

Master's Thesis

Wi-Fi based Localization of Cell Phone

Chengqi Ma
Xiao Hu





Master's Thesis

Wi-Fi based Localization of Cell Phone

by

Chengqi Ma and Xiao Hu

Department of Electrical and Information Technology
Faculty of Engineering, LTH, Lund University
SE-221 00 Lund, Sweden

ABSTRACT

Accurate positioning and tracking system for mobile devices has become very popular in recent years, both in research and practice. In indoor environments, one of the promising approaches is fingerprinting localization technique which is based on received Wi-Fi signal strengths. The objective of this master thesis is to construct an indoor positioning and tracking system with fingerprinting.

The first part of the thesis considers a suitable hardware arrangement based on low cost Raspberry Pis. In a specific indoor environment, five arranged Raspberry Pis are utilized to collect fingerprints – the Wi-Fi signal strengths which are broadcasted from surrounding mobile devices.

The second part introduces how to establish a fingerprint database. Statistical analyses for signal strengths distribution in experiment environment have been discussed. Moreover, since collected data are truncated due to the noise floor, the Expectation Maximization (EM) algorithm has been applied to estimate the mean signal strength at each reference point.

On the positioning and tracking phase, the authors present two different approaches for data processing. The received signal strengths are matched to coordinates on the map of experiment environment by fingerprinting. Furthermore, a self-designed Kalman filter has been applied to increase the tracking accuracy.

Finally, two types of indoor environment tracking systems are implemented. Tracking error analysis for both systems and performance comparisons are presented as well. Analyses of accuracy influence factors such as target movement speed and shadowing are given at the end of the thesis together with possible methods against these factors.

Index terms: **Indoor environment tracking, Fingerprinting technique, Raspberry Pi, Kalman filter, EM algorithm for truncated data, Evil Twins Attack**

ACKNOWLEDGEMENTS

First, we would like to show appreciation to our supervisor Mr. Dimitrios Vlastaras. He helped so much to solve the practical problems and gave us valuable feedback with angelic patience. This research would not exist without his professional guidance and encouragement.

Second, we wish to give thanks to our examiner Prof. Fredrik Tufvesson. He put forward this interesting topic in the beginning and recommended Dimitrios as our supervisor. He also provided great advice on the project and ordered equipment for us generously.

Finally, we would like to thank our friend Dong Wang. She gave us many helps in revising the thesis report.

Chengqi & Xiao

CONTENTS

ABSTRACT	II
ACKNOWLEDGEMENTS	III
CONTENTS.....	IV
LIST OF FIGURES	VII
LIST OF ABBREVIATIONS AND ACRONYMS	IX
1 INTRODUCTION.....	10
1.1 LOCALIZATION TECHNIQUES FOR CELL PHONE.....	10
1.1.1 <i>Global Positioning System (GPS)</i>	10
1.1.2 <i>Wireless Positioning Techniques</i>	10
1.2 WHAT IS FINGERPRINTING?	12
1.3 EUCLIDEAN DISTANCE-BASED MATCHING ALGORITHM.....	13
1.4 PROBLEM DEFINITION.....	13
1.5 THESIS SCOPE.....	14
2 SYSTEM FRAMEWORK	15
2.1 SNOOPY FRAMEWORK.....	15
2.1.1 <i>Brief Introduction of Snoopy Framework</i>	15
2.1.2 <i>Aircrack-ng</i>	17
2.1.3 <i>Tshark</i>	17
2.2 WI-FI PROBE REQUEST	18
2.3 EVIL TWIN ATTACK	18
2.4 SYSTEM FRAMEWORK	19
3 DESIGN AND OPERATION	20
3.1 PREPARATION	20
3.2 OPERATION PROCEDURE.....	22
3.2.1 <i>Local Time Synchronization</i>	22
3.2.2 <i>Rouge AP and Sniffer Setup</i>	23
3.3 LIMITATION	26
4 OFFLINE PHASE OF FINGERPRINTING LOCALIZATION	27

4.1 REFERENCE POINTS ALLOCATION	27
4.2 POSITION OF SNIFFERS	29
4.3 CALIBRATION	29
4.3.1 Fingerprint Database Structure.....	29
4.3.2 Data Collecting Operation.....	30
4.3.3 Estimation of Signal Strength.....	30
5 KALMAN FILTER DESIGN.....	37
5.1 DISCRETE KALMAN FILTER.....	37
5.2 TWO PHASES OF KALMAN FILTER.....	38
5.3 KALMAN FILTER IMPLEMENTATION	39
5.3.1 Process Noise Q	41
5.3.2 Observation Noise R	41
5.3.3 Optimal Values of Q and R	43
5.4 FILTERED TRACKING RESULT	44
6 TRACKING SYSTEM ONE	45
6.1 RULES OF DATA PROCESSING.....	45
6.2 MATCHING PHASE	47
6.2.1 Nearest-Neighbour (NN) Algorithm	47
6.2.2 K-Nearest-Neighbours (KNN) Algorithm.....	47
6.2.3 The Initial Value of the Kalman Filter	48
6.3 ERROR ANALYSIS.....	49
6.4 TRACKING RESULTS AND EVALUATION	50
7 TRACKING SYSTEM TWO.....	52
7.1 TIME AXIS SYNCHRONIZATION	52
7.1.1 Measurement Data Analysis	53
7.1.2 Method of Data Processing.....	55
7.2 MATCHING PHASE	55
7.3 MATCHING RESULTS AND EVALUATION	56
8 INFLUENCE FACTORS.....	59
8.1 SPEED OF TARGET	59

8.2 SHADOWING	60
8.3 TYPE OF MOBILE DEVICE.....	62
8.4 COMPARISON AND ERROR DISTRIBUTION.....	63
8.5 IMPROVEMENT	65
9 CONCLUSIONS AND FUTURE WORKS.....	67
9.1 CONTRIBUTIONS.....	67
9.2 DRAWBACKS.....	68
9.3 FUTURE WORK.....	68
REFERENCE.....	69

LIST OF FIGURES

- Figure 1.1. Illustration of AOA Based Positioning
- Figure 1.2. Illustration of Propagation-time Based Positioning
- Figure 1.3. Wi-Fi Fingerprinting System
- Figure 2.1. Architecture of Snoopy Framework
- Figure 2.2. Process of Evil Twin Attack
- Figure 2.3. Schematic Diagram of System Framework
- Figure 3.1. Model B Raspberry Pi
- Figure 3.2. An Example of Scanning Result
- Figure 3.3. Example of Data Frame
- Figure 4.1. Floor Plan with Reference Points Distribution and Sniffers
- Figure 4.2. Positions of Sniffers
- Figure 4.3. An Example of Fingerprint Database
- Figure 4.4. Structure of Raw Calibration Data
- Figure 4.5. An Example for Signal Strength Distribution by Histogram
- Figure 4.6. Original Measured RSS Distribution with Histogram Estimation
- Figure 4.7. RSS Distribution with EM Algorithm
- Figure 5.1. Operation of Kalman filter
- Figure 5.2. Tracking Result with Kalman Filter Along x-axis
- Figure 5.3. Tracking Result with Kalman Filter Along y-axis
- Figure 5.4. Tracking Result with Kalman Filter Along the Diagonal
- Figure 5.5. Tracking Result with Kalman Filter
- Figure 6.1. Structure of Measurement Data from One Sniffer
- Figure 6.2. Structure of the Processed Data
- Figure 6.3. The Division of the Corridor Map
- Figure 6.4. Tracking Result with EM Algorithm from Door to Door
- Figure 6.5. Tracking Result with EM Algorithm Along the Square
- Figure 7.1. Structures of Measurement Data from Five Sniffers Separately
- Figure 7.2. Structure of Measurement Data After Adjustment

Figure 7.3. Tracking Result with Method One from Door to Door

Figure 7.4. Tracking Result with Method Two from Door to Door

Figure 7.5. Tracking Result with Method Two Along the Square

Figure 8.1. Tracking Result with Tracking System One with Normal Speed 1.3m/s (a) and High Speed 2.5m/s (b)

Figure 8.2. Tracking Result with Tracking System Two with Normal Speed 1.3m/s (a) and High Speed 2.5m/s (b)

Figure 8.3. Tracking Result with Tracking System One with Device in Hand (a) and in Pocket (b)

Figure 8.4. Tracking Result with Tracking System Two with Device in Hand (a) and in Pocket (b)

Figure 8.5. Tracking Result with Tracking System One with Device iPad mini (a) and iPhone 4s (b)

Figure 8.6. Tracking Result with Tracking System Two with Device iPad mini (a) and iPhone 4s (b)

Figure 8.7. CDF of Error Distance with Tracking System One (a) and Two (b)

Figure 8.8. Error Distance with Coefficient K

LIST OF ABBREVIATIONS AND ACRONYMS

RSS	Received Signal Strength
AP	Access Point
GPS	Global Positioning System
AOA	Angle of Arrival
TOA	Time of Arrival
RTOF	Roundtrip Time of Flight
WPS	Wi-Fi-based positioning system
EncryMode	Encryption Mode
SSID	Service Set Identifier
RTC	Real Time Clock
NTP	Network Time Protocol
CSV	Comma-separated Values
CDF	Cumulative Distribution Function
NN	Nearest Neighbour Algorithm
KNN	K-Nearest Neighbour Algorithm
MLP	Multi-Layer Perceptron Neural Network
GRNN	Generalized Radial Neural Network
WEP	Wired Equivalent Privacy
WPA-PSK	Wi-Fi Protected Access - Pre-shared Key
DNS	Domain Name System
ARP	Address Resolution Protocol
RMSE	Root-Mean-Square Error
RMS	Root-Mean-Square

1 INTRODUCTION

In modern society, cell phone tracking systems are demanded in many areas such as security and human behaviour research. Some mature technologies such as Global Positioning System (GPS) has been well developed. However, research about accurate positioning system in indoor environments is continuously being studied. In this thesis, we design a feasible tracking system for mobile devices in a certain indoor environment. The system is based on the Wi-Fi fingerprinting technique. This chapter covers background knowledge about mobile device positioning and tracking. Some popular technologies of cell phone localization are briefly introduced as well.

1.1 Localization Techniques for Cell Phone

1.1.1 Global Positioning System (GPS)

GPS is the most popular technique for cell phone positioning. The GPS project was developed in 1973 to overcome the limitations of previous navigation systems. Nowadays, GPS is already embedded in most mobile devices and it can provide a good positioning accuracy. However, because the GPS signal will be reflected or scattered by walls and roofs of a building, it is not suitable for indoor environments localization [1][2]. Thus, it is necessary to realize an indoor environment tracking system rely on other kinds of radio signals or sensor data.

1.1.2 Wireless Positioning Techniques

For most wireless devices positioning applications, the localization methods normally rely on three aspects - Angle of Arrival (AOA), Received Signal Strength (RSS) and Propagation-time.

Taking the receiver as the reference point, AOA is the angle from which a signal arrives. The target position can be estimated by the intersection of several pairs of angle

direction lines [3]. Figure 1.1 is an illustration of AOA based positioning. Normally, it requires at least two known reference points and directive antennas or antenna arrays.

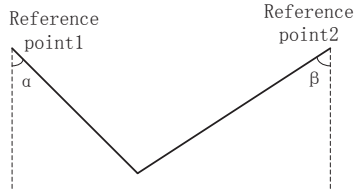


Figure 1.1. Illustration of AOA Based Positioning

The RSS based localization has two types of applications. The first one is based on propagation-loss equations, which requires advanced channel models in complex environments. The second one is fingerprinting which is based on the distribution of measured signal strength and doesn't rely on any channel model.

As for the propagation-time based localization, it can be based on either one-way propagation time i.e. Time of Arrival (TOA) or roundtrip time i.e. Roundtrip Time of Flight (RTOF). The distance between the target and the measurement equipment can be calculated by signal velocity and travel time. This method, especially TOA based, requires precise synchronization of all involved units. Figure 1.2 is an example of triangulation application with TOA. It uses the geometric properties of triangles to estimate the target location. [3]

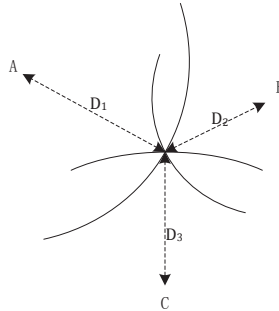


Figure 1.2. Illustration of Propagation-time Based Positioning

All kinds of existing wireless infrastructures, such as Bluetooth, Wi-Fi or specific sensors, have potentials to realize a tracking system. However, the tracking accuracy can be very different based on different devices and tracking algorithms. One of the most widely used techniques is Wi-Fi-based positioning system (WPS). As mentioned before, since GPS is no longer suitable for indoor environments positioning and wireless access points (AP) which normally exist in office, restaurant and other public places, it is a good choice to develop positioning technologies which rely on the Wi-Fi signals. The principle of WPS is based on measuring the intensity of the received signal - the RSS. [4]

1.2 What Is Fingerprinting?

Wi-Fi fingerprinting technique is normally used in indoor environments with multiple APs. It does not rely on the angle or distance measurement from the transmitters, and the channel model may not be considered. Instead, only Wi-Fi signal strengths are utilized for localization. Sensors or measurement units collect the Wi-Fi RSS as fingerprint from surrounding mobile devices. Then, position of target can be estimated by comparing the RSS with a database containing field strength information of different positions [5].

Typically, the fingerprinting system consists of two phases. On the first phase, called “Offline” phase, a “radio map” of the experiment environment must be built. The researchers arrange a large number of reference points, which cover the whole indoor area uniformly. At each reference point, a standard mobile device is used for recording the RSSs broadcasted from surrounding APs. A special fingerprint is a RSS vector including several RSS values from different APs. It is a unique identifier which corresponds to the coordinates of the current reference point. All the RSS vectors compose a fingerprint database. Works in this phase is also called “Calibration”. The second phase is the “Online” phase. When a mobile device enters the region of radio map, it starts to collect RSSs from surrounding APs somewhere. Through comparing the collected RSS vector with those in the established fingerprint database, the best matched fingerprint for the measurement point indicates the corresponding reference point. The coordinate of this reference point is the estimated position of the mobile device [6].

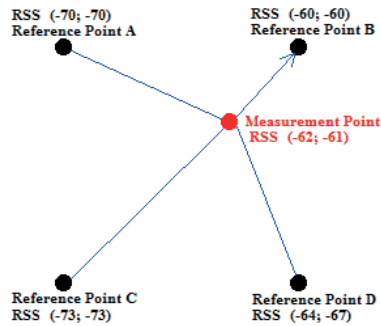


Figure1.3. Wi-Fi Fingerprinting System

A sketch of a typical fingerprinting system is shown in figure 1.3. The vectors of RSS at reference point A, B, C and D are the calibration results in database. The measurement point is the “Online” collected data. The distances marked by blue lines are used for matching the measurement point with a reference point in the calibration database. Obviously, the distance between reference point B and the measurement point is the shortest. It illustrates that point B is the best estimation of the current position.

In our system, the fingerprinting is applied in a reverse way. Instead of using a mobile device to collect APs signal strengths, the APs will collect the broadcasted signal strength of a mobile device. So the target position will be characterized by several RSSs at the AP side. For a traditional fingerprinting system, the tracked target has to be

programmed to collect RSSs from APs. However, in our system, through operating APs, any mobile device can be tracked.

1.3 Euclidean Distance-based Matching Algorithm

The matching algorithm always plays an important role in fingerprinting. For most cases, the measured RSS vector from online phase is not identical with any vector in the fingerprint database. An appropriate matching algorithm is necessary for estimating the user's location through compare the measurement results with the fingerprint database. Also the accuracy of localization can be greatly influenced by different algorithms.

Nearest Neighbour (NN), K-Nearest Neighbours (KNN), Multi-Layer Perceptron Neural Network (MLP) and Generalized Radial Neural Network (GRNN) are common matching algorithms which are based on the Euclidean Distance. Research [7] shows that the KNN algorithm gives the best localization accuracy and generally applied as the matching algorithm of various positioning systems. Details of KNN algorithm and other Euclidean Distance-based Matching Algorithms will be discussed in Chapter 6.

1.4 Problem Definition

In order to establish a complete cell phone tracking system, there are several problems need to be addressed.

First, it is important to select a suitable device to implement data measurement. Both measurement accuracy and equipment costs should be considered. It is obvious that better measurement equipment could lead to a higher accuracy of positioning. However, precision sensor is usually expensive and not easy to buy. The thesis objective is to realize a tracking system with devices which are normal in market and easy to install.

Second, how to establish a good fingerprints database. This project is based on fingerprinting which gets several advantages compare to other methods. As mentioned before, different methods require different crucial measurement data. However, fingerprinting does not require complex channel modelling or precise synchronization. It just requires a fingerprint database to be compared with. So it is extremely important to establish a good fingerprint database.

Third, select or develop appropriate data processing and matching algorithms. It is very common that the collected raw data need statistical analysis and processing. A reasonable data processing algorithm could improve the tracking accuracy significantly. It has been listed that there are several matching algorithm such as NN, KNN and etc. it is necessary to apply different matching algorithm to collected data and obtain an optimal selection. Furthermore, Kalman filter has been widely used in various tracking system, which is applied to our system as well. It has functions of predicting and correcting tracking results.

The last problem is to conduct experiments in different situations. A good tracking system should be able to adapt to different kinds of indoor environments. Testing with different influence factors will greatly contribute to evaluate and improve our research.

1.5 Thesis Scope

In this thesis project, a tracking system is implemented to realize precise localization of mobile device in a certain indoor environment with Wi-Fi signals. The thesis work is divided into five parts:

1) In the first part, taking Snoopy Framework as reference, the system framework is designed and the hardware devices are installed. Five Raspberry Pis are arranged in a certain indoor environment and used to collect the packets broadcasted from the mobile devices. The Evil Twin Attack will be carried out to increase network through put and a rogue AP will be built according to probe requests.

2) In the second part, the purpose is to implement the offline calibration phase of fingerprinting. It contains the reference point arrangement and data analysis. Meanwhile, The Expectation Maximization (EM) Algorithm for truncated data will be used to optimize the calibration results.

3) In the third part, Kalman filter for improvement of system performance is introduced. The parameters of Kalman filter fit for our tracking system are measured and illustrated.

4) In the fourth part, tracking tests will be performed with two data processing methods. The first method is based on the KNN Algorithm and average RSS value in each second. The second one is based on the NN Algorithm and instantaneous RSS value. Error analyses will be performed and a comparison between the tracking results with these two methods will be given.

5) In the last part, factors that influence tracking accuracy such as target speed, types of mobile devices and shadowing will be discussed. Possible future works of this project will be presented as well.

All experiments and analysis work of this project were conducted in Department of Electrical and Information Technology, Lund University, Sweden

2 SYSTEM FRAMEWORK

In this chapter, the framework of our tracking system will be introduced. Based on Snoopy Framework, we design a mobile device tracking system both in hardware and software level. This chapter gives a brief introduction of the Snoopy Framework and explains some basic concepts of hacking technologies. A clear structure of our system is illustrated in the end of this chapter.

2.1 Snoopy Framework

2.1.1 Brief Introduction of Snoopy Framework

Snoopy is a distributed tracking and profiling framework released by Glenn Wilkinson and Daniel Cuthbert on Dec 7th, 2012. [8] The architecture structure of Snoopy Framework is shown in figure 2.1.

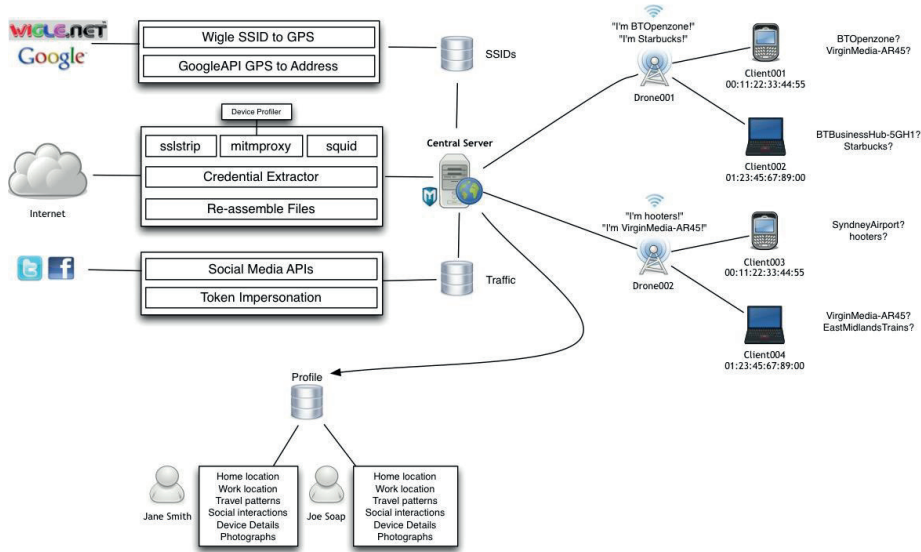


Figure 2.1. Architecture of Snoopy Framework [9]

In Snoopy Framework, there are two packets of code for the client side and the server side. The client side code can be run on any device with Linux operating system which supports for wireless monitor mode. Devices running with the client code are called “*Snoopy Drone*”. A drone is used to collect information of observed probe requests including MAC address, Service Set Identifier (SSID), timestamps, GPS coordinates and signal strengths. Collected data are uploaded to the central server of Snoopy. Then the profiling work will be done on the central server. In our thesis project, the client side consists of five Raspberry Pi single - board computers with Tenda W311M wireless cards.

Snoopy achieves simple tracking of mobile devices by resolving the GPS coordinates. With the inspiration of Snoopy Framework, we developed our indoor tracking system which based on the indoor Wi-Fi signal strength and fingerprinting technique as a complement to it. Taking Snoopy as a reference, we established direct link with cable between the drones and the server. The Shell scripts of Snoopy in Linux system also helped us to complete network configuration on both server and client sides. Following sections give brief introduction of software packages which are used in both Snoopy and our system.

2.1.2 Aircrack-ng [10]

Aircrack-ng is an 802.11 Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access - Pre-shared Key (WPA-PSK) * keys cracking program used by Snoopy Framework. Aircrack-ng has several suites to realize various functions. The main usage of this program is to monitor network activities and establish rouge APs. A rogue AP is a wireless access point which has been installed on a secured network without explicit authorization from a local network administrator. A list of scripts in Aircrack-ng suites which have been used in this project are shown below.

1) Airmon-ng

This script is used to enable the monitor mode on the wireless interfaces.

2) Airodump-ng

Airodump-ng is used to capture raw 802.11 frames. It is able to generate several files, including the detail information of all surrounding APs, clients and the connections between them. It is a significant and convenient tool to monitor the network activities in real time. It also enables us to get the instant display of targets MAC addresses, channels, connections, probe requests and signal strengths.

3) Aircrack-ng

Airbase-ng is a multi-purpose tool which aimed at attacking the clients as opposed to the AP itself. Since this script enable the client act as a full AP, it is used for creating rouge AP with user's familiar SSID and MAC address which they have connected before. It also encourages the mobile devices to associate with the rouge AP.

2.1.3 Tshark [11]

Tshark is a terminal version of Wireshark, which designed for capturing and displaying packets. It is well known that the Wireshark is the world's foremost network protocol analyser and Tshark supports the same options as Wireshark. In Snoopy Framework, Tshark is used to collect probe request packets with information of MAC addresses, SSIDs, timestamps and RSSs. Tshark makes it possible to capture packet data from a live network, or read packets from a previously saved capture file, either printing a decoded form of those packets to the standard output* [12] or writing the packets to a file. Without an instant display of the network activity, Tshark is used for collecting packets and writing them into a capture file.

* WEP and WPA-PSK: two types of a security algorithm for IEEE 802.11 wireless networks. Now WPA2 has replaced WPA as a new version.

* Standard output: sometimes abbreviated stdout, refers to the standardized streams of data that are produced by command line programs (i.e., all-text mode programs) in Linux and other Unix-like operating systems.

2.2 Wi-Fi Probe Request

The Wi-Fi probe request service is a critical element of the IEEE 802.11 standard family. It allows a mobile device with wireless internet access to detect the known APs within its scanning region at regular intervals, only if the mobile users turn on the Wi-Fi function [13]. It is called active discovery. Moreover, the modern smartphone also broadcasts probe requests regularly even when it has already connected to an AP [14].

A broadcast probe request packet contains information of MAC address, operation channel, Encryption Mode (EncryMode) and plaintext SSID of AP which has been connected previously (the other kind of probe request packet does not contain previous connected SSID information). Through capturing the probe requests with users' private information (the information in probe request is not encrypted), hacker realizes the network attacks. In this project, the Wi-Fi Probe request packets have been captured as the measurement data which will be used in fingerprinting. It has special features which are extremely important in synchronization. Details will be presented in the following chapters.

2.3 Evil Twin Attack [15]

Evil Twin Attack is a usual hacking attack. The first type of Evil Twin Attack is aimed at established connection. In order to connect with the wireless AP, a device with Wi-Fi function on keeps broadcasting probe requests for scanning the available APs within a certain region. Even when a connection has been established, the probe requests will be keep broadcasting. If the attacker set up a rouge AP with the same SSID and MAC address as the original AP which has been connected and provides a stronger signal level, the device will connect to the rogue AP instead of the original one.

Another type of Evil Twin Attack is based on the Wi-Fi probe requests with information of previous connected APs. As we have mentioned, the mobile device keeps trying to connect to the AP which has been connected before. According to the probe request, the attacker is able to build rogue APs with user's familiar SSIDs and MAC addresses. The rouge AP has exactly the same special SSID and MAC address in user's home or office. Thus, the user's mobile device will connect to it automatically without any notification. In this situation of Evil Twin Attack, the user may aware something happened and wondering why he connected to the AP at home in a totally strange place.

In fact, these two types of Evil Twin Attack are only applicable for the AP with no password protection or weak WEP EncryMode. A hacker has to crack the security password when he tries to implement the Evil Twin Attack towards a network with WPA/WPA2 EncryMode. By comparison, the network with WPA/WPA2 EncryMode is effectively resistance to Evil Twin Attack and is more secure than the network with no password protection or weak WEP EncryMode.

Through Evil Twin Attack, the attacker is able to implement many attacks, such as Man in the Middle, Domain Name System (DNS) Spoofing, Address Resolution Protocol (ARP) spoofing, and etc. The victims' private information such as passwords, friend list, credit card information and family address will be easy to obtain by the attacker. [16]

The process of Evil Twin Attack is shown in figure 2.2

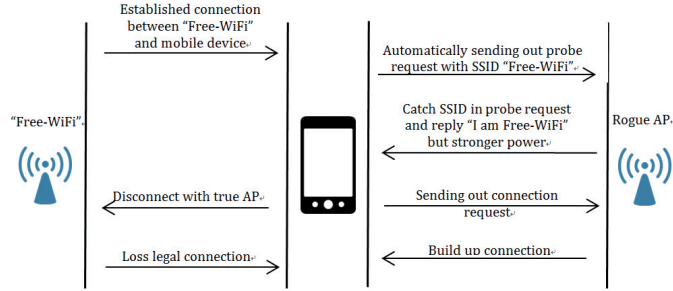


Figure 2.2. Process of Evil Twin Attack

Since the probe request packets are broadcasted regularly with several seconds (7 seconds approximately) pause, limited quantity of RSSs will not meet the demands of a time continues tracking system. The Evil Twin Attack will be used to provide a stable connection between rouge AP and mobile device, which increases the network traffic and ensures that Tshark can capture sufficient packets for continuous tracking. If allowed, it can be used to provide the Internet service and monitor users' internet-based communication.

2.4 System Framework

Snoopy Framework is only interested in information such as SSIDs and Internet cookies. However, the RSS information of the sniffed packets has been totally ignored. The RSS values are exactly the information that we consider as the fingerprints in our tracking system. Units in our system include five Raspberry Pis with Tenda W311M Wireless N150 Nano Adapter, two laptops, one router, one switch, and several patch cables. Details about devices parameters will be introduced in chapter 3. Five Raspberry Pis, we call them "*sniffers*", act as the Snoopy drones to sniff Wi-Fi probe requests and normal data packets from target devices. One of these Raspberry Pis is responsible for creating rouge AP to realize Evil Twin Attack. All units are connected by patch cables and a switch. The network configuration is conducted through the router. Two laptops act as the central server of our system. We remotely control Raspberry Pis through the server and data processing work is carried out in these laptops as well. A schematic diagram of system framework is shown in figure 2.3. More details about how the system works will be given in the following chapters.

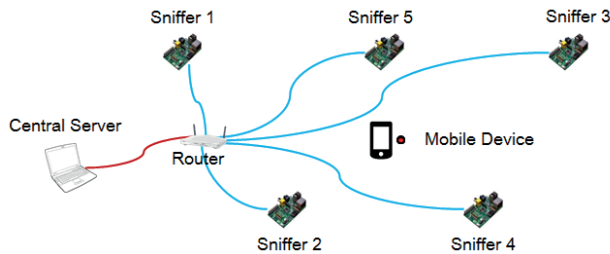


Figure 2.3. Schematic Diagram of System Framework

3 DESIGN AND OPERATION

In this chapter, an introduction to all the hardware devices and software of our indoor tracking system will be given. Furthermore, this chapter describes the operation of setting up sniffers and rogue AP in details. The processing of local time synchronization is illustrated as well.

3.1 Preparation

1) Raspberry Pi

Raspberry Pi is a low cost, credit-card-sized computer (Shown in figure 3.1). It was developed in UK by Raspberry Pi foundation with the intention of simulating the teaching of basic computer science in schools [17]. In this thesis, five model B Raspberry Pis are configured as sniffers and rogue AP. They are marked from #1 to #5.

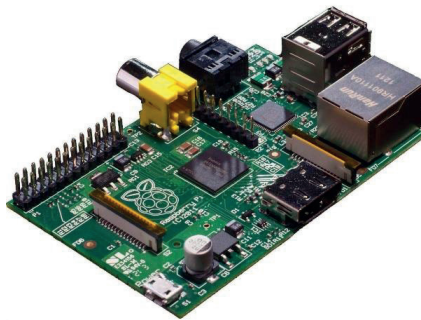


Figure 3.1. Model B Raspberry Pi

2) Tenda W311M Wireless N150 Nano Adapter

The Tenda W311M Wireless N150 Nano Adapter is a kind of network adapter which plugs into a computer through USB port. It is used for supplying the wireless connectivity to computer. The Tenda W311M wireless adapter with chipset Ralink rt2870/3070 supports wireless monitor mode and packet injection, which is important to sniff the RSS and increase network throughput.

3) Router and Switch

In this project, a Dlink-DIR100 4-Port Router and a Netgear GS308 8-Port Switch are used to set up an internal network. It assigns IP addresses to sniffers and allows the Raspberry Pis to access to the Internet for installation purposes. All units of the system are in the same local network through the configuration of the router.

4) Laptop

Two Laptops are used to remote control the Raspberry Pi and act as the central server, which collect measurement data from all sniffers and execute the data analysis.

5) SD Card and Patch Cable

Five 8 GB SD cards are used together with the Raspberry Pis as hard drives. The patch cables are used for connecting the switch with the Raspberry Pis and Laptops.

6) Software

The Raspberry Pis are running Raspbian, version 3.10, which is a Debian-based GNU/Linux operating system specifically tailored for use with the Raspberry Pi.

Through Oracle VM Virtualbox, the Laptops are running Ubuntu-12.04 LTS, which is a Debian-based Linux operating system. Oracle VM Virtualbox is a free virtualization software package for Intel 64-based computers.

Remote control of the sniffers is achieved by PuTTY-0.59, which is a free implementation of SSH* [18] for Windows and UNIX platforms.

Tshark and Aircrack-ng suite are two main programs for tracking. Tshark is a free, terminal-based and open-source packet analyser. Aircrack-ng suite is a set of tools for auditing wireless networks, which creates an interface on monitor mode from the wireless adapter. It observes the connection status of surrounding mobile devices, implements the packets injection work and creates the rogue AP.

* SSH: Abbreviation of Secure Shell, which is a cryptographic network protocol for secure data communication. It is utilized to realize secure network services between two networked computers e.g. remote command-line login and remote command execution.

3.2 Operation Procedure

Since there are five Raspberry Pis in use, one of them which is located in the middle of the experiment environment should be configured as the rogue AP. Meanwhile, all five Raspberry Pis work as sniffers to collect RSSs from surrounding mobile devices.

3.2.1 Local Time Synchronization

Before we start the whole system, the first issue that has to be confronted is the local time synchronization. The Raspberry Pi is designed to be an ultra-low cost computer. Lots of hardware units on a normal computer, such as Real Time Clock (RTC), have been left out. In our tracking system, at each time point, five RSSs are collected by each Raspberry Pi and composed into a fingerprint. Through matching the fingerprint with a database, the position of the tracking target can be estimated. However, the precondition is that all five Raspberry Pis must share the same local time.

RTC is able to keep system clock working even when the power is off. Without it, the local time of Raspberry Pi will be reset to the factory settings when it turns off. So it is necessary to rely on a software clock to realize synchronization. Network Time Protocol (NTP) is widely used to synchronize a computer to Internet time servers or other sources, such as a radio or satellite receiver or telephone modem service. It can also be used as a server for dependent clients [19]. Since the Internet service may not be available, Raspberry Pis cannot update the local time automatically from the global Network Time Protocol directly. It is necessary to synchronize the local time from a time server which is established manually.

First of all, NTP was utilized to establish a time server on one of five Raspberry Pis. Here, the Raspberry Pi #5 was chosen as the local timeserver. Then, the other four updated their respective local time from the timeserver. Packets *ntp* and *ntpd* should be installed on each Raspberry Pi ahead. Details about how to configure an NTP time server and NTP time clients are introduced below.

1) Time Server Side

The commands below should be added into `/etc/ntp.conf` for time server configuration on the Raspberry Pi #5.

```
restrict      127.0.0.1
restrict      192.168.0.0 mask 255.255.255.0 nomodify notrap
server        127.127.1.0
fudge         127.127.1.0 stratum 10
```

where

restrict 127.0.0.1 gives full control to the local host.

nomodify notrap the access control rules, makes all workstations in the internal private network to be able to query time information from this server.

server 127.127.1.0 specifies that a time server is running on the host (own local time).

fudge transfers additional information to the clock driver.

Save and exit the file. Then, restart the NTP service to make sure that the configuration of time server has been applied.

2) Client side

The commands below should be added into `/etc/ntp.conf` on the other four Raspberry Pis for time client side configuration.

```
server 192.168.0.105
restrict default ignore
restrict 127.0.0.1
restrict 192.168.0.105 mask 255.255.255.255 nomodify notrap
noquery
```

where 192.168.0.105 is the IP address of the time server.

Then disable all server setting in this file. Save and exit the file.

Stop the NTP server on each client and use the following commands to query time information from the established time server directly.

```
sudo ntpdate 192.168.0.105
```

After executing this command on each client Raspberry Pi, local time synchronization is completed and all clients share the local time with the central time server.

NTP is able to synchronize all participating computers to within a few milliseconds [20]. Normally, it will meet the requirement of our tracking system. However, some matching algorithms require a more precise synchronization. So another method to achieve high precision synchronization will be introduced in Chapter 7.

3.2.2 Rouge AP and Sniffer Setup

In order to increase the network traffic and guarantee that there are sufficient packets for continuous tracking, Evil Twin Attack was performed by establishing a rouge AP according to the probe requests broadcasted from a specific mobile device or the SSID of AP. Details about how to configure a Rouge AP and setup a sniffer are introduced below.

- 1) First of all, on each Raspberry Pi, we created an interface operating in monitor mode with `airmon-ng` from our wireless adapter, which is named `wlan0`. The command is

```
sudo airmon-ng start wlan0
```

After activated the monitor mode on `wlan0`, a new interface named `mon0` was available for both the Evil Twin Attack and packets collection.

- 2) The connection status of surrounding mobile devices on all the channels was monitored by the command

```
sudo airodump-ng mon0 --ignore-negative-one
```

Sometimes *airodump-ng* might fix on channel -1. Adding *--ignore-negative-one* can simply fix this problem.

Figure 3.2 is an example that shows all the surrounding active APs with SSIDs, authentication methods, channels, MAC addresses and signal power. Airbase-ng can help us to build a rogue AP according to these information.

Moreover, all mobile devices with connection status and MAC address were scanned as well. *airodump-ng* was helpful to decide which channel is suitable to create a rouge AP.

```
CH 1 [[ Elapsed: 2 mins ] [ 2014-05-18 21:58 ] [ display ap+sta+ack ]
```

BSSID	PWR	RXQ	Beacons	#Data	#/s	CH	MB	ENC	CIPHER	AUTH	ESSID
00:26:3E:A9:F2:84	-1	0	0	6	0	1	-1	WPA			<length: 0>
C8:3A:35:CF:2A:47	0	20	1202	120	1	1	54	OPN			eduroam
EE:85:2F:4F:4D:0E	-35	100	883	1100	0	1	54e	WPA2	CCMP	PSK	? " .? s iPhone

BSSID	STATION	PWR	Rate	Lost	Frames	Probe
00:26:3E:A9:F2:84	AC:F7:F3:14:3A:4B	-82	0 - 1	0	32	eduroam
C8:3A:35:CF:2A:47	70:11:24:3A:87:8D	-36	0 -24	0	255	ssss,tttt,Test Only
(not associated)	50:46:5D:85:3C:9B	-68	0 - 1	0	2	
(not associated)	34:23:BA:4E:3C:EB	-84	0 - 1	0	4	eduroam
(not associated)	EC:85:2F:4F:4D:0E	-42	0 - 1	0	4	
(not associated)	50:2E:5C:F1:49:20	-72	0 - 1	0	27	
(not associated)	00:21:6B:38:CC:48	-82	0 - 1	0	3	
(not associated)	BC:CF:CC:0A:18:1C	-82	0 - 1	0	14	eduroam
(not associated)	48:74:6E:68:B2:88	-48	0 - 1	0	2	eduroam
EE:85:2F:4F:4D:0E	60:67:20:DE:FD:1C	-38	54e-54e	0	666	? " .? s iPhone,? " .? s iPh

MAC	CH	PWR	ACK	ACK/s	CTS	RTS_RX	RTS_TX	OTHER
54:E4:3A:EC:CA:24	1	-68	14	0	0	0	0	0
00:0B:0E:F4:8C:44	1	-82	3	0	0	0	0	0
EC:85:2F:4F:4D:0E	1	-32	0	0	221	0	0	0
EE:85:2F:4F:4D:0E	1	-72	64	0	0	0	0	0
60:67:20:DE:FD:1C	1	-32	486	0	0	0	0	0
00:0B:0E:F4:8C:40	1	-80	1	0	0	0	0	0
00:0B:0E:F4:8C:42	1	-82	1	0	0	0	0	0
C8:3A:35:CF:2A:47	1	-68	49	0	0	0	0	0
70:11:24:3A:87:8D	1	-38	0	0	1	0	0	0
00:26:3E:A9:F2:84	1	-84	0	0	0	0	0	2

Figure 3.2. An Example of Scanning Result

- 3) Evil Twin Attack was implemented on Raspberry Pi #5. In order to avoid the others connecting to our rouge AP, our rouge AP for experiment was named as "Test Only" with the command below.

```
sudo airbase-ng -c 1 -P -v -e "Test Only" mon0
```

where

-c 1 represents all APs are running on channel 1

-P responds to all probes, even when specifying SSIDs

-v represents verbose, show more messages

-e "Test Only" specifies a single SSID AP called "Test Only". This AP will be utilized to perform the calibration and tracking test.

All other rouge APs will be automatically created by this command. It creates rouge APs automatically according to every SSID in probe requests from different mobile devices. On the other hand, if airbase-ng received a probe request without SSID, a rouge AP cannot be established automatically.

- 4) In order to keep the link between our rogue AP and clients stable, it is necessary to provide Internet service for the clients who connects to our rouge AP. Here, we simply utilized the bridge mode to implement the Internet service. These commands were executed to deploy a bridge type Internet service.

```
sudo su
brctl addbr br0
brctl addif br0 eth0
ifconfig br0 up
```

brctl tool is required before executing these commands.

- 5) So far, we have finished the configuration of rouge AP. Then Tshark was started to collect the packets with Wi-Fi RSSs. Command below was executed on each Raspberry Pi.

```
sudo tshark -S -l -i mon0 -R "wlan.sa == 00:00:00:00:00:00" -T fields -e
wlan.sa -e wlan_mgt.ssid -e radiotap.dbm_antsignal -e frame.time -E
separator=, -E quote=d >> file_name.txt
```

where

-S set the line separator to be printed between packets.

-l flush the standard output after collected data for each packet is printed

-R "wlan.sa == 00:00:00:00:00:00" represents a filter, Tshark only collects data from the mobile device with specific MAC address.

-T fields the values of fields specified with the *-e* option, in a form specified by the *-E* option.

-e option *-e wlan.sa* the source MAC address

-e wlan_mgt.ssid specific SSID

-e radiotap.dbm_antsignal set the signal strength in dBm

-e frame.time record the time when packet is captured.

-E option *-E separator=, -E quote=d* generates comma-separated values (CSV) file which is suitable for importing into spreadsheet program.

>> *file_name.txt* the data will be wrote into *file_name.txt*

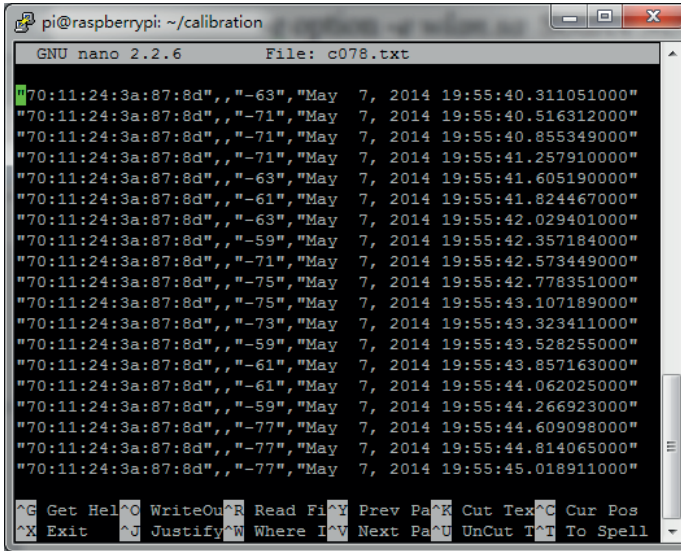


Figure 3.3. Example of Data Frame

Figure 3.3 illustrates the structure of the collected data by Tshark. The command above is only suitable for calibration of offline phase. During the online tracking phase, there is no need to set a filter. Sniffers will automatically collect all wireless network activities including information about MAC addresses, RSSs, probe requests and time stamps.

3.3 Limitation

At present, it seems impossible to host a rouge AP with WPA/WPA2 type of wireless network encryption. The mobile users cannot be forced to connect with a rogue AP automatically when the broadcasted SSID belongs to an AP with network protected by WPA/WPA2. The packets injection will also be denied when a connection exists between mobile user and legal AP. It makes the tracking more difficult, since the captured RSS signals are not sufficient and continuous. Therefore, the user has to be required to connect to the rouge AP manually for increasing the network traffic.

The accuracy of measurement data can influence tracking results significantly. For example, Tenda W311M Wireless N150 Nano Adapter drops packets with RSS below -92 dBm and can just give integer signal strengths. Expectation Maximization (EM) algorithm for truncated data can be used to fix the measurement results. Following Chapter 4 will discuss details of this EM algorithm.

4 OFFLINE PHASE OF FINGERPRINTING LOCALIZATION

The main purpose of “Offline” phase of fingerprinting is to complete the felid strength measurement at each reference point and to establish an original fingerprint database. In this chapter, a detailed description about how to allocate reference points in the experiment environment will be given. The operations of fingerprinting calibration on offline phase will be described as well. Finally, EM Algorithm for truncated data processing will be introduced to optimize the calibration results.

4.1 Reference Points Allocation

The “Offline” phase was performed in the entrance hall of E-huset, which belongs to the Department of Electrical and Information Technology, Lund University, Sweden. The scale of the entrance hall is 10 meters in width and 60 meters in length. The entrance hall also includes a sub-corridor with 3 meters width and several obstructions such as desks, chairs and bonsai. A map of this entrance hall is shown in figure 4.1.

The grid spacing between two reference points significantly affects the performance of the whole system. Generally, larger grid spacing leads to lower localization accuracy [21]. A report [22] shows that the recommended grids spacing is 1-3 meters in indoor environment. Therefore, 210 reference points are allocated in the entrance hall uniformly with a grid about 1.5 meters.

The floor plan with distribution of the reference points (small blue dots) is shown on the map as well. Five Raspberry Pis are configured as sniffers and marked from one to five. The positions of five Raspberry Pis are also marked on the figure 4.1.

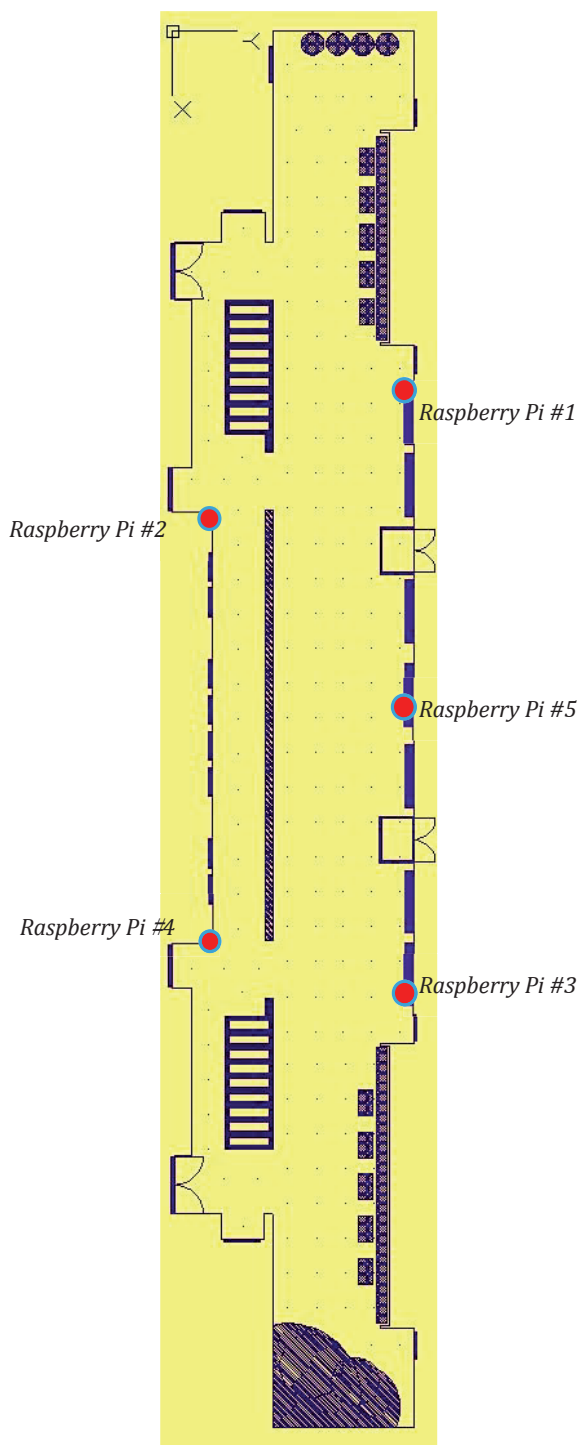


Figure 4.1. Floor Plan with Reference Points Distribution and Sniffers

4.2 Position of Sniffers

The positions of sniffers should be considered to covering the whole area of the experiment environment and selected at place without too many obstructions. In order to reduce the influence of human body obstructions, the height of sniffers should higher than normal height of people. Moreover, the place should be safe and suitable (untouchable by surrounding people and easy to hung Raspberry Pis). Considering the complexity of the environment, (the roofs with different heights and construction of walls), two examples of placing the sniffers are shown below in figure 4.2



Figure 4.2. Positions of Sniffers

Raspberry Pi #1, #3 and #5 were placed at a height of 3 meters as the figure on the left indicates. Raspberry Pi #2 and Raspberry Pi #4 were placed at a height of 2.5 meters in the sub-corridor.

4.3 Calibration

The calibration work on offline phase is to collect RSS data as fingerprints at each arranged reference point and save them into a database. This is the most important phase of fingerprinting. A good result of calibration can significantly benefit to the tracking accuracy. It is necessary to conduct this work during a few or non-people time period. After the data collecting work, the raw data is approached with appropriate statistical methods and the fingerprint database will be established.

4.3.1 Fingerprint Database Structure

The structure of our fingerprint database is a table, shown in figure 4.3, contains 210 rows and 5 columns. Each row represents for RSS values at each reference point and 5 columns represent for data collected from 5 sniffers. Each number in this table is a RSS value and we consider one row with five RSS values as a RSS vector – one single

fingerprint. In order to obtain each number in this table, the sniffer will run for 40 seconds to collect a group of raw data. After statistical analysing and data processing of these groups of the raw data, a complete database will be established.

	Sniffer1	Sniffer2	Sniffer3	Sniffer4	Sniffer5
	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER
1	-87.715	-83.644	-95.784	-103.38	-96.034
2	-84.723	-84.887	-86.625	-156	-80.174
3	-84.07	-91.986	-86.774	-156	-81.276
4	-81.804	-92.496	-94.313	-156	-88.069
5	-78.667	-136.58	-108.91	-156.02	-84.016
6	-84.415	-95.189	-156	-156	-97.856
7	-81.947	-95.293	-92.396	-156	-94.829
8	-82.686	-93.761	-92.948	-156	-103.26
9	-83.242	-101.26	-94.439	-156	-110.42
10	-74.779	-87.12	-89.564	-97.078	-83.504
11	-81.858	-87.831	-91.731	-156	-94.247

Figure 4.3. An Example of Fingerprint Database

4.3.2 Data Collecting Operation

An iPad mini was utilized as our standard equipment to conduct the calibration work. First of all, in the entrance hall with few artificial effects, one tester held an iPad mini horizontally at the height of 1.2 meters, in the direction pointing to the Raspberry Pi #5. After turning on the Wi-Fi function, the iPad connected to the rogue AP “Test Only” automatically (the standard device has connected with this AP before.). Here, the name of rogue AP was set as “Test Only” manually but not according to the probe requests from surrounding device. Meanwhile, the other tester controlled the Raspberry Pis and used Tshark tool to sniff packets broadcasted from the iPad. The sniffed packets contain information with RSSs, MAC addresses, SSIDs of probe requests and timestamps. Data was collected for 40 seconds at each reference point and the raw data was saved into a text file.

Consequently, RSS data at all 210 reference points was collected. For each reference point, there were five raw capture files from five different Raspberry Pis. And the RSS values in these files were needed to be processed before saved into the fingerprint database.

4.3.3 Estimation of Signal Strength

Generally, at each reference point, there are five groups of RSSs caught by five sniffers respectively. The quantity of captured packets in one group is around three hundreds to five hundreds generally. A screenshot of captured data structure is shown in figure 4.4.

	A	B	C	D	E
	MacAddress	ProbeReq...	RSSI	Date	ExactTime
1	"70:11:24:3a:87:8d"		"-83"	"May 18	2014 16:47:32.960687000"
2	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:33.780377000"
3	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:34.804953000"
4	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:35.009832000"
5	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:35.214717000"
6	"70:11:24:3a:87:8d"		"-79"	"May 18	2014 16:47:35.419722000"
7	"70:11:24:3a:87:8d"		"-85"	"May 18	2014 16:47:36.240774000"
8	"70:11:24:3a:87:8d"		"-83"	"May 18	2014 16:47:36.617272000"
9	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:37.587365000"
10	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:37.588636000"
11	"70:11:24:3a:87:8d"		"-79"	"May 18	2014 16:47:37.590330000"
12	"70:11:24:3a:87:8d"	"ssss"	"-85"	"May 18	2014 16:47:37.814028000"
13	"70:11:24:3a:87:8d"	"tttt"	"-83"	"May 18	2014 16:47:37.815380000"
14	"70:11:24:3a:87:8d"		"-83"	"May 18	2014 16:47:37.818842000"
15	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:38.761789000"
16	"70:11:24:3a:87:8d"		"-81"	"May 18	2014 16:47:38.926564000"

Figure 4.4. Structure of Raw Calibration Data

After the data collecting work, it is necessary to analyse the raw signal strengths from every Raspberry Pi at each reference point, and then establish the fingerprint database.

4.3.3.1 Histogram Estimation

In order to obtain the accurate numerical estimation of the RSS value at each reference point, it is necessary to analyse the probability distribution of the raw signal strengths. Histogram estimation is one of the most commonly used algorithms to deal with this kind of problem [23]. According to research [24] and analysis of our experimental data, the RSSs obeys log-normal distribution. With *hist** function and *dfittool** [25] in MATLAB, we fitted the signal strengths at each reference point into a normal distribution, because the data were collected in dBm scales. After checking the probability distribution, it is easy to utilize the mean value of such distribution to represent a group of data. With this algorithm, the RSS value of each reference point can be calculated. Here, in figure 4.5, is the probability distribution for a group of collected data on one Raspberry Pi at a reference point.

* *hist* is a function in MATLAB which is able to create a histogram bar chart. *hist(x,nbins)* was used here, which sorts x into the number of bins specified by the scalar nbins.

* *dfittool* is a command to open Distribution Fitting app which can fit distributions to the data from the workspace and display the fitted distribution over plots of the empirical distributions.

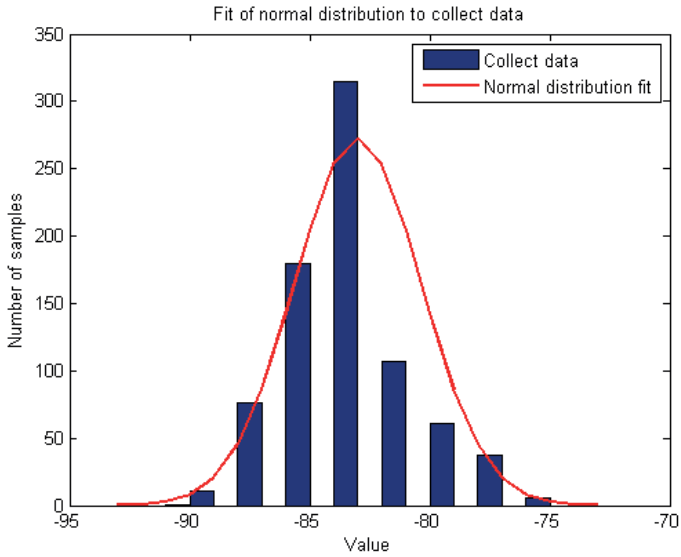


Figure 4.5. An Example for Signal Strength Distribution by Histogram

According to the normal distribution, the mean value of each group of RSSs was calculated as the calibration result at each reference point. The following five figures of figure 4.6 are given to illustrate the RSSs distribution for each Raspberry Pi in the experiment environment.

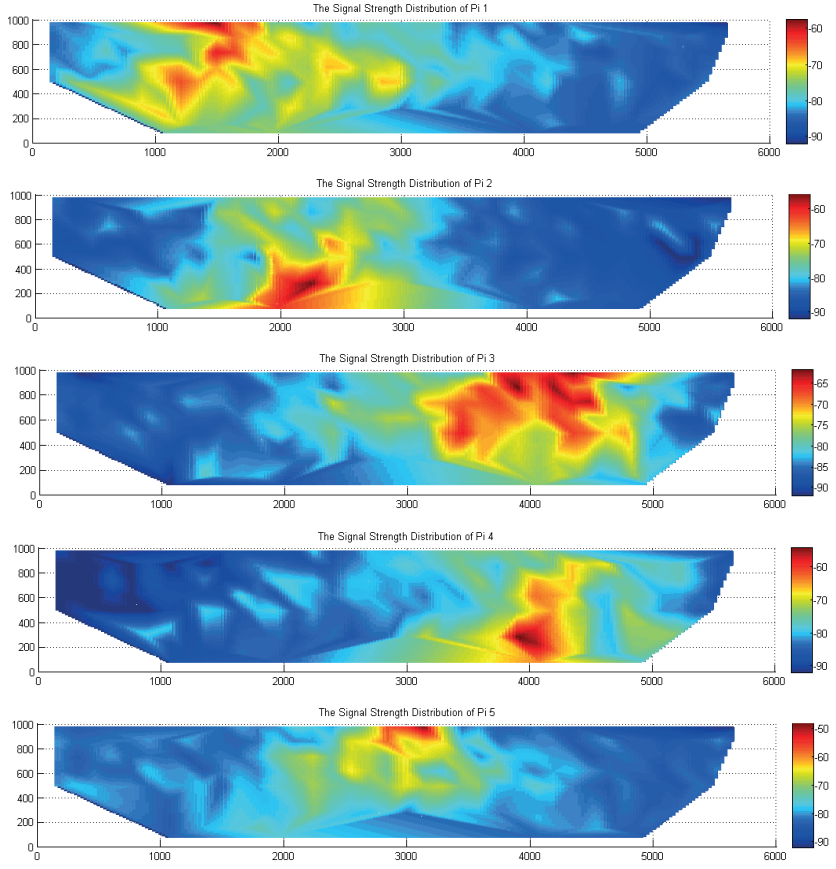


Figure 4.6. Original Measured RSS Distribution with Histogram Estimation

The red area indicates high signal strength level and the blue area indicates low RSS region. These figures show that the closer target gets to the sniffer, the stronger field strength is.

However, influenced by the effect of noise floor, fading, packet loss and some other interference, at some of the reference points, the distribution of collected RSSs is not a complete normal distribution. In another word, the histogram estimation according to the normal distribution only fits for a group of data with a large number of samples with signal strength above the noise floor. Tenda W311M Wireless N150 Nano Adapter drops packets with RSS below -92 dBm actively, which makes the data truncate. Therefore, an advanced estimation algorithm for truncated data is needed.

4.3.3.2 Expectation maximization (EM) algorithm for truncated data

Generally, the larger distance between device and sniffer, the fewer packets could be collected. Sometimes a sniffer even cannot capture any packet from the reference points far from it. With such a huge number of missing data, the histogram estimation no

longer fit for estimating the true RSS value at each reference point. Thus, Expectation maximization (EM) algorithm should be applied for solving the truncated data.

Expectation maximization (EM) algorithm is a good choice to estimate the mean and standard deviation of data with a known number of missing samples [26]. With the calibration data analysing, it can be found that the sniffers did not collect any packets with RSS below -92 dBm and it is reasonable to consider that the noise floor is -92 dBm. If the signal level lower than this value, the packets will be regarded as noise and dropped by the sniffer actively. For each sniffer, since the raw signal strengths are in dBm scales and follow the normal distribution, the probability distribution with missing data points is regarded as the truncated normal distribution.

Suppose a group of RSSs follows such a normal distribution

$$X \in N(\mu, \sigma^2)$$

where μ is the true mean.

σ is the standard deviation.

As the data is truncated, these two values cannot be calculated. For the reason of the noise floor, data points lie within the interval $X \in (a, +\infty)$, where a is the truncation point. The value of a should be set as -92dBm which is the noise floor. Since all packets with RSSs below the truncation point are missing, collected data with a number of missing data points obey the left truncated normal distribution.

During the calibration, since all five Raspberry Pis sniffed packets broadcasted from the same mobile device, the number of packets collected by different sniffers should be the same and equal to the number of broadcasted packets ideally. However, through observation of the raw data, the numbers of data collected from different sniffers were significantly different within the same length of time. We assumed that the sniffer with the largest number of data points is not truncated. Then, it is easy to calculate how many data points were missing.

Assume that a complete distribution has k samples, and l samples have been truncated. All known data points are x_1, x_2, \dots, x_{k-l} , and the missing data points are $x_{(k-l)+1}, x_{(k-l)+2}, \dots, x_k$. In our case, since we arrange the sniffers uniformly, not all the sniffers have the collected data below the truncation point. After we get five groups of data from one reference point, we assume the largest length of the group to be k . It means that the sniffer which got most data points is not missing any data. Then it is easy to calculate how many points have been truncated for other sniffers. From the known data, the mean μ_0 and standard deviation σ_0 for truncated data can be estimated through

$$\mu_0 = \sum_{i=1}^{k-l} \frac{x_i}{(k-l)}$$

and

$$\sigma_0 = \sqrt{\sum_{i=1}^{k-l} \frac{(x_i - \mu_0)^2}{(k-l)}}$$

For this left truncated type distribution, we can give

$$E(X | X < a) = \mu_0 - \sigma_0 \frac{\phi(\alpha)}{\Phi(\alpha)}$$

and

$$Var(X | X < a) = \sigma_0^2 \left[1 - \alpha \frac{\phi(\alpha)}{\Phi(\alpha)} - \left(\frac{\phi(\alpha)}{\Phi(\alpha)} \right)^2 \right]$$

where $\alpha = (a - \mu_0) / \sigma_0$.

$\phi(\cdot)$ is probability density function for standard normal distribution.

$\Phi(\cdot)$ is its cumulative distribution function (CDF).

As the number of missing data points has been known, the iterative EM-algorithm can be utilized to further improve the estimation. [27] The initial μ and σ can be calculated by collected data. And at $j+1$ times of iteration, the new estimation of μ and σ can be calculated by

$$\hat{\mu}_{j+1} = \frac{\sum_{i=1}^{k-l} x_i + \sum_{i=k-l}^k E_j(x_i | x_i < a)}{k}$$

and

$$\hat{\sigma}_{j+1}^2 = \frac{\sum_{i=1}^{k-l} (x_i - \hat{\mu}_j)^2 + \sum_{i=k-l}^k Var_j(x_i | x_i < a)}{k-1}$$

The value of $\hat{\mu}_{j+1}$ will be used as the fixed RSS value.

Finally, for each reference point, we got five means of the RSSs from five different sniffers. Consequently, to establish the fingerprint calibration database, every reference point has a unique identifier RSS vector $\mathbf{s}^{(i)}$ which corresponds to the coordinates of the reference point,

$$\mathbf{s}^{(i)} = (s_1^{(i)}, s_2^{(i)}, s_3^{(i)}, s_4^{(i)}, s_5^{(i)})$$

Where i is the number of reference point and $i \in (1, 210)$, since there are 210 reference points have been arranged. The table of vectors $\mathbf{s}^{(i)}$ composes the necessary calibration database.

For some special cases, our sniffer cannot capture any packet from the mobile device. It means that the RSS at these reference points is extremely weak. It is most likely that the true mean of such points is lower than the RSS on all the other reference points.

Therefore, we give -156 dBm to these reference points, which is the minimum value of all the estimate results with the EM-algorithm. The following figure 4.7 shows the RSS distribution for each Raspberry Pi with iterative EM-algorithm in the entrance hall of E-huset.

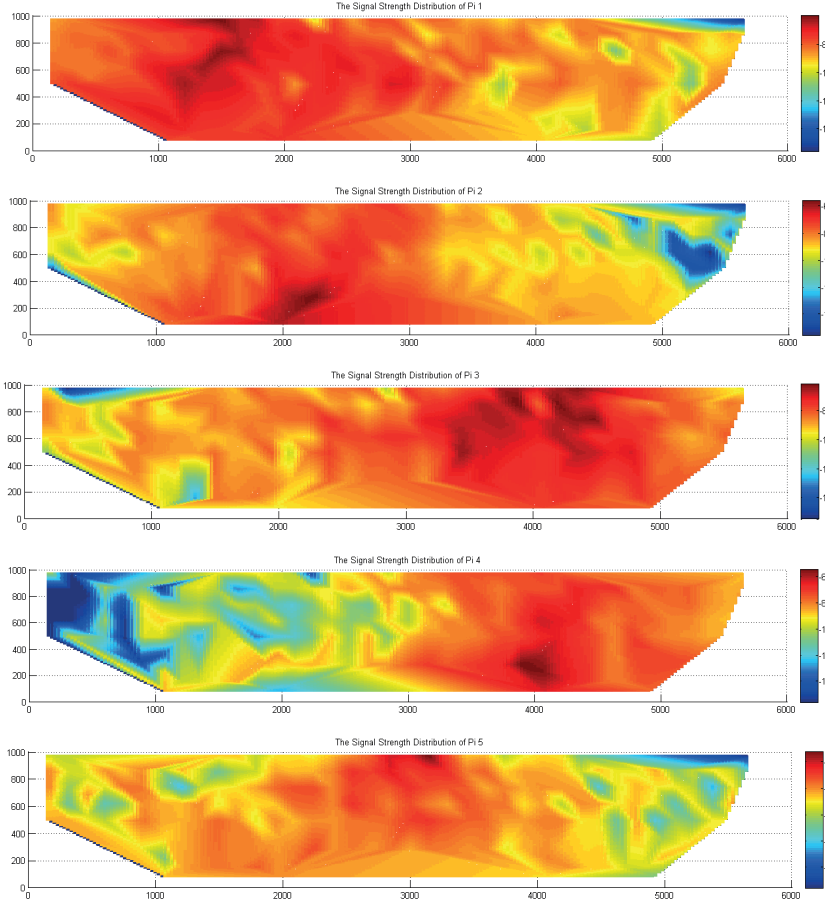


Figure 4.7. RSS Distribution with EM Algorithm

Compared with the Figure 4.6, the RSS distribution has been optimized by the EM-algorithm. And the estimate result is the complete fingerprint database we need in our tracking system.

5 KALMAN FILTER DESIGN

The Kalman filter was originally implemented by R.E. Kalman, who published his famous paper describing a recursive solution to the discrete-data linear filtering problem. It is a set of mathematical equations, providing an efficient recursive mean to estimate the state of a process [28]. It minimizes the mean of the squared error and improves the positioning accuracy on tracking problems. In this chapter, a Kalman filter will be designed according to our indoor tracking environment for optimizing the tracking results. It plays an important role in missing data point prediction and measurement error mitigation. Some experiments in this chapter are involving algorithms which will be introduced in following Chapter 6 and 7, however we would like to bring them out earlier to present a complete explanation of the Kalman filter.

5.1 Discrete Kalman Filter

The Kalman filter model assumes the true state at time k is evolved from the state at time $(k-1)$ according to the linear stochastic difference equation.

$$s_k = A s_{k-1} + B u_k + w_k$$

where s_k is the current state, s_{k-1} is the previous state.

A is the state transition matrix, which relates the previous state at time $k-1$ to the current state at time k .

B is the control-input matrix, which relates the optional control input u_k to the state s_k .

u_k is the control vector.

w_k is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution with covariance Q . $\sigma_w^2 = Q$

$$w_k \sim N(0, Q)$$

At time k , there also is

$$z_k = Hs_k + v_k$$

where z_k is the current observation result.

H is the observation matrix, which relates the state to the measurement z_k .

v_k is the observation noise which is assumed to be zero mean Gaussian white noise with covariance R . $\sigma_v^2 = R$

$$v_k \sim N(0, R)$$

The two random vectors w_k and v_k are uncorrelated. [29]

5.2 Two Phases of Kalman Filter

The Kalman filter estimation is a process that is using a form of feedback control. The filter predicts the process state at some time and then obtains feedback in the form of (noisy) measurements. Regularly, the operation of the Kalman filter is divided into two distinct phases - Prediction and Correction.

- (1) During the prediction phase, the Kalman filter uses the previous optimal state estimate to produce a new estimate of the current state. That is known as discrete Kalman filter time update equations.

$$\hat{x}(k|k-1) = A\hat{x}(k-1|k-1) + B\hat{u}(k)$$

$$P(k|k-1) = AP(k-1|k-1)A' + Q$$

where $\hat{x}(k|k-1)$ is the predicted state using the pervious optimal state estimate.

$\hat{x}(k-1|k-1)$ is the previous optimal state estimate.

$\hat{u}(k)$ is the current control vector.

$P(k|k-1)$ and $P(k-1|k-1)$ are error covariances corresponding to $\hat{x}(k|k-1)$ and $\hat{x}(k-1|k-1)$ respectively.

A , B and Q have been explained in section 5.1.

- (2) During the correction phase, the Kalman filter corrects the estimate which is obtained during the prediction phase by comparing it with the measurement result. That is known as the discrete Kalman filter measurement update equation.

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K_k(z_k - H\hat{x}(k|k-1))$$

$$K_k = \frac{P(k|k-1)H'}{HP(k|k-1)H' + R}$$

$$P(k|k) = (I - K_kH)P(k|k-1)$$

where $\hat{x}(k|k)$ is current optimal state estimate.

K_k is current Kalman gain.

$\hat{x}(k|k-1)$ is the previously predicted state estimate.

$P(k|k-1)$ is the error covariance corresponding to $\hat{x}(k|k-1)$.

$P(k|k)$ is the covariance corresponding to $\hat{x}(k|k)$.

I is the identity matrix.

H , z_k and R have been explained in section 5.1.

A complete illustration of the operation of the Kalman filter is shown in figure 5.1.

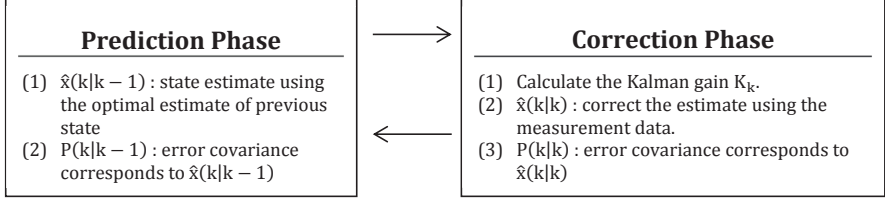


Figure 5.1. Operation of Kalman filter

In each time of iteration, the Kalman filter is replacing the current optimal state estimate $\hat{x}(k|k)$ as the previous optimal state estimate $\hat{x}(k-1|k-1)$ and the $P(k|k)$ as the $P(k-1|k-1)$. The recursive nature is one of the appealing features of Kalman filter. The filter only needs to know the recursive result of the previous state.

In some special cases, there is no observation result z_k with the current state. It happens in tracking system two which will be introduced in Chapter 7. The previous optimal state estimate $\hat{x}(k-1|k-1)$ is regarded as the observation result z_k to continue the iteration. Also, the Kalman gain should be set to zero.

$$\hat{x}(k-1|k-1) = z_k$$

$$K_k = 0$$

5.3 Kalman Filter Implementation [30]

In our self-designed tracking system, the Kalman filter is used to improve the positioning accuracy. It is also used to predict the target position when there is no measurement data. Since the tracking system is in x-y coordinate, the Kalman filter is applied with the x coordinates and the y coordinates separately. In the design of our Kalman filter, there is no control vector which means that the control vector $u_k = 0$.

During the prediction phase, the following operations are performed,

$$\begin{bmatrix} x_k \\ vx_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ vx_{k-1} \end{bmatrix} \times \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad (x_k = x_{k-1} + vx_{k-1}\Delta t)$$

$$\begin{bmatrix} y_k \\ vy_k \end{bmatrix} = \begin{bmatrix} y_{k-1} \\ vy_{k-1} \end{bmatrix} \times \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad (y_k = y_{k-1} + vy_{k-1}\Delta t)$$

where $\begin{bmatrix} x_k \\ vx_k \end{bmatrix}$ and $\begin{bmatrix} y_k \\ vy_k \end{bmatrix}$ corresponds to s_k .

x_k and y_k are the estimated coordinates in the current position.

vx_k and vy_k are the estimate velocities in the current position.

Δt is used as the update time which is the time difference between two contiguous time points (section 7.3)

The state does not change from step to step.

So,
$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

The noisy measurement is taken directly from the state, there is

$$\begin{bmatrix} z_x \\ zv_x \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_k \\ vx_k \end{bmatrix}$$

$$\begin{bmatrix} z_y \\ zv_y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} y_k \\ vy_k \end{bmatrix}$$

where $\begin{bmatrix} z_x \\ zv_x \end{bmatrix}$ and $\begin{bmatrix} z_y \\ zv_y \end{bmatrix}$ corresponds to z_k .

z_x and z_y are measured coordinates of x and y at the current position.

zv_x and zv_y are measured velocities of x and y at the current position.

So,
$$H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The Kalman filter needs the initial values of $P(k-1|k-1)$ and $\hat{x}(k-1|k-1)$ to start the system during the prediction phase and calculate the $\hat{x}(k|k-1)$ and $P(k|k-1)$. Assume the initial value is

$$P(0|0) = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}.$$

The initial value of $P(0|0)$ is used for both x and y dimensions. The initial value of the Kalman filter is not critical with sufficient measurement data. It is fine to choose almost any $(0|0) \neq 0$, and the filter will eventually converge. However, if there are not enough observation values for the Kalman filter to correct the estimates, the initial value can influence the tracking accuracy significantly. In our tracking systems, the coordinates of the first estimate position are regarded as the initial value of $x(0|0)$ and $y(0|0)$. As for the initial values of velocities, since we assume the target is moving with a constant speed, the mean velocity is regarded as the initial values. So,

$$x(0|0) = \begin{bmatrix} \text{first estimate } x \text{ coordinate} \\ \text{mean } x \text{ velocity} \end{bmatrix}$$

$$y(0|0) = \begin{bmatrix} \text{first estimate } y \text{ coordinate} \\ \text{mean } y \text{ velocity} \end{bmatrix}$$

The method of how to determine an accurate position of the start point will be discussed in section 6.3.

5.3.1 Process Noise Q

The covariance matrix of process noise Q should be a two by two matrix.

Assume that

$$Q = q \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The value of q is different for x dimension and y dimension separately. q should be determined by the most standard movement pattern of the tracking object, which is usually uncertain and we do not have the ability to calculate the process noise directly. Therefore, for most cases, Q is obtained by exhaustive search.

5.3.2 Observation Noise R

The covariance matrix of observation noise R should be a 2×2 matrix.

$$R = \begin{bmatrix} Rp & 0 \\ 0 & Rv \end{bmatrix}$$

where Rp is the covariance between the coordinate of matching result and the coordinate of true position. Rv is the covariance between the measured velocity and the true velocity.

The function to calculate covariance is

$$\sigma(x, \hat{x}) = E[(x - E[x])(\hat{x} - E[\hat{x}])].$$

x stands for the true value and \hat{x} stands for the observation value.

First of all, we set a known path and perform the tracking with this route. At each time point, we subtract the coordinates of the true position from the coordinates of the matching result on x-axis and y-axis separately (in units of centimetres). An identical operation is carried out between the true and the matching velocities.

After that, two vectors for the differences on x-axis and two vectors for the differences on y-axis are built. A covariance calculation is performed with the difference vectors on x-axis and y-axis separately. Then the values of measurement noise R_x and R_y are obtained.

$$R_x = \begin{bmatrix} Rpx & 0 \\ 0 & Rvx \end{bmatrix}$$

$$R_y = \begin{bmatrix} Rpy & 0 \\ 0 & Rvy \end{bmatrix}$$

where Rpx is the covariance of coordinate difference vector on x-axis

Rpy is the covariance of coordinate difference vector on y-axis

Rvx is the covariance of velocity difference vector on x-axis

Rvy is the covariance of velocity difference vector on y-axis

With a tracking approach as introduced in Chapter 6, several tracking results along known paths are shown below. The purpose of following experiments is to find an optimal R value which can be applied with all patterns of movement, because the R value for different kinds of movement can be very different. In order to calculate the measurement noise, it is necessary to apply a matching algorithm, the K-Nearest-Neighbours (KNN) Algorithm, which we will introduce in following section 6.2.1. This algorithm is utilized to calculate the estimation error between the true position and matched result (observation result). The error is measured in units of centimetres (cm).

1) The tracking result with known path along the x-axis

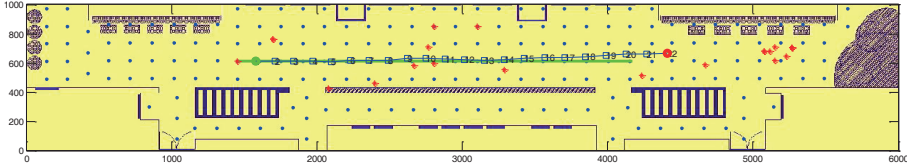


Figure 5.2. Tracking Result with Kalman filter along x-axis

On the figure 5.2, the green line represents the true path of tester walking and the blue line represents the tracking result with Kalman filter. The green point on the figure is the true start point and the red point represents the end point. The red “*” represents the measurement matching result. Marks and lines in following figure 5.3, 5.4, 5.5 have the same meaning in figure 5.2. According to the measurement data, it can be calculated that

$$R_x = 10^5 \begin{bmatrix} 4.0661 & 0 \\ 0 & 0.9997 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 1.2548 & 0 \\ 0 & 2.3266 \end{bmatrix}$$

2) The tracking result with known path along the y-axis

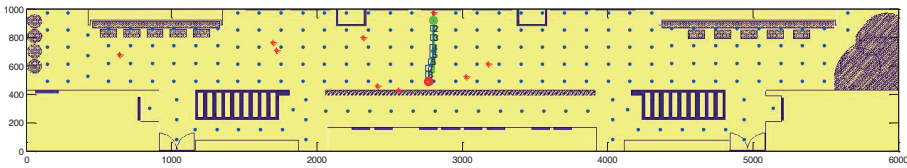


Figure 5.3. Tracking Result with Kalman filter along y-axis

R_x and R_y in this case is

$$R_x = 10^5 \begin{bmatrix} 7.8153 & 0 \\ 0 & 6.0162 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 3.4789 & 0 \\ 0 & 3.1804 \end{bmatrix}$$

3) The tracking result with known path along the diagonal

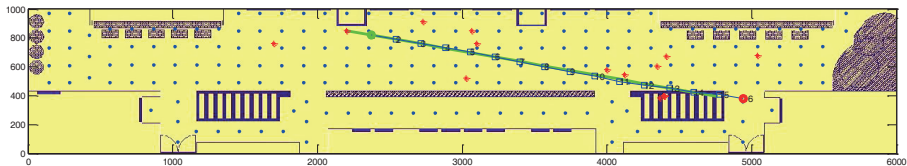


Figure 5.4. Tracking Result with Kalman filter along the diagonal

In this case, the measurement noise is calculated as

$$R_x = 10^5 \begin{bmatrix} 2.8446 & 0 \\ 0 & 4.0448 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 1.9586 & 0 \\ 0 & 4.1317 \end{bmatrix}$$

According to calculations above, the values of measurement noise for different movement patterns are quite different. Applying a certain value of R with other patterns of movement, it can be found that if R is too large and the tracking result will be estimated as a straight path, and details of the tracking route such as turning will be ignored.

Therefore, when a target moves along a circle or square, the measurement noise must be optimized to get a reliable tracking result. As one paper [25] states, *often times superior filter performance (statistically speaking) can be obtained by tuning the filter parameters Q and R*. To improve the accuracy through applying different Q and R to the Kalman filter, we measured R for several movement patterns and obtained an optimal value of R which can be applied in all movement patterns.

5.3.3 Optimal Values of Q and R

1) Move along a straight line

In order to obtain the optimal R values for pattern of moving along a straight line, we took the average of R from several experiments which a tester walks along straight lines in different directions. Since there are two types of tracking system has been designed, it is necessary to calculate R for them separately. Generally, in tracking system one which will be introduced in Chapter 6, the optimized values of R are

$$R_x = 10^5 \begin{bmatrix} 8.67107 & 0 \\ 0 & 6.14465 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 2.5828 & 0 \\ 0 & 2.8698 \end{bmatrix}$$

As for tracking system two which will be introduced in Chapter 7, through applying the mean value of R with different walking patterns, we found that it cannot give a satisfied result. This problem can be solved by multiply a coefficient to the mean value. The optimized values were approximately $\frac{1}{45}$ of the measurement values, which

$$R_x = 10^3 \begin{bmatrix} 7.921 & 0 \\ 0 & 2167.759 \end{bmatrix} \quad R_y = 10^3 \begin{bmatrix} 1.013 & 0 \\ 0 & 246.117 \end{bmatrix}$$

Changing the process noise does not affect the tracking result significantly. So, in both systems, we took the Q value as following.

$$Q_x = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \quad Q_y = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$$

2) Move along a circle or a square (natural movement pattern)

For the natural movement pattern, in tracking system one, the optimized R values are approximately $\frac{1}{50}$ of the mean of measurement values. The values of process noise Q which obtained by exhaustive search were given as well.

$$R_x = 10^5 \begin{bmatrix} 8.67107 & 0 \\ 0 & 6.14465 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 2.5828 & 0 \\ 0 & 2.8698 \end{bmatrix}$$

$$Q_x = \begin{bmatrix} 0.001 & 0 \\ 0 & 0.001 \end{bmatrix} \quad Q_y = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.3 \end{bmatrix}$$

Generally, in tracking system two, the optimal values of R are approximately $\frac{1}{100}$ of the measurement values. The final R and Q are shown below.

$$R_x = 10^5 \begin{bmatrix} 8.67107 & 0 \\ 0 & 6.14465 \end{bmatrix} \quad R_y = 10^4 \begin{bmatrix} 2.5828 & 0 \\ 0 & 2.8698 \end{bmatrix}$$

$$Q_x = \begin{bmatrix} 0.003 & 0 \\ 0 & 0.003 \end{bmatrix} \quad Q_y = \begin{bmatrix} 0.006 & 0 \\ 0 & 0.006 \end{bmatrix}$$

5.4 Filtered Tracking Result

This experiment was designed to indicate the improvement of tracking result with the Kalman filter. We took the coordinates of matching point which is calculated by KNN Algorithm (section 6.2.1) as the observation result z_k of the Kalman filter.

A comparison between the filtered tracking result and the original tracking result is shown below.

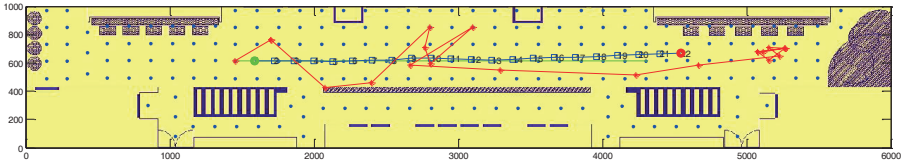


Figure 5.5. Tracking Result with Kalman filter

On figure 5.5, the green line represents the true path of tester walking, the red line represents the estimated route without Kalman filter and the blue line represents the tracking result with Kalman filter. The green point on the figure is the start point and the red point is the end point.

Compared with the true route on the map, there is a significant improvement on the tracking result with the Kalman filter. The filtered result is very close to the true route. The distance between two neighbour reference points is about 1.5 meters. The RMS error, details will be introduced in section 6.3, with the Kalman filter is 1.7 meters. Compare with the grid of 1.5 meters, the RMS error of 1.7 meters is satisfactory.

6 TRACKING SYSTEM ONE

In this chapter, a tracking system based on the average RSS in each second is introduced and tested. The tracking experiment conducted in this way: a tester held an iPad with Wi-Fi function on and walked through the entrance hall of E-huset. The device connected to the rogue AP automatically and all the sniffers were activated to collect packets broadcasted from the iPad. In the first place, a detailed description of the measurement data processing will be given. Then, with the KNN matching Algorithm, the processed data are matched with the fingerprint database. After applying Kalman filter with optimal parameters, the estimated tracking route will be given in the end.

6.1 Rules of Data Processing

During the tracking experiment, when a tester hold an iPad with turned on Wi-Fi function, the Evil Twin Attack is activated. Then a rogue AP is built according to the broadcasted probe request from this iPad. Since a stable wireless connection can increase the network traffic between the AP and the mobile device, this hacking action guarantees sufficient packets with RSS information for tracking and these packets are captured in continuous time. Although mobile devices keep broadcasting probe requests without a stable connection with a rouge AP, there is a few seconds pause between two probe requests. In tracking system one, the captured packets must be formed by a continuous time, which means that there should be at least one RSS value in each second. Thus, Evil Twin Attack is necessary to our tracking system. After the tracking data measurement, data processing will be performed on the server side with MATLAB.

According to the collected data on the server side, an example of the data structure is shown in following figure 6.1.

	A	B	C	D	E
	MAC_Address	Prober_Request	Received_Signal_Strength	Date	Exact_Time
16	"70:11:24:3a:87:8d"	"Test Only"	"-87"	"May 19	2014 15:47:16.358416000"
17	"70:11:24:3a:87:8d"		"-87"	"May 19	2014 15:47:16.358614000"
18	"70:11:24:3a:87:8d"		"-87"	"May 19	2014 15:47:16.360118000"
19	"70:11:24:3a:87:8d"		"-89"	"May 19	2014 15:47:16.361865000"
20	"70:11:24:3a:87:8d"		"-87"	"May 19	2014 15:47:16.362687000"
21	"70:11:24:3a:87:8d"		"-85"	"May 19	2014 15:47:16.363538000"
22	"70:11:24:3a:87:8d"	"Test Only"	"-87"	"May 19	2014 15:47:16.380278000"
23	"70:11:24:3a:87:8d"	"Test Only"	"-87"	"May 19	2014 15:47:16.390460000"
24	"70:11:24:3a:87:8d"		"-89"	"May 19	2014 15:47:16.398095000"
25	"70:11:24:3a:87:8d"		"-87"	"May 19	2014 15:47:16.399496000"
26	"70:11:24:3a:87:8d"		"-83"	"May 19	2014 15:47:16.516903000"
27	"70:11:24:3a:87:8d"		"-83"	"May 19	2014 15:47:16.517507000"
28	"70:11:24:3a:87:8d"		"-85"	"May 19	2014 15:47:16.518958000"

Figure 6.1. Structure of Measurement Data from One Sniffer

This figure illustrates that the measurement data includes five types of information - MAC address, probe request, RSS, date and time stamp. RSSs and time stamps are two required information in our tracking system. Through analysing the time information of the captured packets, the collected data is divided into small groups by one second time slot. As we known, the normal speed of human is about 1.4 meters per second, and the distance between two adjacent reference points is 1.5 meters. It means that the distance of target has moved within one second is approximate equals to the separation of two reference points. Then, in an ideal situation, every second the target can be matched to a new reference point. Thus, it is reasonable to divide the data with a time slot of one second. According to the measurement data, it can be observed that, in general, there are approximately 20 records within a time slot. Sometimes, the number of samples in one second can be up to 50.

In order to match the measurement data with the fingerprint database, it is necessary to find a certain value of RSS in each time slot as a fingerprint. The first method to obtain this value is to calculate the mean of all RSSs within each time slot. However, as mentioned in calibration data processing (Chapter 4), sniffers drop packets with RSS below -92 dBm actively. Therefore, simply matching the mean of RSSs with database could lead a large error distance.

The Expectation Maximization (EM) Algorithm for the truncated data can be used to estimate the true mean value of the RSSs within a time slot. This algorithm has been introduced in section 4.3.3.2. In a special case, within one second, the sniffer doesn't collect any packet. As we did in calibration, it will be given a value of -156 dBm, which is the minimum fixed RSS value with the EM Algorithm. The structure of the processed data is shown in figure 6.2.

	Pi 1 RSSI	Pi 2 RSSI	Pi 3 RSSI	Pi 4 RSSI	Pi 5 RSSI	TIME
	1	2	3	4	5	6
1	-99.9386	-156	-84.5556	-85.5016	-156	131747
2	-104.6748	-156	-84.0714	-84.9706	-85	131748
3	-95.1410	-156	-76.5000	-85.5540	-91.0742	131749
4	-101.1083	-156	-82.9216	-82.7869	-101.0581	131750
5	-95.3691	-89	-75.3750	-81.0625	-91.7513	131751
6	-89.2052	-100.2956	-80.3408	-83.3333	-98.9402	131752
7	-84.8016	-156	-85.4728	-79.2000	-88.9672	131753
8	-109.0691	-101.5661	-75.2632	-89.1871	-96.7883	131754
9	-85.2413	-156	-66.6923	-78.2308	-86.5184	131755
10	-98.8977	-156	-76.5429	-81.6980	-96.8544	131756

Figure 6.2. Structure of the Processed Data

In this figure, the RSS is in dBm scales. The “TIME” column represents the time stamps e.g. the “TIME” column data in row one can be expressed as “13:17:47”.

6.2 Matching Phase

On the matching phase, the processed data is matched with the fingerprint database. In this section, matching algorithms in tracking system one are introduced, including the K-Nearest-Neighbours (KNN) matching algorithm and a method to give initial values of the Kalman filter.

6.2.1 Nearest-Neighbour (NN) Algorithm

Although the Nearest Neighbour (NN) matching algorithm is not used in tracking system one, a simple introduction to NN Algorithm is present here. In our database, as mentioned before, each reference point is characterized by a RSS vector with five RSS values and expressed as $\mathbf{s}^{(i)} = (s_1^{(i)}, s_2^{(i)}, s_3^{(i)}, s_4^{(i)}, s_5^{(i)})$, $1 \leq i \leq 210$. At each time slot, a vector of measurement signal strengths is represented by $\mathbf{t}^{(j)} = (t_1^{(j)}, t_2^{(j)}, t_3^{(j)}, t_4^{(j)}, t_5^{(j)})$. Then, the Euclidean signal distance between the vector $\mathbf{t}^{(j)}$ and all the reference points in database can be calculated. For example, the Euclidean signal distance between time point one and all reference points can be calculated by

$$l_i = \sqrt{(s_1^{(i)} - t_1^{(1)})^2 + (s_2^{(i)} - t_2^{(1)})^2 + (s_3^{(i)} - t_3^{(1)})^2 + (s_4^{(i)} - t_4^{(1)})^2 + (s_5^{(i)} - t_5^{(1)})^2}$$

Find the minimum l_i , the value of i indicates which is the nearest reference point. Then the position of the mobile device at time point one can be estimated as the reference point number i .

6.2.2 K-Nearest-Neighbours (KNN) Algorithm

As mentioned before, KNN is a popular location estimation algorithm which has been widely used in fingerprint technique to increase tracking accuracy. It is also suitable for our tracking system. Based on the NN Algorithm, after l_i calculation, k nearest

neighbours are selected with k minimum values of l_i . Then the average coordinates of these matching reference points is calculated as the estimate of the target position. In our system, we choose k equals to 4, which means that the number of nearest neighbours is four and the nearest reference points with minimum four Euclidean signal distances will be selected. The matching position can be calculated by [31]

$$p = \frac{\sum_{i=1}^k \left(\frac{1}{l_i + \varepsilon} \cdot p_i \right)}{\sum_{i=1}^k \frac{1}{l_i + \varepsilon}}$$

where ε is an extremely small constant number to avoid dividing by zero. p_i is the coordinates of the selected nearest reference points.

6.2.3 The Initial Value of the Kalman Filter

An accurate initial value of the Kalman filter can significantly improve the tracking accuracy. In our system, the initial values contain coordinates of the initial position and the initial velocity. In terms of the initial position, using the first measurement data point as the initial value always leads to a large error, because the first estimate position may have a large error distance. In spite of the Kalman filter has the function of correction, since the collected samples are limited, it may not correct the estimate result in time. In order to solve this problem, a method is developed to decide the true position of the start point which is applied as the initial value of the Kalman filter.

The entrance hall of our experiment environment contains four entrance doors. Every person who comes into this entrance hall has to go through one of them. Thus, the start point must be the coordinates of one of these doors. The map of the entrance hall is divided into four zones and if the matching result falls in a zone, the accurate position should be the door in this zone. An experiment was conducted to find the matching position when a tester stands at each door. Also, the optimal boundaries to divide the corridor can be determined according to this experiment.

Since the purpose of this method is to make the matched point falling in a certain zone instead of finding the correct position, we found that a larger number of nearest neighbours (k) can help increasing the probability of the results falling in the right zone. After multiple experiments, it was proved that the boundaries perform at their best when k equals to 9. Therefore, applying this method to the measurement data within the first second, then an accurate initial position can be inferred.

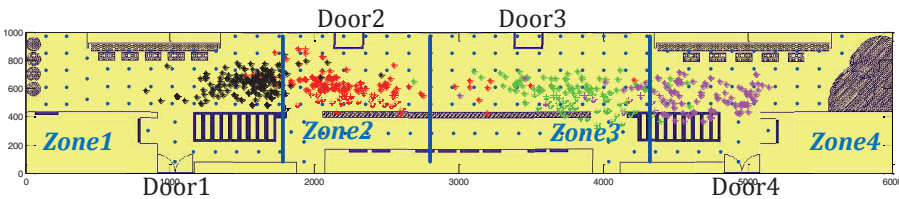


Figure 6.3. Division of the Corridor Map

In figure 6.3, the blue line represents the boundaries of different zones, the black points are the matching results when the tester stands at Door 1, the red points are the matching results when the tester stands at Door 2, the green points are the matching results when the tester stands at Door 3, and the pink points are the matching results when the tester stands at Door 4.

Sometimes, the first measurement point can be matched into a wrong zone, which causes an unacceptable tracking result. A large number of experiments results provide the probabilities of position matched correctly in each zone are

$$q_1 = 94.95\% ; q_2 = 92.75\% ; q_3 = 93.44\% ; q_4 = 91.37\%.$$

With this method, for example, if the first measurement point is matched in zone one, the start point should be at Door 1, and the coordinates of Door 1 will be set as the position initial value of the Kalman filter.

As for the initial values of velocity, the average velocities on x-axis and y-axis during the whole experiment are applied as the initial values.

6.3 Error Analysis

In our tracking system, the root-mean-square (RMS) error is utilized to evaluate the tracking accuracy. The RMS error represents the sample standard deviation of the differences between the filtered result and the true position. We assume that the tracking target is moving with a constant velocity. Within each time slot, one estimated position is obtained. So according to the total number of time slots, the estimated walking route can be divided into several points uniformly and the true route is divided as well. Then, each estimated point corresponds to a true position. With the formula below, the RMS error distance D_{rms} can be calculated.

$$D_{rms,x} = \sqrt{\frac{\sum_{i=1}^n (x_i - \hat{x}_i)^2}{n}}$$

$$D_{rms,y} = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}}$$

$$D_{rms} = \sqrt{D_{rms,x}^2 + D_{rms,y}^2} = \sqrt{\frac{\sum_{i=1}^n [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2]}{n}}$$

where, $D_{rms,x}$ is the error distance in x dimension. x_i is the x coordinate of true position. \hat{x}_i is the x coordinate of estimated position. $D_{rms,y}$ is the error distance in y dimension. y_i is the y coordinate of true position. \hat{y}_i is the y coordinate of estimated position. n is the total number of time slots.

6.4 Tracking Results and Evaluation

In this part, two tracking experiments results are presented. The first experiment is a constant-speed-walking test from door to door. The second one is an experiment of walking along a large square with constant velocity.

Applying the tracking system one as described before, data of target movement is processed with EM algorithm. The start point is decided as the initial value of Kalman filter. Then, the estimated coordinates are taken as the observation values of Kalman filter. Optimal R and Q have been loaded to Kalman filter as well. The experiment conducted in a few people environment and the tracking results are shown below.

1) Walking from door to door

In the first experiment, a tester walked along the green line which shows in figure 6.4 with a constant speed (approximately equals to 1.3 m/s). The start point and the end point are two doors of this entrance hall.

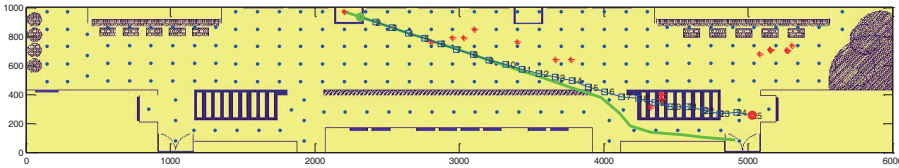


Figure 6.4. Tracking Result with EM Algorithm from Door to Door

In figure 6.4, the red “*” represent the matching results with KNN. The green line represents the true walking route and the blue line represents the tracking result with Kalman filter. The green point on the figure is the start point and the red point represents the end point. Marks and lines in all following figures of tracking results have the same meaning as this figure. The RMS error is 1.58 meters. It can be seen that the tracking result is relatively close to the true walking route.

2) Walking along a square

In this experiment, as the figure shows below, a tester walked along a large square which presents by the green line, with a constant speed. In the square, the shadowing part presents a brick wall. So, in a relatively complicated environment the direction of walking has changed three times. This experiment is used to evaluate the performance of our tracking system when the tracking target walks in a nature pattern. The tracking result is shown in figure 6.5.

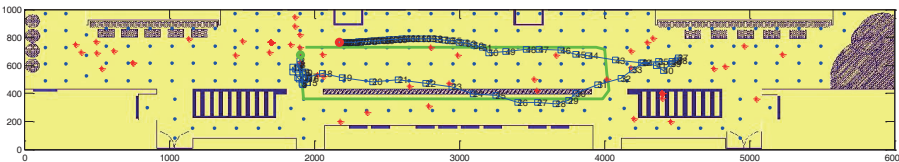


Figure 6.5. Tracking Result with EM Algorithm along the Square

The RMS error is 4.88 meters in this experiment. The tendency of the movement can be reflected by the tracking result. However, the turning points are not matched clearly.

In conclusion, this tracking system works well with a straight-line walking pattern. The result of the first experiment gives an approving RMS error distance. However, if the target direction changes frequently, the tracking result may not reflect the true route accurately. Since the RMS error in the second experiment is about 5 meters, it seems unreliable to use this system to track a square or circle route with small radius. An advantage of this tracking system is that there is no strict requirement on synchronization. NTP can provide a satisfactory synchronization among all units. However, a limitation of this tracking system is that it is hard to track a short time movement. Since we only get one estimated result within each second, a movement with short duration cannot provide sufficient observation values for the Kalman filter.

7 TRACKING SYSTEM TWO

In chapter 6, through the application of expectation maximization (EM) algorithm, a tracking system based on the average RSS within one second is introduced. In this chapter, another tracking system is implemented by utilizing the instantaneous value of RSS. This system involves new methods of time axis adjustment and synchronization, as well as a new matching algorithm. In the end, comparison between the tracking results from our two tracking systems will be presented.

7.1 Time Axis Synchronization

A simple idea to realize a tracking system by instantaneous value of RSS is to match the captured packets one by one. Since a single fingerprint consists of five RSSs from five sniffers, high precision time synchronization among five sniffers is a precondition to implement the tracking system two. As described before, when a packet transmits between the mobile device and the AP, all five sniffers attempt to capture this packet. However, according to our captured packets (captured information includes the RSS, MAC address, probe request and exact time of the received packet), there is no identity information such as packet number. In a good data transmission condition, there will be up to 50 packets collected by a sniffer in one second. It is difficult to distinguish which five RSSs from five sniffers corresponding to a certain transmitted packet. Furthermore, not all transmitted packets will be captured by the sniffers, since every sniffer has a packet loss mechanism and the ability of capturing the Wi-Fi data packets is limited by the wireless card. Also, received packets with weak signal strength are identified as interference noise which will be dropped by sniffer. Therefore, according to the order of packets received, the one-to-one correspondence between different sniffers is unreliable.

Another problem is that the same packet collected by different sniffers could be saved in a different time slot. Although the NTP has been utilized to establish a time server and synchronize the local time between each sniffer (section 3.2.1), due to the packet traffic time and duration of packet processing, the arrival time of a packet in different sniffers

are not exactly the same. Thus, the synchronization by NTP is no longer fulfilling the tracking system two and the time axis adjustment is necessary for the collected data. It mainly aims at establishing accurate one-to-one correspondence between time slot and broadcast packet.

In a word, instead of using the EM Algorithm to estimate RSS in one second, the tracking system two requires a new algorithm to realize the calculation of time difference and the adjustment of time axis. With this system, the movement details in one second can be illustrated.

7.1.1 Measurement Data Analysis

In order to distinguish which five RSSs belong to the same transmitted packet, special identification information must be determined. We found that the probe requests with specific SSID can be used as the packet identification.

597	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.116973000"
598	"70:11:24:3a:87:8d"	"Test Only"	"-79"	"May 19	2014 15:48:19.354943000"
599	"70:11:24:3a:87:8d"		"-79"	"May 19	2014 15:48:19.355206000"
600	"70:11:24:3a:87:8d"	"Test Only"	"-77"	"May 19	2014 15:48:19.365246000"
601	"70:11:24:3a:87:8d"	"Test Only"	"-77"	"May 19	2014 15:48:19.376805000"
602	"70:11:24:3a:87:8d"	"Test Only"	"-75"	"May 19	2014 15:48:19.387060000"
603	"70:11:24:3a:87:8d"		"-77"	"May 19	2014 15:48:19.396105000"
604	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.511849000"
605	"70:11:24:3a:87:8d"		"-77"	"May 19	2014 15:48:19.557523000"
606	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.673115000"

(a)

536	"70:11:24:3a:87:8d"		"-85"	"May 19	2014 15:48:19.153198000"
537	"70:11:24:3a:87:8d"	"Test Only"	"-79"	"May 19	2014 15:48:19.391109000"
538	"70:11:24:3a:87:8d"		"-79"	"May 19	2014 15:48:19.391406000"
539	"70:11:24:3a:87:8d"	"Test Only"	"-81"	"May 19	2014 15:48:19.401364000"
540	"70:11:24:3a:87:8d"	"Test Only"	"-81"	"May 19	2014 15:48:19.412919000"
541	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.423156000"
542	"70:11:24:3a:87:8d"		"-83"	"May 19	2014 15:48:19.432220000"
543	"70:11:24:3a:87:8d"		"-81"	"May 19	2014 15:48:19.548019000"
544	"70:11:24:3a:87:8d"		"-83"	"May 19	2014 15:48:19.593786000"
545	"70:11:24:3a:87:8d"		"-87"	"May 19	2014 15:48:19.709375000"

(b)

455	"70:11:24:3a:87:8d"		"-71"	"May 19	2014 15:48:18.944476000"
456	"70:11:24:3a:87:8d"	"Test Only"	"-67"	"May 19	2014 15:48:19.182407000"
457	"70:11:24:3a:87:8d"		"-67"	"May 19	2014 15:48:19.182703000"
458	"70:11:24:3a:87:8d"	"Test Only"	"-67"	"May 19	2014 15:48:19.192638000"
459	"70:11:24:3a:87:8d"	"Test Only"	"-67"	"May 19	2014 15:48:19.204217000"
460	"70:11:24:3a:87:8d"	"Test Only"	"-65"	"May 19	2014 15:48:19.214491000"
461	"70:11:24:3a:87:8d"		"-65"	"May 19	2014 15:48:19.223510000"
462	"70:11:24:3a:87:8d"		"-65"	"May 19	2014 15:48:19.339294000"
463	"70:11:24:3a:87:8d"		"-63"	"May 19	2014 15:48:19.385004000"
464	"70:11:24:3a:87:8d"		"-63"	"May 19	2014 15:48:19.500583000"

(c)

291	"70:11:24:3a:87:8d"		"-71"	"May 19	2014 15:48:19.172890000"
292	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.410763000"
293	"70:11:24:3a:87:8d"		"-85"	"May 19	2014 15:48:19.411075000"
294	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.420985000"
295	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.432589000"
296	"70:11:24:3a:87:8d"	"Test Only"	"-81"	"May 19	2014 15:48:19.442832000"
297	"70:11:24:3a:87:8d"		"-81"	"May 19	2014 15:48:19.451888000"
298	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.567678000"
299	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.613457000"
300	"70:11:24:3a:87:8d"		"-71"	"May 19	2014 15:48:19.729053000"

(d)

528	"70:11:24:3a:87:8d"		"-79"	"May 19	2014 15:48:19.130831000"
529	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.368779000"
530	"70:11:24:3a:87:8d"		"-85"	"May 19	2014 15:48:19.369179000"
531	"70:11:24:3a:87:8d"	"Test Only"	"-83"	"May 19	2014 15:48:19.379039000"
532	"70:11:24:3a:87:8d"	"Test Only"	"-85"	"May 19	2014 15:48:19.390643000"
533	"70:11:24:3a:87:8d"	"Test Only"	"-85"	"May 19	2014 15:48:19.400882000"
534	"70:11:24:3a:87:8d"		"-81"	"May 19	2014 15:48:19.409959000"
535	"70:11:24:3a:87:8d"		"-75"	"May 19	2014 15:48:19.525729000"
536	"70:11:24:3a:87:8d"		"-73"	"May 19	2014 15:48:19.571407000"
537	"70:11:24:3a:87:8d"		"-69"	"May 19	2014 15:48:19.686942000"

(e)

Figure 7.1. Structures of Measurement Data from Five Sniffers Separately

In figure 7.1, (a) to (e) illustrate parts of measurement data collected by Raspberry Pi #1 to #5. Since mobile device has connected to the AP with SSID "Test Only", we call the captured packets with probe request "Test Only" the ID packets. A mobile device broadcasts this kind of packet every seven seconds. And during each time of broadcasting, it repeats sending the ID packet 4 times. So, if ID packets appear 4 times in all five sniffers within the same time period of one second, we can find a one-to-one corresponding relationship among these captured packets. And with this relationship, the time differences between time axes of different sniffers can be determined and data from different sniffers will be able to be synchronized.

7.1.2 Method of Data Processing

In most case, there are more than one group of ID packets are recorded. The time differences between different sniffers can be obtained by subtracting the time stamps of ID packets. Therefore, the average value of time differences will be calculated by several groups of ID packets, and it will be utilized to adjust the time axis. The time axis of Raspberry Pi #5 is regarded as the standard time axis, since it's in the middle of the indoor environment and the NTP time server is settled on it. After the time axis adjustment, there is still a tiny time difference in hundred-microsecond level (10^{-4} second) between sniffed time of the same packet. In other word, the precision of synchronization is in 10^{-4} second level now.

Next, the duration of time slot should be determined. Generally, the time difference between different sniffers is in hundred-microsecond level (10^{-4}), and the minimum time difference between two received packets is in microsecond level (10^{-6}). Thus, the new time slot should be set as the maximum value of the time differences between sniffers. In a special case, if there is no ID packet for the time axis adjustment has been captured, the time slot will be set to 0.1 second.

After that, the time axis will be evenly divided by the new time slot. Within one slot, the records are regarded as broadcasted from the same packet. The new structure of measurement data is shown below.

42	-87	0	0	0	-87
43	-83	-87	0	0	-89
44	-77	-83	-89	0	-85
45	-77	0	0	0	-81
46	-83	-87	0	0	-83
47	-75	-87	0	0	-81
48	-75	0	0	0	-83
49	-77	-73	-81	-79	-83

Figure 7.2. Structure of Measurement Data after adjustment

In figure 7.2, five RSS values in one row constitute a RSS vector corresponds to a time slot. The duration of time slot will also be used as the update time for the Kalman filter. To be noticed, the “0” in measurement data does not stand for the value of signal strength, it means no data has been collected.

Since not all five RSS values are known in one RSS vector, the matching algorithm we used in Chapter 6 has to be abandoned. A new matching algorithm should be discussed to deal with those partly known RSS vectors.

7.2 Matching Phase

Because of the noise floor, a broadcasted packet may not be captured by all sniffers. In an ideal situation, one packet is captured by all five sniffers. Then we can directly match this point with our fingerprint database by calculating the Euclidean distance and select the minimum one (the NN Algorithm), as we discussed in section 6.2.1. The most likelihood reference point can be selected as the current position of the tracking target.

In another situation, within a time slot, a broadcasted packet is captured by four of five sniffers. It is reasonable to utilize these four RSSs to calculate and select the nearest reference point. The matching result will still be reliable.

However, if a packet is captured by three or two or even one sniffer, a new strategy should be applied to help selecting the matching point. For example, a packet was captured by Raspberry Pi #1 and Raspberry Pi #2. Then the fingerprint can be recorded in form of $(t_1, t_2, x_3, x_4, x_5)$, where t_1, t_2 are the captured RSSs, and x_3, x_4, x_5 are unknown. First, calculate the Euclidean distance between the t_1, t_2 and the fingerprint database by

$$d_i = \sqrt{(s_1^{(i)} - t_1)^2 + (s_2^{(i)} - t_2)^2}$$

where $1 \leq i \leq 210$. Second, the reference point with the minimum Euclidean distance $d_{i_{\min}}$ is selected. However, since only two of the five RSS are known, this selected reference point may not be reliable. In one case, there are several reference points with the same $d_{i_{\min}}$ are selected through this way. In the other case, the d_i of the true reference position cannot be selected since it is not the true $d_{i_{\min}}$ if x_3, x_4, x_5 are known to us. In order to solve these two special cases, a value of α is set as a threshold which helps to select all the possible matching points with d_i close to $d_{i_{\min}}$. The reference points with $d_i < d_{i_{\min}} + \alpha$ are selected as the potential matching points. In the fingerprint database, the rest three values of the potential matching points are known. Since the reason of missing data in most case is that the RSSs are too weak to be sniffed, we calculate the sum of the rest three values of the potential matching points by $s_h^{(i)} = s_3^{(i)} + s_4^{(i)} + s_5^{(i)}$. Then the most likelihood result should be the matching point with the weakest $s_h^{(i)}$. Finally, the potential matching points with the minimum $s_h^{(i)}$ will be regarded as the final matching point.

As for the situation of no sniffer or only one sniffer captures a certain packet within a time slot, the prediction of the Kalman filter from the previous stage should be regarded as the current position of the tracking target.

7.3 Matching Results and Evaluation

In this section, the raw experiment data in section 6.4 are processed with the algorithms we have introduced in this chapter. Evaluation of tracking system two is presented. We also make a comparison between the results we obtained by tracking system one and two.

1) Walking from door to door

Because of the time slot is larger than the minimum time difference between packets, there may be several records within one time slot. Two methods are discussed here to solve this problem.

a) Method One for Solving Several Records in One Time slot

The first method is taking the first record of each time slot. Generally, the normal walking speed of a human being is 3 to 4 miles per hour, which is around 1.34 to 1.8 meters per second. Assuming that the time slot is 0.18 second, it means that in this time slot, the approximate walking distance is around 0.32 meter. Then, we can make a reasonable assumption that the RSS in a single time slot does not change a lot with in a time slot, since the tracking target almost keeps its position within a (10^{-4}) duration. Therefore, we take the first recorded RSS value within each time slot as the observation value of the Kalman filter. With this method, the tracking result is shown in figure 7.3.

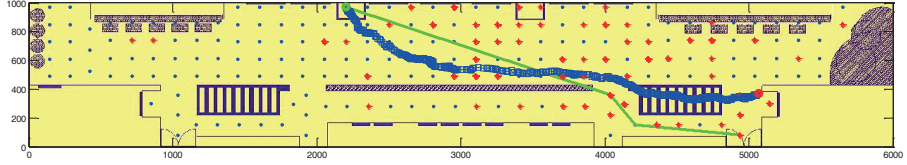


Figure 7.3. Tracking Result with Method One from Door to Door

In this case the RMS error is calculated as 2.17 meters.

b) Method Two for Solving Several Records in One Time slot

The second method is taking the mean value of all RSSs within each time slot. In method one, the data processing is based on the assumption that the RSS is stable within each time slot. However, if the assumption is not correct, method one can result in great error. Thus, in method two, all RSSs within one time slot are averaged as the corresponding RSS vector for each time slot. With the same Kalman filter in method one, the tracking result is shown in figure 7.4.

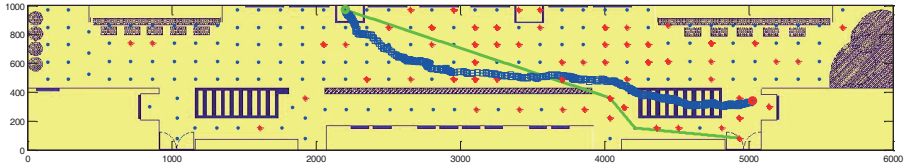


Figure 7.4. Tracking Result with Method Two from Door to Door

The RMS error is 2.11 meters in this situation. Comparing the tracking results with the method one, the method two has a better tracking performance because it has a smaller RMS error. Therefore, in the following tracking test along the square, we only estimate the route with the second method.

2) Walking along the square

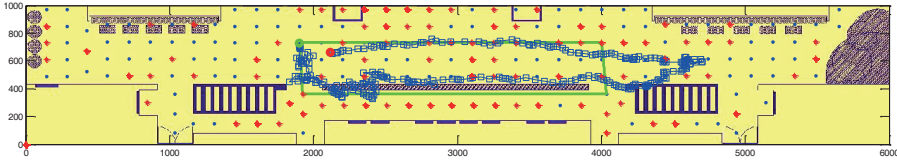


Figure 7.5. Tracking Result with Method Two Along the Square

The RMS error distance is 6.59 meters in this experiment with tracking system two. Comparing with the tracking result of tracking system one, the RMS error is 1.71 meters larger. However, the tracked result seems fixed well to the true route square. Admittedly, the RMS error is a significant evaluation standard, sometimes a good-looking tracking route may give a higher error distance.

In conclusion, the RMS error distance of walking pattern of door-to-door is acceptable but not as good as the result obtained by system one. The RMS error calculated in experiment two is larger than system one as well. However, the good-looking tracking route shows that it has better performance when the movement and environment is complicated. Furthermore, since the time slot is much smaller than one second, we can have sufficient observation values for Kalman filter even if the tracking time is short. Thus, the tracking system two has potential to track a short time movement.

8 INFLUENCE FACTORS

In order to investigate how the influence factors affect the tracking accuracy, three main aspects involves target speed, shadowing and type of mobile device are discussed in this chapter. The movement pattern of a tester holding an iPad mini walking along a known straight line in x-axis direction with a speed of 1.3 m/s was considered as the standard walking pattern. Three control experiments are designed to investigate how these three influence factors affect our tracking systems. Tracking results of experiments for different influence factors are presented from section 8.1 to section 8.3. Then details of performance evaluation with comparisons of all factors will be given in section 8.4. Section 8.5 gives a suggestion against the factor of target speed.

8.1 Speed of Target

In order to compare with the standard walking pattern, a tester with an iPad mini walked along the known path with a higher speed. The standard speed was 1.3 m/s and the higher speed was 2.5 m/s (both speeds were calculated after the experiment). For this group of control experiment, the same raw data were applied with two different tracking systems which were introduced in Chapter 6 and Chapter 7. In this way, we can clearly analyse how serious will the speed factor affect our systems and which system has a better performance.

1) With tracking system one

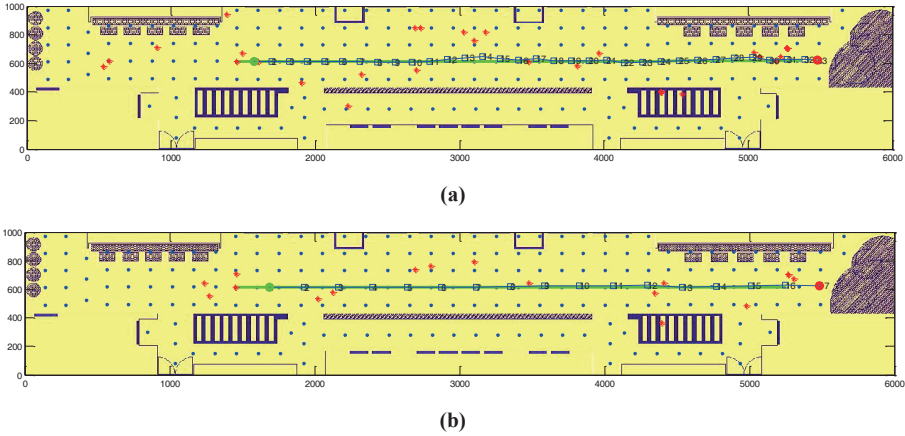


Figure 8.1. Tracking Result with Tracking System One with Normal Speed 1.3m/s (a) and High Speed 2.5m/s (b)

These two figures in figure 8.1 indicate the tracking results with tracking system one with different speeds. The RMS error for (a) is 1.26 meters and for (b) is 2.35 meters.

2) With tracking system two

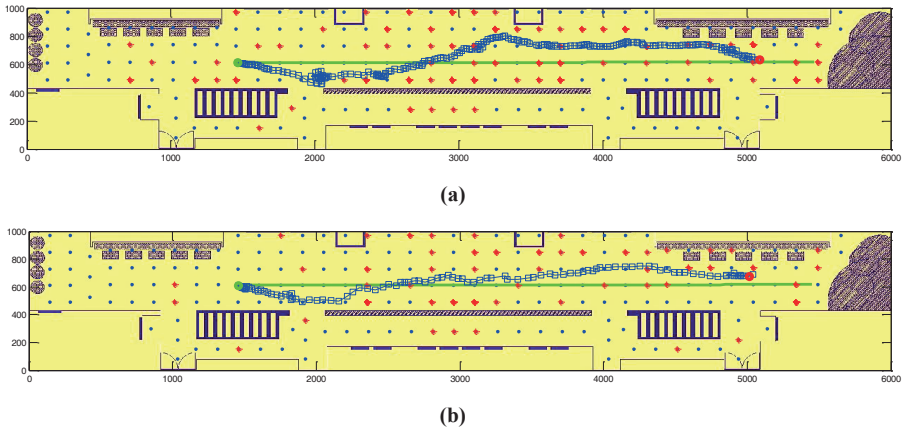


Figure 8.2. Tracking Result with Tracking System Two with Normal Speed 1.3m/s (a) and High Speed 2.5m/s (b)

Figure 8.2 indicates the tracking results with tracking system two with different speeds. The RMS error for (a) is 2.46 meters and for (b) is 3.83 meters.

8.2 Shadowing

As mentioned before, another experiment was designed and performed to indicate the effect of shadowing. A tester put the iPad mini in his pocket and walked along the

known path with a normal speed (approximately equals to 1.3 m/s). Compared with the standard walking pattern, the tracking results from different systems are shown in figure 8.3 and 8.4.

1) With tracking system one

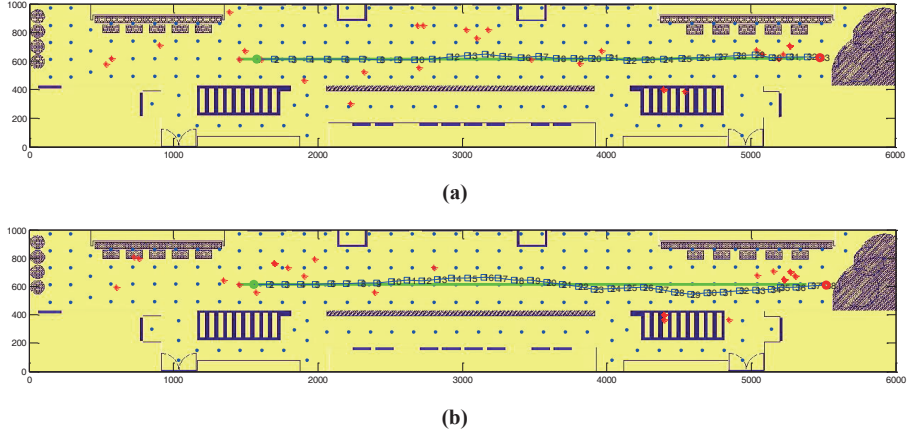


Figure 8.3. Tracking Result with Tracking System One with Device in hand (a) and in Pocket (b)

Figure 8.3 (a) is the result of standard pattern with tracking system one, which is the same as the figure 8.1 (a). Figure 8.3 (b) is the result of the situation which the tester put the iPad in pocket and the RMS error is 1.3 meters.

2) With tracking system two

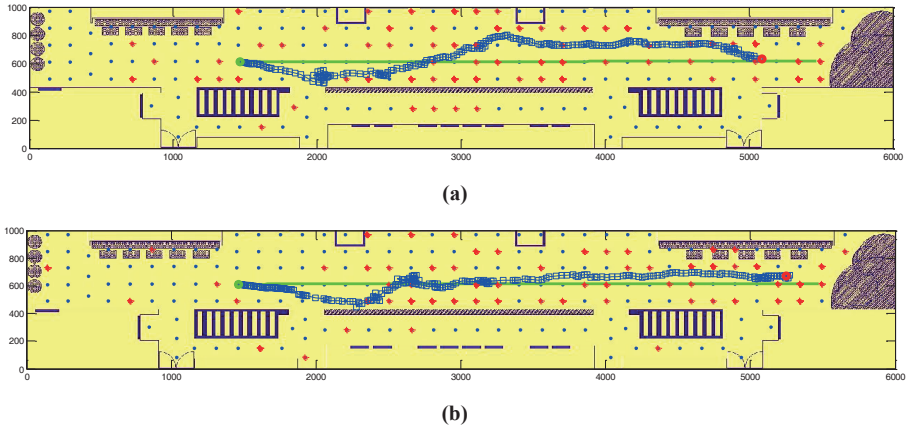


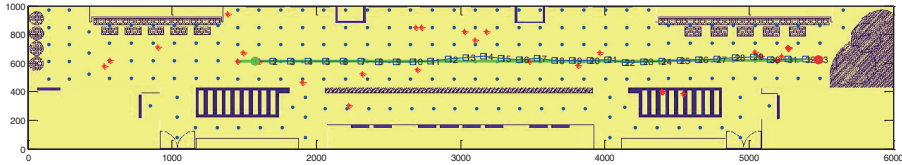
Figure 8.4. Tracking Result with Tracking System Two with Device in hand (a) and in Pocket (b)

Figure 8.4 (a) is the result of standard pattern with tracking system one, which is the same as the figure 8.2 (a). Result in figure 8.4 (b) has a RMS error of 3.08 meters.

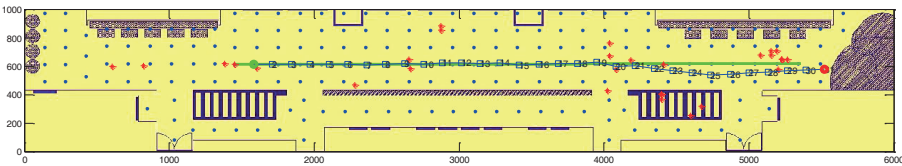
8.3 Type of Mobile Device

In order to figure out whether different types of mobile devices affect the tracking accuracy, we performed an experiment with an iPhone 4s. A tester held an iPhone 4s in hand walked along the known path with a normal speed (approximately equals to 1.3 m/s). The tracking results from different systems are shown in figure 8.5 and figure 8.6.

1) With tracking system one



(a)

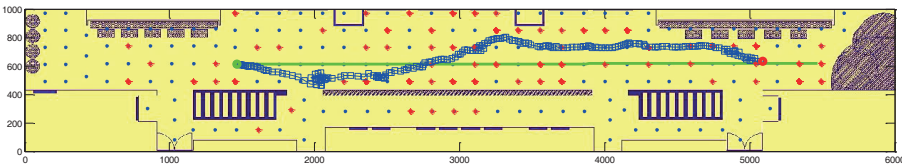


(b)

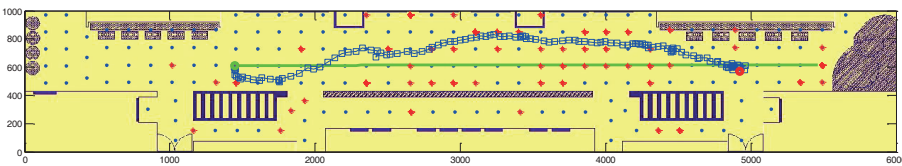
Figure 8.5. Tracking Result with Tracking System One with Device iPad mini (a) and iPhone 4s (b)

Figure 8.5 (a) is the same as figure 8.1 (a). The iPhone 4s tracking result with tracking system one has a RMS error of 1.5 meters.

2) With tracking system two



(a)



(b)

Figure 8.6. Tracking Result with Tracking System Two with Device iPad mini (a) and iPhone 4s (b)

Figure 8.6 (a) is the same as figure 8.2 (a). The tracking result in figure 8.6 (b) has a RMS error of 2.14 meters.

8.4 Comparison and Error Distribution

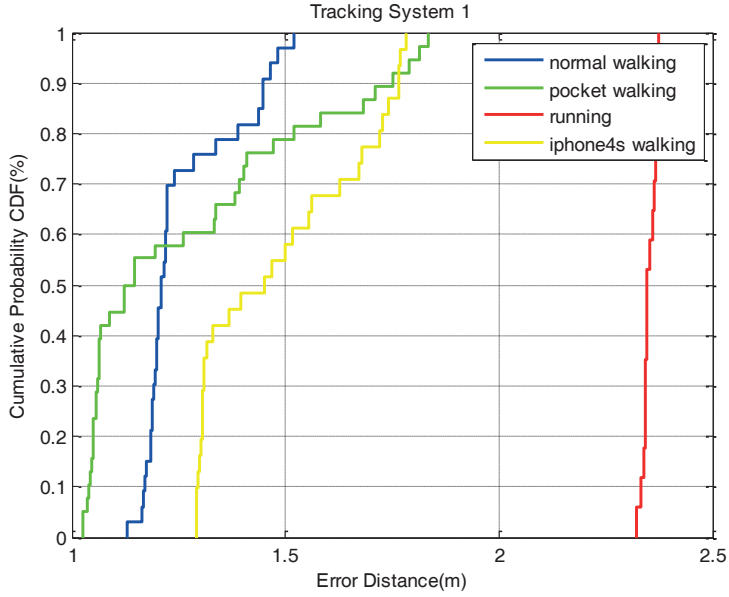
The table 8.1 is given to present the comparisons of RMS error distances with different influence factors. The accuracy changes in percentile compared with the standard walking pattern are also presented in this table.

	Tracking System One		Tracking System Two	
	RMS error	Accuracy change	RMS error	Accuracy change
Standard	1.26 meters		2.46 meters	
High Speed	2.35 meters	-86%	3.83 meters	-56%
Shadowing	1.30 meters	-3.2%	3.08 meters	-25%
Different Type	1.50 meters	-19%	2.14 meters	+13%

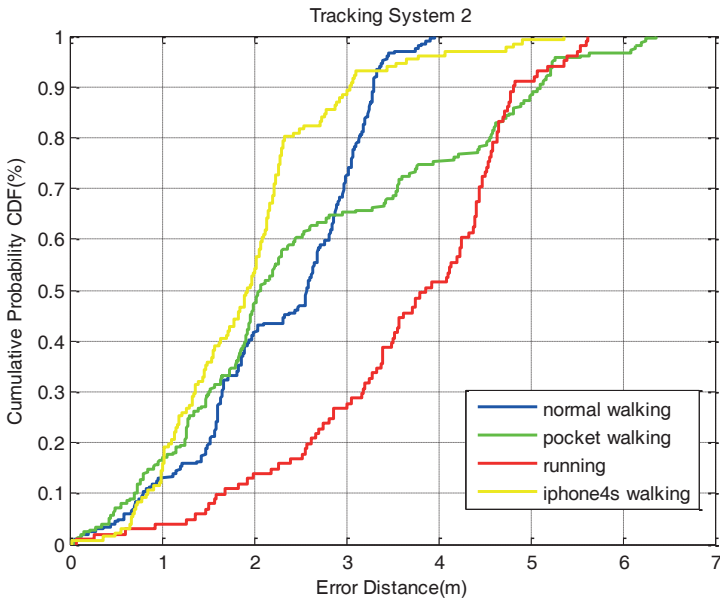
Table 8.1 RMS Errors and Accuracy Change with Different Influence Factors

According to the table, we can conclude that both speed and shadowing factors decrease the tracking accuracy. For tracking system one, it can be seen that it is sensitive to high speed, where the accuracy decreases by 86%. However, it can resist the influence of shadowing very well. The tracking result of iPhone 4s states an accuracy decrease of 19%. As for the tracking system two, it can be found that the ability of resist high speed (-56%) is better than tracking system one. Shadowing factor leads a 25% decrease in accuracy. However, the accuracy for iPhone 4s has increased 13%. Thus, it is hard to conclude how the types of mobile devices affect our tracking systems.

In order to analyse the effects of different factors in details, the CDF of error distribution is given in the following figure 8.7.



(a)



(b)

Figure 8.7. CDF of Error Distance with Tracking System One (a) and Two (b)

Diagrams in figure 8.7 indicate the error distributions with tracking system one and two, where the blue line represents the distribution of standard walking pattern. The green

line is obtained from the test when the tester put the mobile device in pocket, and the red line corresponds to the case when the tester is running instead of walking in the corridor. The yellow line stands for the distribution of using iPhone 4s as the tracking target.

The comparison shows that, all these three influence factors impact the system performance. Speed is the most significant factor, which leads to larger RMS errors in both tracking systems. The shadowing factor reduces the accuracy not serious but still makes the maximum error distance larger. As for different types of mobile devices, it does have some effects on the tracking results. However, in tracking system two, using iPhone 4s instead of the original iPad mini improves the accuracy. Since we do not have enough knowledge of the transmit power and types of antenna of different devices, it makes the influences with different devices uncertain. In general, the tracking system one is more stable and reliable and has a better performance against most kinds of influence factors.

8.5 Improvement

As mentioned before, the speed of target is a major influence factor which reduces the tracking accuracy significantly. In order to improve the tracking result, we adjusted the process noise Q in tracking system two. As [23] states, *we might reduce the magnitude of Q_k if the user seems to be moving slowly, and increase the magnitude if the dynamics start changing rapidly.*

$$Q_x = q_x \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad Q_y = q_y \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

where the original values of

$$q_x = 0.003; q_y = 0.006.$$

In order to study how process noise influence the performance of Kalman filter, we multiplied a coefficient K to the process noise

$$q'_x = K q_x; q'_y = K q_y.$$

Keep changing the coefficient K from 1 to 2000, a curve of error distance is plotted to indicate which K is the optimal value to this high speed movement.

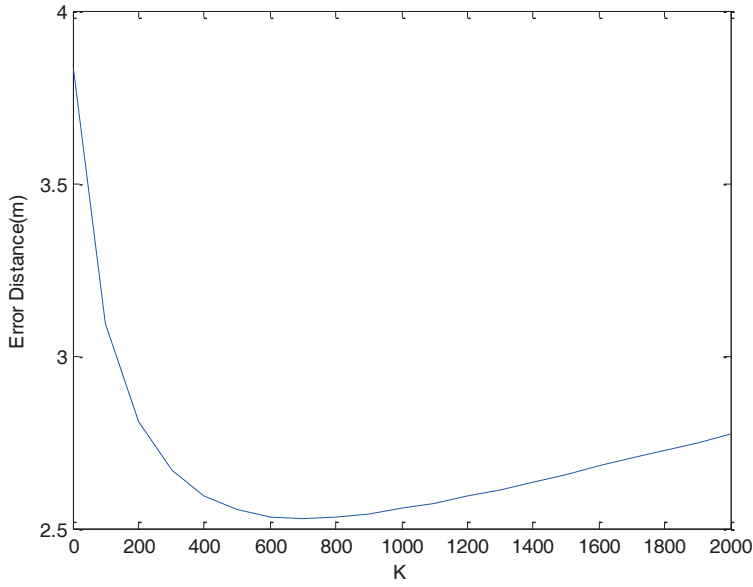


Figure 8.8. Error Distance with Coefficient K

From the figure 8.8, when K equals to 700, we can get the minimum RMS error which is 2.53 meters. The fixed process noise is

$$Q_x = 1.8 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad Q_y = 3.9 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The error distance analysis with coefficient K shows that we can improve the system performance through changing the values of process noise when the speed of target is different. Thus, design a dynamic Kalman filter against the influence of speed could be a good idea. And this could be a good topic to study in the future.

9 CONCLUSIONS AND FUTURE WORKS

In this thesis, a Wi-Fi based tracking system of cell phone in a certain indoor environment has been implemented. Any mobile device with Wi-Fi function on can be tracked by this system. Through the self-designed tracking system and the Kalman filter, the target tracking accuracy is about 2 meters when the target moves in one direction. And when the target moves along a square or in a natural pattern, the RMS error is within 5 meters. The accuracy of the system meets our expectation for this thesis

Snoopy Framework supports the data collection part with five configured Raspberry Pis. The Evil Twin Attack is used to build the rogue AP and increase the network traffic which promises a continuous tracking. In order to solve the packets dropping problem when the RSS below the noise floor -92 dBm, the Expectation Maximization (EM) Algorithm for truncated data has been applied. Finally, an adaptive Kalman filter is used to improve the tracking accuracy.

9.1 Contributions

In this thesis, an indoor tracking system which based on the indoor fingerprinting technique was designed and implemented.

Compared with traditional fingerprinting system, our system is able to track any mobile device in a target unknown condition. Although it was realized in a hacking way, none of private information and law was violated.

In order to improve the tracking accuracy, we applied two data processing methods and compared the tracking results with error analysis. One method utilized the average of RSS values with KNN Algorithm, and the other method utilized the instantaneous value of RSS with NN Algorithm.

We configured the Raspberry Pi on Linux operating system for collecting data from the mobile device. Evil Twin Attack was implemented by establishing a rogue AP, which can be used to increase the network traffic in wireless connection and realize the continuous tracking.

9.2 Drawbacks

It's regrettable that, limited by the equipment, our indoor tracking system cannot achieve high-accuracy localization in crowded environments. It only can be used in indoor environments with fewer people, because of the complexity of environment and the Wi-Fi signal strength is easily influenced by shadowing, speed and other factors. The sharply turning and variable motion also lead to large error distance.

9.3 Future Work

In this thesis, the indoor environment was a two dimensional model, which means we could not locate target in 3D, e.g., we cannot differentiate between a target on the stair case and another target right under it on the ground level.

When the target is out of the region of the experiment environment, the tracking system still estimate the target position based on the existing fingerprint database, which would lead to huge tracking error. A new algorithm to judge whether the target is out of region should be developed in the future.

Currently the sniffers use the arrival time of the broadcasted packet to realize the time axis synchronization. However, the adjustment of time difference is not ideal. If the departure time of the packet is available, the system performance can be improved further.

REFERENCE

- [1] National Research Council (U.S.). Committee on the Future of the Global Positioning System; National Academy of Public Administration (1995). The global positioning system: a shared national asset: recommendations for technical improvements and enhancements. National Academies Press. p. 16. ISBN 0-309-05283-1. Retrieved August 16, 2013., Chapter 1, p. 16
- [2] Wan Bejuri, Wan Mohd. Yaakob, Mohamad, Mohd. Murtadha and Sapri, Maimunah, "Ubiquitous Positioning: A Taxonomy for Location Determination on Mobile Navigation System." *Signal & Image Processing : An International Journal(SIPIJ)* Vol.2, No.1, March 2011
- [3] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu, "Survey of Wireless Indoor Positioning Techniques and Systems", *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS*, VOL. 37, NO. 6, NOVEMBER 2007
- [4] Y. Chen and H. Kobayashi, "Signal strength based indoor geolocation," in *Proceedings of the IEEE International Conference on Communications (ICC '02)*, vol. 1, pp. 436–439, New York, NY, USA, April–May 2002.
- [5] A. Dempster, B. Li, and I. Quader, "Errors in deterministic wireless fingerprinting systems for localisation," in *Proceedings of 3rd International Symposium on Wireless Pervasive Computing*, pp. 111–115, Santorini, Greece, May 2008.
- [6] Jungmin So, Joo-Yub Lee, Cheal-Hwan Yoon, Hyunjae Park, "An Improved Location Estimation Method for Wifi Fingerprint based Indoor Localization," in *International Journal of Software Engineering & Its Applications*; May 2013, Vol. 7 Issue 3, p77, May 2013
- [7] Wassi, G.I. ; Fac. des Sci. et de Genie, Laval Univ., Que. ; Despins, C. ; Grenier, D. ; Nerguizian, C. "Indoor location using received signal strength of IEEE 802.11b access point, Electrical and Computer Engineering", 2005. Canadian Conference on, on page(s): 1367 - 1370
- [8] Snoopy public blog, "Snoopy: A distributed tracking and profiling framework". Available in <http://www.sensepost.com/blog/7557.html>
- [9] The diagram of Snoopy architecture on sensepost. Available in http://research.sensepost.com/conferences/2012/distributed_tracking_and_profiling_framework
- [10] Introduction of Aircrack-ng. Available in <http://www.aircrack-ng.org/doku.php?id=Main>

- [11] Description of Tshark. Available on <http://www.wireshark.org/docs/man-pages/tshark.html>
- [12] Definition of standard output Available on: http://www.lininfo.org/standard_output.html
- [13] M. Cunche and R. Boreli, "I know who you will meet this evening! Linking wireless devices using Wi-Fi probe requests," *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–9, Jun. 2012.
- [14] Bonne, B.; Barzan, A.; Quax, P.; Lamotte, W. "Wi-FiPi: Involuntary tracking of visitors at mass events", *World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2013 IEEE 14th International Symposium and Workshops on a, On page(s): 1 - 6
- [15] Kevin Bauer; Harold Gonzales; Damon McCoy. "Mitigating Evil Twin Attacks in 802.11", Performance, Computing and Communications Conference, 2008. IPCCC 2008. IEEE International
- [16] Jose Maria Briones, Mario Alejandro Coronel Patricia and Chavez-Burbano, MSEE. "Case of study: Identity Theft in a University WLAN Evil twin and cloned authentication web interface" *Computer and Information Technology (WCCIT)*, 2013 World Congress on. June 2013.
- [17] Patibandala, B. ; Santhanam, H. ; Gaddam, S. ; Alla, V.K. ; Prakash, G.R. ; Chandracha, S.C.V. ; Boppana, S. ; Conrad, J.M. Sundaram, G.S.. "Bluetooth Communication using a Touchscreen Interface with the Raspberry Pi". Southeastcon, 2013 Proceedings of IEEE April 2013
- [18] Network Working Group of the IETF, January 2006, RFC 4252, The Secure Shell (SSH) Authentication Protocol
- [19] Introduction of NTP, available on <http://www.eecis.udel.edu/~mills/ntp/html/index.html#intro>
- [20] David L. Mills. "Computer Network Time Synchronization: The Network Time Protocol". Taylor & Francis. pp. 12–. ISBN 978-0-8493-5805-0.
- [21] Yungeun Kim; Hyojeong Shin; Hojung Cha. "Smartphone-based Wi-Fi pedestrian-tracking system tolerating the RSS variance problem", *Pervasive Computing and Communications (PerCom)*, 2012 IEEE International Conference on, On page(s): 11 - 19
- [22] Ullah, K. ; ICMC, Univ. of Sao Paulo, Sao Carlos, Brazil ; Custodio, I.V. ; Shah, N. ; dos Santos Moreira, E. "An Experimental Study on the Behavior of Received Signal Strength in Indoor Environment", on *Frontiers of Information Technology (FIT)*, 2013 11th International Conference

- [23] Nicolas Le Dortz; Florian Gain; Per Zetterberg, 'WIFI FINGERPRINT INDOOR POSITIONING SYSTEM USING PROBABILITY DISTRIBUTION COMPARISON', *Acoustics, Speech and Signal Processing (ICASSP)*, 2012 IEEE International Conference, On page(s): 2301 – 2304.
- [24] Siladitya Sen; "Qualitative analysis on Log-Normal Indoor Propagation Model for WLAN 1", on *Journal of Computer Science and Technology*; May 21, 2014; ISSN : 0976-8491
- [25] Explanation of MATLAB functions available on <http://se.mathworks.com/help/>
- [26] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, vol. 39, no. 1, pp. 1–38, 1977
- [27] Taimoor Abbas, Carl Gustafson, Fredrik Tufvesson, 'Pathloss Estimation Techniques for Incomplete Channel Measurement Data', COST IC1004 10th Management Committee and Scientific Meeting, 2014-05-26/2014-05-28, available on: <https://lup.lub.lu.se/search/publication/4442929>
- [28] Greg Welch and Gary Bishop, "An Introduction to the Kalman Filter" TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, NC 27599-3175, July 24, 2006
- [29] Hamilton, J. (1994), *Time Series Analysis*, Princeton University Press. Chapter 13, 'The Kalman Filter'.
- [30] Dimitrios Vlastaras, Taimoor Abbas, Daniel Leston and Fredrik Tufvesson "Vehicle Detection through Wireless Vehicular Communication", *Vehicle Detection through Wireless Vehicular Communication*
- [31] G. I. Wassi , D. Grenier , C. Despins and C. Nerguizian "Indoor location using received signal strength of IEEE 802.11b access point", *Proc. 18th Annu. Can. Conf. Elect. Comput. Eng.*, pp.1367 -1370



LUND
UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2015-423

<http://www.eit.lth.se>