, we want to explore different approaches of Wi-Fi fingerprinting methods for dealing with such types of training data. Based on the performance of these approaches, several ensemble strategies are proposed to improve the overall positioning performance. By combining a number of uncorrelated models, it is expected to reduce the errors in cases of lacking training data.

Ensemble methods for improving learning models are a well-known approach in the field of machine learning. For Wi-Fi fingerprinting-based positioning, the works of Torres-Sospedra et al. [2016] propose an ensemble framework based on KNN models. The authors vary a set of parameters and then combine all of the generated models to have a more stable performance. These parameters include the number K of considered neighbors, the distance/similarity measure of signal feature vector, the data representation and the method of filtering out weak signals. However, because all the models are derived from KNN model, their learning capability could be highly correlated. Thus, it leads to an overfitting model when the ensemble step is carried out, especially in the context of limited training data. We share the similar idea of ensemble models but in a different way. Mostly, we aim to reduce the dependent on KNN model by varying the Wi-Fi signal based features and the learning models.

In our approach, the Wi-Fi fingerprinting problem is first transformed into a standard learning problem. The learning problem includes a set of feature vectors and its desired labels. The label in this case is the coordinates of each sample. Two other features sets, which are filter-based features and hyperbolic-based features, are derived directly from the default raw signal features. The two additional features are reported with good performance for building Wi-Fi fingerprinting models. Three different learning models, which are KNN, Random Forest and Extreme Gradient Boost, are then used in the learning phase. The KNN model is a well-known model for working with Wi-Fi fingerprinting data. Both Random Forest and Extreme Gradient

Boost are tree based methods which have good performance against various types of data. The two additional learning models are added to make the following ensemble step less rely on the KNN method. The learning targets are also used selected between the regression and the classification objective functions.

All the proposed methods are tested against a published dataset, which is used in the competition data of the IPIN 2016 Conference with offsite track (track 3) [Torres-Sospedra et al. 2017]. As described in the work, the database is collected in a similar way to daily phone usage. The setup only requires an ordinary user to carry a phone along some predefined paths. It did not exist a specific set of sample points as in a standard Wi-Fi fingerprinting database in building the signal map. Other issues of Wi-Fi fingerprinting database are also included in the dataset such as out-of-training samples and device diversity.

Our experiment results on the published dataset shows that the proposed approaches yield significant performance. For detecting the user's floor, the accuracy can reach 0.93 of accuracy on the testing data. For computing the user's position, standard learning models could reach around 6m in average distance error. By combining these models, the mean distance error can be reduced to around 5.12m.

3.2 Literature Review

The smartphone's API generally supports the scanning available Wi-Fi access points within the environment. The received data includes the access point identifiers (SSID) and the RSS or RSSI information. Both the RSS and RSSI provide values that can be used as an indicator specifying how strong the signal from the access point is received by the mobile device. Because the difference between RSS and RSSI for localization purpose is not significant, it is able to use the only term RSS for referring the signal strength information. Traditional approaches rely on mathematical models to calculate the distance between device's position and the access point from the RSS value. For example, Dao et al. [2013] use the well-known LDPL model in Equation (3) for building the probabilistic propagation model from a set of access points. For measuring the wall attenuation component in the LDPL equation, the thickness of the walls

and the arrival angles are included. Genetic algorithms are then used to find the optimal values for those parameters. In general, it is difficult to model the complex properties of radio signal propagation in indoor environment.

Fingerprinting approaches rely on statistical models for dealing with the complexity of radio signal propagation. The feature space is the RSS feature vector and the target is the device's position. For the learning model, KNN and its variations are preferred. For example, weighted KNN is used in various works [Ma et al., 2015; Liu et al., 2016; Mathisen et al., 2016; Zhang et al., 2016]. In Retscher and Joksch [2016], the authors provide a comprehensive study about different distance metric to use in learning KNN model. Signal filtering could be added to boost the position accuracy. Moreira et al. [2015] split the training data into smaller subsets. Each subset is characterized by the strongest access points. Torres-Sospedra et al. [2016] introduce several ways to vary the KNN parameters and end up in over 2000 sets of parameters. Approximately, each set results in a different KNN model. By removing bad configurations from over 2000 sets, the ensemble model could reach 6.19m in average distance error.

In general, fingerprinting techniques are widely accepted to get errors in the range of 5m to 7m on mean distance. The results, though, are site-dependent. The work in Mathisen et al. [2016] collects data in 160,000m^2 with 589 installed access points within the experimental area. Six different smartphones are used in the experiments. The mean distance errors are reported in the range of 5.47m to 15.42m, which depends on the specific trips and the tested phones. Two published datasets, Potortì et al. [2015] and Torres-Sospedra et al. [2017] address the issues of comparison between various the Wi-Fi fingerprinting techniques. The first dataset contains nearly 20000 training samples which are collected with more than 25 different devices. The provided data are split into training data, validation data and testing data. It also illustrates the change of Wi-Fi infrastructure where the set of visible access points are different between each subset. Several works in Moreira et al. [2015], Berkvens et al. [2015] and Knauth et al. [2015] have the mean distance error ranging from 6m to 8m. The second one includes 4 different buildings with a limited amount of training data for Wi-Fi fingerprinting. For Wi-Fi fingerprinting-based approach, the overall accuracy is reported at 6.33m mean distance errors [Moreira et al., 2016]. Though, the results of the Wi-Fi fingerprinting models across all the test buildings vary from 4m to 12m.

3.3 Wi-Fi Fingerprinting Features

For building the fingerprinting models, it is necessary that the training Wi-Fi signal data are collected at multiple points in the environment. Based on the collected data, signal characteristics are determined for each specific point. In our approach, we start with the raw features as the signal characteristics. The raw features only include the RSS value of all nearby access points. The raw RSS value is expected to have a stable feature space for learning the target positions. Besides that, as our target is to vary learning spaces, two feature sets, the Filtering-based features [Marques et al. 2012] and the Hyperbolic Location Fingerprinting features [Kjaergaard and Munk, 2008], are added. Generally, the two latter feature sets could be considered as an improvement from the standard raw feature. We add some minor modifications to make the added features applicable with the lack of training Wi-Fi RSS data.

3.3.1 **Raw Features**

From the scanning process, the Wi-Fi access points in the surrounding environment ire usually presented in the form of pair MAC address and RSS value. The MAC address is used as the access point's identifier and the RSS value is the access point's signal quality at the position of the mobile device. Each Wi-Fi data package within a short time window is grouped in a same scanning cycle. The raw features are built from a list of these two values of a scanning cycle. The length of a specific scanning cycle depends on the smartphone's operating system. For example, in our experimenting data from the IPIN 2017 track 3 competition, the scanning times likely vary from 4s to 8s. Practically, one could use a fixed time interval for defining a complete Wi-Fi scan process of smartphones.

In order to create a matrix representation of the raw features, it is necessary to collect all the appeared access points within the collected data. With D access points, namely AP_1, AP_2, ..., AP_D, the feature vector has exactly D dimensions. The AP_1, AP_2, ..., AP_D are the access points appeared in the training data only. If there is any access point, which appears only in test data, it should be completely ignored. Assume at time t, there is a complete scan cycle at point P. Each scan cycle is then used to create a fixed-length feature vector r as follows:

$$r = (r_1, r_2, \ldots, r_D) \quad (9)$$

Each value r_i is assigned with the RSS value of the access point AP_i if AP_i appears in the scan at time t. In the other case, when AP_i does not appear in the scan, the r_i^t value is assigned to a minimum value MIN_{rss}. The MIN_{rss} is selected to be lower than any other scanned RSS values, which are appeared in the data. The selection of MIN_{rss} could affect several types of learning model, such as KNN-based approaches. More specifically, the KNN models use the distance metrics for comparing the similarity between two feature vectors. Therefore, the value of MIN_{rss} can be selected to put more emphasize on the difference between the list of unseen access points from each feature vectors.

3.3.2 **Filtering-based Features**

The filtering-based approach could be considered as a direct improvement from the standard raw features. In the original version, Marques et al. [2012] split the whole feature space is split into sub-regions based. The split criteria is the indices of 2 strongest access points of the raw signal vector r. In the later version [Moreira et al., 2015], the number of considered strongest access points is extended to 3. Both versions have good results when they are combined with KNN-based learning models. However, the two results are tested in large training data set (nearly 10000 samples). The large training samples allow the filtering step be able to split the feature space to sufficient smaller subspaces. With a little training data, its performance is moderate [Torres-Sospedra et al. 2017].

In our context, as there is only a small amount of available data, it is difficult to follow the original approach. Instead, the feature space could be split indirectly by adding features to the standard RSS feature vector. From the starting D dimensions, each corresponding to an appeared access points, an additional D features $s = (s_1, s_2, \ldots, s_D)$ is added to the standard $r = (r_1, r_2, \ldots, r_D)$. The value of s_i is computed by following rule:

$$s_i = \begin{cases} INF_{rss} & \text{if } r_i \text{ is one of two highest values of r,} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

Then, the new feature vector q is a combination of the two vectors, r and s:

$$q = (r_1, r_2, \ldots, r_D, s_1, s_2, \ldots, s_D) \quad (11)$$

The purpose of using INF_{rss} instead of a simple value constant 1 is to let the combining vector q can work with the several types distance metric. By selecting an appropriate value of INF_{rss}, it is able to put more weight to the differences in the s component rather than the differences in r component.

The resulting vector doubles the length of the standard vector. The number of strongest access points could be selected as a tuning parameter. Normally, a value of 2 or 3 is chosen for having an effective splitting of the standard RSS feature spaces. In comparison to the original filter version, there is a minor modification in our implementation version. The authors in [Marques et al. 2012] use an additional threshold for deciding the grouping process. More specifically, if an access point B_i is the strongest access point in two signal characteristics vectors r^a and r^b, the difference between r_i^a and r_i^b should be less than a given threshold. If the condition is not met, r^a and r^b would not be grouped into the same feature vector subspace. The condition is not included in our implementation version. Instead of giving the explicit threshold on the RSS value difference, our modified feature set depends on the later learning model for identifying the difference.

3.3.3 **Hyperbolic Location Fingerprinting Features**

For building the Wi-Fi RSS database, the collecting data process and the later testing phase usually evolve more than one mobile devices. Because the RSS value is highly dependent on the specific device, several methods are proposed to overcome the hardware dependent issue. Those methods including mean normalization, applying linear transformations and using signal calibration free approaches. Roughly speaking, all the three methods rely on the linear relationship of RSS values between different devices. The calibration free approaches could be considered more general approaches as it does not need additional calibrating data for finding the linear transform. The methods encode the transforms' parameters directly into the feature vector. Popular methods include the different of signal strength [Dong et al., 2009] and hyperbolic location fingerprinting (HLF) features [Kjaergaard and Munk, 2008. In our approach, we select the HLF features with the aim to deal with the presence of multiple smartphones.

For reducing the effect of noisy RSS value between different devices, the HLF takes relative difference for every pair of access points. Other than that, the RSS values is transformed into logarithm space. More specifically, from the standard raw features r with D dimension, the HLF is a vector h which has the size of $\mathrm{D} * (\mathrm{D} - 1)/2$. The explicit values of h are given as:

$$h = \begin{pmatrix} diff(r_1, r_2), \dots, diff(r_1, r_D), \\ diff(r_2, r_3), \dots, diff(r_2, r_D), \dots, \\ diff(r_{D-1}, r_D) \end{pmatrix} \quad (12)$$

The different operation between two RSS values r_i and r_j is defined as:

$$diff(r_i, r_j) = \log\frac{r_i}{r_j} - \log\frac{1}{rmax} \quad (13)$$

with the $rmax$ is the maximum RSS value.

When there are hundreds of access points which can be appeared in the environment, the transformation in Equation (12) results in thousands of dimensions in the feature spaces. Thus, collecting enough data for training the HLF features could be problematic. Therefore, we apply a further dimension reduction operator to get a suitable feature representation from the generated HLF features. Several possible approaches are Random Tree Embedding [Geurts et al., 2006] or Principle Component Analysis based methods. In our experiments, we employ the Truncated-Single Value Decomposition, which is proposed by Halko et al. [2011], to reduce the dimensions of from $D * (D - 1)/2$ to D'. The value of D' can be selected as a small value to make learning process in later training phase feasible.

After the feature selection step, the collected data could be represented as the lists of pairs:

$$S = (< X^1, P^1 >, < X^2, P^2 >, \dots, < X^M, P^M >) \quad (14)$$

with $i \in [1, M]$, X^i is the correspondent $N - dimensional$ vector, P^i is the position of the sample i^{th}. For simplicity, the notation X^i could be in one of three types of features, the raw features r, the filters-based features s or the HLF-based features h. At this stage, it could justify each sample's position P^i as triplet representing the coordinates in 3 dimensions: floor, latitude and longitude.

3.4 Learning Models

For learning the mapping from RSS-based vector to position, three family of models are selected, which are Nearest Neighbors, Random Forest [Breiman, 2001] and Extreme Gradient Boost [Chen and Guestrin, 2016]. While KNN model is a popular choice to work with the Wi-Fi fingerprinting data, the others propose are tree based models. They are capable of learning on a small amount of training data. Each type of model is tested with both options, i.e., classifier and regressor.

3.4.1 K-Nearest-Neighbor Model

The K-Nearest Neighbors (KNN) model requires a distance function between two vectors in the feature space. The work in Retscher and Joksch [2016] compares the positioning results of nine different distance functions. Similar distance errors could be achieved with a group of distance functions including Cosine, Euclidean, Hellinger and Chi-square. We choose the Euclidean function for measuring the distance between two feature vectors. Specifically, let $a = \{a_0, a_1, \ldots, a_N\}$ and $b = \{b_0, b_1, \ldots, b_N\}$ be the two vectors in the N-dimension feature space, the distance $d^{euclide}(a, b)$ between a and b is calculated as:

$$d^{euclide}(a, b) = \sqrt{\frac{\sum(a_i - b_i)^2}{N}} \quad (15)$$

For an input X^{new}, the standard KNN algorithm finds K samples in S which have the smallest distance to X^{new}. The distance between X^{new} and X^i is computed by Equation (15). Assume the list of K samples is:

$$nearest(X^{new}, d^{euclide}) = (< X^{u_1}, Y^{u_1} >, < X^{u_2}, Y^{u_2} >, \ldots, < X^{u_K}, Y^{u_K} >) \quad (16)$$

The corresponding P^{new} is computed as the centroid of all P^{u_i} in the above list:

$$P^{new} = \frac{\sum_{j=1}^{K} P^{u_j}}{K} \quad (17)$$

The parameter K defines how many nearest samples in the feature space are taken for the computing of P^{new}. A small value of K tends to make to model overfitting. For example, with $K = 1$, the P^{new} is the position with the nearest sample in the training set S. When the value

of K is increased, more nearest samples are taken into the computing process, thus make the output more stable. However, if there are only a few data, the position of P^{new} could be fluctuated between those nearest samples. In the specific task of Wi-Fi fingerprinting, K is usually chosen from in the range of $[1,10]$.

There are some variants of the KNN model. The most popular one is the weighted-KNN approach. In weighted-KNN, the position output P^{new} is calculated by a weighted sum of each P^{u_j}. The specific weight is a function, which is computed from the similarity of between X^{new} and X^{u_j}.

3.4.2 **Random Forest Model**

Random Forest (RF) model bases on building a list of decision trees. Each tree would split the feature space into random subspaces. Each subspace is expected to contain the samples which have identical or similar target positions. The work in Lin and Jeon [2002] shows that RF model can be justified as an adaptive weighted KNN model.

For training a standard decision tree, the training data S is split from top down by a pair of feature and values until some criterion are met. The criterion could be based on the depth of the tree or errors within for defining a leaf node. Table 3 shows a set of three training samples on the raw RSS data. A decision tree could be built as in Figure 13. In this example, the building process stops when every leaf contains only one sample. There are also other decision trees with different node splitting criteria which can split the tree samples X_1, X_2 and X_3.

Table 3. An example dataset with raw RSS feature for 3 access points

Samples	AP_1	AP_2	AP_3
X_1	-50dBm	-70dBm	-75dBm
X_2	-65dBm	-80dBm	-52dBm
X_3	-70dBm	-65dBm	-76dBm

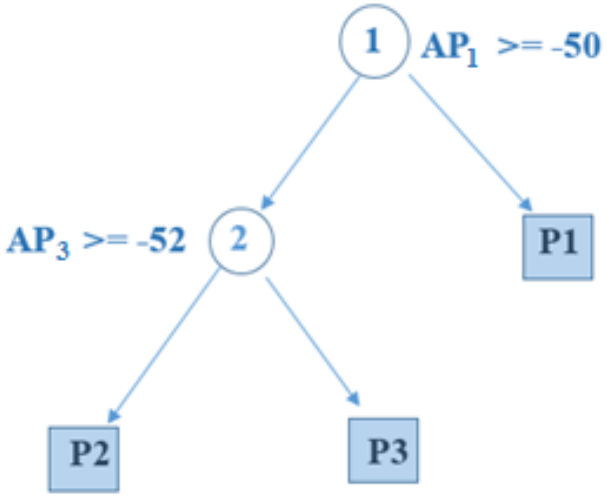


Figure 13. A decision tree with two internal nodes and three leaf nodes for splitting X_1, X_2, X_3 in Table 3

Given a training data set, there are several well-known methods for building a decision tree such as ID3 [Quinlan, 1986] or CART [Breiman et al., 1984]. Because the building decision tree process always seeks for an optimal way to split the feature space, the resultant tree is likely overfitting. The RF model is introduced for overcoming the overfitting issue by building a set of decision trees. In general, all the trees are not an optimal one but have a predictive capability. Bagging methods and random subset of features are employed for lowering the correlation between each tree in the forest. The output of RF model is the mean of all the predictions of each decision tree in the forest.

3.4.3 **Extreme Gradient Boost Model**

Extreme Gradient Boost (XGB) model is also a tree-based learning method. The training method includes several rounds. Starting from a simple prediction, trees are added to the model based on a boosting approach, which tries to reduce the error of the objective function of the previous step. Popular objective functions for XGB model includes the mean square error for regression or logistic function for classification. Apart from the objective function, additional components which measure the model complexity are also added. The objective function is chosen in a way that it is able to compute gradient based on features values. Therefore, at each splitting step, the built tree could choose the right ways to split the feature

space. The model is also included with other techniques for prevent overfitting such as random subset of features or early stopping criteria.

3.5 Experiments

The IPIN Competition 2016 track 3 propose tasks of tracking smartphones through a large, multi-floor building. The data is separated into training log files and test log files. The training log files can be used to train several types of models such as Wi-Fi fingerprinting model and the test log files provide the data for the evaluating purpose. Both training and test log files are collected with a same setup. A user is asked to carry the smartphone along predefined paths. An Android-based application is used to collect several type of sensors such as Wi-Fi scanning samples, accelerometer sensor, magnetic sensor and gyroscope sensor. In this chapter, only the Wi-Fi scanning samples are mandatory for the experiments.

Apart from the smartphone's sensors data, the log file is provided with the moving path of the user. The moving path is present in the form of a list of checkpoints and their arrival timestamp. Firstly, the checkpoints along the moving path are selected. In the data collecting step, every time the user reaches a specific checkpoint, he then notices to the application. From the checkpoints' position and the time stamps, the full user's moving path can be approximated by a linear interpolation step. From the approximate moving path, the user's position at a specific timestamp are matched with the Wi-Fi scanning data for creating Wi-Fi fingerprinting database. The main advantage of using path interpolation is its low acquisition time. In fact, the data collecting process could integrate into daily smartphones easily. The disadvantages are the limited amount of samples and the noise which is generated by the moving of user during the scanning process. A standard collecting data process usually divides the entire area into a dense grid of points. The user is required to stay at a specific point for several minutes to. This ensures the captured Wi-Fi scanning data is good enough for modeling the variances of radio signal in indoor environment. However, this method would take much effort for setup in a large area. By employing the path approximation setup, the cost of building such database could be reduced. The proposed method is also integrated into users' daily life activity easily.

3.5.1 **Data Preprocessing**

The dataset is collected at four buildings. Because the nearest pair of buildings is around 300m in distance, it is trivial to identify the buildings. In our experiments, we select the UAH building which has the largest size of training and testing data. The UAH building is provided with six train log files and other four log files for testing. The training files are split into three separated paths. Each path is recorded with one of the two phone models (Samsung Galaxy S3 and/or Samsung Galaxy S4). Figure 14 illustrates the route number 1 in the floor 1. There are two separated paths in the test log files. The two paths are also collected with two above phone models. The moving paths in training and in testing set are generally different.

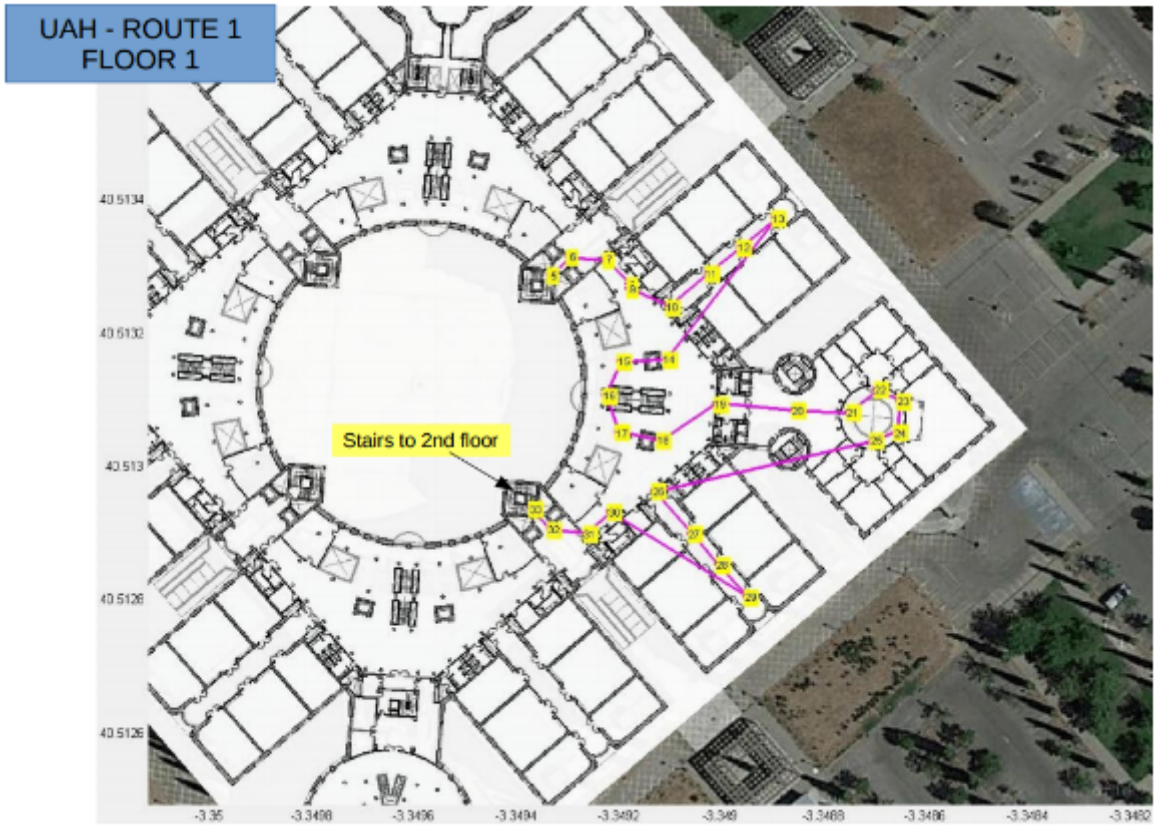


Figure 14. An example of moving part given by the competition organizers

Among different data types in the log files, the Wi-Fi scanning information and the checkpoint information are the two interested data for building the fingerprinting model. Both training and testing files have a same data format. The Wi-Fi scanning data is marked with the WIFI

tag and the checkpoint data is marked with the POSI tag in the log files. Table 4 and Table 5 present the format of the Wi-Fi scan data and checkpoint data in the log files.

Table 4. Samples of Wi-Fi data stream in one of the provided log files

AppTimestam	SensorTimeStamp	SSID	BSSID	RSS
2.742	16169.500	congresos	04:bd:88:50:4b:60	-77.0
2.742	16169.500	eduroam	04:bd:88:50:4b:61	-78.0
2.742	16169.500	eduroam	04:bd:88:50:3d:c1	-70.0
2.742	16169.500	matematicasUAH	00:1c:f0:62:62:d3	-83.0
2.742	16169.501	eduroam	04:bd:88:50:4a:31	-81.0

Table 5. Samples of user input checkpoint in one of the provided log files

AppTimestamp	Latitude	Longitude	Floor	Building
22.206	40.51360786	-3.34883	0	20
34.676	40.51355224	-3.34892	0	20
68.927	40.51335552	-3.34923	0	20
92.021	40.51328754	-3.34934	0	20
132.294	40.51328445	-3.34934	1	20

Each Wi-Fi record is provided with application level timestamp (*AppTimeStamp*), sensor level timestamp (*SensorTimeStamp*), the names of the seen access point (*Name_SSID),* the MAC address of the seen access point (*MAC_BSSID*) and the received signal strength (*RSS*) to the access point. For the Wi-Fi data only, there is not significant different between the application level timestamps and the sensor level ones. We decide to take the application time stamp for the next preprocessing step.

The path is recorded by a list of checkpoints. Each checkpoint record is provided with application timestamp (*AppTimeStamp*) and the checkpoint's position, which includes latitude, longitude, floor and building identifier. The building identifier contains only one constant which indicates the UAH building. Therefore, it can be excluded without the loss of information. There are total four floors, which are assigned the values from 0 to 3. The moving paths contain multiple floors. The users could use the stairs or the elevators for changing the floor purpose. However, in the data, all of the appeared checkpoints are selected to have integer values.

From Table 4, it can be seen that the Wi-Fi records are only provided for a specific Wi-Fi access point at a given time stamp. An additional grouping step is needed to create the standard raw RSS feature vector. Generally, the reachable access points with the same or closed timestamps could be grouped into one feature vector. In order to identify the time threshold for grouping the closed Wi-Fi scans, one could base on the time differences of two consecutive scans. For instance, Figure 15 plots rounded time differences from one of the training log files. Apart from the peak at 0 second, there is another peak at around 4 seconds. A time threshold of 4 or 5 seconds would be suitable for defining a complete scan Wi-Fi cycle for the log file. There are also some long intervals which the data does not contain any Wi-Fi data. These intervals last from one minute to several minutes. They are treated as the noisy intervals in this step.

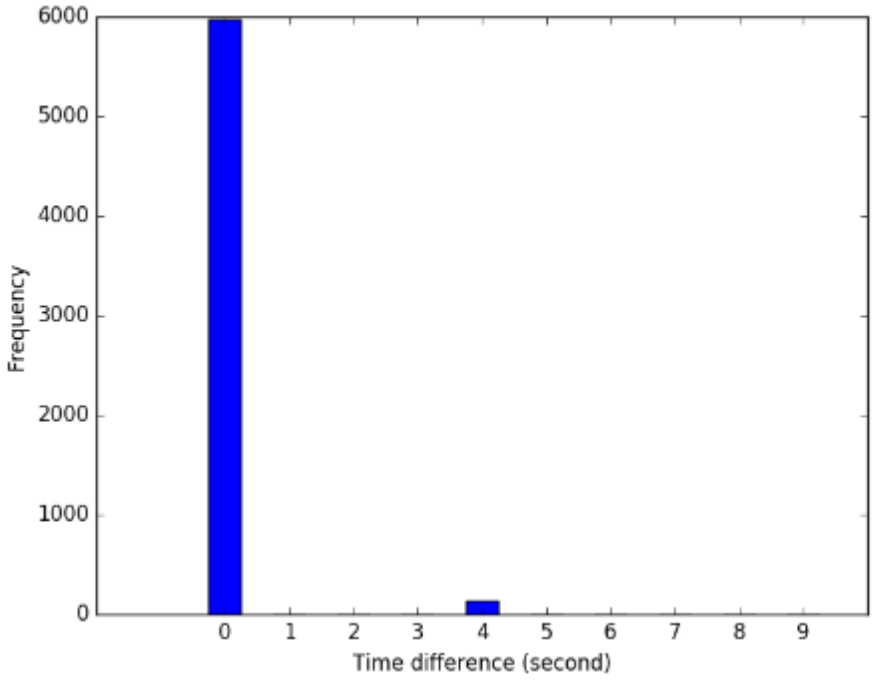


Figure 15. Histogram of time difference between consecutive Wi-Fi scans in a training log file

Table 6 gives the detailed information for the training Wi-Fi data from the provided log files. The grouping process results in 878 training samples across 4 floors. Along the moving path, there is less than one sample per meter. Average scanning time illustrates the time between consecutive scans. The longest scanning time denote the longest interval which do not have

any available Wi-Fi data. Nevertheless, it does not take much time for collecting the whole data set. As reported by the timestamp in the log files, the training data of the UAH building is collected in less than 2 hours.

There are 353 seen MAC addresses in total which results in a raw feature vector of D = 353 dimensions. For computing the default value of unseen access points, we plot the RSS value distribution for the train data. Figure 16 illustrates the histogram distributions of RSS values over all the access points of the data. As the RSS values hardly exceed -100dBm, it is reasonable to pick the value of $MIN_{rss} = -120dBm$. In addition, we use a cut-off threshold at -95dBm. Every access point which has the RSS value less than -95dBm would be assigned the MIN_{rss} value. Another remark from this figure is that the range from -80dBm to -90dBm contains the most RSS values. In practice, the region is considered as the low signal region where the signal strength could fluctuate greatly and is highly affected by indoor obstacles.

Table 6. Summary of training data on UAH dataset

Number of access points	353
Average scanning time	5.25 seconds
Longest scanning time	168.5 seconds
Total number of training	878 samples
Total appeared floors	4
Total moving distance	2640.6m

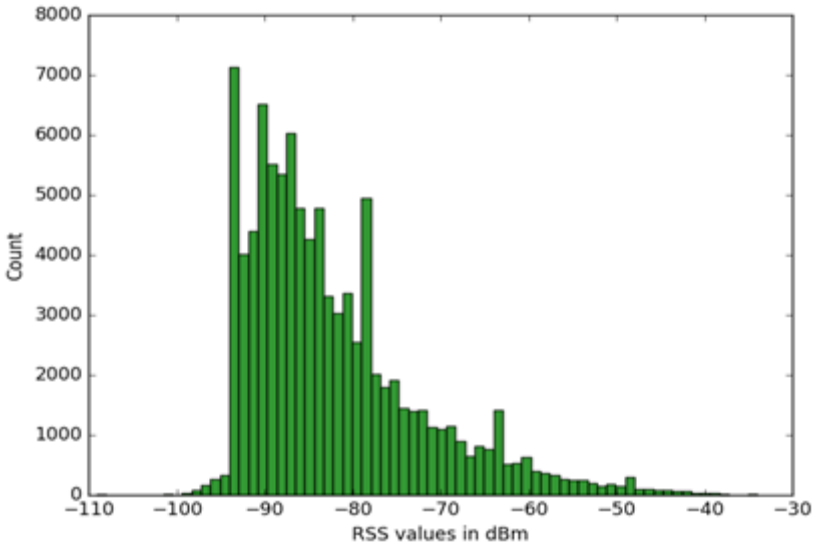


Figure 16. Distribution of the RSS values over the collected data

For each grouped Wi-Fi scan from the above step, we assign it a position as the learning target. The position is a triplet with floor, latitude and longitude. The checkpoints, which are manually input from user, are used to calculate the user's trajectory. Because the log files only contain the timestamp when a checkpoint is arrived, an approximation is needed step to compute the entire user's moving path at a given time t. More specifically, the complete Wi-Fi scan could be created by Wi-Fi records in the interval $[t - dt, t + dt]$ with dt is half value of the scanning time. Then, the position $P(t)$ can be computed by a linear interpolation between two consecutive checkpoints before and after t, which are P_i and P_j, respectively. The position of P_i and P_j are $(floor_i, latitude_i, longitude_i)$ and $(floor_{j,}\ latitude_j, longitude_j)$. The position $(floor_t, latitude_t, longitude_t)$ of $P(t)$ is calculated by:

$$\begin{cases} floor_t = floor_i + \left(floor_j - \ floor_i\right) * (t - i)/(j - i) \\ latitude_t = latitude_i + \left(latitude_j - \ latitude_i\right) * (t - i)/(j - i) \\ longitude_t = longitude_i + \left(longitude_j - \ longitude_i\right) * (t - i)/(j - i) \end{cases} \quad (18)$$

The $floor_t$ would need a rounding to integer value if the user changes the floor in the process. The linear approximation would produce some degree of noise for the output position. Within a scanning cycle, the Wi-Fi data of all the seen access points are not likely arrival at the same time. From the plot in Figure 15, a time window length of 1 second is expected. The specific position for each seen access points, therefore, could be varied. The linear approximation position in Equation (18) is an approximation from all the available positions. Assuming that the user moves at 1.0m per seconds, the noise can be excluded.

For the test data, Table 7 gives information on the 4 testing log files. All of the testing files come with a pair of same moving path but with different smartphone models. The test file 1 and test file 3 are one pair and test file 2 and test file 4 are the other pair. Because of the hardware dependency, the number of complete Wi-Fi scans differ greatly for Samsung Galaxy S3 (S3) and Samsung Galaxy (S4). The joined percentage with train data checks the sharing points between the train data and the test data approximately.

Table 7. Summary of 4 testing log files on UAH data

	Test file 1	Test file 2	Test file 3	Test file 4
Phone model	S3	S4	S4	S3
Total time	1477 seconds	899 seconds	1477 seconds	899 seconds
Total travelled	763m	370m	763m	370m
Number of Wi-Fi scans	159	174	291	96
Joined percentage with train	94%	88%	94%	88%
Number of appeared floors	4	3	4	3
Number of changing floors	7	5	7	5

3.5.2 **Floor Classification Results**

For user localization, on one hand, one can use the position in 2.5-D as a training target. The target includes a tuple of three values, which are floor, latitude and longitude. The floor is given in integer type while the other values are floats. The KNN-based model could handle such type of target naturally. With tree based model, some difficulties arise when creating a suitable objective function. The function should be designed to handle the miss match in scale between floor and the other two latitude and longitude values. An alternative approach is to split the learning position task into two separated sub-tasks. The first sub-task is to identify which floors the user is standing and the second sub-task is to compute the latitude and longitude. Both approaches are expected to have similar performance.

In our experiments, we select the second approach. The Table 8 provides the number of training and testing samples of the building UAH. The number of training samples and testing samples depend on the specific user's walking part and the used phone model. The floor 0 has the highest training samples and the floor 3 has the least amount of samples. The test data has the most samples in floor 1. There are two floors, floor 1 and floor 3, which have the test samples larger than the train samples. Thus, it could result in a significant out-of-samples issue for training the floor classification models.

Table 8. Summary of training data for the floor classification task

Floor Id	Train Samples	Test Samples
0	334	126
1	233	298
2	215	147
3	96	149

3.5.2.1 *Finding Baseline Model*

The KNN model is selected as the base model for our testing, due to its effectiveness with the Wi-Fi RSS data. Among the three types of features, we rely on the raw RSS value feature for selecting the KNN model. There are many ways to vary the model parameters for a KNN model approach. In our experiment, we select the K - the number of neighbor parameters as the tuning parameters. The metric distance for the KNN model is the standard Euclidean distance.

The results are shown in Figure 17. The best result on cross validation test is with by set the number K of neighbors being 1, with 0.97 of accuracy. However, on test data, the best result on test can be reach with K = 4 with around 0.929 of accuracy. With high value of K, both performance on cross validation and test data are decrease. The mismatching between cross validation results and the test results comes from the new paths in test set. The paths lead to unseen samples in train data. It is reasonable that the baseline model should be choose as the best cross validation performance, which is achieved with K = 1 in this case. However, the value $K = 1$ could make the model bias easily because the floor output depends only on the nearest RSS finger printing in the feature space. In the later experiments, we decide to choose $K = 3$, which achieves good results on both cross validation and test data, and as the baseline model.

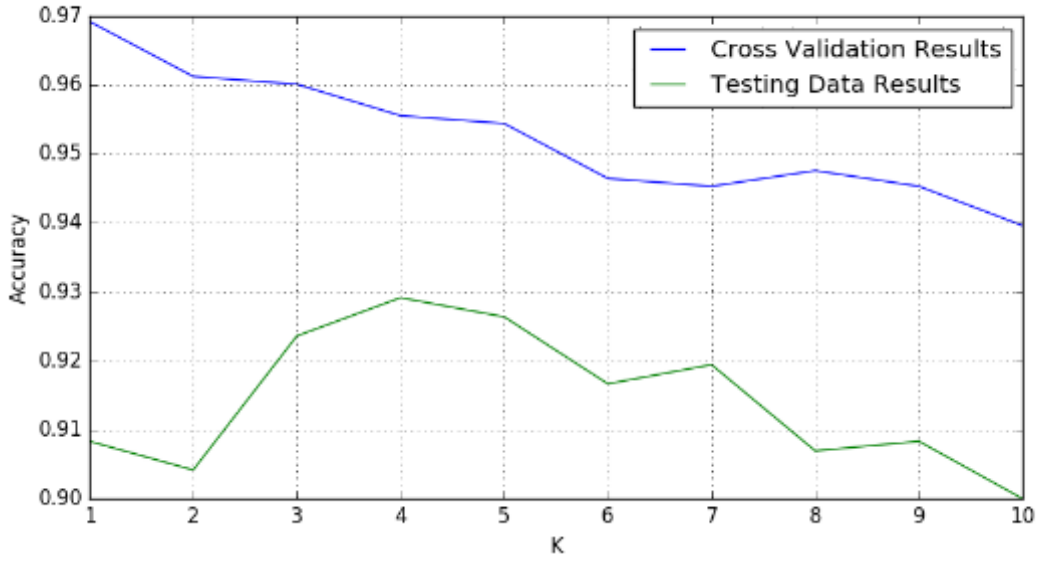


Figure 17. Accuracy of floor prediction by varying the number K of neighbors in cross validation results and test results

In the Figure 18, we plot the mean square distance for different values of K against this baseline model. With the KNN model, it is clear that the set of nearest neighbors will be extended when K increases. Therefore, we ensure a high level of similarity between all of the models when K is varied from 1 to 10. The high correlation makes the later combining step have insignificant improvement. For example, the ensemble model, which takes the average prediction from $K = 1$ to $K = 5$, reaches 0.925 of floor hit rate and does not improve the result.

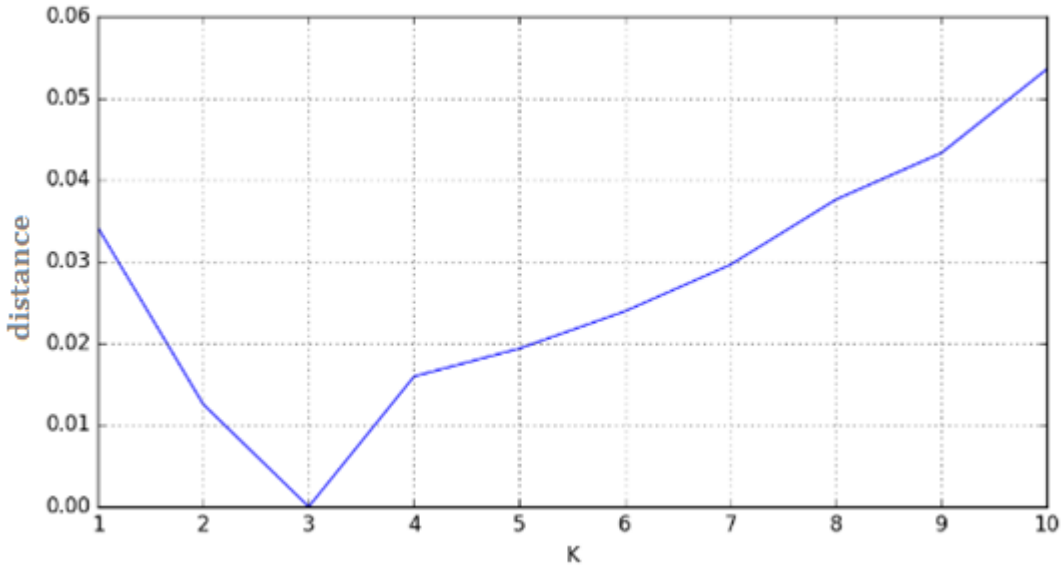


Figure 18. The similarity measure of 10 KNN models again the based line K = 3

Similar experiments are extended to RF model and XGB model. The number of threes in RF model and XGB model are selected using the same approach as in the KNN model. Figure 19 and Figure 20 present the result of the RF model and XGB model respectively.

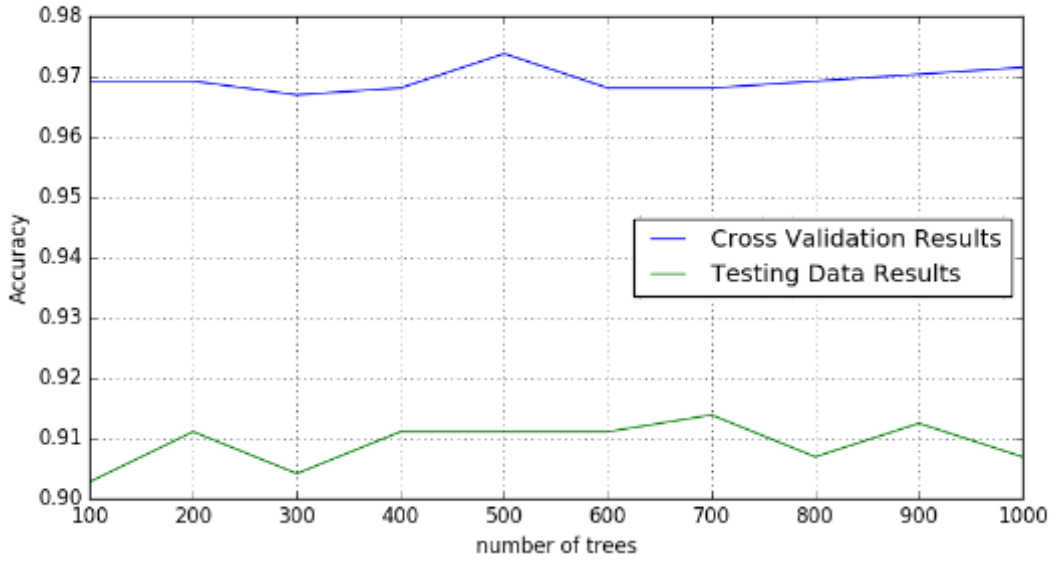


Figure 19. Accuracy of the RF model by varying the number of trees in cross validation results and test results

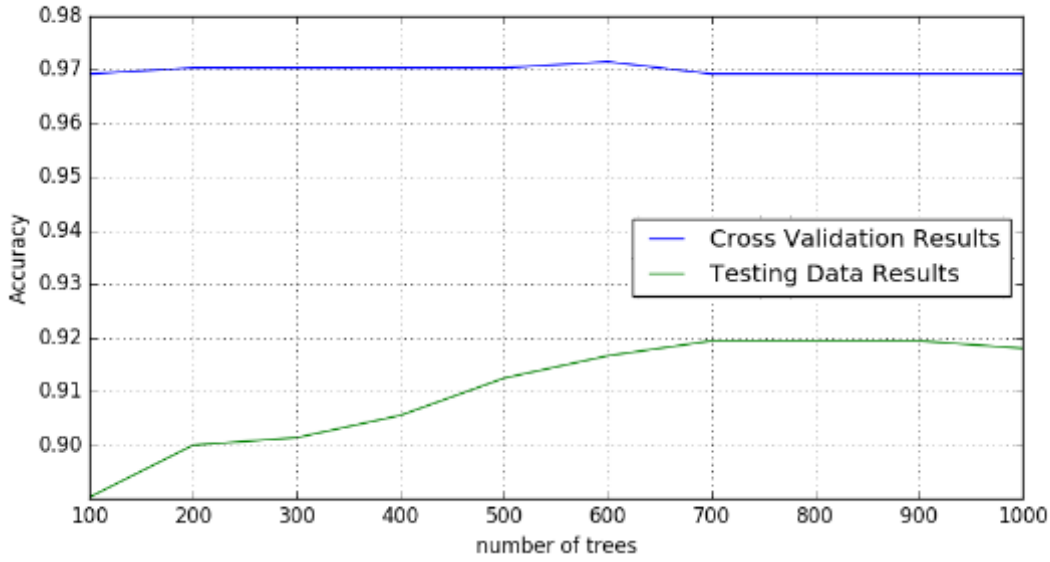


Figure 20. Accuracy of the XGB model by varying the number of trees in cross validation results and test results

A similar pattern could be seen for the two select models. Both cross validation results and testing results remain stable when the number of trees is varied. There is an accuracy bias

between cross validation results and testing results. The cross validation results always stay at 0.97 of accuracy while the test results are around 0.91 of accuracy. The highest accuracy on test data can be reach with 700 trees for both RF model and XGB models. The accuracy of both models, which is around 0.92, is slightly lower than the accuracy of the KNN model's best configuration. At this stage, combining all the models does not change much the outcome. For instance, taking the mean all the prediction of the ten RF models results in an accuracy of around 0.91. It is likely that the high correlation between the set of models make the results decrease. Although there are other tuning parameters for both the RF and XGB models, it takes a great effort to find the right combination. This is the reason for which we introduce a different way to vary the set of models for floor classification task.

3.5.2.2 *Floor Classification Results with Different Target Functions and Feature Spaces*

In the previous section, for the KNN model, it is showed that combining models with different value of K is unable to improve the outcome. The similar difficulty could be observed in combining RF and XGB models. In the work of Torres-Sospedra et al. [2016], the authors suggest to alternate the distance similarity measures, the data representation and the filter on weak signal. The results show that an improvement could be achieved with a proper number of models. We apply the same idea, thus, with a different set of models. In our approach, based on the three base line models, KNN, RF and XGB, the target function and feature space are varied.

For the target function, it is straightforward to train a classifier for differing between floors. Besides that, one could view the floor as a continuous value. The target is a real number indicating the floor, and the output is also a real value x in the range of $[0,3]$. Then, regression models can be trained for predicting the floor values. At later step, in order to convert from the real value x to a floor number, we use a cut vector $C = \{c_1, c_2, c_3\}$. A value x is classified as the floor number i if i is the smallest value which satisfies $x \leq c_{i+1}$, otherwise, x is classified as the Floor 3. The value of vector C could be computed directly on training data as a sub optimization step. After the regression model is trained, we get the model's prediction values

on all the training data. Then, the cut vector C is selected in a way that maximizes the prediction accuracy of the prediction values and the training targets.

For the feature space, apart from the standard RSS features, we add the filtering-based features and HLF features. The filtering-based features are created by adding the indexes of two strongest access points from the raw RSS features. With the HLF features, the value of D' after the dimensional reduction step is set to 25.

Table 9. Floor accuracy on the UAH dataset with different target functions and feature spaces.

Target Function	Feature Type	KNN		RF		XGB	
		Train	Test	Train	Test	Train	Test
Classifier	Raw	0.960	0.924	0.969	0.915	0.970	0.913
Regressor	Raw	0.959	0.925	0.967	0.890	0.954	0.892
Classifier	Filtering-based	0.942	0.863	0.971	0.907	**0.975**	0.925
Regressor	Filtering-based	0.938	0.861	0.962	0.882	0.960	0.875
Classifier	HLF	0.957	**0.933**	0.965	0.918	0.966	0.912
Regressor	HLF	0.958	0.931	0.942	0.868	0.911	0.831

The Table 9 provides the floor hit rate on both train and test data. The result of train data is reported by using a 5-fold cross valid setup. The best results on train data is 0.975 accuracy when the XGB model is trained with the classifier target function on the filtering-based features. Meanwhile KNN with HLF features has the highest accuracy on test data. The KNN model with both target functions classifier and regressor reach 0.93 of accuracy approximately. Overall, the KNN model family has the best performance on test data, though, its performance in the train data is less than the other two tree-based models. The tree-based models likely have some overfitting issues. They have high accuracy on train cross validation setup and low accuracy on the test results. In terms of correlation, all the tested models have a high correlation between the train cross validation results and test results. Besides that, the regressor approach performs less than the corresponding the classifier approach. While those performance drops are insignificant with KNN, the drop can be seen clearly for the two tree bases models. Given the size of training data, it is difficult to find a robust cut vector C.

In terms of feature, the raw features are clearly the most stable features. They achieve good performances in train and test regardless the models and the target functions. The other two feature spaces result in different outcomes for each specific learning model. The filtering-based features add more discriminant power for the tree-based models, RF and XGB models. However, the features make the performance of KNN decrease significant. It seems the added component of the filtering-based features affect the similarity measurement of the distance functions. With the classifier target function, the HLF features have small effects on the tree-based models. In term of the KNN model, the HLF features can be used for reducing the over-fitting issue. The features perform well on test results but have a decrease performance in train cross valid setup.

In order to improve the results on test data, Table 10 presents the accuracy of several ways for combining models. From the 18-available approaches, four ways of combination are tested, which include *All*, *Remove Noise*, *All Classifiers* and *All Regressors*. The *All* configuration includes all the models. The *Remove Noise* ensemble approaches are constructed by excluding models have a low accuracy on the cross valid result such as using the RF model with the regressor objective and HLF features. *All Classifiers* and *All Regressors* use all the model with respective objection functions. By using all the available configurations, the ensemble results reach around 0.925 of floor hitting rate. The best performance is 0.938 of accuracy if the noisy models are removed from the ensemble result. For selection the noisy models, a threshold of 0.95 with the accuracy on cross valid result is used. Moreover, the *All Classifiers* setup results in a small improvement. The *All Regressors* has a significant lower performance. Nevertheless, it is likely that the regression objective functions add more useful predictions to the results. Two combining functions, mean and median, are tested and it can be concluded that there is not significant difference between the two ones.

Table 10. Accuracy on test data with several ways of ensemble models

Ensemble Method	Used Function	Accuracy
All	Mean	0.923
All	Median	0.925
Remove Noise	Mean	**0.938**
Remove Noise	Median	0.936
All Classifiers	Mean	0.926
All Classifiers	Median	0.926
All Regressors	Mean	0.906
All Regressors	Median	0.904

Table 11. Performance of the *Remove Noise* combination setup with mean function on each test log file

Filename	Phone Model	Accuracy
Test file 1	S3	0.918
Test file 2	S4	0.938
Test file 3	S4	0.959
Test file 4	S3	0.909

Table 11 gives the floor accuracy of each test file with the best combining configuration. The performance of the approach could depend on the tested phone model slightly. The S3 phone model has lower accuracy than the S4 phone model. Within the for test files, the test file 4 has the lowest floor hit rate. We plot the path in the Floor 1 and Floor 2 of the test file 4 in Figure 21. It is interesting that all of the wrong prediction floors are near the stairs or elevators area. In the plot, there are one point in the left figure and two points in the right figure. Similar wrong prediction patterns could be seen in the three remaining test files. It can be explained by the high amount of Wi-Fi RSS noise in those area. When the user changes the floor, the signal patterns from one floor to the next floor are usually overlap. Moreover, in case of using elevators, the signal in the area the elevators could be blocked. Therefore, it makes the RSS values change in an unexpected way.

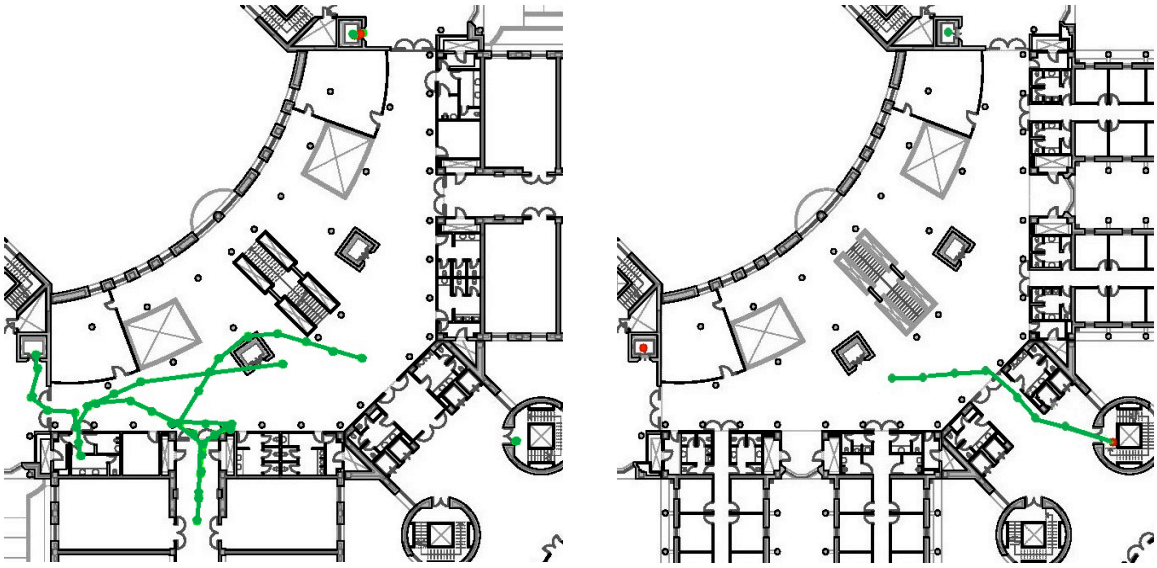


Figure 21. The floor prediction in Floor 1 (left) and Floor 2 (right) of test file 4. The green dots are true predictions and the red dots are false ones.

3.5.3 **Positioning Results**

In the section, we present the results on the Wi-Fi based positioning given the same training and test data. From the previous sections, the floor can be classified given the data. Therefore, instead of working with the positions in 2.5D (position and floor), we are now able to work with the position in 2D only. From this point, there are two ways for training the models. The first one is to train only one model on all training data. In this case, the model should have capability to distinct between different floors, given the same 2D coordinate. This could introduce difficulty to create the right separating plane. Nevertheless, the model has quite a good amount of data to work on. The second approach is to train a separate model for each different floor. Noisy samples between different floors could be avoided with this approach. The disadvantage of this approach is a significant reduction of training data for each model. For example, from the data in Table 8, there are only around 200 samples per model in this case even if we expected the same performance. For this reason, we selected the first approach, which needs to train only one model on the data.

3.5.3.1 *Finding Baseline Model*

From the three model families, KNN, RF and XGB, we run similar setup with 5-fold cross validation on the train data. For test data, the models are trained with all the train data and

predict against the test data. The results are reported with in mean error and its standard deviation (*std*). Metrics such as root mean square error, median error or distance error at percentile 90th, are relevant for comparing positioning output. All of those metrics though have correlation at some degrees. Therefore, it is more convenient to rely only on mean error and its *std* for evaluating a positioning model's performance.

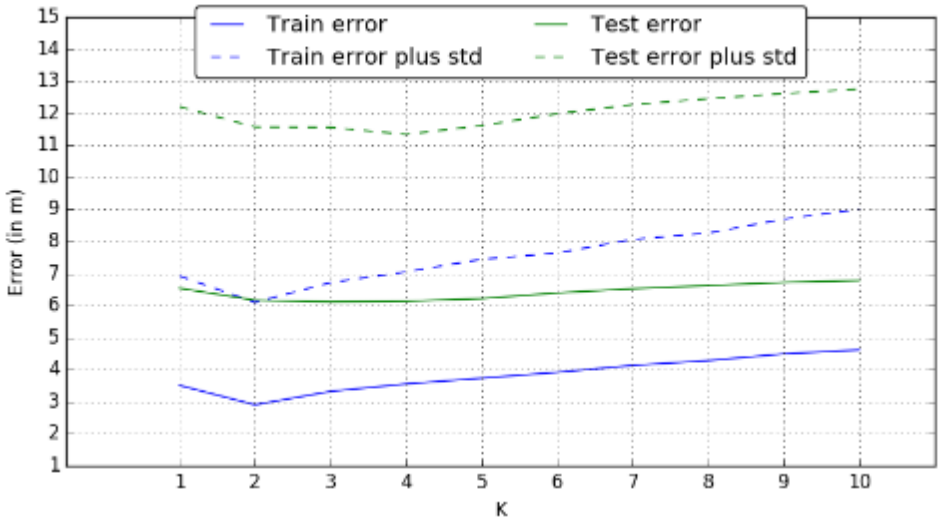


Figure 22. Error distance of KNN model with std on both training and testing data when the number of neighbors is varied

For KNN, the number of nearest neighbors, K, is varied from 1 to 10 and the results are compared with each other. Figure 22 plots the results of the KNN model with those values of K. The best result on train data has the errors around 3.0m with a *std* value of 3.20m. The result can be reached by selecting the number of neighbors $K = 2$ for the KNN model. Meanwhile, the best result on test data is twice this value at around 6.0m in mean error and has a *std* value of 5.40m. Those results can be achieved with $K = 3$ or $K = 5$. Both train and test data have better results with smaller values of K than larger ones. In the experiment, the highest number of neighbors (K = 10) results in the worst performance.

Clearly, the trained model does not generalize well between train and test data. There is a large gap of around 4m between the mean errors of cross-validation results on train and the results on test data. If the *std* is taken into account, the expected error on test results stays near the error region of 12m, while the errors on train data is around 7m. The cross validation setup represents the situation where the test data is drawn randomly from the train data. In

a practical context, if the collected data contains enough training points over the entire floor and enough samples per point, the positioning errors are around 3.50m. When the training data is not large enough to cover the test area, the results would decrease significant. In this case, the errors of a trained KNN model increase from 3.50m to over 6.00m. Without adding more data to the training step, it is difficult to fix the overfitting issue. A simple straightforward combination between 10 KNN models with different value of K does not affect much to the overall results.

For identifying the baseline parameters of RF and XGB models, the number of trees is varied in order to select the best configuration. The results for RF model and XGB model are illustrated in Figure 23 and Figure 24, respectively. Both models have more stable results then the KNN model when the number of trees changes from 100 to 1000. In term of errors, they have fewer errors than the KNN model, in both type of errors, train data and in test data. With RF model, the best performance on train data and test data are around 4m and 5.75m, respectively. The XGB models have a similar performance on train data and a higher error on test data. In general, the number of trees does not affect much the models' performance. The two models also have an overfitting issue between cross validation errors and test errors.

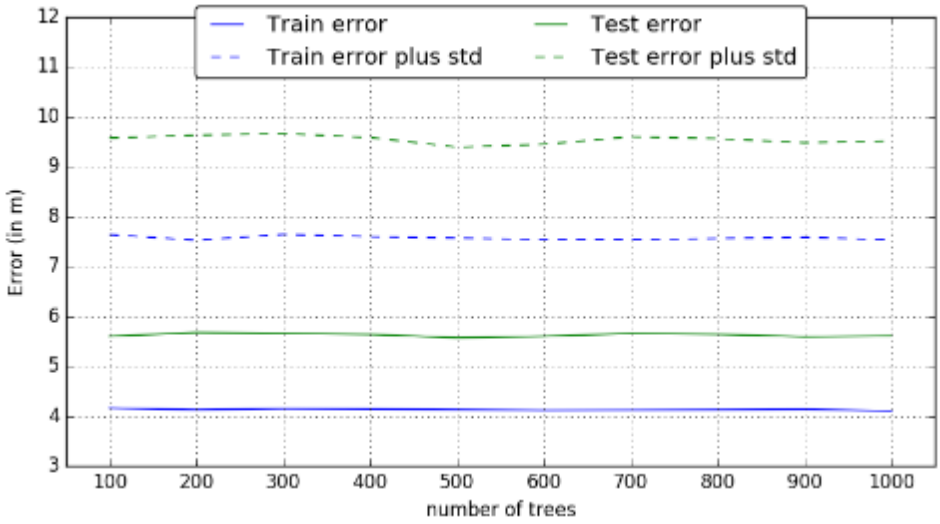


Figure 23. Error distance of RF model with std on both train and test when the number of trees is varied

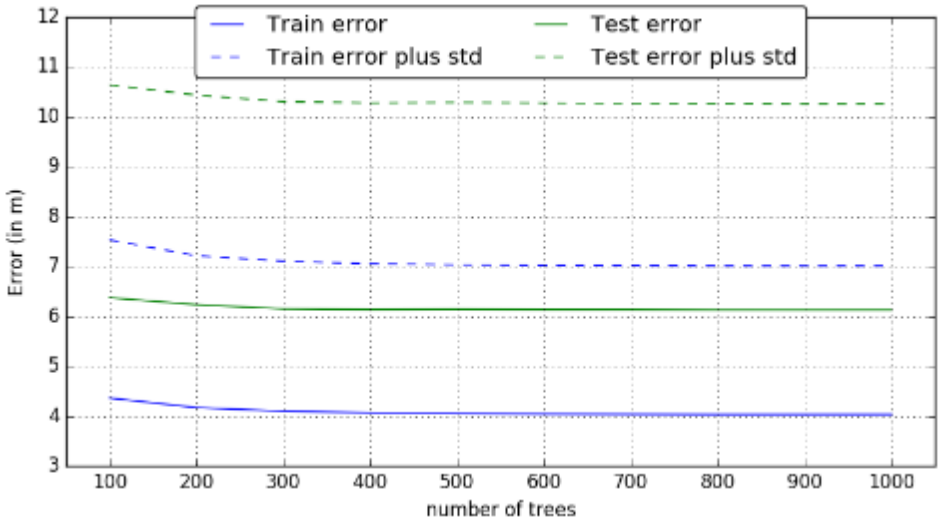


Figure 24. Error distance of XGB model with std on both train and test when the number of trees is varied

The cumulative distribution errors for the best configuration of three models on train data and test data are plotted in Figure 26 and Figure 25. The three models have a quite closely behavior regardless using train or test data. In the train data, the median values of three models stay around 3m, while the median values of the errors in test data are around 5m. In terms of generalization, the results of RF model are slightly better than the other twos. 90% of time, the errors of RF model are under 10m for both train and test data.

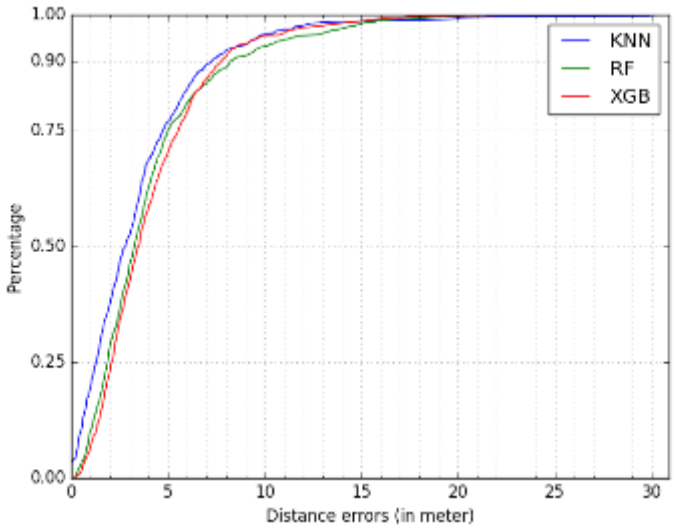


Figure 25. Cumulative distance error distribution on training data for the three models with their best configuration

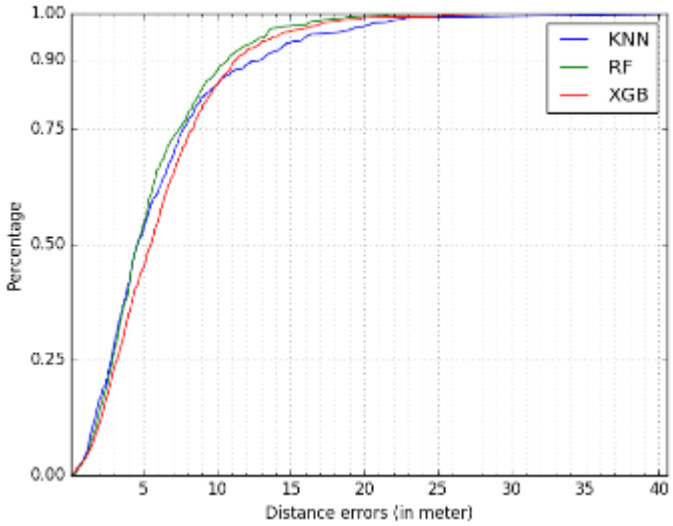


Figure 26. Cumulative distance error distribution on testing data for the three models with their best configuration

From the plot, it can be seen that there exists small amount of samples, which have exceptionally high errors value, which are well above 30m distance errors. By looking closely at those samples, they are mostly recorded when the user is inside the elevators. In this case, the signal from nearby access points is often blocked and the output positions are predicted at the other part of the map.

3.5.3.2 *Positioning Results with Different Target Functions and Feature Spaces*

Different combinations between models and feature spaces are tested in this section. Although the nature of localization is a regression task, a classifier-based task could be constructed by assigning a number which represents the two values latitude and longitude. From all the possible user positions, closed positions can be aggregated into one group. Then, the group identification could be used to represent the classification label of all the members in the group.

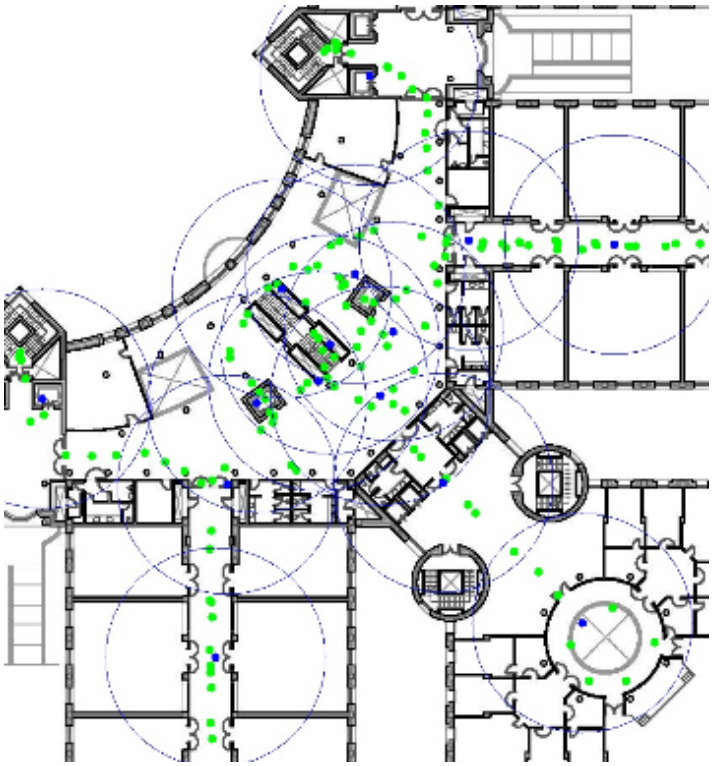


Figure 27. The green dots are the training WIFI points, and the blue dots denote the center of the clusters. The radius of the clusters circle is 10m for visualizing purpose.

There are several ways to group the points. The most popular way is to divide the area into smaller squares (cells). All points within a square would belong to one group. Another approach could involve manual picking of group by using the provided map. In our experiment, we employ the standard K-means clustering for this step. From all the appeared positions in the training data, the points are clustered into different centers based on the distance. Figure 27 illustrates the process of K-mean clustering. Then, for each training point, the corresponding cluster number is used to replace the regression target of latitude and longitude. The classifier is trained with the cluster number the learning target. In the later prediction phase, after get the output from the classifier as the label c, the position of the center c^{th} is then used as the position output.

The errors on cross validation train data and test data with different combination of models, features and objective functions are presented in Table 12 and Table 13, respectively. In the process of clustering all training points, we select 50 clusters for the K-mean process. For the two added features set, the same configuration from the floor classification is used. The filtering-based features are created by adding the indexes of two strongest access points. With

the HLF features, the full HLF features vector is reduced to 25-dimensional vector by applying the Truncated-Single Value Decomposition method.

Table 12. Positioning results on the training set with 5-fold cross validation with different combinations

Target Function	Feature Type	KNN	RF	XGB
Classifier	Raw	4.47m ± 3.78	5.20m ± 3.99	5.04m ± 4.45
Regressor	Raw	**3.32m ± 3.39**	4.21m ± 3.44	4.61m ± 3.42
Classifier	Filtering-based	6.35m ± 5.79	5.36m ± 4.03	5.11m ± 4.54
Regressor	Filtering-based	5.38m ± 5.73	4.34m ± 3.50	4.60m ± 3.26
Classifier	HLF	4.59m ± 3.75	5.52m ± 4.42	6.23m ± 5.51
Regressor	HLF	3.59m ± 3.48	4.42m ± 3.96	5.37m ± 4.20

Table 13. Positioning results on the testing set with different combinations

Target function	Feature type	KNN	RF	XGB
Classifier	Raw	6.49m ± 5.45	6.46m ± 4.44	6.56m ± 5.34
Regressor	Raw	6.11m ± 5.44	**5.75m ± 3.94**	6.10m ± 4.10
Classifier	Filtering-based	7.27m ± 5.83	6.58m ± 4.49	6.40m ± 4.99
Regressor	Filtering-based	7.01m ± 6.05	5.88m ± 4.04	6.12m ± 4.07
Classifier	HLF	6.33m ± 5.63	6.59m ± 5.20	7.41m ± 5.90
Regressor	HLF	5.92m ± 5.61	5.95m ± 4.71	6.43m ± 4.94

From the two tables, it can be seen that the results are varied across all the configurations. The best result on cross validation setup of train data can be achieved with the KNN regressor model on raw RSS features, while the best result on test data is the RF regressor model with raw RSS feature. From the three selected feature spaces, it is clear that the default *Raw* feature has the best perform overall. In term of based line model, the KNN model has the best performance on train data but its results decrease on the test data. The RF model has a more stable performance. By the changing the target function from the standard regressor to the classifier, the distance errors have an increment from 0.5m to 1.0m in term of errors. All the configurations have a high correlation between the mean error and the *std* value.

On the cross validation setup, the KNN model can reach the mean error of around 3.5m with raw feature and HLF features. It outperforms the results of RF and XGB by a large margin. The best result of RF and XGB model are 4.21m and 4.61m, respectively. Both results are achieved

with raw features and the standard regression approach. With the filtering-based feature space, the KNN model's performance has a big increase of 2m in mean errors, which make its performance worse than the other two tree-based models.

For the test data, the results are quite stable between different models and feature space. The mean errors are distributed in the range from 6.00m to 6.50m with some exceptions. The best mean distant error is 5.75m with a *std* value of 3.94m by using the RF regressor on the raw features. Changing the feature space to F or HLF makes the errors increase slightly. The worst combinations are the XGB classifier with the HLF features with a mean error of 7.41m. The combination of KNN with filtering-based features also have mean errors above 7.00m.

Based on the above results, several ways of ensemble model is introduced in Table 14. The *All* configuration is formed by taking the mean of all the 18 models above. The *Remove Noisy Model*s is almost the same as the *All* configuration but exclude the bad combination of model and feature set. Specifically, the combinations which have the mean distant error on test data above 7.00m are excluded. The *Regression Models Only* configuration is to ensemble the nine regression-based models. The *Classifier Models Only* configuration uses the same rule, but with classifier-based models. The last configuration takes the combination of all the six RF-based models. The combination function is the mean function. From a list of output positions, we take the mean of each coordinate axis as the new coordinate in the output position.

Table 14. Distance errors on several way of ensemble the position output

Ensemble Method	Train Errors	Test Errors
All	3.97m $\pm$ 3.11	5.45m $\pm$ 3.98
Remove Noisy Models	**3.39m $\pm$ 2.80**	**5.12m $\pm$ 3.68**
Regression Models Only	3.71m $\pm$ 2.98	5.38m $\pm$ 3.92
Classifier Models Only	4.62m $\pm$ 3.54	5.94m $\pm$ 4.31
RF Based Models	4.65m $\pm$ 3.65	5.97m $\pm$ 4.32

The results show that a simple taken mean of all the models could decrease the errors on test data set. On the train data, the approach reach can also increase the performance overall, however, does not outperform a single KNN model. It has a slightly higher mean distant error but a significant lower *std* value. The second approach of ensemble could reduce the mean

distant errors on test data to 5.12m and has an improvement in the *std* value. The *Remove Noisy Models* approach also matches the best performance on cross validation test. Both the approaches of *Regression Models Only* and *Classifier Models Only* increase the results of individual models. On the other hand, the RF Based Models ensemble configuration decreases the performance of the best RF models. It indicates that basing solely on one model could make the ensemble results overfitting easily.

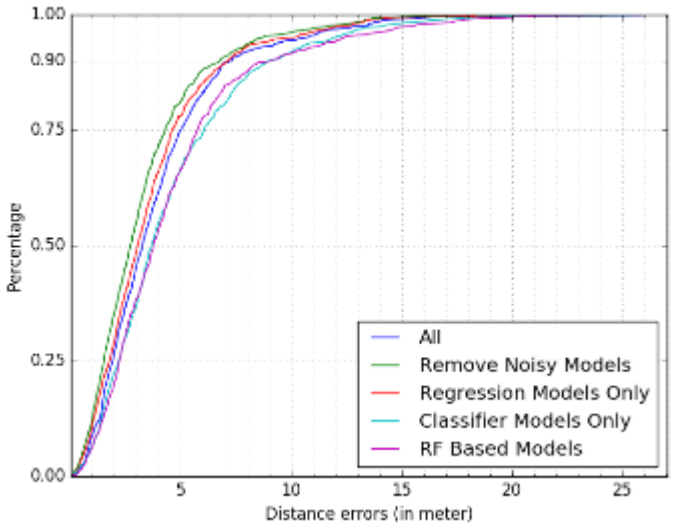


Figure 28. Cumulative error distribution of five ensemble approaches on training data

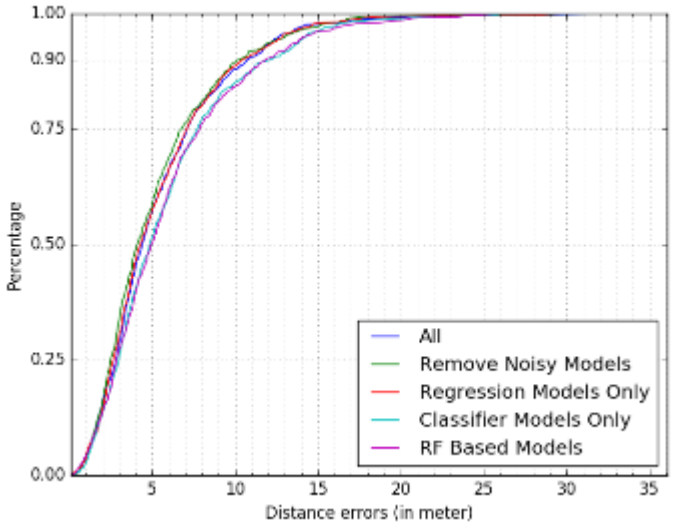


Figure 29. Cumulative error distribution of five ensemble approaches on testing data

Figure 28 and Figure 29 illustrate the error distribution of five ensemble approaches on train and test data. All of the approaches have a close performance in the low errors region. There is a divergence range after the 75^{th} percentile where the *Remove Noisy Models* approach constantly has a better performance than the others. The best model has 90^{th} percentile errors is around 7.0m on the train data and 10.0m on the test data. At the higher-errors regions, all of the models suffer from large distance errors as high as 20m.

The errors distribution of the *Remove Noisy Models* ensemble for each test file are illustrated in Figure 30. The detailed information for each test file can be seen in Table 7. The test file 1 and test file 3 are the same route with different smartphones. By changing the collecting phones from Samsung Galaxy S4 to Samsung Galaxy S3, the fingerprinting model's performance decreases significantly, especially at the higher errors regions. Similar conclusion can be drawn by looking at the pair of test file 2 and test file 4.

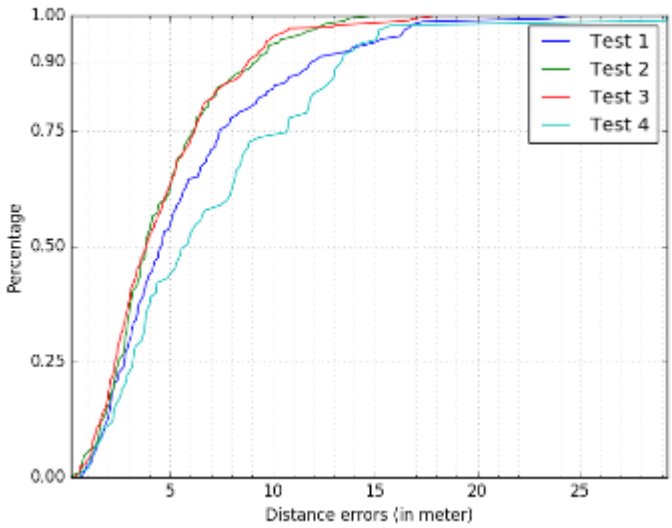


Figure 30. Cumulative error distribution for each specific test file

3.6 Summary

In this chapter, we study the Wi-Fi finger printing approach for indoor positioning. The work focuses on reducing the labor-cost of collecting data with smartphones. As the Wi-Fi fingerprint data is expected to be collected in a simple way, its disadvantages are high noise rate and inadequacy.

Given the data, our approach is to vary feature spaces and learning models for dealing with the noise of radio signals. Three types of features have been tested, including Raw RSS features, Filter-based Features and HLF features. KNN, RF and XGB models are the selected training models. The propose approaches are tested on both train data with cross validation setup and published test data. From the performance of a single model, several ensemble approaches are introduced. The best result achieved in test data is around 93.8% accuracy for floor classifications and 5.12m mean distance error with a *std* of 3.68m for position output. However, there are also several remaining problems. The floor predictions are unstable when the user is in specific area such as stairs or elevators. In case of position output, there are several high distance errors within our results, which make the proposed methods unreliable for indoor applications. Moreover, the device-dependent issue contributes a large proportion to the positioning errors. In order to improve the performance, combining the Wi-Fi data with other available technology in smartphones should be considered.

Chapter 4 Improving Inertial Sensors Tracking Results on Smartphone with WIFI

4.1 Introduction

The accelerometer, gyroscope and magnetic sensors are widely integrated into smartphones and tablets nowadays. Besides Wi-Fi based positioning system, the data from these sensors provides a feasible solution to track the user's movement in a GPS-denied environment. The accelerometer sensor presents the acceleration data of the phone motion. The gyroscope sensor contains the turning angular rate. The magnetic sensor contains the data of the magnetic field. By combining the three sensors, it is able to approximate the phone's motion. Then, the user's movement could be derived from these results.

The two popular approaches for tracking the user by the inertial sensors could be split into Inertial Navigation Systems (INS) and Step-and-Heading Systems (SHS). Both systems are able to compute the relative position of the users from a starting point. While the former approach focuses on tracking the movement of the users by a moving path in 3D, the latter evolves the step and heading calculation of the movement. For the tracking user by their smartphone, the SHS only requires finding the user's speed and velocity in 2D coordinates. It could be considered as a simpler version of the INS. However, for most of the SHS approaches, the relative smartphone's position to the user's body plays an important role. A standard position is that the smartphone is handled in front of the user. In this case, there exists a great correlation between the phone's direction and the user's moving direction. Therefore, it is easy to derive the user's direction from inertial data of the smartphone. Positions such as the phone in the user's pocket are still challenging for building the SHS system.

SHS approaches split the user movement into the moving distance and moving direction. There are several ways to extract the speed, such as using the accelerometers or gyroscopes.

Popular approaches include integrating accelerometer or detecting the moving pattern of the user. Usually, additional information such as the user's step length is needed for inferring the moving distance. With the user's direction, in a noise-free environment, the magnetic sensors can use directly because it provides the direction to the North-magnetic of the Earth. Integrating the gyroscope sensor values is another standard way for computing the heading values. In practice, the two methods are error sensitive. There are several sources of noise, which come from the environment and the device itself. For example, the presence of wall and electronic devices in indoor environment could alternate the magnetic field. The gyroscope sensor in the smartphone often measures the device's angular rate with noise. It makes integrating the gyroscope's angular rate become unreliable. In order to calculate the user's heading in a stable way, filtering methods are introduced to take care of the measurement errors and fuse different sources of heading calculation. Popular filters are Complimentary filter [Mahony et al., 2008], Kalman filter [Choukroun et al., 2006], and Madgwick filter [Madgwick, 2010].

Practically, it is difficult to have an error-free heading estimation methods. Moreover, with the characteristics of relative tracking methods, the drifting effect from the errors at each individual step could be a serious issue. Designing a stable SHS on smartphones for a long working period requires a lot of calibration efforts. Other sources of information are usually needed for adjusting the drifting errors. For example, several methods of opportunistic calibrating are introduced in the works of Zhou et al. [2014] and Qian et al. [2015]. In a calibrating-free environment, the SHS tracking would suffer a great decrease in performance. In many cases, an additional positioning method is preferred such as GPS-based or Wi-Fi based. In additional to that, the usage of additional position methods can be justified because the SHS system needs an additional method to extract the starting position of the user effectively.

In our work, we study the performance of a standard SHS for such context. The work could be divided into two phases. The first phase is illustrated in Figure 31. For determining the speed, two popular methods, step counting and moving window approach, are employed. For determining the heading, three methods, which are the direction cosine matrix, Complimentary Filter and Madgwick Filter are employed. Before combining the moving speed and the head-

ing into a complete tracking solution, the performance of each individual approach are analyzed and the best approach for each part is selected. Then, a particle filter approximation process is used to create the tracking results. We evaluate our system on the data from IPIN 2016 competition, track 3.

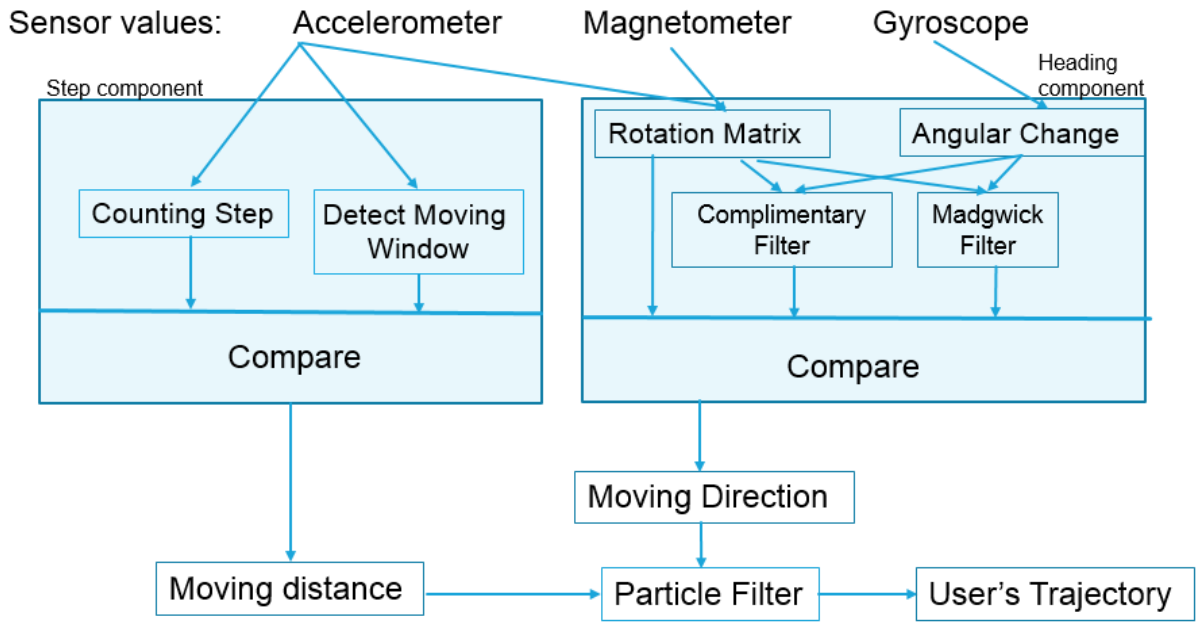


Figure 31. Our propose framework for the SHS tracking

In the second part, the SHS tracking path is combined with the Wi-Fi positioning output. The Wi-Fi data is introduced for two purposes. It provides an estimation of the starting point of the user, which is needed in the SHS system. For reducing the drifting errors from the inertial sensors tracking, the output positioning from Wi-Fi fingerprinting approach can be combined with the SHS output. Two combination approaches are tested (Figure 32). The first approach is to use directly the Wi-Fi output as a pivot point for fixing the SHS tracking part. In the second approach, we rely on the Wi-Fi signal to build an observation model, which is then combined with the above particle filter.

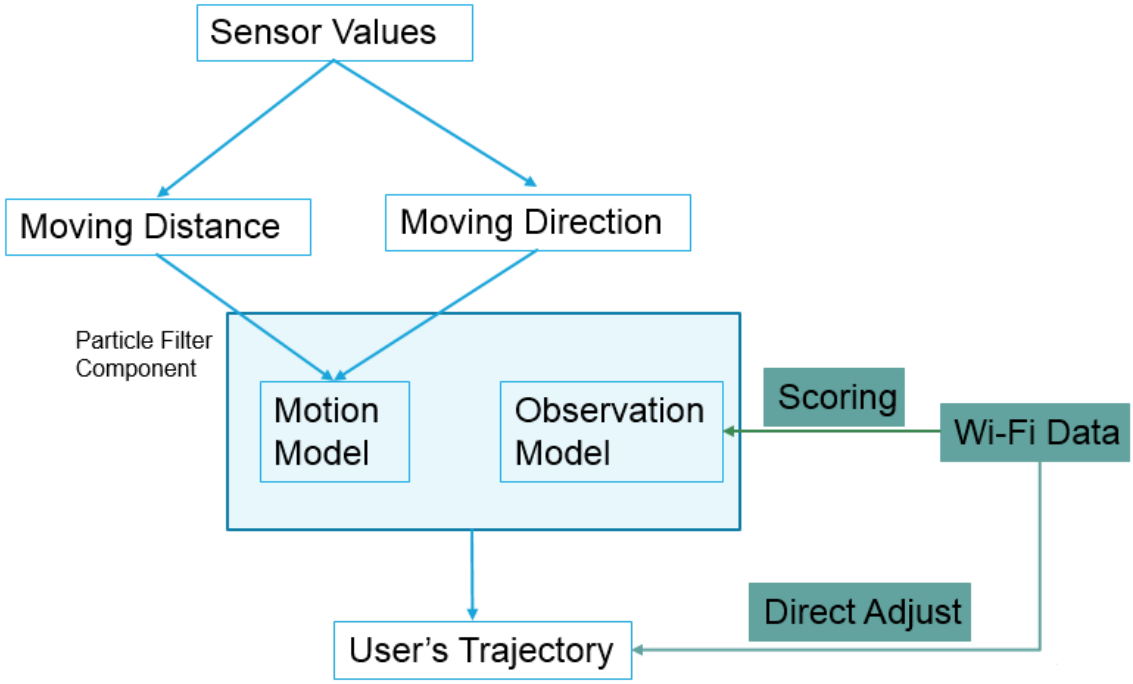


Figure 32. Combine Wi-Fi data with SHS-based tracking

4.2 Literature Review

User tracking through smartphone sensors have been widely studied in recently years. The work in [Kothari et al. 2012] provides a detailed study about the topics for several aspects. They first build a SHS system, and then later, fusing with Wi-Fi and map information to calibrate the drifting errors. For the user's moving speed, the approaches of counting step and continuous moving window detection are used. For the user's heading, the standard Complimentary Filter is used. An additional particle filter approximation is used for dealing with the noise in the moving distance and heading output. The distance results on the SHS system alone is around 5.0m ± 3.0m. By fusing with the Wi-Fi output, the error could be reduced to 3.0m ± 3.0m in the online setup. Additional map information is used to remove bad particles. In the paper, the phone's position is handled and pointed forward throughout the walk. In order to deal with other holding device positions, it is necessary to find the moving axis of the smartphone. Steinhoff and Schiele [2010] attempt to track user's movement with an in-pocket device. For deriving the user's direction, principal component analysis approach is used to find the appropriate user's motion direction. The work, though, is carried out with an IMU, not a smartphone.

Qian et al. [2015] propose a complex architecture for indoor tracing using smartphones (Figure 33). The architecture includes a calibrated module, context awareness module and a positioning error correction module. Several key challenges are addressed in this work. Acceleration data is used to classify four possible phone usages, which are Texting, Calling, In-hand and In-pocket. Specific rules are then built to find the expected step length for each usage. The work also addresses the unstable magnetic field within indoor variations. An additional local magnetic reference is added for detecting magnetic anomalies. The authors use a calibrating process for switching between a standard Complimentary Filter and a simple six-degree heading calculations. From the computed moving distance and heading, particle filters approximation and a vector graph of the area are employed for building the user's moving path.

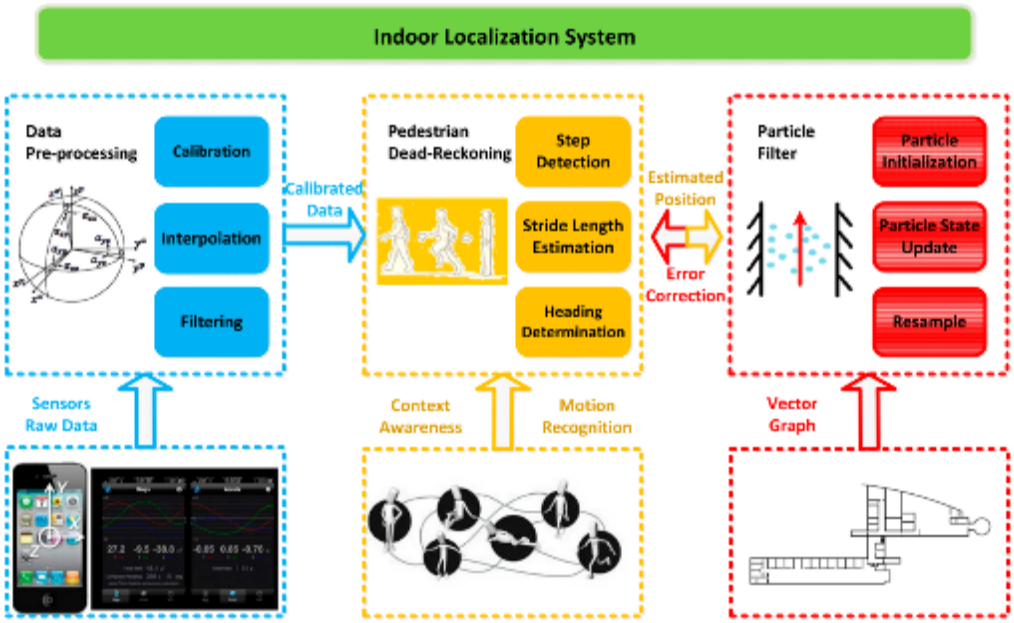


Figure 33. A proposed architecture for smartphone tracking in indoor environment by Qian et al. [2015]

Between the user's moving speed and the user's heading, the latter is considered to contribute more to the drifting errors. As reported in the work of Zhou et al. [2014], angle tracking errors could reach to 40° for over 3 minutes heading tracking. The authors propose A^3 approach, which is an opportunistic calibration algorithm, for stabilizing the heading estimation. The proposed approach can reduce the angle errors to under 10°. Figure 34 presents their results against the standard heading output of Android API. While the Android API's path is

completely lost after several turning points, the A^3 can provide the user's position over four walking cycles. The A^3 approach, however, requires a very specific tuning process and can only work with a specific type of sensor device. In the work, the ADIS1626x series of MEMS gyroscopes are selected for the experiment.

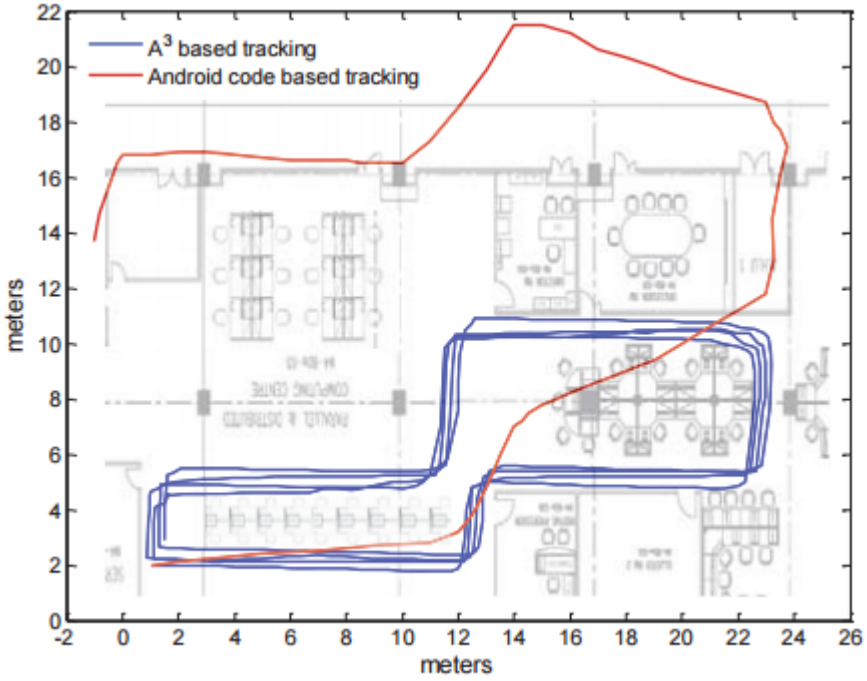


Figure 34. Tracking results comparison between direction from standard Android API (the red line) and A^3(the blue line) *[Zhou et al., 2014]*

Performances of different angle estimation methods on smartphone are reviewed in Mourcou et al. [2015]. The Madgwick Filter [Madgwick, 2010] and Mahony Filter [Mahony et al., 2008] are tested against the default filter of several smartphone's models. The result shows that all the filters have errors less than one degree for each rotation. A robot arm is used to replicate the angle movement. The experiment, however, only evaluates the reliability of smartphone's sensors for angle calculation purpose. Other sources of errors, such as from environment, are not in the discussion. Afzal et al. [2011] study the effect of noisy magnetic data or calculating the phone direction. The authors attempt to derive the heading based solely on the smartphone's magnetic sensor. Several parameters, such as sum of three mag-

netic axis values and the inclination angle, are used to measure the magnetic field perturbation. Then, a fuzzy combiner is built to compute the heading output. An error of over 16° could be seen through the testing.

Reducing the SHS errors by fusing with Wi-Fi data is a quite popular approach. Evennou and Marx [2006] improve the SHS output by the position output from KNN model. The work employ the standard particle filter approach with the motion model is derived from the inertial data. The observation model is a Gaussian kernel function, which is built on the Wi-Fi output position. The particles are then scored based on the distance between the particles' position and the Wi-Fi output position. Wall information from the environment is also employed for adjusting the bad particles. The result of fusing system is 1.53m, which is better than the accuracy of 1.86m from the SHS-based only approach. Similar observation models, which is built from Wi-Fi positioning output are used in [Wang et al., 2007]. Chen et al. [2014] propose another approach in combining SHS output with Wi-Fi output. After the trajectory is computed by the inertial sensors data, it is shifted to the Wi-Fi output positions. The authors employ a gradient descent-based search to find the most suitable shifted value. The proposed method has significant improvements in comparison with the fusion algorithm in Evennou and Marx [2006].

4.3 Standard SHS Approach

Generally, the SHS approach is split into two separated parts. The first part is to compute the moving distance and the second path is the moving heading. The accelerometer can be used to derive the moving distance efficiently. The heading is computed from the all three available sensors, including accelerometer, gyroscope and magnetic sensors. In this section, we only discuss the SHS tracking in the context of the standard phone holding position. In the standard position, the phone is handed and pointed forward when the user moves.

4.3.1 **Speed Calculation**

The user's moving distance can be calculated by using the *Counting Step* or *Moving Window* approaches. In the *Counting Step* approach, by assuming that the smartphone is handled when the user is moving, the *Z-axis* has the most fluctuated values over the 3 axes, as seen in

Figure 35. The steps are defined as the walking patterns in the Z-axis values. A round trip from a local maximization to the next local maximization could be counted as a step. When the user stops, the pattern on Z-axis become more stable and less derivation from the gravity accelerometer value. Therefore, the algorithm uses an additional threshold to differ between standing and walking interval. From the number of step, it requires the step length for computing the moving distance between walking intervals. The step length, on the other hand, could not be derived from the accelerometer data alone. In our approach, we choose to rely on the training data with the assumption that the user is the same for training and testing.

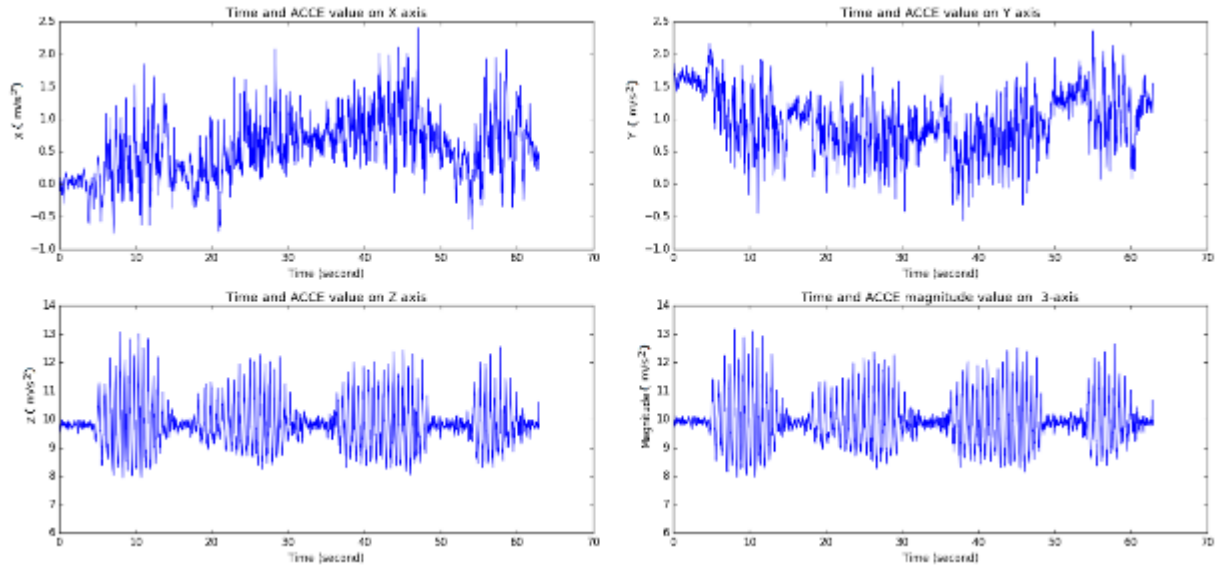


Figure 35. Accelerometer sensor values overtime when the phone is handled

The *Moving Window* approach bases on the assumption that the accelerometer values of a walking-action segment are higher than the accelerometer values of a standing-action segment. It then calculates the user's speed from the *std* of Z-axis over a fixed length of time interval. The computed *std* value is then compared with a threshold to differ whether the user moves or stands for the whole interval. Figure 36 illustrates the cumulative density function of the *std* over a fixed interval length of 0.5s. The point around 0.1 makes a significant change for the *std* pattern. The value thus can be considered as a splitting point between moving and standing actions of the user. If the pattern is categorized as moving, it is possible to assume that the user moves with an average speed. From the provide training data, the *std* threshold and the approximate user's average speed could be estimated.

Both *Counting Step* or *Moving Window* approaches are quite similar in a way that they can only infer some explicit characteristics of the user's movements. They have to depend on additional information for computing the exactly moving distance. The biggest difference between the two approaches is that the average speed parameter in the *Moving Window*, which is not as reliable as the step length parameter. For a user, the step length is usually stable in normal walking action. In a more complicated context, advantage techniques could be employed to differ the walking patterns between short and long steps. However, it requires a large number data to have a meaningful conclusion in practice.

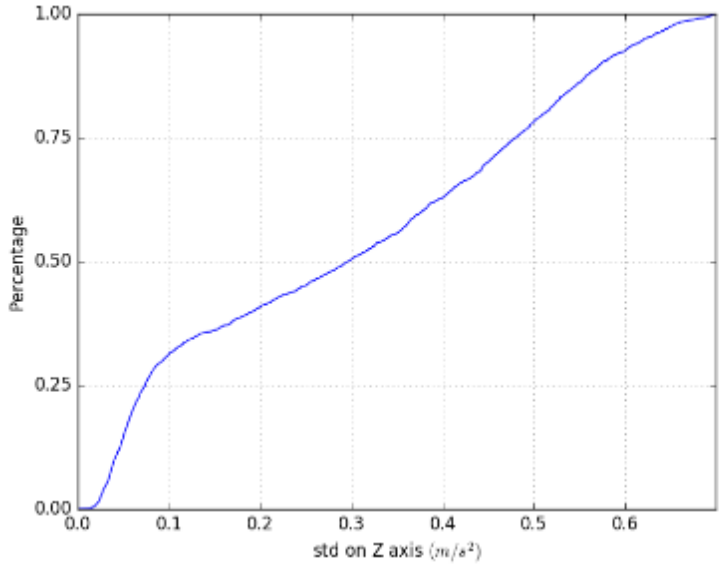


Figure 36. Cumulative distribution of std values for each interval length of 0.5 second

4.3.2 **Heading Calculation**

In a SHS tracking approach, the heading estimation usually involves the calculation of device's attitude in the global frame. Figure 37 illustrates the global frame and the phone frame. Both frames have the same origin O at the device's position. For simplicity, we could assume that a vector v in phone frame is given by ${}^{P}v = (v_x, v_y, v_z)$ and in global frame is given by ${}^{G}v = (v_X, v_Y, v_Z)$. In the global frame, the OXYZ coordinates have the Y-axis point to the North direction and Z-axis point to the Earth center. Roughly speaking, in the phone frame, the Y-axis is given by the magnetic sensor output and the Z-axis is given by the accelerometer sensor

output. The task of attitude estimation is to compute the rotation matrix which transforms coordinates in phone frame to global frame.

There are several ways to represent the rotation operator in this case such as Euler angles, direction cosine matrix and quaternion. The Euler angles method represents the phone attitude in global frames by the rotation around each axis X, Y, Z:

$$^{G}w = (\,^{G}yaw,\,^{G}pitch,\,^{G}roll) \quad (19)$$

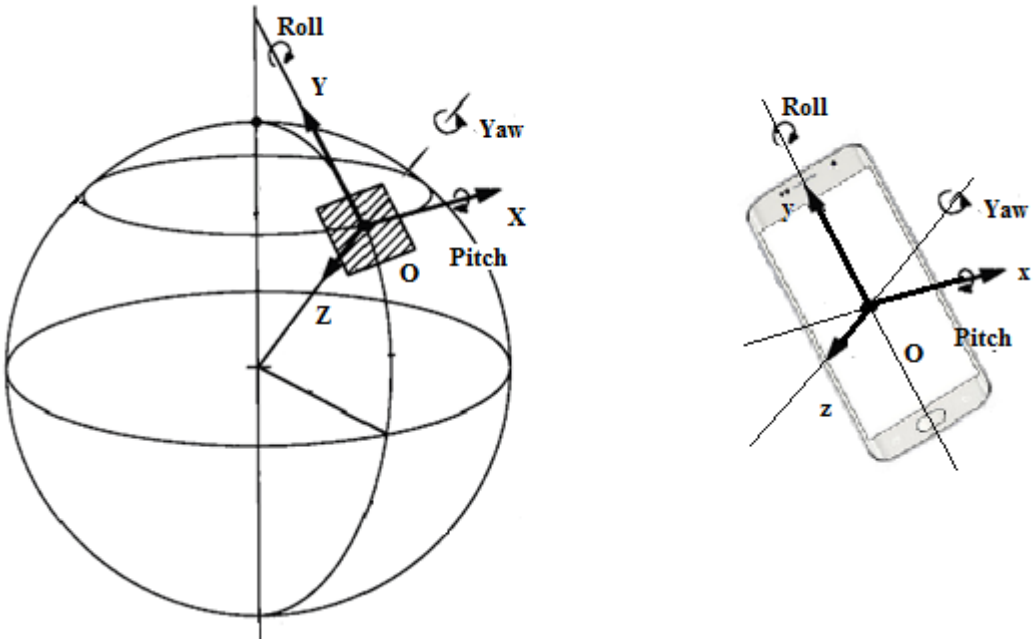


Figure 37. Global frame (left) and device frame (right)

The matrix representation uses a 3x3 R for translating a vector from phone frames to global frames. The relationship between ^{G}v and ^{P}v is expressed as:

$$^{G}v = R\,^{P}v \quad (20)$$

Quaternion representation use a unit-length vector $q = (q_0, q_1, q_2, q_3)$ as the rotation vector. The transformation between ^{G}v and ^{P}v could be expressed by:

$$^{G}v = q \otimes {}^{P}v \otimes q^{-1} \quad (21)$$

In the above equation, the notation $\otimes$ represents the quaternion multiplication operator between a quaternion and a 4-dimnesional vector. All the three representations have advantages and disadvantages in describing the device's attitude and are able to convert from one form to other forms. Diebel [2006] gives a details view in mathematical aspect of the three representations.

In our work, three approaches for computing the heading direction are tested. The first one is to calculate the rotation matrix from accelerometer and magnetic sensors directly. In a noise-free environment, the value of device's accelerometer is aligned with the Z-axis of the global frame and the value of device's magnetic is aligned with the Y-axis of the global frame. Therefore, it is straightforward to calculate the rotation matrix from the above 6 values. The second approach is a complimentary filter-based approach. The integration values of gyroscope sensor are added to the angular vector from the first methods. The third one is Madgwick Filter [Madgwick, 2010]. Madgwick Filter provides an adjustment for the gyroscope errors and magnetic distortion by gradient descent algorithm.

4.3.2.1 *Use Accelerometer and Magnetic*

Assume that at time t, the magnetic sensor value is ${}^{P}m$, and the accelerometer value is ${}^{P}a$, which are both normalized to unit length. A way of computing rotation matrix R is given in Kothari et al. [2012]. Three reference vectors ${}^{P}I$, ${}^{P}J$, ${}^{P}K$ which point along three axes in global frame are calculated as:

$$ {}^{P}I = {}^{P}m * {}^{P}a \quad (22)$$

$$ {}^{P}J = {}^{P}I * {}^{P}K \quad (23)$$

$$ {}^{P}K = {}^{P}a \quad (24)$$

$$ R = \begin{pmatrix} {}^{P}I \\ {}^{P}J \\ {}^{P}K \end{pmatrix} \quad (25)$$

4.3.2.2 *Complimentary Filter*

There are several approaches for Complimentary Filter such as Mahony et al. [2005] or Fourati [2015]. In our approach, we simply combine the angular vector from the above approach with the angular change from gyroscope.

Let the quaternion at time $t-1$ be q_{t-1}, at time t the gyroscope output is ${}^{P}\omega = (\omega_x, \omega_y, \omega_z)$. Then we have the quaternion derivative with respect to the gyroscope as:

$$\dot{q}_{\omega,t} = \frac{1}{2} q_{t-1} \otimes {}^{P}\omega \quad (26)$$

Let $q_{R,t}$ be the quaternion derived from the R at time t and λ be the fusing weight, then the angular vector is computed by:

$$q_t = \lambda q_{R,t} + (1-\lambda)\, \dot{q}_{\omega,t} * \Delta t \quad (27)$$

with Δt is the updated interval.

4.3.2.3 *Madgwick Filter*

Madgwick filter adds the measurement errors from accelerometer sensors and magnetic sensors as an additional component to the computing process of quaternion derivative. The previously calculated quaternion q_{t-1} is first used for constructing an error function of m_t and a_t. A gradient descent step is employed for minimizing the errors.

The Earth magnetic field in global frame, basing on the quaternion in previous step q_{t-1}, is:

$${}^{G}h = q_{t-1} \otimes {}^{P}m \otimes q_{t-1}^{-1} \quad (28)$$

The magnetic field is then normalized to have a horizontal axis and a vertical axis:

$${}^{G}b = [0, \sqrt{h_x^2 + h_y^2}, 0, h_z] \quad (29)$$

The gravity vector within the Earth frame has the normalize value: ${}^{G}g = [0, 0, 0, 1]$

From ${}^{G}b$ and ${}^{G}g$, we can establish a function for computing the magnetic errors and accelerometer errors:

$$f = \begin{bmatrix} q_{t-1}^{-1} \otimes & {}^{G}g \otimes q_{t-1}^{-1} {}^{P}_{-}a \\ q_{t-1}^{-1} \otimes & {}^{G}b \otimes q_{t-1}^{-1} {}^{P}_{-}m \end{bmatrix} \quad (30)$$

From the objective function f, it is able to calculate the Jacobian matrix J of f (as described in Madgwick [2010]), and then the objective function gradient ∇f as follow:

$$\nabla f = J^T f \quad (31)$$

In his work, Madgwick proposes a one-step gradient descent. The needed adjustment for minimizing the error function f from step t to $t-1$ is:

$$\dot{q}_{error,t} = \frac{\nabla f}{||\nabla f||} \quad (32)$$

The total quaternion derivative includes two components, angular rate from gyroscope $\dot{q}_{\omega,t}$ from Equation (26) and the above adjustment quaternion $\dot{q}_{error,t}$:

$$\dot{q}_t = \dot{q}_{\omega,t} - \beta\, \dot{q}_{error,t} \quad (33)$$

with β could be considered as the fusing weight between two types of quaternion derivatives. The quaternion q_t is derived from q_{t-1} and $\dot{q}_t$:

$$q_t = q_{t-1} + \dot{q}_t\, \Delta t \quad (34)$$

4.3.3 **Path Approximation with Particle Filters**

The speed and direction calculation process result in the speed v^t and the heading value h^t at time t. Let (x^0, y^0) be the starting point of the phone. We can apply a standard Particle Filter for removing noise from the calculation step. There are N particles from p_1, p_2, ..., p_N which is initialize at (x^0, y^0). At time t_1, for each particle i, the position $(x_i^{t_1}, y_i^{t_1})$ of $p_i^{t_1}$ will be compute from the p_i^t, which is at (x_i^t, y_i^t) as follow:

$$\begin{cases} x_i^{t_1} = x_i^t + v_i^t * cos(h_i^t) * (t_1 - t) \\ y_i^{t_1} = y_i^t + v_i^t * sin(h_i^t) * (t_1 - t) \end{cases} \quad (35)$$

The value of v_i^t and h_i^t will be drawn from a Gaussian distribution having mean values v^t and h^t respectively.

4.4 Combining Step-and-Heading Output with Wi-Fi Position

There are two reasons for combining SHS output and WIFI-based position output. The first one is to build a complete SHS for user tracking, one needs to find the starting position (x_0, y_0). For indoor environment, WIFI-based output provides this position with some degree of errors. The second one is the drifting error issue of the SHS output. The noise from low cost MEMS sensors would affect the speed and direction computing process. An uncalibrated gyroscope sensor, for example, could make the integration computing differ from the real value. Another source of noise is the unreliable magnetic field within the indoor area. Moreover, the errors, which are generated from those noisy values, can be added up overtime. Therefore, it is necessary to calibrate the output position of SHS-based approach after some time.

The issue between SHS output and WIFI-based output is the mismatched sampling rates. Normally, the update rate of SHS positioning is aligned with the lowest update rate from the three types of sensor, accelerometer, magnetic sensor and gyroscope sensor. The rate could be around 10 to 20 position updates per second. Additional resampling process could be added to reduce the update rate. Meanwhile, the update rate of WIFI scanning depends on a scanning cycle time. In general, a time interval from 4 seconds to 6 seconds is expected for completing a scanning cycle. Therefore, it requires some techniques to combine two different update rates.

4.4.1 Direct Adjustment based on Wi-Fi Output

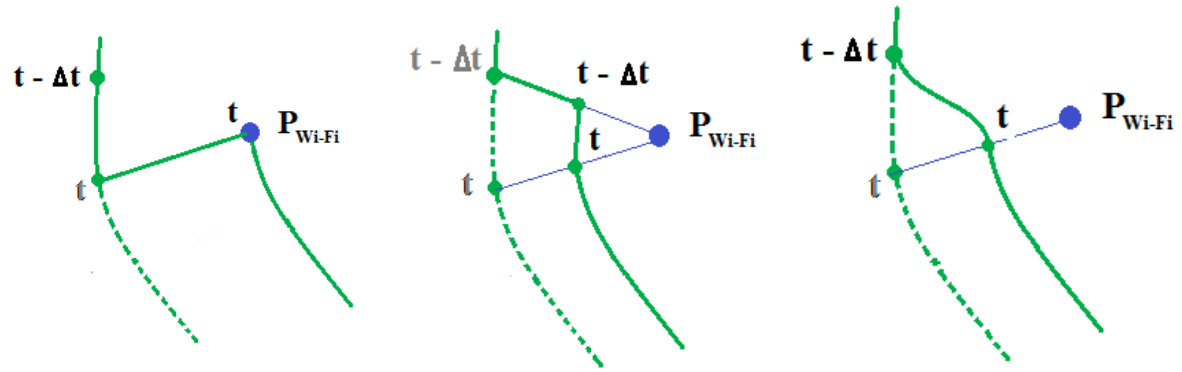
Absolute adjustment could resample the particle around the output position from Wi-Fi finger printing model. However, the errors would depend highly on the performance of Wi-Fi. The contribution from the inertial sensors is minimal in this case. The step and heading output only have effects on the path between the two consecutive Wi-Fi scans. Moreover, the tracking path is likely to be broken down into discrete segments.

In order to create smoothing combination between two methods, a fusing constant w is added. The weighting w defines how far the particles would move to the direction of the Wi-

Fi output. Besides that, there is also temporal information that should be added to the adjusting process. Assume that at step t, the Wi-Fi position output is P_{Wi-Fi}. Then, P_{Wi-Fi} only has impact to the particles within a time window Δt. The adjustment process should not update the particles, which stay out of the time window $[t-\Delta t, t]$. Let p be the old position of a particle at time t_1, before the updating. The value of t_1 should be in the range of $[t-\Delta t, t]$. The new position p^{new} after the update is calculated as:

$$p^{new} = (1 - w * (1 - \frac{t - t_1}{\Delta t}) * p + w * (1 - \frac{t - t_1}{\Delta t}) * P_{Wi-Fi} \quad (36)$$

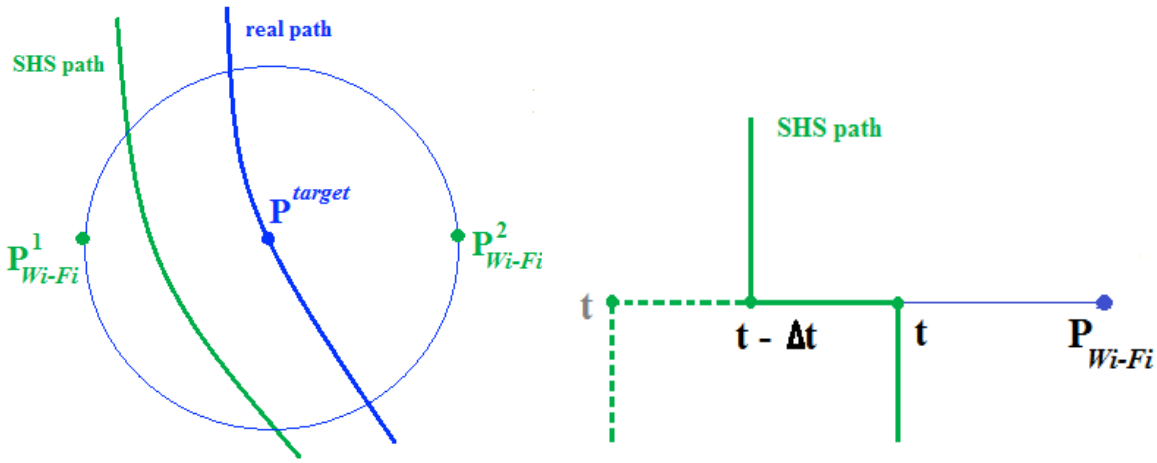
In equation (36), we add a smoothing function $f(x) = 1 - \frac{t-x}{\Delta t}$ over the constant w. The smoothing function makes the particle move slowly to the direction of P_{Wi-Fi} within the predefined time window. At the start of the interval $[t-\Delta t, t]$, the p^{new} is the same as p. At the end of the interval, p^{new} would be at the position of $(1-w) * p + w * P_{Wi-Fi}$, which is a combination between the inertial positioning and Wi-Fi position with weight w. The effect of the adjustment process is illustrated in Figure 38. The green path is the approximation path based on fusing between Particle Filter approach and Wi-Fi output P_{Wi-Fi} at time t. The green dot is the pre-adjusted path. The adjusting process would affect the green path in the interval $[t-\Delta t, t]$. The gray labels of t and $t-\Delta t$ means that the old position points which would be moved to the new position by the adjustment process. In Figure 38a, when the SHS position at time t and the point P_{Wi-Fi} are distant to each other, the path is moved to P_{Wi-Fi} which would completely remove the dependence of the sub-path after t and the sub-path before t. In Figure 38b, by adding the weight w, the contribution between the SHS path and the Wi-Fi output can be balanced. However, at time $t-\Delta t$, there is an immediately jump between the old position and the new position. The resulting path would become discontinuous. In the Figure 38c, when the smoothing function is added to the combined path, the jumping issue could be reduced.



(a) Direct Adjust without weight *(b) Use w = 0.5 and time window Δt* *(c) Add smoothing function over t*

Figure 38. Three possible ways of adjusting the SHS path based on the position of P_{Wi-Fi}

The direct adjustment approach has some drawbacks. Firstly, it depends on the relative position of P_{Wi-Fi} to the real path. Regardless the distance errors, the addition of P_{Wi-Fi} could make the path go in a wrong way. For example, in Figure 39(a), while both ${P^1}_{Wi-Fi}$ *and* ${P^2}_{Wi-Fi}$ have the same distance error to the real output P^{target} at time *t*, they have different outcomes after the adjustment process. The point ${P^1}_{Wi-Fi}$ could make the fusion path go far away the real path (blue path), thus, increase the distance errors in general. The point ${P^2}_{Wi-Fi}$ would make the fusion path closed to the real path and decrease the distance errors. Without additional knowledge, it is difficult to identify the type of behaviors Secondly, the shape of the path could be changed completely after the adjustment step. In Figure 39(b), the path could change from turn left to turn right due to different values of P_{Wi-Fi}. In general, when the output prediction from the Wi-Fi is too far from the SHS tracking path, it would control the path's characteristic. We want to address the problem by making the Wi-Fi fingerprinting model be more informative. Instead of getting only one position for each Wi-Fi scan, the output of the fingerprinting model is representing as a list of possible positions. From the direct combination, the fusion step is changed into a voting scheme based on the observation model of the particle filter step.



(a) The position of $P^1{}_{Wi-Fi}$ and $P^2{}_{Wi-Fi}$ affect the output path

(b) The shape of the path is changed at time $t - \Delta t$

Figure 39. Two cases where the adjusting process could produce unstable results

4.4.2 **Local Observation Model**

Building an observation model for the SHS tracking is a well-known approach to fix the drifting error. In our work, we base on the Wi-Fi positioning model for scoring each particle at each step. The observation model is constructed for dealing with the two weaknesses of the direct adjustment methods. For removing the strong effects of a single output position P_{Wi-Fi}, the output probability of the classifier is used. When the position problem changes from regression to classification, it is able to build a model, which predicts the likelihood of a list of points. For the training part, a grouping step is used to group near training points into a cluster. The cluster index can be used for the target input of the later learning phase (Section 3.5.3.2). In the positioning step, given an input raw RSS feature vector X and N training cluster centers, the classifier can produce the likelihood vector:

$$prob_X = \{a_1, a_2, \ldots, a_N\} \quad (37)$$

where each a_i is the probability of the output position is near the i^{th} cluster. By using the probability output vector, the Wi-Fi positioning is certainly more descriptive than only output the position P_{Wi-Fi}.

One could build a scoring function for the position p of a particle based on $prob_X$. The main idea of the scoring function is to put high score for the particles which is stay near the high

probability centers. For example, our original approach in Ta et al. [2016] focuses on the relative distance of p to three nearest centers. However, the function takes a long time to adjust the path as it is lack of discriminative capability. Because the three nearest centers are likely the same for each particle, the scores would be approximately identical. It thus makes the later resampling step likely be a random sampling.

In order to fix the low discriminative power issue, the new function attempts to give each particle a score based on its relative distance to a specific center. For the i^{th} center, let C_i be its position. C_i then adds to the score of p a value of:

$$score_{C_i}(p) = a_i * (1 - \frac{d(p,C_i) - dmin_{C_i}}{dmax_{C_i} - dmin_{C_i}}) \quad (37)$$

where $dmax_{C_i}$ and $dmin_{C_i}$ are the maximum and minimum distances from C_i to all the particles. In the equation, a particle is scored based on the distance from it to the center C_i. The value is then linearly scaled against the maximum and minimum distances. The nearest particles, which have $d(p, C_i) = dmin_{C_i}$, would get the highest score a_i with respect to C_i. The farthest particles, which have $d(p, C_i) = dmin_{C_i}$, would receive a zero score for the center C_i.

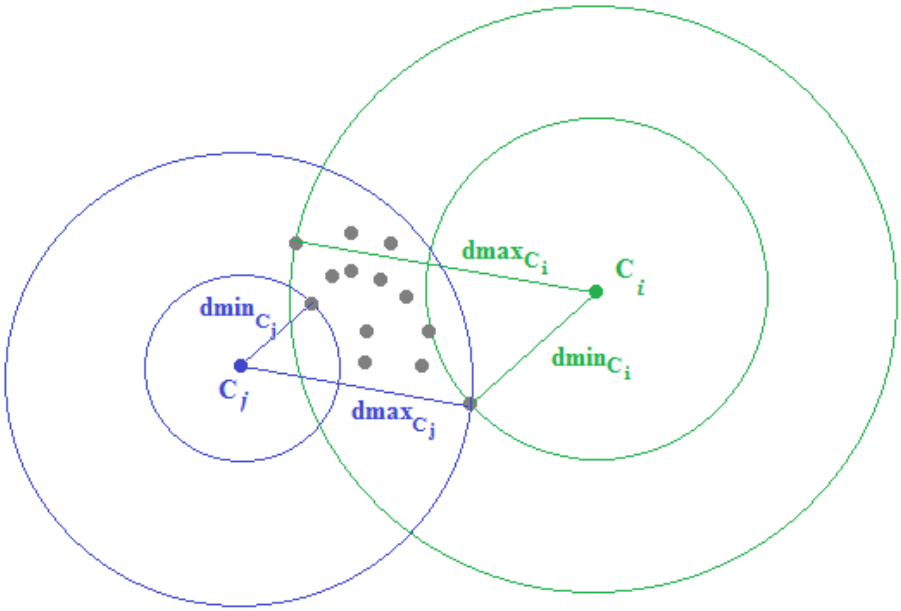


Figure 40. The scoring function for particles (gray dot) with two centers C_i and C_j. The maximum and minimum distances are user to scale the observation probability

The score of the particle *p* is the sum of each $score_{C_i}(p)$ for all *N* centers:

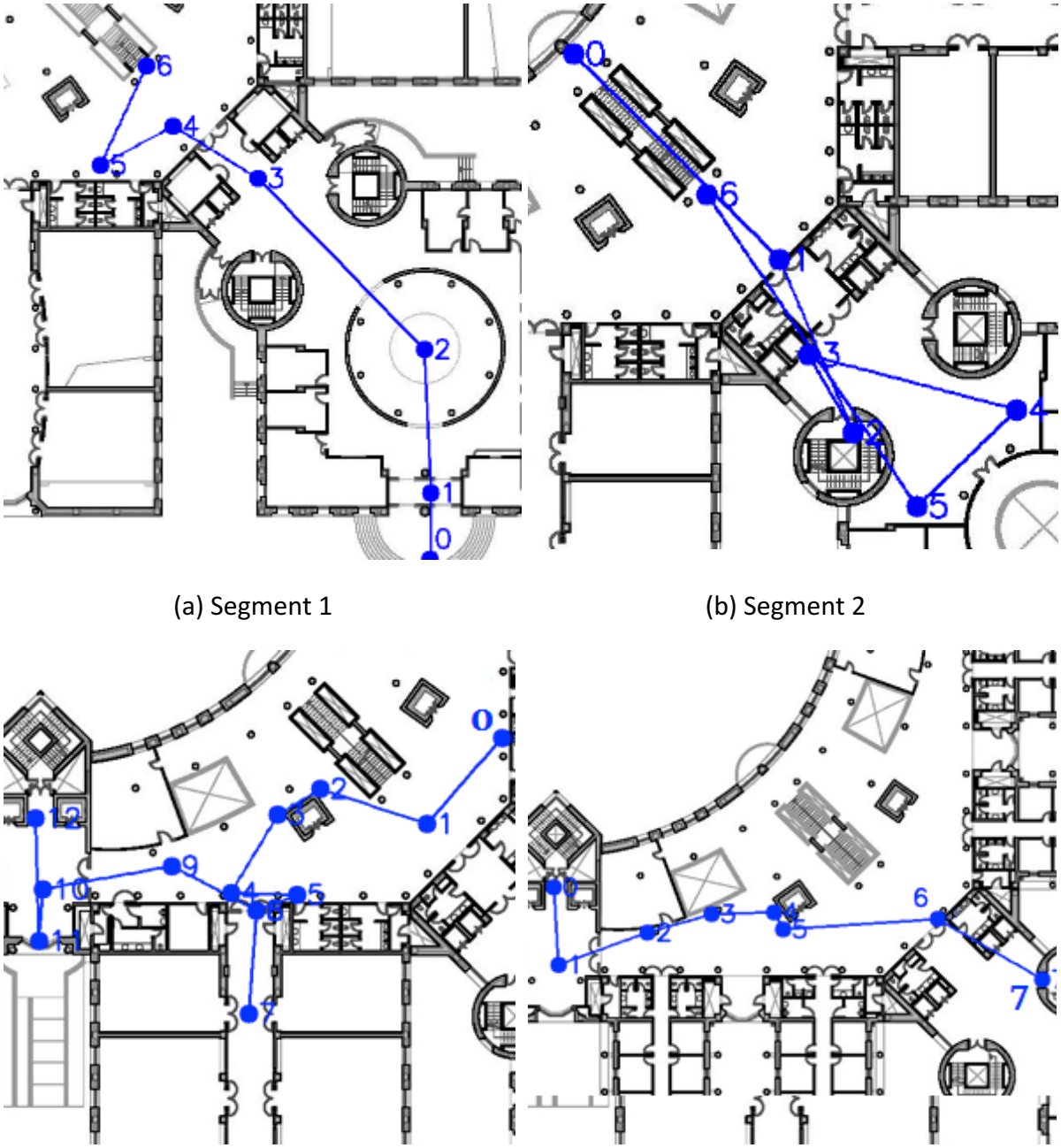
$$score(p) = \sum_{i=1}^{N} score_{C_i}(p) \quad (38)$$

Figure 40 shows how multiple centers could affect the scoring function of each particle. Based on the specific distance from the position p to each center, the contribution of each specific element in $prob_X$ could be varied. In general, a particle, which is closer to high probability centers than other particles, would have the highest score.

The scores of all particles are normalized to unit vector and then used to resample the particles in the next step. The effect of the $prob_X$ to the SHS path is limited within a time window of $[t, t + \Delta t]$, where t is scanning time of X and Δt is the effective window time of the Wi-Fi scan *X*. When there is no Wi-Fi scan available, the scores of all particles are set to a constant. Then, the next resampling step would be likely a random sampling step.

4.5 Experiments and Results

We perform our evaluation on the IPIN 2016 competition dataset. There are four test files in total. Because each test file is recorded during quite a long time, around 900s and 1500s, we split each test file into separate small segments where the user moves entirely within a floor. The small segments make the SHS tracking less affected by drifting errors. In addition, as a result from Wi-Fi floor classification, the changing floor point could be detected easily and thus reset the localization process. Several segments then are selected for testing our approaches. The specific selected segments are presented in Figure 41. Each segment is recorded with two different devices, Samsung Galaxy S3 (S3) and Samsung Galaxy S4 (S4). There are eight testing segments in total.



(a) Segment 1 (b) Segment 2

(c) Segment 3 (d) Segment 4

Figure 41. The four selected testing segments from test data, with the number represent the checkpoint visiting order of the user

4.5.1 **Moving Distance Error**

For computing the moving distance, The *Counting Step* or *Moving Window* methods requires additional input values, which are the step length and the moving speed. We use the provided training data for calculating the necessary parameters. Each users' moving path in the data

are recorded by clicking the specific checkpoint when it is reached. There are two assumptions. The first assumption is that the user moves directly between two checkpoints. With this assumption, it is possible to calculate the moving distance from the two checkpoints' coordinates. The second one is that both the users' step length and users' moving speed do not change much from one checkpoint to the next checkpoint. For a stable estimation, we remove the segment when there is a floor change.

To determine the step length in the *Counting Step* approach, we rely on each sub-moving part from one checkpoint to the next checkpoint. A threshold for local maximum on Z-axis is used. The standing-action and walking-action are separated by this threshold. From the calculated the moving distance and the number of steps between two consecutive checkpoints, the approximated step length of the specific moving part could be calculated.

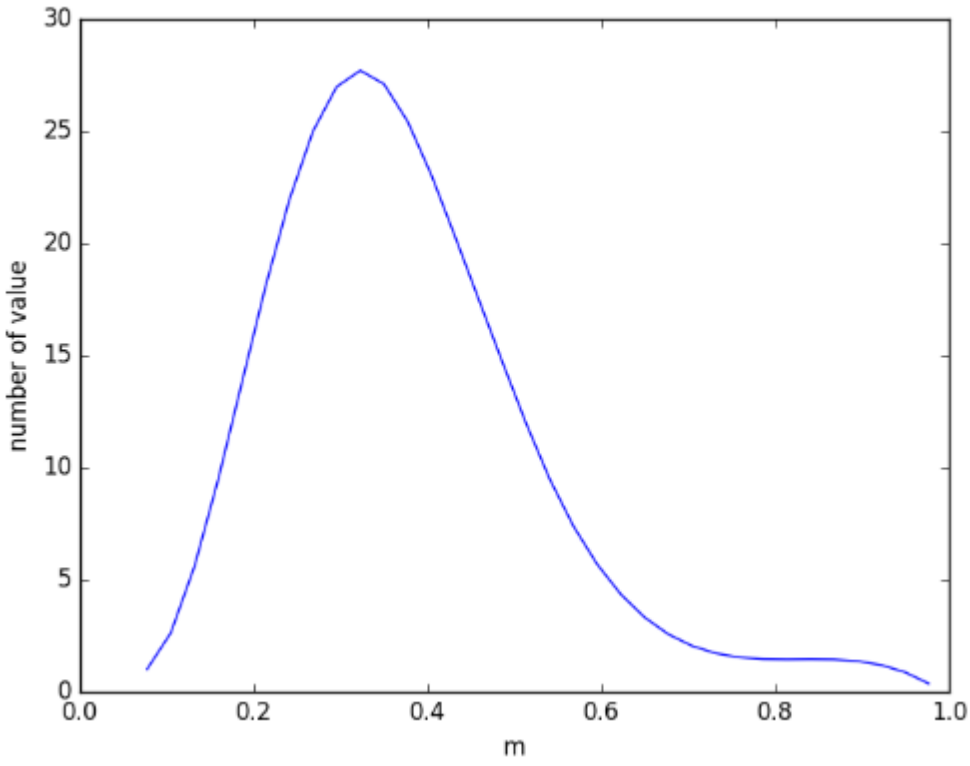


Figure 42. The step length distribution over the training data set

Figure 42 illustrates the distribution of step length with the counting step approach. By removing the noisy tail of the greater domain, the step length could be seen as a normal distribution with the mean value of $\mu_{StepLength} = 0.36m$ and the *std* value of $\delta_{StepLength} =$

$0.12m$. It can be seen clearly that the value of step length varies a lot through the training data. Therefore, it is reasonable to take the step length mean value for computing the distance.

To determine the average moving speed in the *Moving Window* approach, the full moving paths are also split into segments between each consecutive checkpoint. For each segment, a moving window with length of 0.5s is used for identifying the *std* on the Z-axis of the accelerometer. The computed *std* is then compared with a threshold to find out whether the user was moving or standing within the specific time window. Similar patterns can be seen with the average speed (Figure 43). The average speed values also vary significantly through different walking segments. Its central is around 1.0m, with a little skew to the right of the central. There is a tail range of the domain greater than 1.5m, which should be removed. It can assume that the speed average has a normal distribution with the mean of $\mu_{AvgSpeed} = 0.97m$ and the *std* of $\delta_{Speed} = 0.28m$.

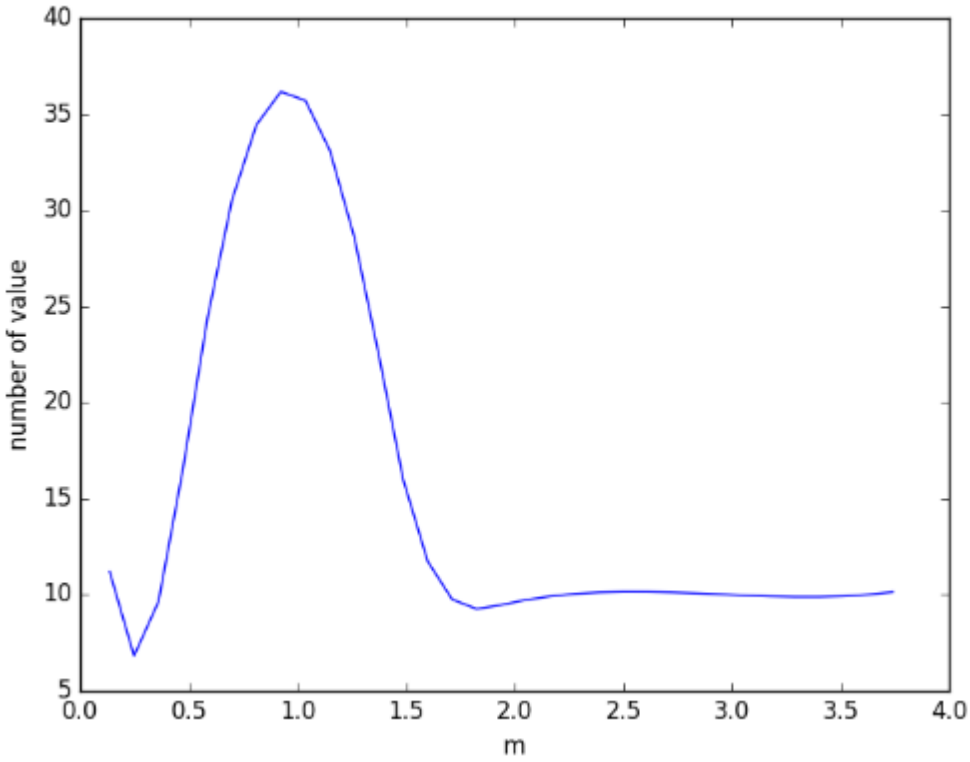


Figure 43. The speed average distribution over the training data set

After finding the average of step length and average speed, we test the calculated values against the eight selected sub-paths. The test sub-paths are also divided into several walks. The user is assumed to move in a straight line from one checkpoint to the next checkpoint. The ground truth length is calculated from the two checkpoints coordinates. The length is then compared to the moving distance output from the *Step Count* method and the *Moving Window* method.

Figure 44 shows the cumulative distribution of the absolute errors from two methods. The *Counting Step* method performs better than the *Moving Window* method. There is an exception at the highest regions, where the *Moving Window* method is slightly better than the *Counting Step* method. The maximum error of *Moving Window* is around 6.0m while the *Counting Step* reaches 8.0m. The relative error with respect to the moving distance is 8% per meter distance for the *Counting Step* approach and 12% per meter distance for the *Moving Window* approach. Both approach suffers a similar error pattern. The reason is the usage of fixed constants for calculating the distance, which are the average step length and the speed average. In fact, the two values are varied through the entire data. However, with 3.0m error at percentile 90th, it is acceptable to select the Counting Step for the later phase of particle filter approximation.

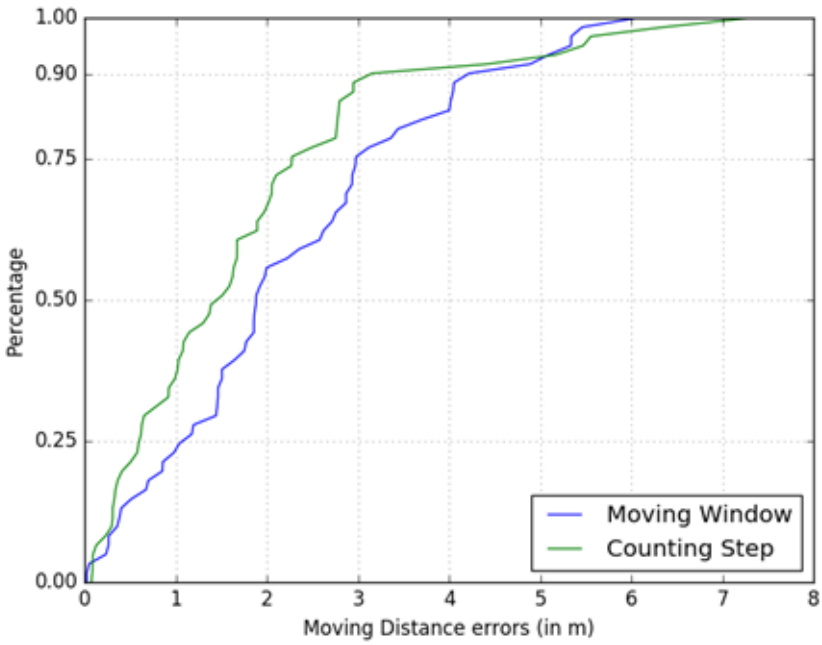


Figure 44. The absolute errors distribution of moving distance of two methods

4.5.2 **Heading Calculation Error**

Three approaches, Rotation Matrix, Complimentary Filter and Madgwick Filter are used for calculating the phone's orientation. We assume that the azimuth axis of the phone is the same as the user's moving direction. The heading ground truth is then calculated based on the two consecutive checkpoints in a similar way of the moving distance ground truth from the above section.

Several pre-processing steps for the raw sensors' data are used. First, it is necessary to down sampling the entire three data streams to a proper update rate together because the raw sensor values are provided with different updating rates. Usually, gyroscope sensor has the highest rate which is around 150Hz - 200Hz. Accelerometer sensors and magnetic sensor are updated with the frequency of around 50Hz. In this case, all the three sensors are resampled at 50Hz. Second, the magnetic sensor of the smartphone is highly affected by hard and soft iron noise. In practice, at the beginning, a calibration should be carried out to reduce the errors. However, in the provided testing data, there is no such process. In our processing step, we apply a zero-mean normalization step for the magnetic sensor data to reduce the effects of the noise.

Figure 45 shows the distribution of absolute errors for the three selected approaches, the accelerometer and magnetic heading (*AccMag Heading*), *Complimentary Filter* and *Madgwick Filter*. Both the *AccMag Heading* and *Complimentary Filter* have a closed performance. Meanwhile the *Madgwick Filter* performs slightly worse than that. The gyroscope integration component in *Complimentary Filter* could improve the results from the standard *AccMag Heading* output. It also reduces the maximum angle error. In case of *Madgwick Filter*, the gradient fixing step introduces more noise in the output heading.

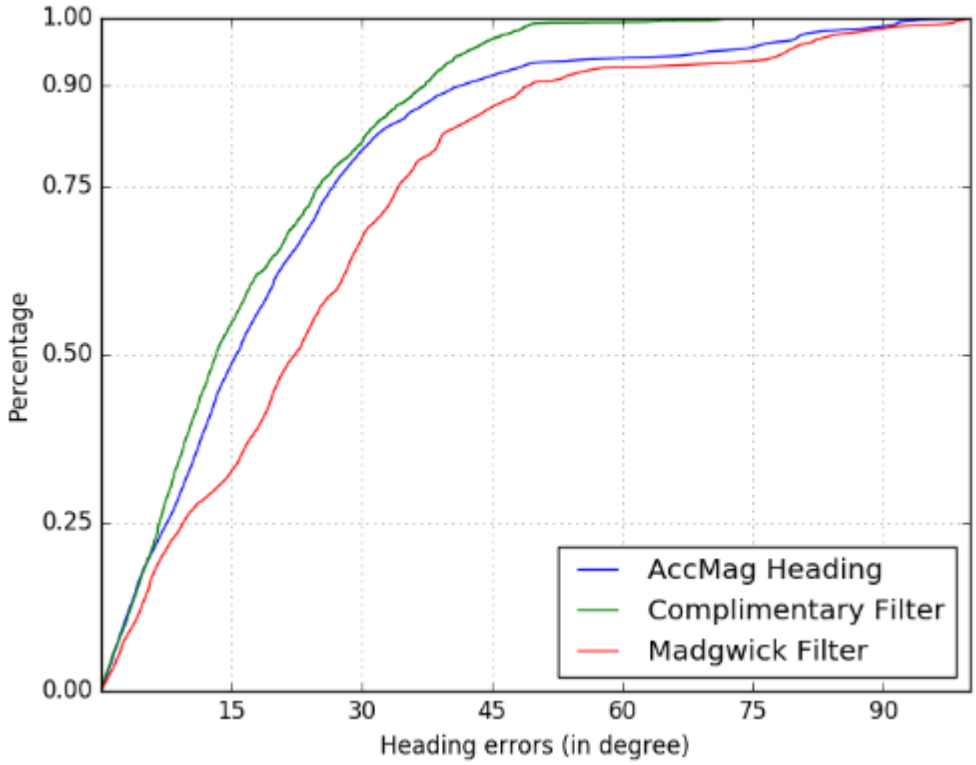
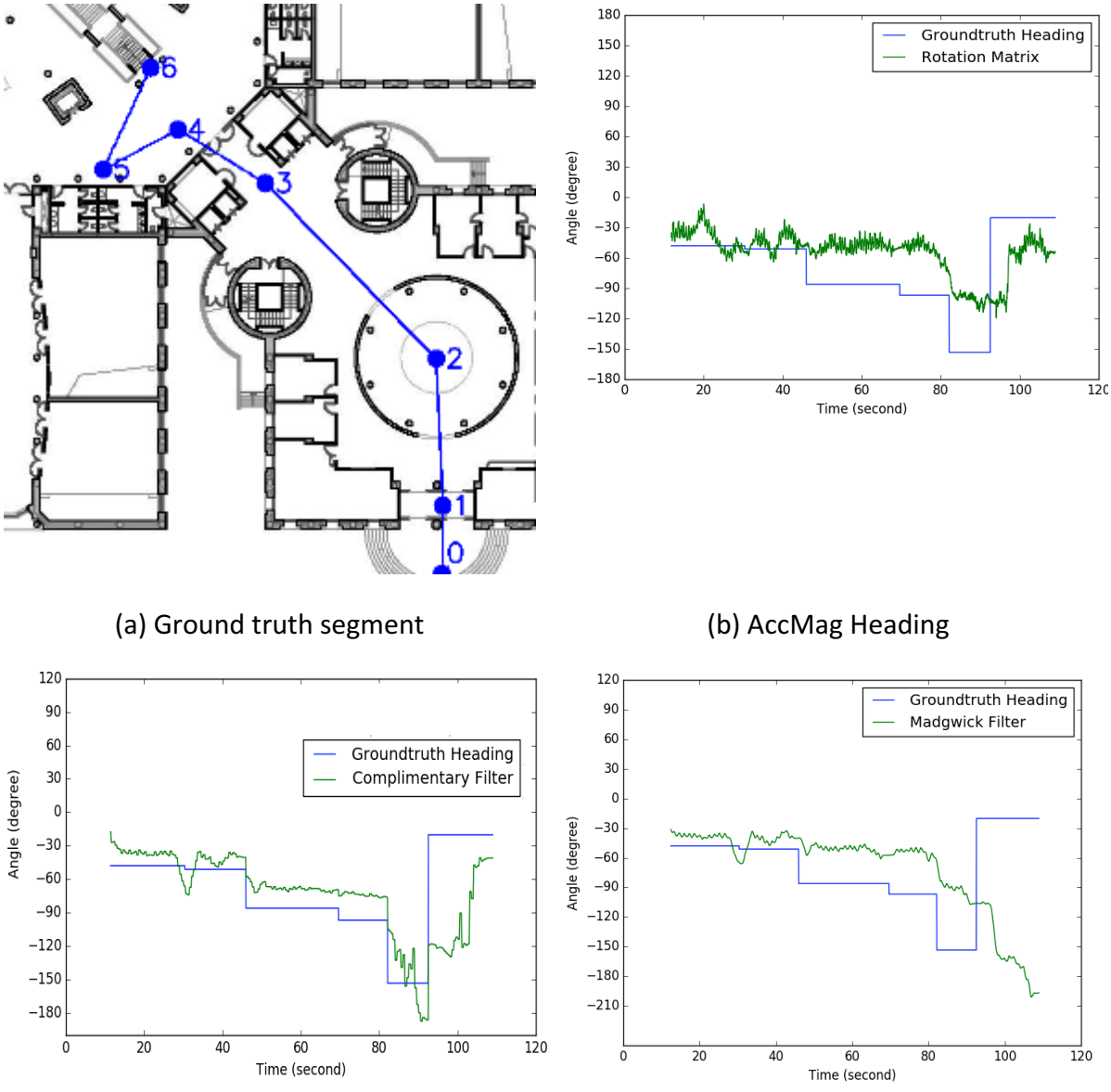


Figure 45. The cumulative distribution absolute errors of three approaches

Apart from the noisy magnetic sensors and gyroscope sensors, there are a number of other reasons which contribute to the high heading errors over all. Firstly, some turning angles does not reflect in the ground truth calculation. Throughout the collecting data process, there are several ambiguous parts where the user needs to change the direction slightly to avoid obstacles. The second reason is that the turning moment in the ground truth calculation is considered to be happened immediately. When the user marks his arrival at a new checkpoint, the angle is changed to the next direction after that. The new angle should appear in the sensor data stream within the interval of several seconds. It depends on the angular speed of the user.

The Figure 46 plots the respected path and the calculated heading for the segment 1 of the phone model Samsung Galaxy S3. Three methods are plotted as well as the ideal heading output. From the plot, it can be seen that the *AccMag Heading* and *Complimentary Filter* can keep up with the user's changing direction to some extents. *Complimentary Filter* is more stable against the noisy sensor data. However, it has an issue for the large turning point at

the checkpoint number 5. The output of *Madgwick Filter* approach is quite unstable against the direction change throughout the path. Nevertheless, there is also the checkpoint 1 in the path, where all the algorithms recognize a pulse of changing direction which does not appeared in the ground truth heading direction. Overall, the three methods are affected by small drifting angles, around less than 10° in average. Using a simple zero-mean normalization is unable enough to fix the indoor magnetic abnormal issues in this case.



(a) Ground truth segment

(b) AccMag Heading

(c) Complimentary Filter

(d) Madgwick Filter

Figure 46. The moving path (a) and three heading calculation methods: AccMag Heading (b), Complimentary Filter (c) and Madgwick Filter (d)

The Figure 47 plots the respected path and the calculated heading for the segment 3 of the phone model Samsung Galaxy S4. The path of segment 3 is more complicated than the segment 1. There are a total of 13 checkpoints with large turning angles. The path also goes through places near the elevators. From the heading output, it can see clearly the highly fluctuation when the user moves near the elevators from checkpoint number 1 to checkpoint number 3. There are also two 180° turning points at checkpoints number 7 and number 11, which adds up to a high variations of errors.

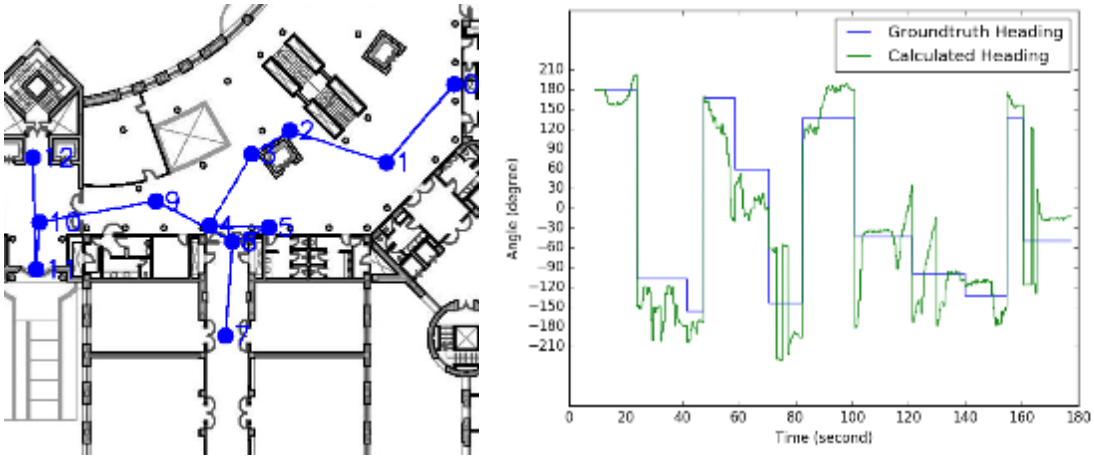


Figure 47. The moving path of testing segment 3 (left figure) and the Complimentary Filter output (right figure)

Overall, the mean heading errors of *AccMag Heading* and *Complimentary Filter* stay around 30°. As reported by several works, the type of high errors is expected in uncontrolled environment. General methods as *Complimentary Filter* and *Madgwick Filter* have a limited capability to fix the errors from the noisy sensors output of the smartphone. In the future works, opportunistic calibrating methods such as in Qian et al. [2015] and Zhou et al. [2014] could be employed for reducing the errors.

4.5.3 **Path Construction with Particle Filters**

Basing on the results of the moving distance and phone's heading, a particle filter approximation is used to create full moving path of each testing segment. We select the *Step Counting* method for calculating the distance and *Complimentary Filter* for calculating the heading. Again, as the individual results of two methods are not time aligned, an additional step of

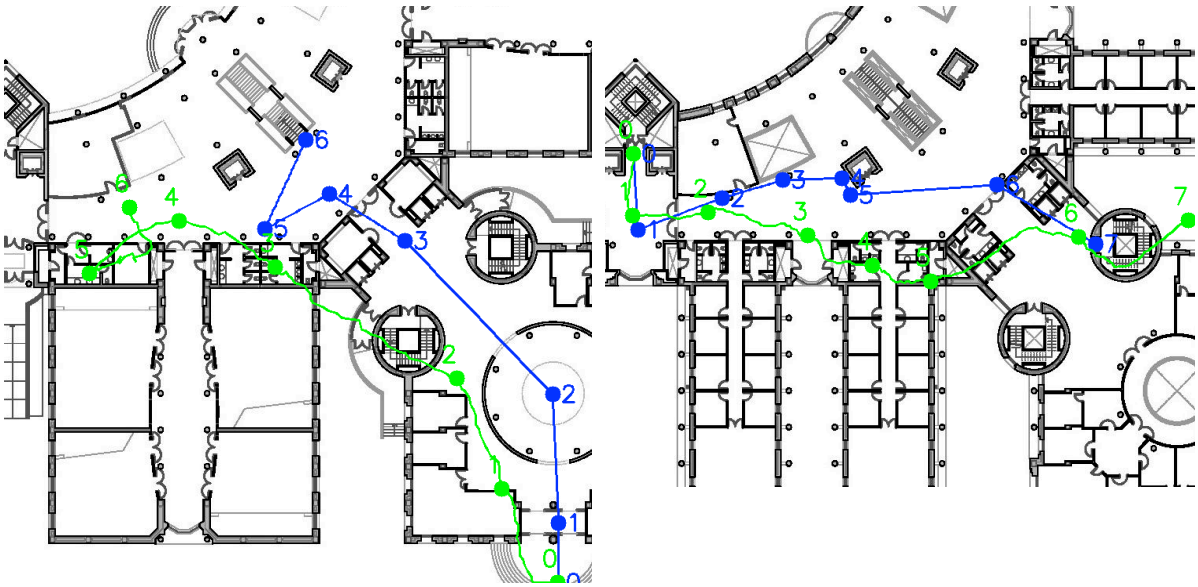
resampling both heading and moving distance values is required. We use the interval of *0.5* second as the new updating interval of the both values. At this stage, we assume that the starting point of each path is known. The initial particles therefore are randomly created around the starting point. The approximation step used 5000 particles. Each particle is then updated with the calculated moving distance and heading values. The Gaussian noise is added to model the calculation errors. With moving distance, the mean error per meter of the *Counting Step* method is used as the *std* of the Gaussian distribution. With heading, the percentile 50^{th} error of the *Complimentary Filter* is used for the same purpose. After each step, the output position is taken as the mean position of all the available particles.

Table 15. Distance errors for each test segment file

Segment	Phone Model	Distance error
Segment 1	S3	10.70m $\pm$ 5.44
Segment 1	S4	10.29m $\pm$ 6.34
Segment 2	S3	6.82m $\pm$ 3.56
Segment 2	S4	9.71m $\pm$ 5.11
Segment 3	S3	6.55m $\pm$ 3.61
Segment 3	S4	7.16m $\pm$ 3.19
Segment 4	S3	6.61m $\pm$ 3.56
Segment 4	S4	**5.59m $\pm$ 3.18**

Table 15 presents the mean distance errors and its *std* across the eight testing sub-path. The best segment results are *Segment 4*, with the mean error of 6.61m and 5.59m for the S3 phone and S4 phone, respectively. *Segment 1* has the worst results with the errors over 10m for both tested phones. It is interesting that the most complicated segment, *Segment 3*, has a better performance than the simple path, *Segment 1*. Figure 48(a) represents the worst mean distance errors path and Figure 48(b) represent the best mean distance errors distance errors path. For *Segment 1*, it is highly affected by the drifting errors. While the heading errors stay around 10° for a large part of the segment, the moving straight action of the user makes the distance errors grow overtime. In case of *Segment 4*, it is as simple as *Segment 1*. The

heading errors make the tracking path diverge from the truth path from checkpoint 2. However, due to the path's characteristics, the distance errors are kept around 6m. The two paths stay closer at the end of the segment (checkpoint number 6 and 7).



(a) Segment 1, S3 Phone (b) Segment 4, S4 Phone

Figure 48. The blue path is the real user moving path and the green path is the approximation path by particle filter method

4.5.4 **Fusing with Wi-Fi Data Stream**

From the previous results, a Wi-Fi fingerprinting model is fused for with the output. The Wi-Fi fingerprint model is not only used to calibrate the drifting errors but also to localize the user at the starting point. The starting position of the user corresponds to the first completed scan cycle of the Wi-Fi data. The calculated starting point, however, is not guaranteed to be accurate because there are also some degrees of errors for the Wi-Fi fingerprinting positioning model. When the Wi-Fi output position is not present at the start of the testing segment, an additional particle filter is carried out to construct the previous missing part. As there is no additional Wi-Fi data for the missing part, the particle process would use the moving distance and heading output.

From the first available Wi-Fi output position to the end of the testing segment, the tracking progress is straightforward as more Wi-Fi information would be available. From the results

shown in Table 14 of combining Wi-Fi fingerprinting models, we select the *Remove Noisy Models* for the position output and *Classifier Models Only* for building the observation model. The *Remove Noisy Models* has a mean distance error of 5.12m while the *Classifier Models Only* reaches around 6.00m of mean distance error. Nevertheless, the latter approach is able to output the prediction as a list of probability, which is mandatory to build the observation model in particle filter step. In the *Direct Adjust* method, the Wi-Fi output positions would be combined with the SHS output part. In Equation (36), the fusing weight *w* is selected as 0.5, which equalizes the contribution from the Wi-Fi data and SHS data. The window effect length is selected at 5s, which is the approximated time of a Wi-Fi scan cycle. For the observation model, with each input scan *X*, the nine probability outputs $prob_X$ from all the classifier models are used. Because the cluster centers are identical across all the configurations, it is able to take the sum of all the nine probabilities for each center. After that, the sum probability $prob_X^{sum}$ is normalized to unit vector. The window effect is the same as the *Direct Adjust*.

Table 16 shows the results of the two approaches together. The Wi-Fi only output serves as a reference solution. For constructing the Wi-Fi only output path, simple linear interpolation is used for each pair of consecutive complete Wi-Fi scans. Compared with the results in Table 15, the fusing approaches could out-perform the inertial-sensors based approach. Without the need of starting point, both fusing approach have better mean distance errors and *std*. In general, the two fusing-based approaches depend mainly on the performance of Wi-Fi fingerprinting model as it provides the first output position for the user's starting point. From the results in Section 3.5.3, it is shown that the Wi-Fi fingerprinting could reach a low distance error of 5m in average. However, it also has to deal with the high distance errors, for example the errors in Segment 1 of Phone Model S3. By combining with the SHS output, these high distance errors could be reduced significantly. In addition to that, the combination approach provides more stable tracking results across the tested segments. The linear approximation step in *Wi-Fi Only* setting only outperforms the fusing-based approaches when the fingerprinting outputs have good tracking results, for example the result of Segment 1 with S4 phone model. When the errors increase, information from the inertial sensors provide more useful data of the user's movement between two consecutive Wi-Fi scans. Among the fusing-

based approaches, the *Use Observation Model* performs slightly better than the *Direct Adjust*. Figure 49 presents the distribution of distance errors of the three approaches in Table 16. The low error region, which is less than 5m distance error, illustrates the tracking parts where the *Wi-Fi Only* approach output positions give good tracking outputs. The three approaches have a closed performance in the low regions. However, in the high error region, the two fusion-based approaches outperform the *Wi-Fi Only* significantly.

Table 16. Distance errors for each test segment file

Segment	Phone Model	Wi-Fi Only	Direct Adjust	Use Observation Model
Segment 1	S3	15.18m ± 12.62	7.26m ± 4.25	**4.74m ± 1.79**
Segment 1	S4	**4.15m ± 2.48**	4.51m ± 2.72	4.78m ± 2.95
Segment 2	S3	6.48m ± 3.55	5.23m ± 2.17	**4.04m ± 2.00**
Segment 2	S4	3.84m ± 1.44	4.05m ± 1.57	**3.81m ± 2.09**
Segment 3	S3	6.79m ± 5.21	5.25m ± 3.17	**5.08m ± 3.10**
Segment 3	S4	5.15m ± 3.02	**5.13m ± 3.05**	5.35m ± 3.05
Segment 4	S3	15.95m ± 7.27	8.73m ± 2.50	**7.48m ± 1.64**
Segment 4	S4	5.73m ± 3.70	5.60m ± 3.29	**4.06m ± 1.73**

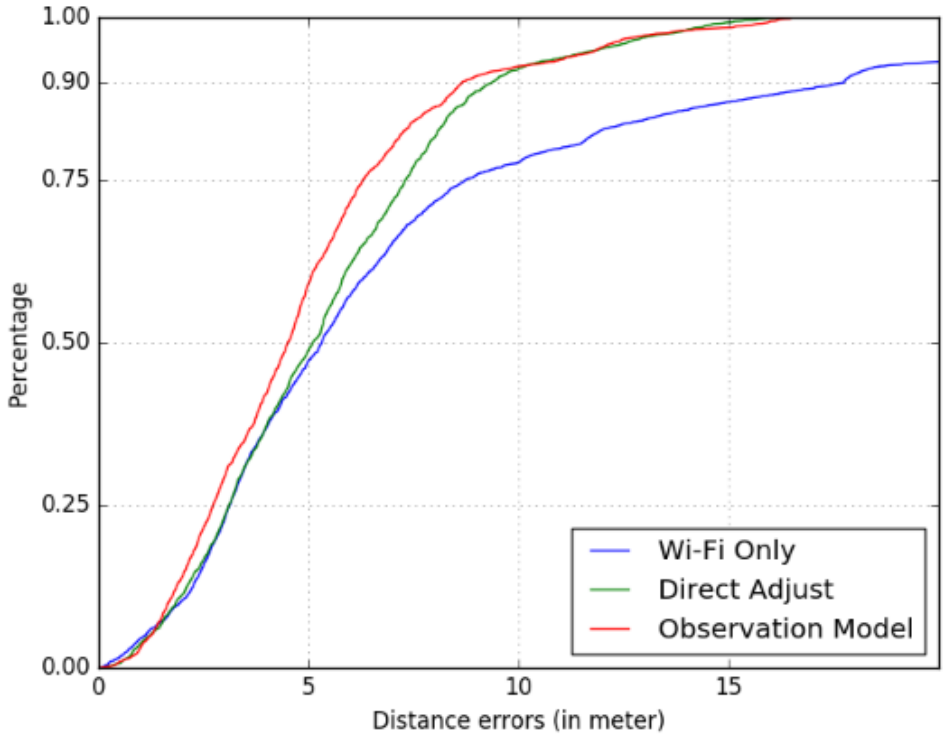
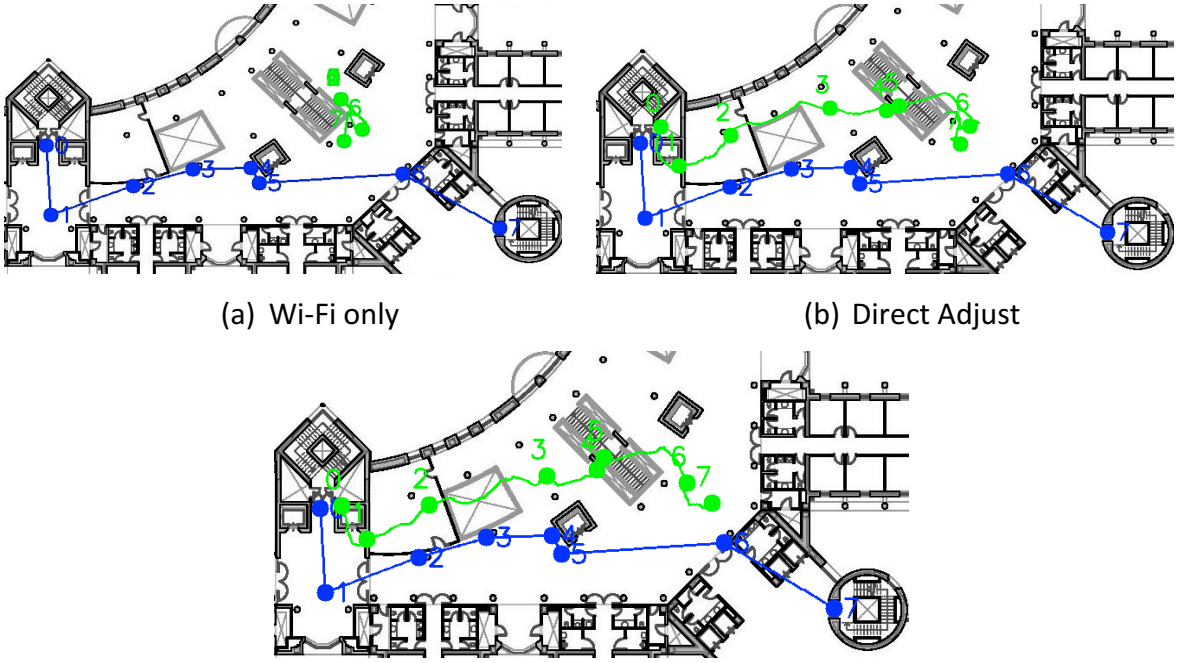


Figure 49. The distribution of distance errors of the three approaches present in Table 16

The biggest improvements are the capability of fixing the noisy fingerprinting errors in Segment 1 - S3 and Segment 4 - S4. For example, Figure 50 illustrates the in Segment 4 - S3. Nearly half of the path is missing Wi-Fi data. The missing Wi-Fi scanning causes the errors increase to as high as 15m in the *Wi-Fi Only* approach. The approach provides the tracking outputs around a small area (Figure 50a). However, both fusing methods are able to reduce those errors to around 8m (Figure 50b and Figure 50c). The sub-path has 120 seconds but has only 6 complete Wi-Fi scans. There is a lagging interval of around 90 seconds at the beginning of the segment which is needed a long back tracking process. The *Use Observation Model* approach has a better approximation because it is more independent of the Wi-Fi output.



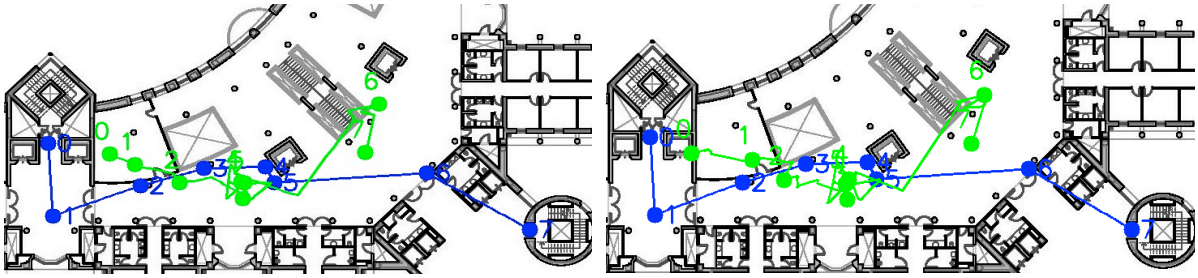
(a) Wi-Fi only (b) Direct Adjust

(c) Use Observation Model

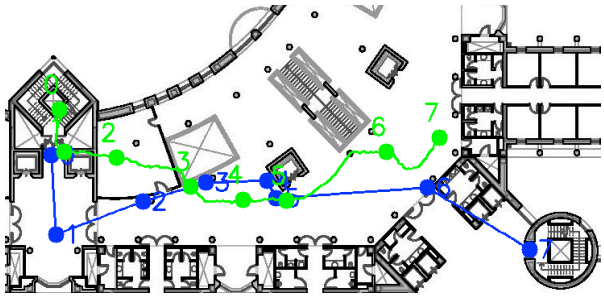
Figure 50. Tracking results on the Segment 4, with Samsung Galaxy S3 model. The blue path is the ground truth path and the green path is the approximation path

Figure 51 illustrates the same path with a good number of Wi-Fi scans. The complete Wi-Fi scan are distributed uniformly along the moving path. The *Wi-Fi only* approach has a mean distance error of 5.73m with a *std* value of 3.70. Although the method has a good positioning result, the linear approximation path results in a completely noisy line. It is difficult to identify critical points such as turning point from the approximation path (Figure 51a). Providing a

good tracking results, the raw Wi-Fi output positions is then capable for improving the SHS output in both fusion methods.



(a) Wi-Fi only (b) Direct Adjust



(c) Use Observation Model

Figure 51. Tracking results on the Segment 4, with Samsung Galaxy S4 model. The blue path is the ground truth path and the green path is the approximation path

In the *Direct Adjust* method, it results in a similarity performance, which has a distance error of 5.60m. Because of a large numbers of Wi-Fi output positions, the SHS path is the same as the linear approximation path. Instead of following the heading direction from the inertial sensors data, the tracking path would move from one Wi-Fi output position to the next Wi-Fi output position. We could reduce this effect by changing the value of the fusing constant w in Equation (36). The second method of fusing, *Use Observation Model*, results in a much better path approximation (Figure 51c). The structure of the SHS approximation path is preserved mostly and aligned with the probability output from the Wi-Fi classifier models. It results in a mean distant errors of 4.06m with a *std* value 1.73m. Compare with the result of using particle in Table 15, the fusing-based approach outperforms the standard result in both distant errors and the stability of the output. Moreover, the fusing method do not need to know the starting point for initializing the particles. In Figure 51c, the starting point of the approximate path is miss-identified to stay inside the stair (the point is numbered with 0).

However, there are some other regions where the Wi-Fi fingerprinting output could add confusion to the SHS path. In the region near the point number 6 and 7, the Wi-Fi output suggests that the user would stay in the area while the real path go to the stair for moving to the next floor.

4.6 Summary

In this chapter, we study several aspects of the SHS tracking for indoor environment. While the moving distance tracking results are quite stable, the heading output is clearly affected by noisy data. The errors in the heading contribute the tracking results mostly. In the later phase, by combining with the Wi-Fi fingerprinting output, the drifting errors could be reduced significant. The *Direct Adjust* is a weighted-combination between the inertial-sensors-based tracking and Wi-Fi fingerprinting based tracking. In the *Use Observation Model* combination method, we focus on finding an agreement between the two tracking approaches in a more general way. The Wi-Fi data is incorporated into the tracking path through the observation model. From the experiment results, the *Use Observation Model* method provides more stable tracking paths than the *Direct Adjust* method.

There are several limitations in the works. For example, the phone's position is restricted to be handled and pointed forward. In a more general phone handling setup, the standard Complimentary Filter is unable to match the device's heading to the user's heading. Besides that, complicated walking patterns, such as moving in elevators and stairs, are not considered. Apart from these limitations, the performance of two fusion approaches depends on several parameters, which control the contribution between the SHS output and Wi-Fi output. We would like to address the issues in future works.

Chapter 5 Using Bluetooth-based Distance for Improving Wi-Fi Fingerprinting Tracking

5.1 Introduction

In a GPS denied environment, Wi-Fi and Bluetooth could be considered as alternative wireless-based solutions for positioning purpose. Novel Wi-Fi-based positioning methods on smartphones can find the position by scanning the available Wi-Fi access points in the surrounding environment. Due to the unreliable characteristics of the Wi-Fi signal propagation in indoor environment, it is expected that there is a mean distance error of around 5m (results from Section 3.5.3). In case of Bluetooth-base positioning, the Bluetooth technology available on smartphone nowadays is much similar to the Wi-Fi technology in terms of underlying radio physics characteristics and application level. Therefore, it is possible to create a similar positioning system similar to the Wi-Fi ones. However, the Bluetooth communication range is smaller than the range of Wi-Fi. Therefore, it requires a higher number of static beacons for deployment in a large area. Moreover, the standard Bluetooth usage on smartphones only focuses on pair-to-pair communication. Thus, an additional network infrastructure could make the positioning system costlier.

Apart from the short-range communication, the Bluetooth protocol on smartphones is designed to focus on pair-to-pair communication, which can be employed in the context of multiple users in the area. When each user moves with his/her smartphone within a public area, it is able to let the smartphone's Bluetooth stay at visible mode. Other smartphones then could acquire the Bluetooth RSS signal value from the Bluetooth enquiry process. The RSS could give an approximation of the relative range between the two devices in the area. By

combining the computed Bluetooth-based distance with the Wi-Fi positioning output, it is able to refine the devices' position. This approach does not require to install additional infrastructures and can work on the standard Bluetooth protocol which is supported by standard on smartphones

There remain several key challenges of the mentioned approach. The first difficulty is that the Wi-Fi positioning output is not accurate. The errors from the Wi-Fi output is expected to have a mean value in the range from 5m to 7m. In addition, the distance error depends on specific users' location in the area. The second difficulty is that the inferred distance from the RSS value in the Bluetooth enquiry process also suffers some degree of error. The source of Wi-Fi and Bluetooth output errors is the noisy propagation characteristics in indoor environment. For example, None-Line-of-Sight and multipath effect are among the main reasons of the noise. The third difficulty is the non-synchronization between the Wi-Fi scanning process and Bluetooth scanning process. Specifically, the scanning cycle for each technology in the smartphone are not guaranteed to start and finish at the same time. In case of moving users, the time difference causes more noise to the estimated users' position and relative distance.

In this work, we try to overcome these problems by considering both non-temporal and temporal approaches. In the non-temporal approach, the distance error of the Wi-Fi output position is modeled by a Gaussian distribution. Similarly, another Gaussian distribution is used to describe the distance error between two devices from the Bluetooth enquiry process. Wi-Fi and Bluetooth outputs within a short time period are treated as if they happen in a same time window. An error function is then created to measure the mismatch between the two distributions. By neutralizing the mismatch, it is able to improve the position results of Wi-Fi output. In the temporal approach, the time component of the users' movement is incorporated into the error function. The error function uses the position of all the users as its parameters. We employ particle-filter-based tracking to minimize the error function. The particle filter has the observation model as a combination between Wi-Fi and Bluetooth scanning data.

The experiments were conducted with real scenarios with up to four users. Both the non-temporal and temporal approach results are tested against the standard Wi-Fi fingerprinting

model. Our results show that it is able to make use of Bluetooth signal to improve the positioning output of the Wi-Fi fingerprinting model.

5.2 Literature Review

In the state of the art, the most straightforward approach is to deploy static Bluetooth devices in the environment, which act with a similar role to wireless access points of Wi-Fi based technology. Bandara et al. [2004] use up to four Bluetooth antennas as the static station. The proposed system is able to locate a Bluetooth tag within a room with area of 4.5m x 5.5m. The RSSI value is used to classify the tag's position between different areas of the room. Pei et al. [2010] employ fingerprinting based approach for tracking a moving phone. The setup includes only three Bluetooth beacons in a corridor-like space of 80m long, approximately. The horizontal error is reported at 5.1m. For comparison, the Wi-Fi based solution has an error of 2.2m in the same area. However, these results are possible thanks to the 8 installed WLAN access points. More recent works employ the new *BLE* technology. The BLE beacons are smaller and more energy efficient. They are able to power up for a long period of time [Gomez et al., 2012]. Thus, it is more convenient to create Bluetooth beacon networks for positioning purpose. Faragher and Harle [2014] provide an in-depth study of using BLE for indoor localization purpose. The distance error of Bluetooth-based approach could reach as low as 2.6m for 95% of time. However, a high number of beacons should be deployed to reach the above performance. The study also addresses some issues of the BLE signal such as the scanning cycle, fast fading effects and Wi-Fi scanning interference. A similar performance for BLE-based indoor positioning is reported in Zhuang et al. [2016]. The authors employed fingerprinting based approach with the RSS value from the installed BLE beacons.

In the aspect of presence of multiple devices, there are several works on collaborative localization. Those works rely on some specific wireless technologies, which support the peer-to-peer communication. The technologies include Bluetooth, Wi-Fi Direct and Sound. These technologies are capable to discover the existence of nearby neighbors. In Liu et al. [2014], the task of detecting face-to-face proximity is researched. The smartphones are used to scan nearby visible Bluetooth devices in daily usage. From the received RSSI, relative distance between two devices is calculated. The distance is then used to detect whether the two users

are closed to each other. For dealing with noisy Bluetooth signals, additional techniques such as RSSI smoothing and light sensor data are introduced for calculating a more accurate distance. Jun et al. [2013] propose the Social-Loc system, which uses Wi-Fi Direct technology for detecting two events: Encounter and Non-Encounter between each pair of users. In the work, the authors find the RSSI peak for separating Encounter and Non-Encounter events. These detected events are then used to improve the Wi-Fi fingerprinting and Dead Reckoning tracking. The drawback of Wi-Fi Direct technology is that it does not allow the regular Wi-Fi scanning. Therefore, the proposed Social-Loc is more suitable for improving the Dead Reckoning tracking than the Wi-Fi fingerprinting tracking. Sound-based ranging is also useful for detect the relative distance between two devices. In Liu et al. [2012], the authors use the Sound-based distance to improve Wi-Fi fingerprinting positioning system. The acoustic ranging is designed with TOA method for calculating the distance between devices. The estimated ranges are then used to form a graph between participated devices. The graph's vertices are derived from the Wi-Fi positioning output. A search progress is then performed to find the best match position for the graph. The searching task aims to find an agreement between the vertices' position and the edges' length. The proposed approach has a mean error of around 1.6m, depending on specific setups. However, the study only mentions the cases when all the devices are in static position.

5.3 Using Bluetooth Data to Improve Wi-Fi Positioning

Within the context of multiple users, the available data from the environment includes the absolute positions of each device and the relative distances between each pair of devices. In order to implement our idea of data fusion, the participated devices need to know each other absolute positioning. In this step, there are two feasible approaches. The first one is each pair of devices maintains a direct communication, which used to exchange their positions. The pair communication could be carried out by using a Bluetooth-based exchange information. However, it would be inefficient because a participated device is required to communicate to several other devices within a short period of time. The devices, therefore, should be able to maintain several Bluetooth connections at the same time. Moreover, standard Bluetooth API requires the user intervention for accepting incoming connection, which make the approach

less user-friendly. The second approach is to employ an additional central sever which keeps all the available information about the devices' position and their distance to the neighbor devices. The server and the devices could communicate by using Wi-Fi connections. The second approach is more user-friendly, energy efficient and has higher computing capability than the first approach. Hence, we choose the second approach for fusing the Wi-Fi positioning and Bluetooth pair distance.

5.3.1 **Centralized Positioning Framework with Wi-Fi and Bluetooth**

The usage of a central server for indoor positioning is a popular approach. For example, Dao et al. [2014] use a server-based solution for combining different information to improve the localization results. The additional server acts as a central node which gathers all the available information from each participated smartphones. They also employ the server to carry intensive complexity computation in the positioning computing process.

In our task of fusing Wi-Fi and Bluetooth data, the server-based approach can be implemented easily because we can minimize the need for exchange information between the participated smartphones and the server. The information mostly includes the device identifiers and the RSS values. The upload rate is in the range of several seconds. There are two types of information which are required to send for the server side. The first type is the scanned Wi-Fi information of access points. For each completed scan cycle, the device sends the identifier (Wi-Fi MAC address) of the seen access points and their RSS values. Roughly, there is a five-second interval between two consecutive scans. The second type is the Bluetooth scanned information. The needed data is also the Bluetooth MAC addresses of the seen devices and their RSS values. The time of a complete Bluetooth scan is not clearly defined. When a new Bluetooth device is seen, it could be sent to the server immediately. In practice, there can be many visible Bluetooth devices within the environment such as wireless headphones or wireless mice. The server side maintains a list of active devices. From the list, only the Bluetooth information from the participant devices is used for the positioning purpose.

On the server side, the data from Wi-Fi scans and Bluetooth scans gives different ways for calculate the user's positions. Figure 52 illustrates the principle of our approach. For simplicity, we consider the context of positioning within a floor. Each user can be characterized by

his smartphone. The example context involves two users, namely the i^{th} and j^{th} users. The two users' devices keep gathering Wi-Fi access points data and Bluetooth inquiry data within the environment and send to the server. The server receives the data and stores as an Event, which could be of Wi-Fi type or Bluetooth type. In Figure 52, there are three Events: two Wi-Fi scans and a Bluetooth scan. The real position of user i^{th} and j^{th} are denoted as $(x^i_{truth,t}, y^i_{truth,t})$ and $(x^j_{truth,t}, y^j_{truth,t})$, respectively. The subscript t is the time parameter. The real distance between the two users is $d^{ij}_{truth,t}$.

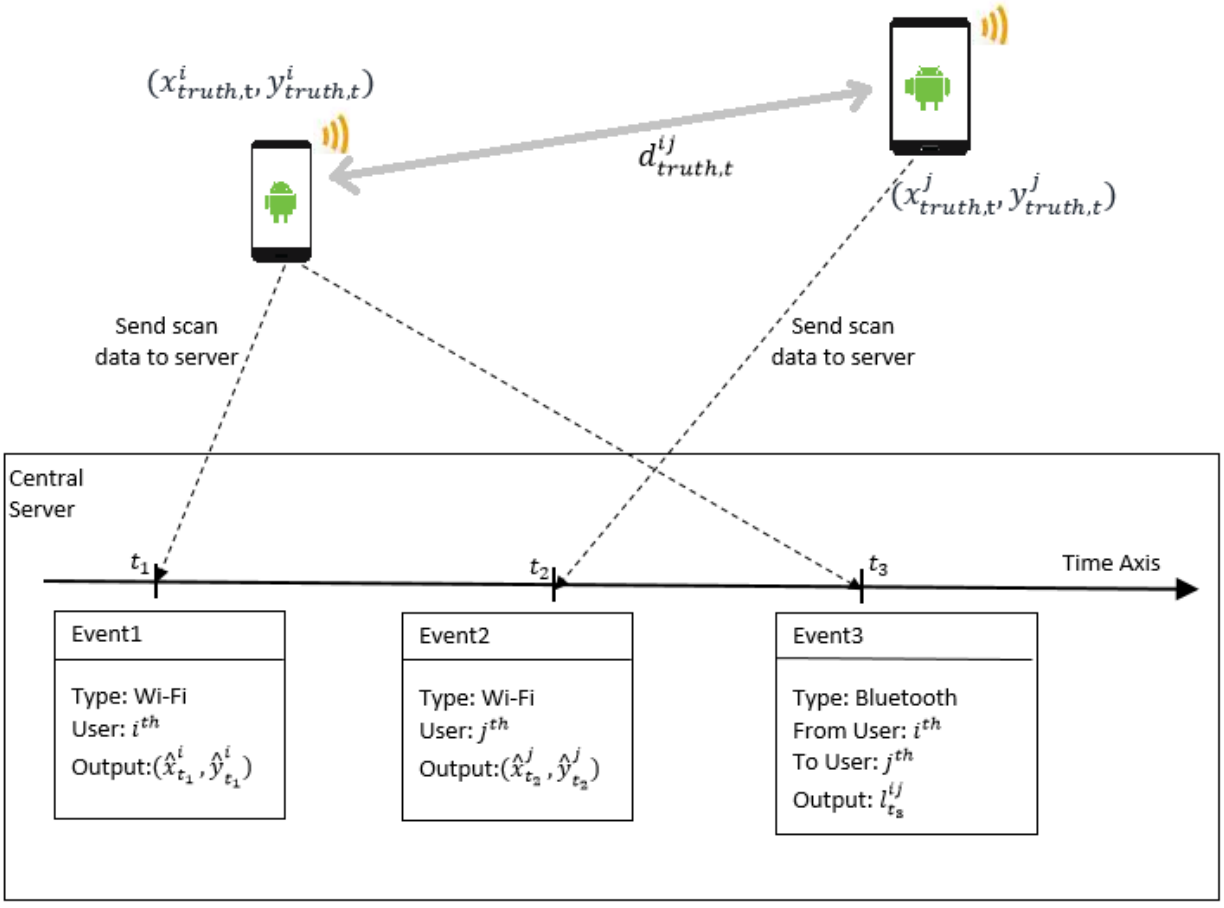


Figure 52. Example of using both the WLAN scan and Bluetooth scan for user positioning system

At time t_1, we can determine the position of user i^{th} is $(\hat{x}^i_{t_1}, \hat{y}^i_{t_1})$ from the WLAN scan information of the i^{th} device. The Wi-Fi position output, however, could be different to the real position $(x^i_{truth,t_1}, y^i_{truth,t_1})$ the user. Similarly, at time t_2, when the device j^{th} completes a

Wi-Fi scan, we can compute the Wi-Fi position output of user j^{th}. Let the result of this computation be $(\hat{x}^i_{t_2}, \hat{y}^i_{t_2})$. Besides that, the Bluetooth scanning process could give an estimation between the two users. At time t_3, if the two users i^{th} and j^{th} are within the Bluetooth scanning range, we could find the relative distance $l^{ij}_{t_3}$ from the RSS value of Bluetooth scanning process. The value of $l^{ij}_{t_3}$ is an approximation of the real distance between the two users at time t_3, d^{ij}_{truth,t_3}. An alternative way for calculating d^{ij}_{truth,t_3} is using the output position from the Wi-Fi data of the users i^{th} and j^{th}. For example, if both t_1 and t_2 are closed enough to t_3, the distance $\hat{d}^{ij}_{t_3}$ between $(\hat{x}^i_{t_1}, \hat{y}^i_{t_1})$ and $(\hat{x}^i_{t_2}, \hat{y}^i_{t_2})$ is a measurement of the value d^{ij}_{truth,t_3}. Therefore, a constraint between $\hat{d}^{ij}_{t_3}$ and $l^{ij}_{t_3}$ could be established such as both values should be as closed as possible. When there are more devices and scanning data, more constraints could be included. To use these constraints effectively, it is necessary to deal with the noisy output of the two technologies. Besides that, the unsynchronized events would add more ambiguity to the problems.

In order to employ the relationship between Wi-Fi and Bluetooth for improving positioning results, we propose two different approaches. The first one is a non-temporal based approach. We remove the temporal relationship between successive events. Closed events within a time interval window would be treated as if they happened at the same time. Wi-Fi fingerprinting approach is used for finding user's positioning from Wi-Fi scan. The LDPL model in Equation (3) is used to find the distance from the Bluetooth RSS value. A simple likelihood function is built with the users' position as parameters. Based on the function of non-temporal approach, we extend the likelihood function to include the time parameters in the temporal approach. The error function includes the devices' position at each timestamp as its parameters. A simple motion model is added to establish the positioning relationship between closed events. The error function is then computed by using a particle filter approximation.

5.3.2 **Non-temporal Approach**

In the non-temporal approach, a sliding window of length Δt is used. All of events from t to $t + \Delta t$ are considered happened at the same time. Without loss of generality, we assume that from t to $t + \Delta t$, it is possible to receive the Bluetooth scan, from the mobile device user i^{th} which contains the enquiry information from the mobile device of user j^{th}. The Wi-Fi scans from the mobile device of the user i^{th} and j^{th} are aslo available within the time interval $[t, t + \Delta t]$. Let w^i and w^j be the Wi-Fi scans from the two users, and rss^{ij} is the RSS value of the Bluetooth scan. For the non-temporal based approach, we remove the time variable from the parameters. It is able to create a likelihood function with the two users' position (x^i, y^i) and (x^j, y^j) as the parameters $P(x^i, y^i, x^j, y^j | w^i, w^j, rss^{ij})$:

$$P(x^i, y^i, x^j, y^j | w^i, w^j, rss^{ij}) = P_W(x^i, y^i | w^i) P_W(x^j, y^j | w^j) P_B(x^i, y^i, x^j, y^j | rss^{ij}) \quad (39)$$

In the right hand side of the equation, the two first terms measure the likelihood of the position with respect Wi-Fi scan for each user. The third term provides the likelihood of the relative distance between two users with respect to the Bluetooth scan.

For measuring the Wi-Fi part, for example $P_W(x^i, y^i | w^i)$, one could assume the Wi-Fi output position from a fingerprinting model has a Gaussian distribution, like the approach in Evennou and Marx [2006] . Let $(\hat{x}^i, \hat{y}^i)$ be the calculated position from the scan w^i, $P_W(x^i, y^i | w^i)$ is measured by:

$$P_W(x^i, y^i | w^i) \sim P_W(x^i, y^i | \hat{x}^i, \hat{y}^i) = \frac{1}{\sqrt{2\pi}\delta_w} e^{-\frac{(x^i - \hat{x}^i)^2 + (y^i - \hat{y}^i)^2}{2\delta_w^2}} \quad (40)$$

with δ_w is a constant indicating the reliability of the Wi-Fi fingerprinting model

For measuring the Bluetooth part of the estimated likelihood, we first calculate the l^{ij} from rss^{ij} by using the well known LDPL model:

$$l^{ij} = l0 * 10^{(rss^{ij} - rss_{l0})/(10n)} \quad (41)$$

where rss_{l0} is the RSS value at the distance l_0, n is the path loss exponent. The three values rss_{l0}, n, and l_0 are known constants. The value of l^{ij} is an approximation of the real distance, which comes directly from the real position of users i^{th} and j^{th}, (x^i, y^i) and (x^j, y^j):

$$d^{ij} = \sqrt{(x^i - x^j)^2 - (y^i - y^j)^2} \quad (42)$$

By assuming l^{ij} have a Gaussian distribution around d^{ij}, the Bluetooth likelihood can be estimated by another Gaussian kernel:

$$P_B(x^i, y^i, x^j, y^j | rss^{ij}) \sim P_B(d^{ij} | l^{ij}) = \frac{1}{\sqrt{2\pi}\delta_b} e^{-\frac{(d^{ij} - l^{ij})^2}{2\delta_b^2}} \quad (43)$$

with δ_b is a constant indicating the reliability of the LDPL on the RSS Bluetooth signal.

From Equations (40) and (43), the likelihood function in Equation (39) could be rewritten as:

$$P\left(x^i, y^i, x^j, y^j \middle| w^i, w^j, rss^{ij}\right) = C * e^{-g(x^i, y^i, x^j, y^j)}$$

$$g\left(x^i, y^i, x^j, y^j\right) = \frac{(x^i - \hat{x}^i)^2 + (y^i - \hat{y}^i)^2 + (x^j - \hat{x}^j)^2 + (y^j - \hat{y}^j)^2}{2\delta_w^2} + \frac{(\sqrt{(x^i - x^j)^2 - (y^i - y^j)^2} - l^{ij})^2}{2\delta_b^2} \quad (44)$$

with C, δ_b and δ_w are constants.

Our purpose is to find the value of (x_i, y_i) and (x_j, y_j) which minimize the value for g, and thus, maximize the likelihood probability P. If g is rewritten with $r^i = \sqrt{(x^i - \hat{x}^i)^2 + (y^i - \hat{y}^i)^2}$, $r^j = \sqrt{(x^j - \hat{x}^j)^2 + (y^j - \hat{y}^j)^2}$ and d^{ij} from above, the equation of g is changed into a simpler form:

$$g(x^i, y^i, x^j, y^j) = \frac{(r^i)^2 + (r^j)^2}{2\delta_w^2} + \frac{(d^{ij} - l^{ij})^2}{2\delta_b^2} \quad (45)$$

The range of d^{ij} is $[max(\hat{d}^{ij} - r^i - r^j, 0), \hat{d}^{ij} + r^i + r^j]$ with $\hat{d}^{ij}$ is the distance between $(\hat{x}^i, \hat{y}^i), (\hat{x}^j, \hat{y}^j)$. In general, g could reach the minimum value at various points in the domain as we could swap the value of r^i and r^j without changing the value of g. Moreover, g is symmetric by the line between $(\hat{x}^i, \hat{y}^i), (\hat{x}^j, \hat{y}^j)$. Therefore, we add two constraints to make the minimum point of g unique. The first constraint is that the four points $(x^i, y^i), (x^j, y^j), (\hat{x}^i, \hat{y}^i), (\hat{x}^j, \hat{y}^j)$ are aligned. The second constraint is that the distance $r^i = r^j = r$. The function $g(x^i, y^i, x^j, y^j)$ is then rewritten as a function of r:

$$g(r) = \frac{r^2}{\delta_w^2} + \frac{(\hat{d}^{ij} - 2r - l^{ij})^2}{2\delta_b^2}$$

(46)

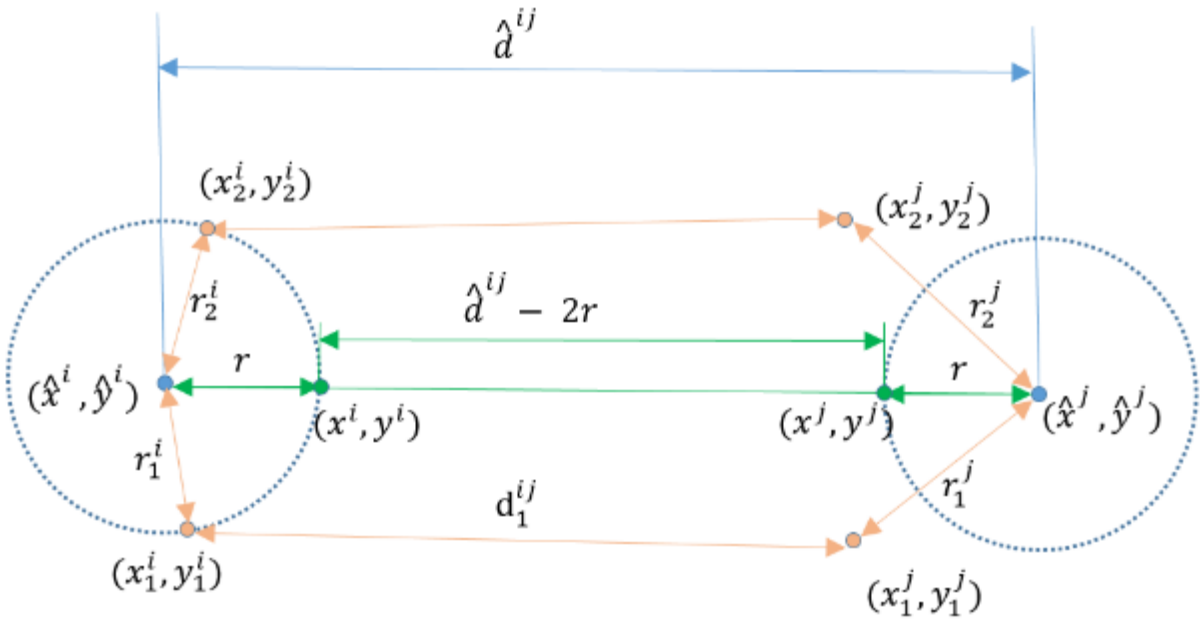


Figure 53. Several ways for evaluating g. Two pairs of points $< (x_1^i, y_1^i), (x_1^j, y_1^j) >$ *and* $< (x_2^i, y_2^i), (x_2^j, y_2^j) >$ *are symmetric by the line through* $(\hat{x}^i, \hat{y}^i)$ *and* $(\hat{x}^j, \hat{y}^j)$. *The pair* $< (x^i, y^i), (x^j, y^j) >$ *is used to calculate the minimized value of g*

Figure 53 gives an illustration of the above transforming process. The pair of position (x^i, y^i) and (x^j, y^j)can be used for finding the minimum value of $g(x^i, y^i, x^j, y^j)$. It also shows the symmetric property of g. The value $g(x_1^i, y_1^i, x_1^j, y_1^j)$ is equal to $g(x_2^i, y_2^i, x_2^j, y_2^j)$ if r_1^i is equal

to r_2^i and r_1^j is equal to r_2^j. It is straightforward to find the minimum value of the function $g(r)$ in (46). The updated positions of user users i^{th} and j^{th} are then calculated from the value of r by using the two added constraints.

The first drawback of the above method is to exclude the temporal-component of the user's movement. The WIFI-based position and the Bluetooth-based distance are received at different moments. It is also difficult to determine the time interval length Δt. If it is too large, noisier output could be added to the adjustment process. If it is too small, there is not enough related Wi-Fi scans and Bluetooth scan. The other drawback is the use of Gaussian distribution to model the likelihood $P_W(x^i, y^i|w^i)$. The later temporal approach is designed to improve those drawbacks of the temporal approach.

5.3.3 **Temporal Approach**

In our temporal approach to the problem, we attempt to use the temporal relationship in the establishment of the likelihood function P. Instead of relying only on the position of two users at specific timestamp to measure the errors, the likelihood function could be extended to include the moving path of all the users. Each moving path is considered as a sequence of points. The new likelihood function would receive all of the points as its parameters.

We first construct the likelihood function F, which is a more completed form of P based on three probability functions. A motion model M is used to establish the relationship between the position at time t and the position at time $t + 1$ when the user moves within the area. A probability distribution function W describes the distribution probability from Wi-Fi scan results. The B function describes the distance distribution based on the Bluetooth RSS value from each pair of devices.

Assume that there are N users to track in T seconds. Let call the position of user i^{th} at second as (x_t^i, y_t^i). It is required that the time index t contains all the Event timestamps from Wi-Fi and Bluetooth of all the participant devices. Approximately, all the float-typed timestamps could be rounded to the nearest integer values. The motion model for each user i^{th} is defined as a probability function between the previous position and present position,

$M(x_t, y_t|x_{t-1}, y_{t-1})$. The moving component for T seconds for each user i^{th} is then calculated by:

$$F_i^M = \prod_{t=1}^{T} M\,(x_t^i, y_t^i|x_{t-1}^i, y_{t-1}^i) \quad (47)$$

For each specific user i^{th}, assume that there are K timestamps among the T seconds which have the Wi-Fi scan results. Let call the timestamp for Wi-Fi events $u_1, u_2, \dots u_K$. Then for each u^k, $w_{u^k}^i$, the function W is used to estimate the likelihood probability $W(x_{u_k}^i, y_{u_k}^i, |w_{u_k}^i)$. Then, the Wi-Fi component F_i^W of the user i^{th} is built from all the available Wi-Fi scans:

$$F_i^W = \prod_{k=1}^{K} W\,(x_{u_k}^i, y_{u_k}^i, |w_{u_k}^i) \quad (48)$$

Similarly, we can build the Bluetooth component F_{ij}^B from the Bluetooth inquiry process for each pair of for each pair of users i^{th} and j^{th}. From the Bluetooth data of the user i^{th}, it can be assumed that the Bluetooth scan process results in L timestamps which are $v_1, v_2, \dots, v_L$. Each scan receives a signal strength $r_{v_l}^{ij}$. We are then able to estimate the probability $B(x_{v_l}^i, y_{v_l}^i, x_{v_l}^j, y_{v_l}^j|r_{v_l}^{ij})$. The Bluetooth components for L timestamps are then calculated by:

$$F_{ij}^B = \prod_{l=1}^{L} B\,(x_{v_l}^i, y_{v_l}^i, x_{v_l}^j, y_{v_l}^j|r_{v_l}^{ij}) \quad (49)$$

Then, our probability likelihood function F could be written as:

$$F = \left(\prod_{i=1}^{N} F_i^M\right)\left(\prod_{i=1}^{N} F_i^W\right)\left(\prod_{i=1,j=1}^{N} F_{ij}^B\right) \quad (50)$$

At this step, one could select the explicit form of M, W and B and process to find the maximum value of F. The number of estimated parameters in F totally depends on the number of users and the tracking time. As the F function includes a motion model M, particle filter-based approximation is a natural way for approximating the maximum value of F. In addition to that, the particle filter process would have a more flexible way for selecting the explicit form of W and B.

For each user i^{th} at time t, there is a set of particles S_i^t, which represents the position distribution probability. For the motion model M, without the additional information from inertial

sensors, the movement of the user could take random values as the moving speed v and the heading direction h. While there is no constraint on the value of h, the moving speed v should be suitable with a typical indoor movement. In our specific implementation, we generate the moving speed from a normal distribution around a speed average value. The speed average is chosen according to the walking action in indoor environment. The heading is generated from the uniform distribution in the range $[0, 2\pi]$. An additional wall-crossing checking step is added for removing bad particles. Figure 54 gives an example of the motion model M for generating the new particles. The center black dot is the initial particle. The wall is represented with the black lines. New particles are then generated with a normal distribution moving speed around the speed average value and a uniform heading direction. The green particles are kept. The gray ones which cross the wall are removed.

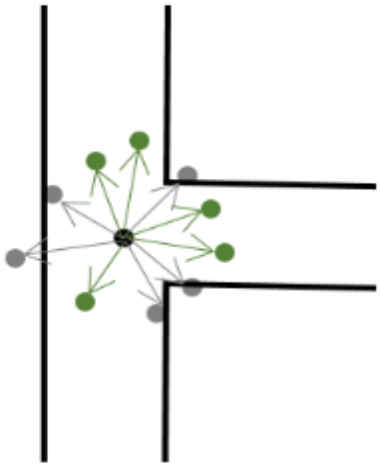


Figure 54. An example of the motion model M. The black dot in the center is the original particle which is used to generate the gray and the green dots. The gray dots are removed because they cross the walls (black lines).

With the particle filter-based approximation, the likelihoods given by Wi-Fi component F^W and Bluetooth component F^B could be transformed into the observation model. The score a specific $p_i^t \in S_i^t$ is calculated by:

$$score(p_i^t) = score_W(p_i^t) + score_B(p_i^t) \quad (51)$$

with $score_W(p_i^t)$ is the Wi-Fi component and $score_B(p_i^t)$ is the Bluetooth component.

The $score_W(p_i^t)$ is calculated by using the method described in Section 4.4.2 if there is a scan at around time $[t - \Delta_1 t, t + \Delta_1 t]$ for user i^{th}. Otherwise, its value is set to 0. The constant $\Delta_1 t$ is the effective window length for each Wi-Fi scan. Similarly, the $score_B(p_i^t)$ can be calculated if there is any Bluetooth scan involved the user i^{th} around time t. Without loss of generality, we assume the available Bluetooth scan is rss_{ij}^t that specifies the RSS value from the device i^{th} and j^{th}. The update rule for $score_B(p_i^t)$ is defined as follow:

$$score_B(p_i^t) = \sum_{p_{j,k}^t \in S_j^t} score_W(p_{j,k}^t) * B(p_i^t, p_{j,k}^t | rss_{ij}^t) \quad (52)$$

The subscript k indicates the need to calculate repeatedly for each $p_{j,k} \in S_j^t$. The likelihood $B(p_i^t, p_{j,k}^t | rss_{ij}^t)$ is computed by using similar process as the computing of the likelihood $P_B(d|l)$ in Equation (43). The distance d is the distance between two particles $p_i^t, p_{j,k}^t$ and the distance l is derived from rss_{ij}^t by the LDPL model. A constant $\Delta_2 t$ is added to define the effective interval length for a Bluetooth scan.

5.4 Experiments and Results

To evaluate the performance, we select up to four devices which are participating in the positioning scenario. Each device is set to scan Wi-Fi access points and available Bluetooth devices in the environment. The users are instructed to carry the devices and move around the experiment area. When a checkpoint is reached, the time is registered. The checkpoint's position and its reaching time are then used to calculate the user's trajectory as the ground truth movement. The testing area is an office environment which includes two floors of the MICA institute. There is only one tested trajectory which is composed mostly by the corridor and office rooms. The path is illustrated in Figure 55. The length of the path is around 200m, which usually takes 300s of walking with average speed. There are four devices in use, including two smartphones and two tablets. All of them run the Android operating system and use the same application for collecting the Wi-Fi and Bluetooth data. Table 17 gives some details information about the four devices.

In our experiment, we selected an offline approach. Instead of sending the scan results to a server, they are stored as text log files and processed later. In general, this approach's results are expected to be identical with the online processing.

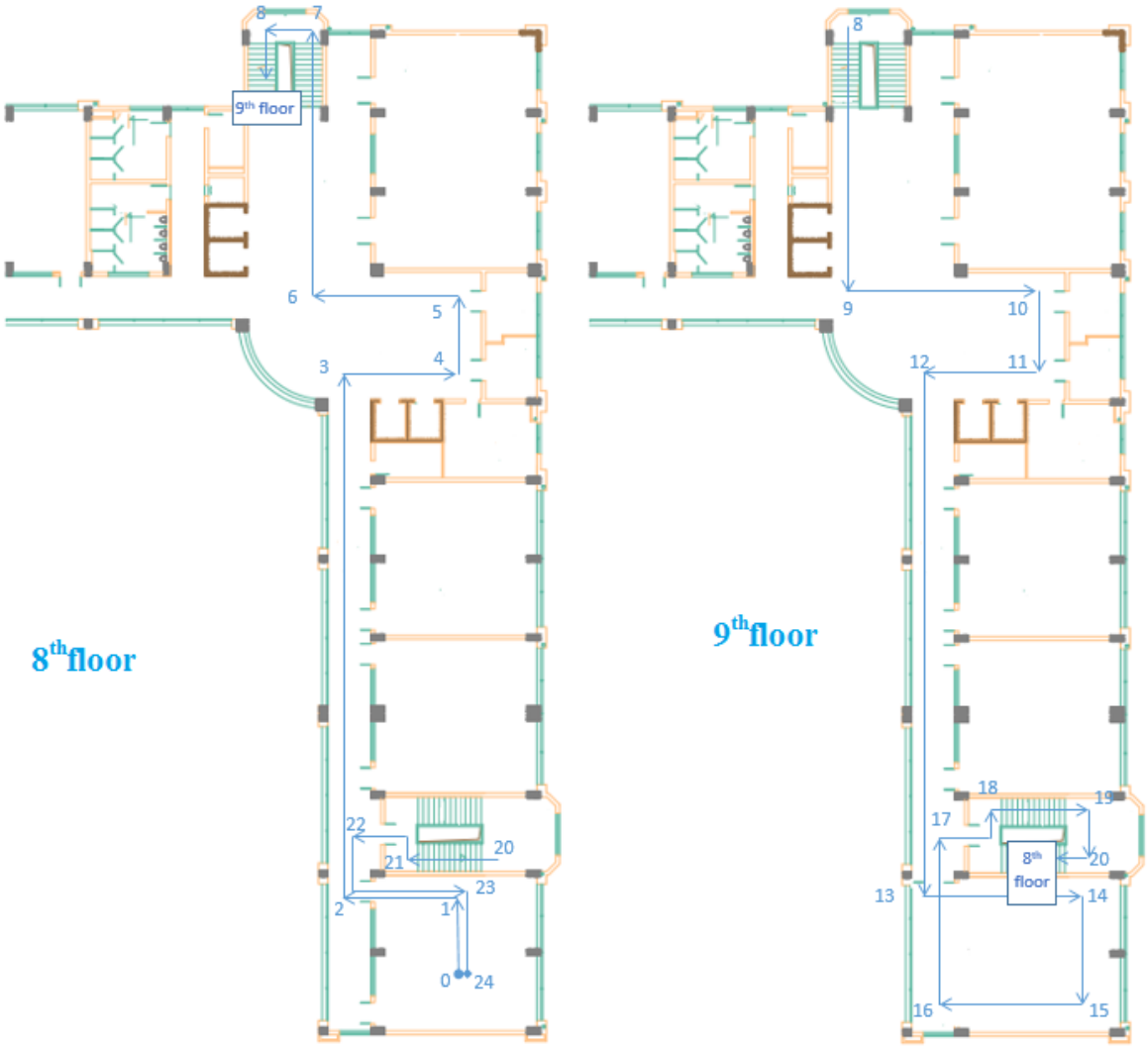


Figure 55. Testing path in two floors of MICA institute

Table 17. Detailed information for each devices

Device ID	Name	Type	Operating System
1	Samsung Galaxy Note 4	Smartphone	Android 6.0.1
2	HTC One ME	Smartphone	Android 5.0.2
3	Asus ME	Tablet	Android 4.2.2
4	Samsung Galaxy Tab	Tablet	Android 4.2.1

5.4.1 **Relationship between Bluetooth RSS Value and Distance**

In the first part, the parameters for estimating the distance from RSS value are computed. From Equation (43), it is mandatory to find the appropriate $d0$ and the RSS value at $d0$, namely rss_{d0}. In this experiment, we select the path loss exponent $n = 1.8$ because of the indoor office environment as listed in Table 1. A test device is set to scan the visible Bluetooth connection of another device in the environment. There is no obstacle between the two devices, thus, the effect of non-line-of-sight propagation is ignored. For each distance, the collection time is set to 300 seconds. The completed inquiry cycle of Bluetooth is around 12 seconds in average. This results in around 25 to 30 samples per distance. However, when the distance is too large, the number of the inquiry samples starts to reduce. At the distance of 20m, there are only 10 inquiry samples. Figure 56 illustrates the experiment results.

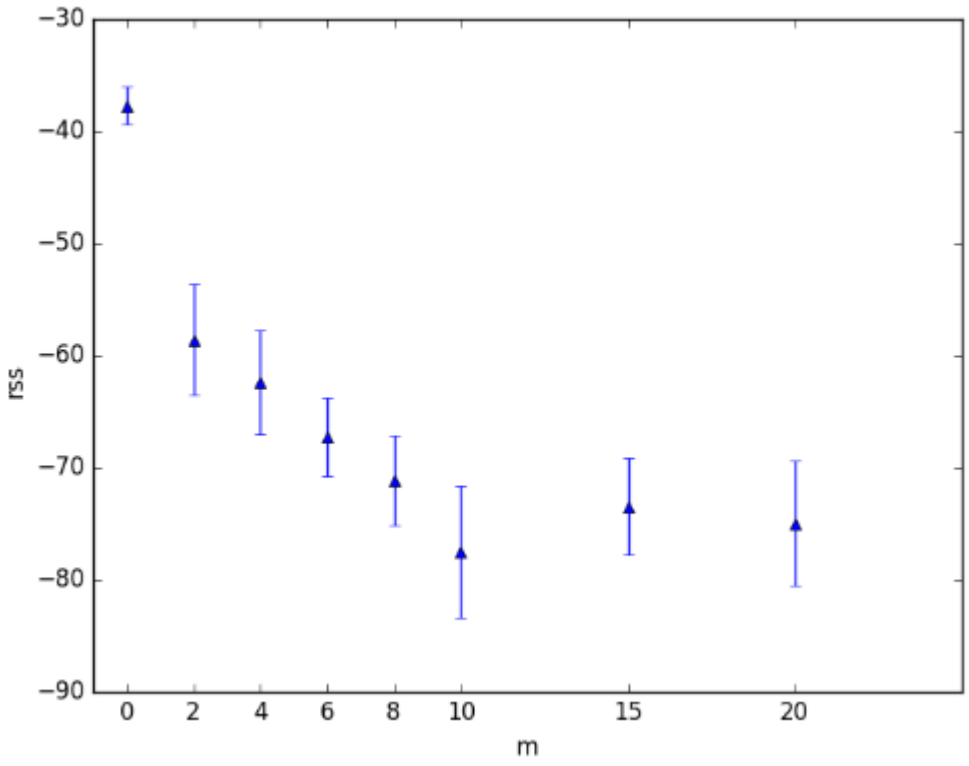


Figure 56. The mean RSS and its std values for selected distances between two smartphones

The distance $d = 0m$ has the highest RSS value and also lowest *std*. The decrement pattern become more stable for the distance from 2m to 10m. Outside the range of 10m, it starts to

fluctuate and drop packages. It is the reason why both mean distance at 15m and 20m is higher than at the 10m. Knowing the results, we selected $d0 = 2m$ and $rss_{d0} = -60$dBm.

In order to estimate the distance error, the two selected values $d0$ and rss_{d0} are substituted into Equation (41) to estimate the distance from the received signal strength. The cumulative distribution distance error is illustrated Figure 57. From the plot, we set the value of ϵ_d in Equation (43) to 3.0m, which is near the 90th percentile of the distance errors.

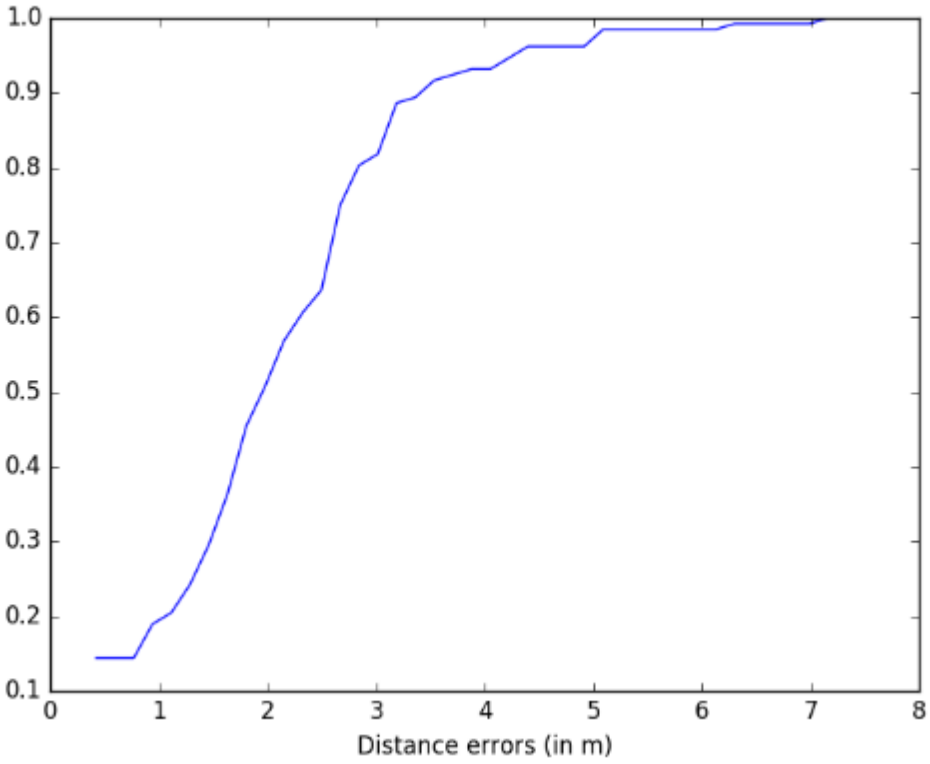


Figure 57. Cumulative distance errors with the selected parameters

5.4.2 **Wi-Fi Baseline Model**

A Wi-Fi fingerprinting model is trained as a baseline model for the later fusing step with Bluetooth data. For building the Wi-Fi fingerprinting database, two the Device 1 (Samsung Galaxy Note 3) and Device 3 (Asus ME) are used. The database is built with the same method described in Chapter 3. One user is asked to carry the device and walk along the test path. Every time a checkpoint is reached, the timestamp is registered and then used to interpolate the full moving path in a later stage. The building database process therefore does not take much

time. There are six training log files in total. Given the specific testing path in Figure 55, all the data are collected in less than one hour.

Table 18 gives a detailed view of the collected Wi-Fi data. A time threshold of 5s is used to grouping RSS signals into one complete scan. There are around 400 fingerprinting across 6 training files. We noticed that the number of completed scan Wi-Fi per user's walks fluctuate in the range of 50-100 scans. Moreover, the number of seen Wi-Fi access points also depends on the user's position and the underlying device running processes. In our case, the average number of seen Wi-Fi access points per scan is around 7. The minimum number of access points is 3 and maximum number of access points is 12.

Table 18. Data for training Wi-Fi finger printing model

Number of train files	6
Path Length	220m
Average time length	300 seconds
Average of scan for one path	63 scans
Total of seen access points	138
Average seen access points	7

The distributions of RSS value from two devices are plotted in Figure 58. There are some variations between distributions of the two devices. The variation is noticeable in the region of less than -85dBm and around -60dBm. In a multiple-device context, those types of variants could affect the generation of fingerprinting models. In order to reduce the effects of device diversity, we firstly filter out the RSS values of the region less than -85dBm. Second, a mean normalization step is used. For each participated device, it is assumed to know the mean RSS value over the test area. The raw RSS value is then subtracted by this value. Other approaches such as finding the linear transformation [Bernardos et al, 2010] or using HLF features [Kjaergaard and Munk, 2008] would be suitable for improving the later training fingerprinting model step.

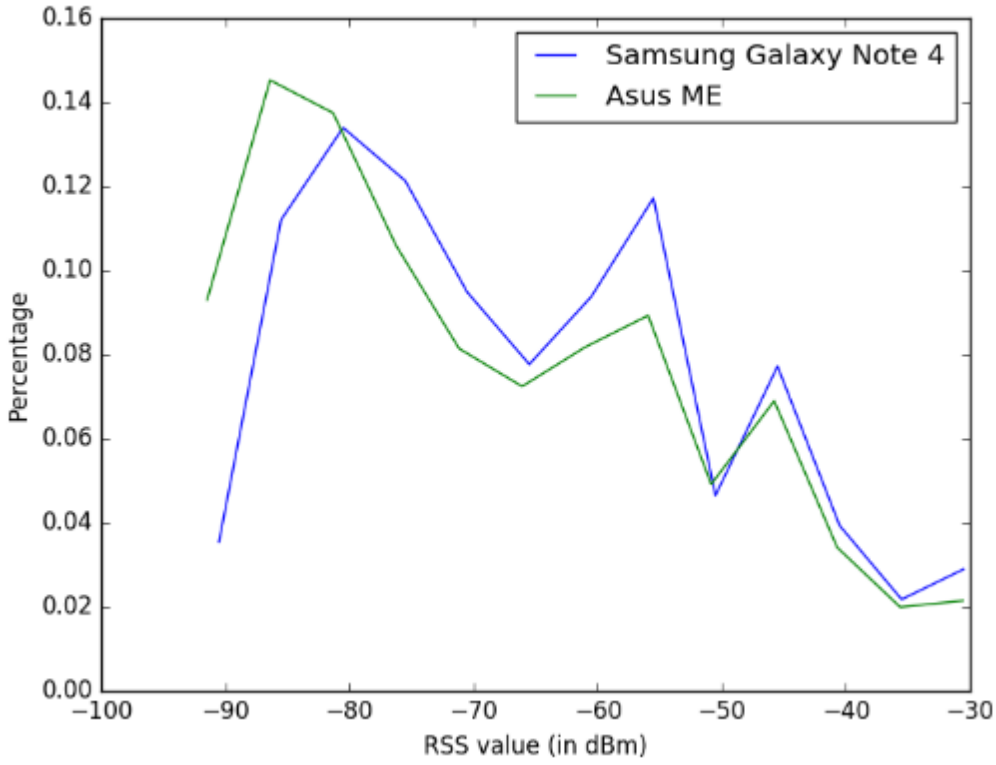


Figure 58. RSS value distribution of collected dataset for training fingerprinting model

For the fingerprinting model, we select the Random Forest based model as the training model. The Random Forest is trained with 500 trees. Both the raw RSS feature and the mean normalization RSS feature are used in the K-fold cross validation testing with K = 5. As the path contains two floors with some segments within the stairs, the target position is treated as a 2.5D coordinates. For a triplet (x, y, z), the z value is normalized to receive only one of three values in $[0, 0.5, 1]$. The values 0 and 1 represent the points in the 8^{th} floor and 9^{th} floor while the value 0.5 is used to indicate that the user moves within the stairs. The distance error between the target point $p_{target} = (x_{target}, y_{target}, z_{target})$ and the output position $p = (x, y, z)$ is calculated by:

$$d(p_{target}, p) = \sqrt{(x_{target} - x)^2 + (y_{target} - y)^2} + 10 * abs(z_{target} - z) \quad (53)$$

In the experiment, we use 10 as the weight of wrong floor prediction.

The performance of the two preprocessing feature approaches are shown in Figure 59. The standard raw features result in a mean distance error of 4.21m while the mean normalization

one has mean error of 3.61m. The former has a *std* of 2.65m and the latter has a *std* of 2.53m. There is a slightly improvement by changing from the raw features to the preprocessed one. In terms of maximum distance error, both approaches suffer from large errors of over 10m. Compared with the published dataset of IPIN 2016, the performance of RF model with the preprocessing features is comparable. In order to reduce the error further, it would require a large amount of training data.

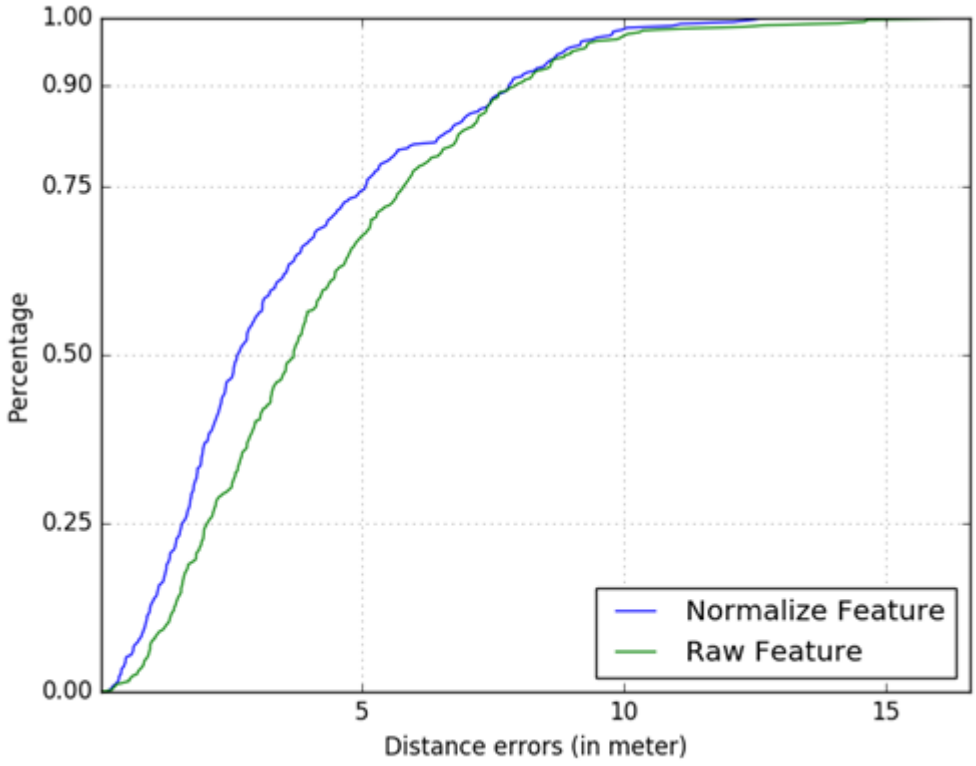


Figure 59. Comparison between the raw feature and the normalization feature in training with RF model

5.4.3 **Positioning Results**

The two proposed methods are tested with different scenarios. The scenarios were designed in such a way that the additional Bluetooth scanning data could provide useful information for smoothing the Wi-Fi positioning output. Each scenario involves from 2 to 4 users. They walk in the same path as shown in Figure 55, with different relative distances between each other. The *Wi-Fi only*, *Non-temporal* and *Temporal* approaches are used to localize the users. We used a tracking interval of 0.5 second for tracking the users.

In the *Wi-Fi only* approach, the output of Wi-Fi fingerprinting method is provided as the reference tracking results. First, the RF regressor model is trained on the six training files. In the testing phase, with each completed Wi-Fi scan from the tested devices, the RF model is used to produce the position output. The full tracking path for each 0.5 second interval is interpolated from the outputs.

In the *Non-temporal* approach, the output positions from the pre-trained RF regressor model are calculated for each device. The Bluetooth scanned information is then used for adjusting the positioning by the updating rule at *Non-temporal* updating rule. To solve the problem of non-simultaneous between Wi-Fi scans and Bluetooth scans, we use a time window of length $\Delta t = 10$ seconds for grouping successive events into the same timestamp. In practice, within a specific time window Δt, there could exists multiple Bluetooth inquiry information. Therefore, multiple ways of adjusting the Wi-Fi output positions would be possible. The resulting position is calculated as the mean value of these adjusted positions.

In the Temporal approach, a RF classifier model is trained from the six training files. To transform the real world coordinates to label index, we perform a K-means clustering of all the available training positions, similar to what is described in Section 3.5.3.2. The new learning targets are the indices of the corresponding clusters. In our experiments, we use K = 30 for clustering all the available points in the tested area. The radius of each cluster in this configuration is 4.0m approximately. The probability output of the classifier model is then used to update the Particle Filter in the time window of 10 seconds. If there are multiple completed scans within time window of 10 seconds, the nearest completed scan is selected. The Bluetooth data has the effective range set to 2.0 seconds. For the moving model, the average speed of each particle is set to 1 m/s. In the simulation step, the number of particles is set to 1000.

5.4.3.1 *With Two Users*

The two specific devices, Device 1 and Device 3, are used. Each user carries one of the two devices and walk along the trajectory. In the first scenario, they walk together. The distance between the two users in this case is kept under one meter. In the setup, they walk separately. The distance between the two users is kept in the range from 5m to 10m.

Table 19 shows the distance error of the three approaches. With the *Wi-Fi only* approach, we have similar results as in the 5-fold cross validation test above. Because both the devices are used in collecting the data, the Wi-Fi fingerprinting model is unaffected by the device diversity in this case. When the information from Bluetooth scanning process is added, the distance error is reduced in both *Non-temporal* and *Temporal* approaches. The *Non-temporal* has a clear improvement in the *One group* scenario. However, the impact of Bluetooth data when discarding the temporal relationship is insignificant in the *Two groups* scenario. With the *Temporal* approach, we have stable improvements across the two scenarios.

Table 19. Positioning results when there are two users

Scenario	Device	Wi-Fi only	Non-temporal	Temporal
One group	1	3.69m $\pm$ 2.13	2.58m $\pm$ 1.56	2.18m $\pm$ 1.61
	3	3.72m $\pm$ 1.79	2.31m $\pm$ 1.58	2.18m $\pm$ 1.45
Two groups	1	3.77m $\pm$ 2.81	3.89m $\pm$ 2.63	2.87m $\pm$ 2.23
	3	3.05m $\pm$ 1.90	2.65m $\pm$ 1.94	2.12m $\pm$ 1.84

5.4.3.2 *With Three Users*

In this experiment, three users are asked to carry three devices. Device 1, Device 3 and Device 4 are used. The walking path is the same as the training data. In the first scenario, both three users are asked to walk together. In the second test, the three users are split into 2 separated groups. The user carrying the Device 3, is asked to walk as a separated group. The other two users are asked to walk as the same group. In the third scenario, all the three users walk in three separated groups When there is more than one group, the distance between groups is also kept from 5m to 10m approximately throughout the experiments.

Table 20 shows the distance error of the three approaches for each scenario. With the Wi-Fi only approach, the RF regressor model results in similar distance errors for Device 1 and Device 3. The positioning result for the Device 4 is not as good as the other two devices, because the data does not contain training samples for Device 4. Compared to the *Wi-Fi only* approach, there is a slightly improvement by using the *Non-temporal*. Across all the devices and scenarios, the average improvement is around 0.5m in mean distance errors. The appearance of the Device 4 with high distance error clearly affects the performance of the other two devices.

The *Temporal* approach provides more stable improvement from the results of *Wi-Fi only* methods. It outperforms the two other approaches in all scenarios. The highest improvements are the case of device 1 in *One group* and device 3 in *Three groups* settings. With the contribution of the motion model and the Bluetooth-based distance, the Temporal could make the effect of non-training data on the Device 4 minimal.

Table 20. Positioning results when there are three users

Scenario	Device	Wi-Fi only	Non-temporal	Temporal
One group	1	3.83m $\pm$ 2.08	2.99m $\pm$ 2.18	2.23m $\pm$ 2.09
	3	3.49m $\pm$ 2.29	3.29m $\pm$ 2.13	2.00m $\pm$ 1.68
	4	4.77m $\pm$ 2.93	4.32m $\pm$ 2.19	2.18m $\pm$ 1.69
Two groups	1	2.99m $\pm$ 2.05	2.77m $\pm$ 1.85	2.31m $\pm$ 1.72
	3	3.41m $\pm$ 2.60	3.28m $\pm$ 2.32	2.75m $\pm$ 2.36
	4	3.93m $\pm$ 2.55	3.27m $\pm$ 2.29	2.13m $\pm$ 1.78
Three groups	1	3.28m $\pm$ 2.37	2.97m $\pm$ 2.39	2.23m $\pm$ 1.90
	3	3.50m $\pm$ 2.03	2.67m $\pm$ 1.88	2.11m $\pm$ 1.68
	4	3.86m $\pm$ 2.05	3.75m $\pm$ 2.39	2.38m $\pm$ 1.99

5.4.3.3 *With Four Users*

There are two separated scenarios for the four-user setup. In the first scenario, all of the users are asked to move together as a group. Compared to the previous *One group* scenarios, it is difficult to keep a closely relative distance in this step. The group is supposed to move within a circle of 2m radius. In the second scenario, the four users are split into two separated groups. The first group is composed of two users carrying Device 1 and Device 2. For the second group, users carrying Device 3 and Device 4 are included. The distance between two groups is then kept in the range from 5m to 10m, similar to the previous scenarios.

Table 21 illustrates the results for each scenario. Similar patterns as the 3-user setup can be seen in the results of the *Wi-Fi only* approach. Both new devices, Device 2 and Device 4, have larger distance errors than the two remaining devices. Device 1 and Device 3 both have similar performance as the 5-fold cross validation testing. It is clearly that the RF fingerprinting model has difficulties for tracking Device 2 and Device 4, which are not present in the training data. The high error from RF finger printing model add some degrees of noise into the later fusion step with Bluetooth information. The *Non-temporal* approach can improve the results of Wi-

Fi only in most of cases. However, its impact is quite low. There are several exceptions which using the Bluetooth data makes the error increase. For example, the results from Device 1 increases from 3.05m to 3.46m in the *One group* setting. The *Temporal* approach results in lower errors than the other two approaches. In general, good results are obtained on the Device 1 and Device 3, which RSS data are present in the training phase. There is an exception on the results of Device 2 in the *One group* scenario. The *Temporal* 's mean error is slightly higher than that of the *Non-temporal*. Nevertheless, it is able to reduce to errors from the *Wi-Fi only* method.

Table 21. Positioning results when there are four users

Scenario	Device	Wi-Fi only	Non-temporal	Temporal
One group	1	3.05m $\pm$ 1.90	3.46m $\pm$ 2.33	1.91m $\pm$ 1.48
	2	4.95m $\pm$ 3.59	4.29m $\pm$ 3.23	4.62m $\pm$ 3.11
	3	3.48m $\pm$ 2.59	3.04m $\pm$ 2.25	2.40m $\pm$ 1.82
	4	3.54m $\pm$ 2.35	3.27m $\pm$ 2.37	2.30m $\pm$ 1.63
Two groups	1	3.22m $\pm$ 2.25	2.64m $\pm$ 1.74	2.74m $\pm$ 2.02
	2	4.15m $\pm$ 2.46	3.51m $\pm$ 2.21	3.16m $\pm$ 2.26
	3	3.52m $\pm$ 2.24	3.87m $\pm$ 2.27	1.73m $\pm$ 1.58
	4	4.24m $\pm$ 2.69	3.89m $\pm$ 2.87	2.80m $\pm$ 2.47

Figure 60 illustrates the distance error for three approaches over all the scenarios. Both the *Wi-Fi only* and the *Non-Temporal* have a closed performance. For 75% of time, the distance errors of two approaches are around 5m. The Bluetooth-based relative distance are employed more efficiently in to Tem*poral* approach. It has a significant improvement from the Wi-Fi-based tracking. For 75% of time and 90% of time, the errors of *Tempora*l approach stay around 3.0m and 5.0m respectively. Beside the Bluetooth information, the adding of map-based information and moving model constraint also reduce much noisy output from the standard Wi-Fi fingerprinting model.

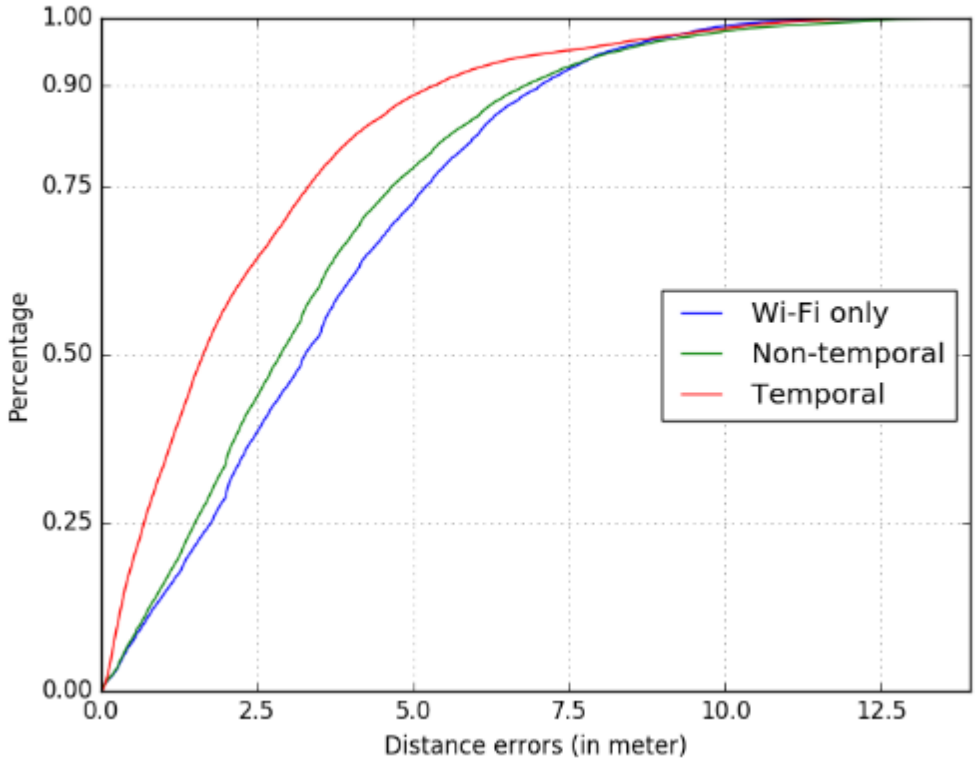


Figure 60. Performance comparison between three approaches

Individual distribution error for each tested device is given in Figure 61. Both smartphones, Samsung Galaxy Note 4 and HTC One ME, have a similar distribution. The *Non-temporal* approach is a slightly improvement from using only Wi-Fi data and the *Temporal* approach can reduce the error significantly for the regions less than 7.5m. However, the addition of Bluetooth data is unable to reduce the errors at higher region. It even adds more noise to the tracking results of Device 2. In the case of Device 3 and Device 4, both *Non-temporal* and *Wi-Fi only* have nearly identical distributions and the *Temporal* one outperforms the two others. The *Temporal* has the biggest improvement with Device 4, which can overcome the issue of non-training data.

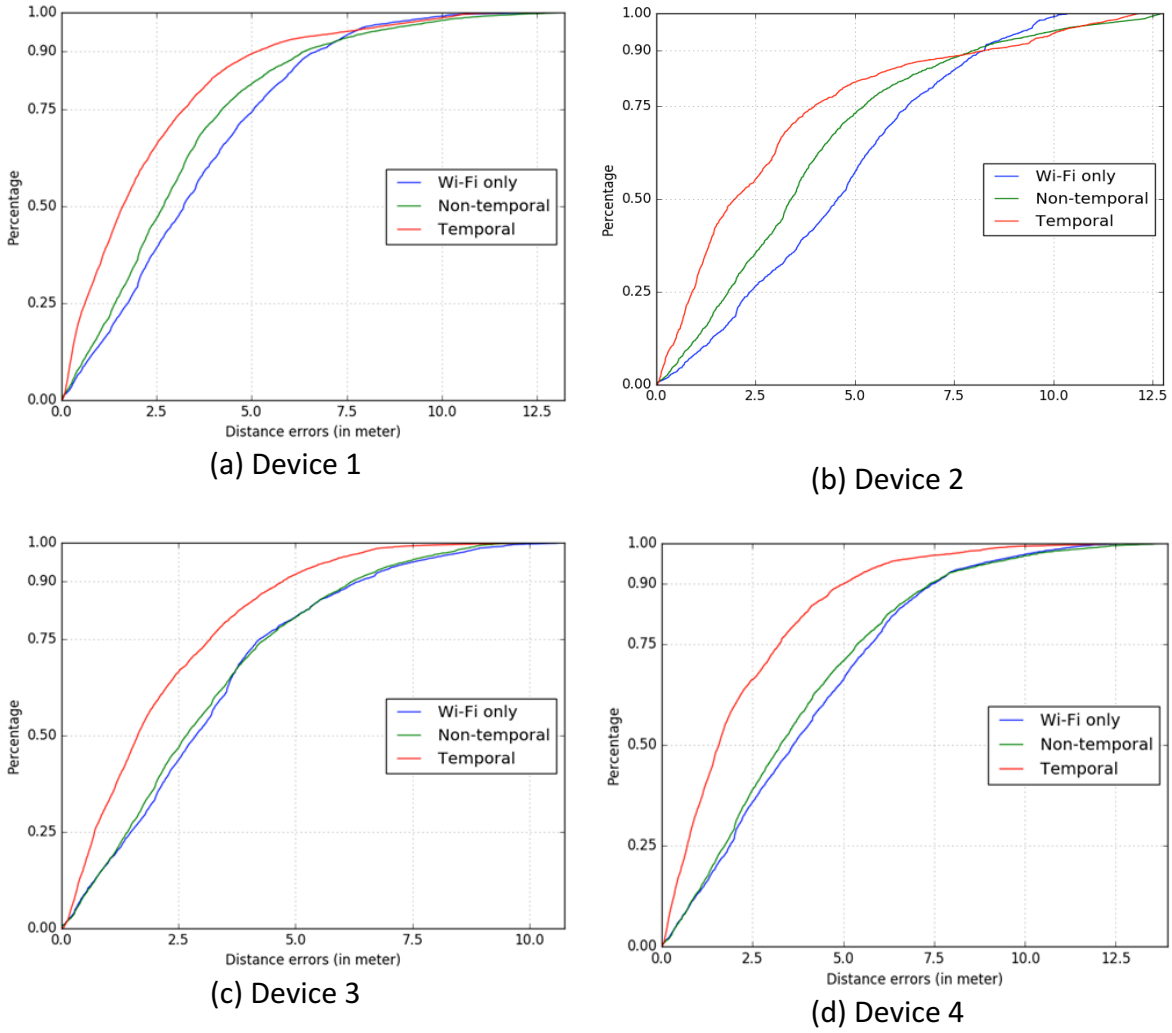


Figure 61. Performance comparison between three approaches for each device

5.5 Summary

In this chapter, we have presented a collaborative tracking framework based on the smartphone's Wi-Fi and Bluetooth scanning data. The Wi-Fi data is used as a raw positioning output, which is then improved by the relative distance from Bluetooth inquiry RSS signals. Two combination approaches are introduced, which is the *Non-temporal* approach and *Temporal* approach. The *Non-temporal* approach attempts to simplify the information fusion task by removing the time-relationship between different Wi-Fi scan and Bluetooth scan. The *Temporal* approach takes a more direct way to establish the conditions between the two types of data. Two approaches have been tested and compared with the standard Wi-Fi fingerprinting model. From the testing results, while the *Non-temporal* is only applicable in some specific scenarios, the *Temporal* approach outperforms the Wi-Fi fingerprinting models significantly.

The study has shown that the collaborative positioning would be applicable in a multi-user context. However, the testing scenario are still only evolved simple contexts of multiple users. There are also some remaining issues on the technical aspects, such as the communication between the users and the server, energy impact on the smartphone and signal inference. We are going to discuss these issues in future works.

Chapter 6 **Conclusion**

In this thesis, several aspects in the field of indoor localization by using smartphones have been presented. Among available technologies on the smartphones, Wi-Fi, inertial sensors and Bluetooth are selected.

The Wi-Fi data with fingerprinting approach is well-known for locating the devices within buildings. Its disadvantage is the need of large amount of training data for a stable performance. When there is a limited amount of data, the performance is likely to decrease. Our approach employs several learning models, features and targets to overcome the issue of limited training Wi-Fi data. The learning models are selected from the KNN, RF and XGB models. The features are selected from the raw RSS features, filtering-based features and HLF features. The learning target is chosen between classification learning and regression learning. The task of finding the smartphone's position is split into identifying its floor and its position in 2D space. By combining different models, features and learning targets, we have a wide range of models. The best performance on floor classification is 93.3% in accuracy and on the positioning is 5.75m in mean distance error. In order to improve the performance, several ways to develop different models are introduced. Thus, the floor accuracy has a slightly improvement, from 93.3% to 93.8%. Meanwhile, the mean distance error has a reduction over 0.5m. Although our approach has prominent results, the unstable characteristics of Wi-Fi signal still present in the results. The floor classification has issues with the areas near stairs and elevators. There are a number of positioning outputs with high distance error. Therefore, approaches employing inertial sensors and Bluetooth should be considered as alternative solutions.

Inertial sensors on smartphones include the accelerometer sensor, gyroscope sensor and magnetic sensor. The three types of data are used for building a SHS-based tracking. We first start from the standard SHS which attempts to identify the user's moving speed and heading direction. The accelerometer data is used to find the user's moving speed by the *Counting*

Step method and *Moving Window* method. Three approaches, including direction cosine matrix from accelerometer and magnetic data, *Complimentary Filter* and *Madgwick Filter* are employed for finding the heading direction. The two calculated values, moving speed and heading, are combined by a particle filter component. Based on the output from SHS, the tracking path is combined with the Wi-Fi output position with the aim to reduce the effect of SHS's drifting errors. The *Direct Adjust* combination shifts the SHS path the Wi-Fi output positioning. The combination weight is introduced to find an agreement between the SHS path and Wi-Fi output. The second combination approach attempts to build an observation model in the particle filter step. The observation model is used to score each particle by how far from it to the Wi-Fi's prediction probability. From the results on the experiment data, both the approaches could improve the SHS path and Wi-Fi output to some extends. The latter approach results in more smooth paths and lower mean distance error. It could avoid the extreme adjustment from the Wi-Fi output positions. However, the two combination approaches are unable to fix the drifting errors from the noisy sensors data in a systematic way. In future works, the usage of map information and opportunistic calibration process could be employed for stabilizing the tracking results over long time. We also want to address the effect of smartphone's positions to the performance of SHS systems.

Beside the Wi-Fi technology and inertial sensors technology, the Bluetooth technology is applicable for positioning purpose. In a multiple-user context, the Bluetooth data provides useful information for estimating the relative distance between different users. When each user could be localized by the Wi-Fi scan data, it is able to employ the relative distance to improve the Wi-Fi positioning output. In the *Non-temporal* combination, the approach attempts to remove the effect of the user's movement between different timestamps. It assumes that the Wi-Fi data and Bluetooth data are arrived at a same time. Therefore, for each pair of users, an error function is built to estimate the mismatch between the Wi-Fi output positions and the relative distance. The adjusted positions are the points which correspond to minimal error values. In the *Temporal* approach, the above error function is extended to include the users' movement through different timestamps. A particle filter based tracking is used to find the minimal error. A simple motion model with map information is used for replicating the movement in indoor environment. The observation model is built as a combination from the Wi-Fi

positioning output and Bluetooth relative distance. For evaluating the effectiveness of our proposed approaches, real experiments are carried out. The scenarios are varied from two to four users. The results indicate that both the *Non-temporal* and *Temporal* could improve the raw Wi-Fi positioning output to some extends. While the former only has significant improvement when the users move within one group, the latter is applicable across a wide range of scenarios. In general, the *Temporal* approach could reduce the mean distance error of the Wi-Fi output from 1m to 1.5m. In future works, we plan to address several aspects of the problems which include communication methods between the individual device and the server, the energy impact when using both Wi-Fi and Bluetooth scanning at the same time and the effects of signal interference of multiple devices. We also plan to evaluate the approach in larger areas.

References

Abel, J. S. and Chaffee, J. W. (1991). Existence and uniqueness of gps solutions. IEEE Transactions on Aerospace and Electronic Systems, 27(6):952–956.

Afzal, M. H., Renaudin, V., and Lachapelle, G. (2011). Magnetic field based heading estimation for pedestrian navigation environments. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, pages 1–10. IEEE.

Ali, A. A. and Omar, A. (2005). Time of arrival estimation for wlan indoor positioning systems using matrix pencil super resolution algorithm. In Proceedings of the 2nd Workshop on Positioning, Navigation and Communication(WPNC'05), pages 11–20.

Atiya, S. and Hager, G. D. (1993). Real-time vision-based robot localization. IEEE Transactions on Robotics and Automation, 9(6):785–800.

Aulinas, J., Petillot, Y. R., Salvi, J., and Lladó, X. (2008). The slam problem: a survey. CCIA, 184(1):363–371.

Bahl, P. and Padmanabhan, V. (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, volume 2, pages 775–784 vol.2.

Bandara, U., Hasegawa, M., Inoue, M., Morikawa, H., and Aoyama, T. (2004). Design and implementation of a bluetooth signal strength based location sensing system. In Radio and Wireless Conference, 2004 IEEE, pages 319–322.

Bargh, M. S. and de Groote, R. (2008). Indoor localization based on response rate of Bluetooth inquiries. In Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments, MELT '08, pages 49–54, New York, NY, USA. ACM.

Beauregard, S., Klepal, M., et al. (2008). Indoor PDR performance enhancement using minimal map information and particle filters. In Position, Location and Navigation Symposium, 2008 IEEE/ION, pages 141–147. IEEE.

Berkvens, R., Weyn, M., and Peremans, H. (2015). Localization performance quantification by conditional entropy. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–7.

Bernardos, A., Casar, J., and Tarrio, P. (2010). Real time calibration for rss indoor positioning systems. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2010, pages 1–7.

BH-Wellenhof, H. L. and Collins, J. (1997). Global Positioning System: theory and practice. Springer-Verlag.

Bird, J. and Arden, D. (2011). Indoor navigation with foot-mounted strapdown inertial navigation and magnetic sensors [emerging opportunities for localization and tracking]. IEEE Wireless Communications, 18(2):28–35.

Blewitt, G. (1997). Basics of the gps technique: observation equations. Geodetic applications of GPS, pages 10–54.

Bo, L., Ren, X., and Fox, D. (2010). Kernel descriptors for visual recognition. In Advances in neural information processing systems, pages 244–252.

Bose, A. and Foh, C. H. (2007). A practical path loss model for indoor wifi positioning enhancement. In Information, Communications & Signal Processing, 2007 6th International Conference on, pages 1–5. IEEE.

Breiman, L. (2001). Random forests. Machine learning, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). Classification and regression trees. CRC press.

Brunato, M. and Battiti, R. (2005). Statistical learning theory for location fingerprinting in wireless lans. Computer Networks, 47(6):825–845.

Cadman, J. (2003). Deploying commercial location-aware systems. In Proceedings of the 2003 Workshop on Location-Aware Computing (held as part of UbiComp 2003), pages 4–6.

Chen, L.-H., Wu, E. H.-K., Jin, M.-H., and Chen, G.-H. (2014). Intelligent fusion of wi-fi and inertial sensor-based positioning systems for indoor pedestrian navigation. IEEE Sensors Journal, 14(11):4034–4042.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 785–794. ACM.

Chen, Y.-C., Chiang, J.-R., Chu, H.-h., Huang, P., and Tsui, A. W. (2005). Sensor-assisted wi-fi indoor location system for adapting to environmental dynamics. In Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems, pages 118–125. ACM.

Cheng, Y.-C., Chawathe, Y., LaMarca, A., and Krumm, J. (2005). Accuracy characterization for metropolitan-scale wi-fi localization. In Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, MobiSys '05, pages 233–245, New York, NY, USA. ACM.

Cheung, K. W., So, H.-C., Ma, W.-K., and Chan, Y.-T. (2004). Least squares algorithms for time-of-arrival-based mobile location. Signal Processing, IEEE Transactions on, 52(4):1121–1130.

Chintalapudi, K., Padmanabha Iyer, A., and Padmanabhan, V. N. (2010). Indoor localization without the pain. In Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking, MobiCom '10, pages 173–184, New York, NY, USA. ACM.

Chon, H. D., Jun, S., Jung, H., and An, S. W. (2004). Using rfid for accurate positioning. Journal of Global Positioning Systems, 3(1-2):32–39.

Choukroun, D., Bar-Itzhack, I. Y., and Oshman, Y. (2006). Novel quaternion kalman filter. IEEE Transactions on Aerospace and Electronic Systems, 42(1):174–190.

Correa, A., Munoz Diaz, E., Bousdar Ahmed, D., Morell, A., and Lopez Vicario, J. (2016). Advanced pedestrian positioning system to smartphones and smartwatches. Sensors, 16(11):1903.

Dao, T.-K., Nguyen, H.-L., Pham, T.-T., Castelli, E., Nguyen, V.-T., and Nguyen, D.-V. (2014). User localization in complex environments by multimodal combination of gps, wifi, rfid, and pedometer technologies. The Scientific World Journal, 2014, no. 814538, 4-2014.

Dao, T. K., Pham, T. T., and Castelli, E. (2013). A robust wlan positioning system based on probabilistic propagation model. In 9th International Conference on Intelligent Environments, 2013, pages 24–29.

Della Rosa, F., Paakki, T., Leppakoski, H., and Nurmi, J. (2010). A cooperative framework for path loss calibration and indoor mobile positioning. In 7th Workshop on Positioning Navigation and Communication (WPNC), 2010, pages 86–92. IEEE.

Deng, Z.-A., Wang, G., Qin, D., Na, Z., Cui, Y., and Chen, J. (2016). Continuous indoor positioning fusing wifi, smartphone sensors and landmarks. Sensors, 16(9):1427.

Deretey, E., Ahmed, M. T., Marshall, J. A., and Greenspan, M. (2015). Visual indoor positioning with a single camera using pnp. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–9. IEEE.

Dinh-Van, N., Nashashibi, F., Thanh-Huong, N., & Castelli, E. (2017, March). Indoor Intelligent Vehicle localization using WiFi received signal strength indicator. In Microwaves for Intelligent Mobility (ICMIM), 2017 IEEE MTT-S International Conference on (pp. 33-36). IEEE.

Diebel, J. (2006). Representing attitude: Euler angles, unit quaternions, and rotation vectors. Matrix, 58(15-16):1–35.

Dippold, M. (2006). Personal dead reckoning with accelerometers. In Applied Wearable Computing (IFAWC), 2006 3rd International Forum on, pages 1–6. VDE.

Dong, F., Chen, Y., Liu, J., Ning, Q., and Piao, S. (2009). A calibration-free localization solution for handling signal strength variance. In Mobile Entity Localization and Tracking in GPS-less Environnments, pages 79–90. Springer.

Driusso, M., Marshall, C., Sabathy, M., Knutti, F., Mathis, H., and Babich, F. (2016). Indoor positioning using lte signals. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–8. IEEE.

Edel, M. and Köppe, E. (2015). An advanced method for pedestrian dead reckoning using blstm-rnns. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–6. IEEE.

Evennou, F. and Marx, F. (2006). Advanced integration of wifi and inertial navigation systems for indoor mobile positioning. Eurasip journal on applied signal processing, 2006:164–164.

Fang, S.-H. and Lin, T.-N. (2008). Indoor location system based on discriminant-adaptive neural network in ieee 802.11 environments. IEEE Transactions on Neural Networks, 19(11):1973–1978.

Faragher, R. and Harle, R. (2014). An analysis of the accuracy of bluetooth low energy for indoor positioning applications. In Proceedings of the 27th International Technical

Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2014), Tampa, FL, USA, volume 812.

Feliz Alonso, R., Zalama Casanova, E., and Gómez García-Bermejo, J. Pedestrian tracking using inertial sensors. Journal of Physical Agents, vol. 3 (1), pp. 35-42, 2009.

Fernandez-Prades, C., Presti, L. L., and Falletti, E. (2011). Satellite radiolocalization from gps to gnss and beyond: Novel technologies and applications for civil mass market. Proceedings of the IEEE, 99(11):1882–1904.

Ferris, B., Fox, D., and Lawrence, N. (2007). Wifi-slam using gaussian process latent variable models. In Proceedings of the 20th International Joint Conference on Artifical Intelligence, IJCAI'07, pages 2480–2485, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Filonenko, V., Cullen, C., and Carswell, J. (2010). Investigating ultrasonic positioning on mobile phones. In 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), , pages 1–8. IEEE

Fontana, R. J. and Gunderson, S. J. (2002). Ultra-wideband precision asset location system. In IEEE Conference on Ultra Wideband Systems and Technologies, 2002. Digest of Papers. 2002, pages 147–150. IEEE.

Fourati, H. (2015). Heterogeneous data fusion algorithm for pedestrian navigation via foot-mounted inertial measurement unit and complementary filter. IEEE Transactions on Instrumentation and Measurement, 64(1):221–229.

Foxlin, E. (2005). Pedestrian tracking with shoe-mounted inertial sensors. IEEE Computer graphics and applications, 25(6):38–46.

Fuentes-Pacheco, J., Ruiz-Ascencio, J., and Rendón-Mancha, J. M. (2015). Visual simultaneous localization and mapping: a survey. Artificial Intelligence Review, 43(1):55–81.

Gallagher, T., Li, B., Kealy, A., and Dempster, A. G. (2009). Trials of commercial wifi positioning systems for indoor and urban canyons. In IGNSS Symposium, 2009, Gold Coast, Australia, 1-3 December, CD-ROM procs.

Gentner, C. and Jost, T. (2013). Indoor positioning using time difference of arrival between multipath components. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013, pages 1–10. IEEE.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. Machine learning, 63(1):3–42.

Gezici, S., Tian, Z., Giannakis, G. B., Kobayashi, H., Molisch, A. F., Poor, H. V., and Sahinoglu, Z. (2005). Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. IEEE signal processing magazine, 22(4):70–84.

Godha, S. and Lachapelle, G. (2008). Foot mounted inertial system for pedestrian navigation. Measurement Science and Technology, 19(7):075202.

Golden, S. A. and Bateman, S. S. (2007). Sensor measurements for wi-fi location with emphasis on time-of-arrival ranging. Mobile Computing, IEEE Transactions on, 6(10):1185– 1198.

Gomez, C., Oller, J., and Paradells, J. (2012). Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. Sensors, 12(9):11734–11753.

Gozick, B., Subbu, K. P., Dantu, R., and Maeshiro, T. (2011). Magnetic maps for indoor navigation. IEEE Transactions on Instrumentation and Measurement, 60(12):3883–3891.

Griswold, W., Shanahan, P., Brown, S., Boyer, R., Ratto, M., Shapiro, R., and Truong, T. (2004). Activecampus: experiments in community-oriented ubiquitous computing. Computer, 37(10):73–81.

Günther, A. and Hoene, C. (2005). Measuring round trip times to determine the distance between wlan nodes. In International Conference on Research in Networking, pages 768–779. Springer.

Gustafsson, F. and Gunnarsson, F. (2005). Mobile positioning using wireless networks: possibilities and fundamental limitations based on available wireless network measurements. Signal Processing Magazine, IEEE, 22(4):41–53.

Haeberlen, A., Flannery, E., Ladd, A. M., Rudys, A., Wallach, D. S., and Kavraki, L. E. (2004). Practical robust localization over large-scale 802.11 wireless networks. In Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, MobiCom '04, pages 70–84, New York, NY, USA. ACM.

Hahnel, D., Burgard, W., Fox, D., Fishkin, K., and Philipose, M. (2004). Mapping and localization with rfid technology. In IEEE International Conference on Robotics and Automation, 2004. ICRA'04. Proceedings, volume 1, pages 1015–1020. IEEE.

Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM review, 53(2):217–288.

Harle, R. (2013). A survey of indoor inertial positioning systems for pedestrians. IEEE Communications Surveys and Tutorials, 15(3):1281–1293.

Harter, A., Hopper, A., Steggles, P., Ward, A., and Webster, P. (2002). The anatomy of a context-aware application. Wireless Networks, 8(2/3):187–197.

Hata, M. (1980). Empirical formula for propagation loss in land mobile radio services. IEEE transactions on Vehicular Technology, 29(3):317–325.

Hay, S. and Harle, R. (2009). Bluetooth tracking without discoverability. In Choudhury, T., Quigley, A., Strang, T., and Suginuma, K., editors, Location and Context Awareness,

volume 5561 of Lecture Notes in Computer Science, pages 120–137. Springer Berlin Heidelberg.

Hazas, M. and Hopper, A. (2006). Broadband ultrasonic location systems for improved indoor positioning. IEEE Transactions on mobile Computing, 5(5):536–547.

He, S. and Chan, S.-H. G. (2016). Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons. IEEE Communications Surveys & Tutorials, 18(1):466–490.

Henk Muller, Cliff Randell and Chris Djiallis (2003). Personal position measurement using dead reckoning. In Proceedings of the Seventh IEEE International Symposium on Wearable Computers (ISWC03), volume 1530, pages 166-173.

Hu, W., Tan, T., Wang, L., and Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 34(3):334–352.

Huang, J., Millman, D., Quigley, M., Stavens, D., Thrun, S., and Aggarwal, A. (2011). Efficient, generalized indoor wifi graphslam. In Robotics and Automation (ICRA), 2011 IEEE International Conference on, pages 1038–1043.

Ido, J., Shimizu, Y., Matsumoto, Y., and Ogasawara, T. (2009). Indoor navigation for a humanoid robot

Ikegami, F., Yoshida, S., Takeuchi, T., and Umehira, M. (1984). Propagation factors controlling mean field strength on urban streets. Antennas and Propagation, IEEE Transactions on, 32(8):822–829.

Jedari, E., Wu, Z., Rashidzadeh, R., and Saif, M. (2015). Wi-fi based indoor location positioning employing random forest classifier. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–5. IEEE.

Jimenez, A. R., Seco, F., Prieto, C., and Guevara, J. (2009). A comparison of pedestrian dead-reckoning algorithms using a low-cost mems imu. In Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on, pages 37–42. IEEE.

Jun, J., Gu, Y., Cheng, L., Lu, B., Sun, J., Zhu, T., and Niu, J. (2013). Social-loc: Improving indoor localization with social sensing. In Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems, page 14. ACM.

Juri, A. A., Arslan, T., Du, Y., and Wang, Z. (2016). Dual scaling and sub-model based pnp algorithm for indoor positioning based on optical sensing using smartphones. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–5. IEEE.

Kaemarungsi, K. and Krishnamurthy, P. (2004). Properties of indoor received signal strength for wlan location fingerprinting. In MobiQuitous, pages 14–23. IEEE Computer Society.

King, T., Kopf, S., Haenselmann, T., Lubberger, C., and Effelsberg, W. (2006). Compass: A probabilistic indoor positioning system based on 802.11 and digital compasses. In Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization, pages 34–40. ACM.

Kjaergaard, M. and Munk, C. (2008). Hyperbolic location fingerprinting: A calibration-free solution for handling differences in signal strength (concise contribution). In Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on, pages 110–116.

Kontkanen, P., Myllymaki, P., Roos, T., Tirri, H., Valtonen, K., and Wettig, H. (2004). Topics in probabilistic location estimation in wireless networks. In Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004. 15th IEEE International Symposium on, volume 2, pages 1052–1056. IEEE.

Kothari, N., Kannan, B., Glasgwow, E. D., and Dias, M. B. (2012). Robust indoor localization on a commercial smart phone. Procedia Computer Science, 10:1114–1120.

Krach, B. and Roberston, P. (2008). Cascaded estimation architecture for integration of foot-mounted inertial sensors. In Position, Location and Navigation Symposium, 2008 IEEE/ION, pages 112–119. IEEE.

Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernández, G., Vojir, T., Hager, G., Nebehay, G., and Pflugfelder, R. (2015). The visual object tracking vot2015 challenge results. In Proceedings of the IEEE international conference on computer vision workshops, pages 1–23.

Li, B., Salter, J., Dempster, A. G., and Rizos, C. (2006). Indoor positioning techniques based on wireless lan. Technical report, School of Surveying and Spatial Information Systems, UNSW, Sydney, Australia.

Lin, Y. and Jeon, Y. (2002). Random forests and adaptive nearest neighbors. Journal of the American Statistical Association, pages 101–474.

Link, J. A. B., Smith, P., Viol, N., and Wehrle, K. (2011). Footpath: Accurate map-based indoor navigation using smartphones. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, pages 1–8. IEEE.

Liu, H., Darabi, H., Banerjee, P., and Liu, J. (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.

Liu, H., Gan, Y., Yang, J., Sidhom, S., Wang, Y., Chen, Y., and Ye, F. (2012). Push the limit of wifi based localization for smartphones. In Proceedings of the 18th Annual International Conference on Mobile Computing and Networking, Mobicom ’12, pages 305–316, New York, NY, USA. ACM.

Liu, S., Jiang, Y., and Striegel, A. (2014). Face-to-face proximity estimationusing bluetooth on smartphones. IEEE Transactions on Mobile Computing, 13(4):811–823.

Liu, W., Fu, X., Deng, Z., Xu, L., and Jiao, J. (2016). Smallest enclosing circle-based fingerprint clustering and modified-WKNN matching algorithm for indoor positioning. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–6. IEEE.

Luca, D. G. and Alberto, M. (2016). Towards accurate indoor localization using ibeacons, fingerprinting and particle filtering. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016. IEEE.

Luhmann, T., Robson, S., Kyle, S., and Harley, I. (2006). Close range photogrammetry: principles, techniques and applications. Whittles.

Ma, R., Guo, Q., Hu, C., and Xue, J. (2015). An improved Wi-Fi indoor positioning algorithm by weighted fusion. Sensors, 15(9):21824–21843.

Madgwick, S. O. (2010). An efficient orientation filter for inertial and inertial/magnetic sensor arrays. Report x-io and University of Bristol (UK).

Madhavapeddy, A. and Tse, A. (2005). A study of bluetooth propagation using accurate indoor location mapping. In Proceedings of the 7th International Conference on Ubiquitous Computing, UbiComp'05, pages 105–122, Berlin, Heidelberg. Springer-Verlag.

Mahony, R., Hamel, T., and Pflimlin, J.-M. (2005). Complementary filter design on the special orthogonal group so (3). In Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on, pages 1477–1484. IEEE.

Mahony, R., Hamel, T., and Pflimlin, J.-M. (2008). Nonlinear complementary filters on the special orthogonal group. IEEE Transactions on automatic control, 53(5):1203–1218

Marouane, C., Ebert, A., Linnhoff-Popien, C., and Christil, M. (2016). Step and activity detection based on the orientation and scale attributes of the surf algorithm. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–8. IEEE.

Marques, N., Meneses, F., and Moreira, A. (2012). Combining similarity functions and majority rules for multi-building, multi-floor, wifi positioning. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2012, pages 1–9. IEEE.

Mathisen, A., Sørensen, S. K., Stisen, A., Blunck, H., and Grønbæk, K. (2016). A comparative analysis of indoor wifi positioning at a large building complex. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–8. IEEE

Mautz, R. and Tilch, S. (2011). Survey of optical indoor positioning systems. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, pages 1–7.

Michel, T., Fourati, H., Geneves, P., and Layaïda, N. (2015). A comparative analysis of attitude estimation for pedestrian navigation with smartphones. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–10. IEEE.

Miller, M. M., Soloviev, A., Uijt de Haag, M., Veth, M., Raquet, J., Klausutis, T. J., and Touma, J. E. (2010). Navigation in gps denied environments: feature-aided inertial systems. Technical report, DTIC Document.

Misra, P. and Enge, P. (2006). Global positioning system: Signals, measurements and performance second edition. Massachusetts: Ganga-Jamuna Press

Moore, T. (2002). An introduction to differential gps. Lecture Notes, Institute of Engineering Surveying and Space Geodesy (IESSG) – University of Nottingham (UK).

Moreira, A., Nicolau, M. J., Meneses, F., and Costa, A. (2015). Wi-fi fingerprinting in the real world - rtls@um at the evaal competition. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2015, pages 1–10.

Mourcou, Q., Fleury, A., Franco, C., Klopcic, F., and Vuillerme, N. (2015). Performance evaluation of smartphone inertial sensors measurement for range of motion. Sensors, 15(9):23168–23187.

Mulloni, A., Wagner, D., Barakonyi, I., and Schmalstieg, D. (2009). Indoor positioning and navigation with camera phones. IEEE Pervasive Computing, 8(2).

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. IEEE Transactions on Robotics, 31(5):1147–1163.

Nam, H. and Han, B. (2016). Learning multi-domain convolutional neural networks for visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4293–4302.

Ni, L. M., Liu, Y., Lau, Y. C., and Patil, A. P. (2004). Landmarc: indoor location sensing using active rfid. Wireless networks, 10(6):701–710.

Niculescu, D. and Nath, B. (2003). Ad hoc positioning system (aps) using aoa. In INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, volume 3, pages 1734–1743 vol.3.

Park, J.-g., Charrow, B., Curtis, D., Battat, J., Minkov, E., Hicks, J., Teller, S., and Ledlie, J. (2010). Growing an organic indoor location system. In Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys '10, pages 271–284, New York, NY, USA. ACM.

Park, S. and Hashimoto, S. (2009). Indoor localization for autonomous mobile robot based on passive rfid. In IEEE International Conference on Robotics and Biomimetics, 2008. ROBIO 2008., pages 1856–1861. IEEE.

Pei, L., Chen, R., Liu, J., Kuusniemi, H., Tenhunen, T., and Chen, Y. (2010). Using inquiry-based bluetooth rssi probability distributions for indoor positioning. Journal of Global Positioning Systems, 9(2):122–130.

Pirker, K., Rüther, M., and Bischof, H. (2011). Cd slam-continuous localization and mapping in a dynamic world. In Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pages 3990–3997. IEEE.

Popleteev, A. (2011). Indoor positioning using FM radio signals. PhD thesis, University of Trento.

Potortì, F., Barsocchi, P., Girolami, M., Torres-Sospedra, J., and Montoliu, R. (2015). Evaluating indoor localization solutions in large environments through competitive benchmarking: The evaal-etri competition. In Indoor Positioning and Indoor Navigation (IPIN), 2015 International Conference on, pages 1–10. IEEE.

Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. (2000). The cricket location-support system. In Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, MobiCom '00, pages 32–43, New York, NY, USA. ACM.

Qian, J., Pei, L., Ma, J., Ying, R., and Liu, P. (2015). Vector graph assisted pedestrian dead reckoning using an unconstrained smartphone. Sensors, 15(3):5032–5057.

Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1):81–106.

Rai, A., Chintalapudi, K. K., Padmanabhan, V. N., and Sen, R. (2012). Zee: Zero-effort crowdsourcing for indoor localization. In Proceedings of the 18th annual international conference on Mobile computing and networking, pages 293–304. ACM.

Rappaport, T. S. et al. (1996). Wireless communications: principles and practice, volume 2. prentice hall PTR New Jersey.

Renaudin, V. and Combettes, C. (2014). Magnetic, acceleration fields and gyroscope quaternion (magyq)-based attitude estimation with smartphone sensors for indoor pedestrian navigation. Sensors, 14(12):22864–22890.

Retscher, G. and Joksch, J. (2016). Analysis of nine vector distances for fingerprinting in multiple-ssid wi-fi networks. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 4–6.

Saha, S., Chaudhuri, K., Sanghi, D., and Bhagwat, P. (2003). Location determination of a mobile device using ieee 802.11 b access point signals. In Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE, volume 3, pages 1987–1992. IEEE.

Se, S., Lowe, D. G., and Little, J. J. (2005). Vision-based global localization and mapping for mobile robots. IEEE Transactions on robotics, 21(3):364–375.

Seco, F., Plagemann, C., Jiménez, A. R., and Burgard, W. (2010). Improving rfid-based indoor positioning accuracy using gaussian processes. In Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on, pages 1–8. IEEE.

Shen, J. and Oda, Y. (2010). Direction estimation for cellular enhanced cell-id positioning using multiple sector observations. In Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on, pages 1–6. IEEE.

Smeulders, A. W., Chu, D. M., Cucchiara, R., Calderara, S., Dehghan, A., and Shah, M. (2014). Visual tracking: An experimental survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(7):1442–1468

Soloviev, A., Bates, D., and GRAAS, F. (2007). Tight coupling of laser scanner and inertial measurements for a fully autonomous relative navigation solution. Navigation, 54(3):189– 205.

Steinhoff, U. and Schiele, B. (2010). Dead reckoning from the pocket-an experimental study. In Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on, pages 162–170. IEEE.

Strasdat, H., Davison, A. J., Montiel, J. M., and Konolige, K. (2011). Double window optimisation for constant time visual slam. In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 2352–2359. IEEE.

Ta, V. C., Vaufreydaz, D., Dao, T. K., and Castelli, E. (2016). Smartphone-based user location tracking in indoor environment. In 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN), pages 1–8.

Tayebi, A., Gomez Perez, J., Adana Herrero, F. M. S. d., and Gutierrez, O. (2009). The application of ray-tracing to mobile localization using the direction of arrival and received signal strength in multipath indoor environments. Progress In Electromagnetics Research, 91:1–15.

Thi-Thanh-Thuy Pham, Thi-Lan Le, T.-K. D. (2016). Person re-identification for nonoverlapping cameras in multimodal person localization. International Journal on Advances in Systems and Measurements, 9(1–2):102–111.

Torres-Sospedra, J., Jiménez, A. R., Knauth, S., Moreira, A., Beer, Y., Fetzer, T., Ta, V.-C., Montoliu, R., Seco, F., Mendoza-Silva, G. M., Belmonte, O., Koukofikis, A., Nicolau, M. J., Costa, A., Meneses, F., Ebner, F., Deinzer, F., Vaufreydaz, D., Dao, T.-K., and Castelli, E. (2017). The smartphone-based offline indoor location competition at IPIN 2016: Analysis and future work. Sensors, 17(3).

Torres-Sospedra, J., Mendoza-Silva, G. M., Montoliu, R., Belmonte, O., Benitez, F., and Huerta, J. (2016). Ensembles of indoor positioning systems based on fingerprinting: Simplifying parameter selection and obtaining robust systems. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–8. IEEE

Trevisani, E. and Vitaletti, A. (2004). Cell-id location technique, limits and benefits: an experimental study. In Mobile computing systems and applications, 2004. WMCSA 2004. Sixth IEEE workshop on, pages 51–60. IEEE.

Trucco, E. and Plakas, K. (2006). Video tracking: a concise survey. IEEE Journal of Oceanic Engineering, 31(2):520–529.

Tsui, A. W., Chuang, Y.-H., and Chu, H.-H. (2009). Unsupervised learning for solving rss hardware variance problem in wifi localization. Mob. Netw. Appl., 14(5):677–691.

Van Diggelen, F. S. T. (2009). A-GPS: Assisted GPS, GNSS, and SBAS. Artech House.

Varshavsky, A., de Lara, E., Hightower, J., LaMarca, A., and Otsason, V. (2007). Gsm indoor localization. Pervasive and Mobile Computing, 3(6):698–720.

Wang, C.-H., Kao, T.-W., Fang, S.-H., Tsao, Y., Kuo, L.-C., Shih-Wei, K., and Lin, N.-C. (2013). Robust wi-fi location fingerprinting against device diversity based on spatial mean normalization. In Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific, pages 1–4.

Wang, H., Lenz, H., Szabo, A., Bamberger, J., and Hanebeck, U. D. (2007). Wlan-based pedestrian tracking using particle filters and low-cost mems sensors. In Positioning, Navigation and Communication, 2007. WPNC'07. 4th Workshop on, pages 1–7. IEEE.

Wang, H., Sen, S., Elgohary, A., Farid, M., Youssef, M., and Choudhury, R. R. (2012). No need to war-drive: Unsupervised indoor localization. In Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, MobiSys '12, pages 197–210, New York, NY, USA. ACM.

Wang, J. (2002). Pseudolite applications in positioning and navigation: Pro-gress and problems. Positioning, 1(03):0.

Wang, L., Ouyang, W., Wang, X., and Lu, H. (2015). Visual tracking with fully convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, pages 3119–3127.

Wang, Q., Zhang, X., Chen, X., Chen, R., Chen, W., and Chen, Y. (2010). A novel pedestrian dead reckoning algorithm using wearable emg sensors to measure walking strides. In Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010, pages 1–8. IEEE.

Wang, X., Jiang, M., Guo, Z., Hu, N., Sun, Z., and Liu, J. (2016). An indoor positioning method for smartphones using landmarks and pdr. Sensors, 16(12):2135.

Wang, Y. and Lu, K. (2017). Challenge of multi-camera tracking. CoRR, abs/1702.01507

Woodman, O. J. (2007). An introduction to inertial navigation. Technical report, University of Cambridge, Computer Laboratory.

Wu, C.-L., Fu, L.-C., and Lian, F.-L. (2004). Wlan location determination in e-home via support vector classification. In Networking, sensing and control, 2004 IEEE international conference on, volume 2, pages 1026–1031. IEEE.

Xiang, Z., Song, S., Chen, J., Wang, H., Huang, J., and Gao, X. (2004). A wireless lan-based indoor positioning technology. IBM Journal of research and development, 48(5.6):617– 626.

Xu, Q., Zheng, R., and Hranilovic, S. (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, pages 307–318. ACM.

Yadav, N. and Bleakley, C. (2014). Accurate orientation estimation using ahrs under conditions of magnetic distortion. Sensors, 14(11):20008–20024.

Youssef, M. and Agrawala, A. (2005). The horus wlan location determination system. In Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, MobiSys '05, pages 205–218, New York, NY, USA. ACM.

Zandbergen, P. A. (2009). Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning. Transactions in GIS, 13(s1):5–25.

Zhang, J., Li, B., Dempster, A. G., and Rizos, C. (2010). Evaluation of high sensitivity gps receivers. In 2010 Int. Symp. On GPS/GNSS, pages 410–415.

Zhang, W., Hua, X., Yu, K., Qiu, W., and Zhang, S. (2016). Domain clustering based wifi indoor positioning algorithm. In International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, pages 1–5. IEEE.

Zhang, Z., Scanlon, A., Yin, W., Yu, L., and Venetianer, P. L. (2009). Video surveillance using a multi-camera tracking and fusion system. Multi-Camera Networks: Principles and Applications, pages 435–456.

Zhou, M., Xu, Y., and Tang, L. (2010). Multilayer ann indoor location system with area division in wlan environment. Journal of Systems Engineering and Electronics, 21(5):914– 926.

Zhou, P., Li, M., and Shen, G. (2014). Use it free: Instantly knowing your phone attitude. In Proceedings of the 20th annual international conference on Mobile computing and networking, pages 605–616. ACM.

Zhuang, Y., Yang, J., Li, Y., Qi, L., and El-Sheimy, N. (2016). Smartphone-based indoor localization with bluetooth low energy beacons. Sensors, 16(5).