A STUDY OF LOCALIZATION METHODS ON MOBILE PLATFORM
AND WIFI-BASED USER MOVEMENT DETECTION

by

Yongwei Ma

A thesis submitted to the Faculty of the University of Delaware in partial
fulfillment of the requirements for the degree of Master of Science in Electrical and
Computer Engineering

Fall 2014

© 2014 Yongwei Ma
All Rights Reserved
ACKNOWLEDGMENTS

I would like to express the deepest gratitude to my advisor Stephan Bohacek for his guidance and encouragement. Without his supervision and help, this thesis would not have been possible.

I would also like to thank my parents, girlfriend and other friends. I sincerely thank you all for your endless support and love.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>viii</td>
</tr>
</tbody>
</table>

Chapter

1 INTRODUCTION ................................................................... 1

1.1 Localization on Mobile Phones ...................................... 1
1.2 WiFi-Based Movement Detection ..................................... 3

2 FUNDAMENTAL POSITIONING TECHNIQUES FOR MOBILE DEVICES .......... 5

2.1 Global Positioning System ........................................... 5
2.2 WiFi Positioning ...................................................... 7
2.3 Cell-ID Positioning .................................................. 8

3 MORE OPTIMIZED LOCALIZATION SYSTEMS FOR MOBILE DEVICES ....... 11

3.1 Methods Focusing on Reducing Energy Consumption ................. 11

3.1.1 Some General Principles .......................................... 11
3.1.2 GAC ........................................................................ 12
3.1.3 Enloc ...................................................................... 13

3.2 Ambience Fingerprinting for Logical Locations ................... 15

3.2.1 SurroundSense ....................................................... 15

3.3 Methods Focusing on Indoor Localization ........................... 16

3.3.1 UnLoc ..................................................................... 17
3.3.2 Zee

4 WIFI-BASED MOVEMENT DETECTION

4.1 Observations of WiFi Signals
4.2 Methods

4.2.1 Majority Agreement
4.2.2 Top K

4.3 Results

5 CONCLUSION

BIBLIOGRAPHY
LIST OF TABLES

1.1 A list of sensors on smartphones. Power consumption is provided in [1] .................................................................................................................. 2

4.1 Accuracy of majority agreement at different locations. The thresholds are configured from 1dBm to 9dBm. ......................................................... 33

4.2 Accuracy of Top k for k = 5 using different thresholds ................. 34

4.3 Accuracy of Top K for k = 3, 4, 5, 6. The preset thresholds are 10.39, 12, 13.4 and 14.7 .......................................................... 34
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Satellite constellation</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>A simple abstraction of a cellular network</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>An example LMT with four uncertainty points, the edges represent the time departing from the uncertainty point</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>The structure of SurroundSense</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>The structure of UnLoc</td>
<td>17</td>
</tr>
<tr>
<td>3.4</td>
<td>The gyroscope’s bias $\theta$ can be achieved when two landmarks are encountered</td>
<td>18</td>
</tr>
<tr>
<td>3.5</td>
<td>An intuitive example of Zee. The user takes the path ABCD. The dark shadow is the possible location of this user. As this user walks, the dark shadow is getting smaller, and eventually this user is located</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>The fluctuation of three different signals over two hours. The strength level of signal 1 is strong while signal 2 and 3 are medium and weak respectively</td>
<td>23</td>
</tr>
<tr>
<td>4.2</td>
<td>The difference of signal strength between continual scans</td>
<td>24</td>
</tr>
<tr>
<td>4.3</td>
<td>Received signal strength at different locations</td>
<td>25</td>
</tr>
<tr>
<td>4.4</td>
<td>A 2-of-3 system</td>
<td>26</td>
</tr>
<tr>
<td>4.5</td>
<td>Estimating the accuracy of Majority agreement</td>
<td>29</td>
</tr>
<tr>
<td>4.6</td>
<td>The average value of the calculated distances(meter)</td>
<td>33</td>
</tr>
</tbody>
</table>
ABSTRACT

Modern smartphones can provide various features to enrich people’s everyday life. Localization is one feature that has gained more popularity recently. The fundamental techniques to support localization are GPS, WiFi and Cell-ID. Each technique is different from another in terms of accuracy, energy consumption and availability. Although GPS, WiFi and Cell-ID come in handy for localization on mobile phones, there are still some problems that limit their usage, such as low battery capacity and the high cost to locate a user indoors. Therefore, some new positioning systems are required to improve the performance of localization. In this paper, the main positioning techniques (GPS, WiFi, Cell-ID) along with their advantages and disadvantages are introduced. Several existing positioning systems are then presented.

Users’ mobility is also very useful for positioning systems. One common idea is to use a user’s mobility to control when to turn on/off the location sensing mechanism. For example, it is not necessary to track a user when he is standing still so the location sensing mechanism should be turned off in this case. In addition to the localization, I suggest two WiFi-based movement detection methods, Majority agreement and Top K, in this paper. The evaluation shows that both methods are able to provide approximately 90% accuracy to distinguish between absolute standing still and moving at walking speed around the UD area.
1.1 Localization on Mobile Phones

The modern smartphone now plays an important role in our daily life. Besides the basic service of voice communication, a smartphone is able to locate itself and provides location-based service, which is highly useful in many areas from location tracking, context-aware games, public safety, navigation and more [2].

The widely used fundamental techniques to support localization on mobile devices are Global Positioning System, WiFi positioning and Cell-ID positioning. GPS relies on satellites to calculate the location of a GPS receiver while WiFi and Cell-ID positioning are based on existing WiFi access points and the cellular network, respectively. Any single technique could outperform another under a certain environment since the level of accuracy, availability and energy consumption are different. For example, GPS is usually the most accurate for outdoor localization but it consumes the most energy. Cell-ID, on the other hand, works both indoors and outdoors with less energy consumption, but only provides an inaccurate location. WiFi is in the middle in terms of accuracy and energy consumption.

Although GPS, WiFi and Cell-ID are useful for location-aware application development on mobile phones, there are still some challenges and problems that need to be solved. The first challenge is that modern smartphones suffer from limited battery capacity. Continuously sampling GPS will deplete the battery within a few hours (data provided in [3, 4]). Consider that a user also uses their phone for calling or texting, wisely using the battery energy for positioning becomes even more important. Secondly, fundamental positioning techniques (GPS, WiFi and Cell-ID) are not
Table 1.1: A list of sensors on smartphones. Power consumption is provided in [1]

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Power Consumption(mW)</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerometer</td>
<td>96</td>
<td>Measures the acceleration applied on x, y, z axes of the device</td>
</tr>
<tr>
<td>Compass</td>
<td>N/A</td>
<td>Provides the angle between the orientation of the device and magnetic north</td>
</tr>
<tr>
<td>Gyroscope</td>
<td>N/A</td>
<td>Measures the rotation(angular velocity) of the device on x, y and z axes</td>
</tr>
<tr>
<td>Camera</td>
<td>1258</td>
<td>Takes pictures or records videos</td>
</tr>
<tr>
<td>Microphone</td>
<td>329</td>
<td>Captures sound signals</td>
</tr>
<tr>
<td>GPS</td>
<td>623(outdoor), 383(indoor)</td>
<td>Obtains the location on earth</td>
</tr>
<tr>
<td>WiFi</td>
<td>661</td>
<td>Captures IEEE 802.11 signals</td>
</tr>
</tbody>
</table>

class context-aware. Context can be the ambience information (light, sound, etc.) around a user or the current behavior of the user. Typically, contexts can be very different for two different physical locations. For example, it is very quiet and bright in a library while it can be very loud and dark in a movie theater. However, GPS, WiFi and Cell-ID do not benefit from users’ context. Lastly, fundamental positioning techniques cannot perform indoor localization (e.g., GPS) or cannot do it in an inexpensive way (e.g., WiFi). Locating a user who is indoors using WiFi requires a process to collect WiFi signals and the corresponding location (at where the WiFi is heard) in advance. This process will take a lot of time and money, which means that the scalability is poor.

Fortunately, modern smartphones are equipped with many sensors, which provide numerous opportunities to build new positioning systems to cope with the problems mentioned above. Table 1.1 describes the function of a list of sensors and their power consumption in mW if available (power consumption is obtained from [1], the data may be different for different phones). The most important features provided by sensors could be dead-reckoning (used in [5, 6]) and ambience fingerprinting [7]. Dead-reckoning is a method that estimates users’ location using an accelerometer and
compass based on Newton’s law of motion. Ambiance fingerprinting is a set of sensor data that indicates where a user is logically at. The details of several positioning systems and sensors’ usage are described in Chapter 3.

1.2 WiFi-Based Movement Detection

The mobility of a device carrier can be very valuable to a positioning system. Firstly, a user’s mobility can help to put the user in the right context. For example, a stationary user is more likely to be on the bed rather than in the middle of a street. Users’ mobility also can control when to turn on/off the location sensing mechanism. That is, location sensing only needs to be performed when the user is moving while it can be idle if the user is stationary. Previous works such as [8, 9] both use the built-in accelerometer on mobile as a binary detector to distinguish between moving and standing still. However, the mobile device usually has to be carried or handled properly to ensure accurate detection. Shaking the mobile in hand while sitting still on a sofa will be erroneously considered as moving.

A coarser-grained method is needed to assist in detecting users’ movement. Since most mobile smartphones have a WiFi sensor built-in, it is possible to actively scan signals broadcasted by nearby WiFi access points (AP) at the user end. The signals produced by a certain AP contain the unique identifier of that AP and also signal strength indicates the distance between the source AP and user’s device to some extent [10, 11]. Those valuable observations naturally imply that users’ mobility can be detected using WiFi. In this paper, I suggested two WiFi-based movement detection methods, Majority agreement and Top k, which reflect a mobile carrier’s mobility (moving or standing still) based on the change of the nearby WiFi environment. The basic idea is to periodically collect all the nearby APs and their corresponding signal strength. The result of the current scan can be compared with the previous one. For Majority agreement, the difference of signal strength for each AP that is obtained from both scans is calculated. This algorithm will return ”moving” if the majority of those calculated differences is greater than a threshold and return ”staying” otherwise.
Top K uses the distance (in signal space) of k most changed signals as an indicator of users’ mobility, and a threshold is also needed to distinguish movement and stillness. Both algorithms are tested at six different locations around the UD area. The results show that they have around 90% accuracy to distinguish between absolute stillness and movement (walking speed).

The rest of the paper is organized as follows: Chapter 2 presents current fundamental positioning techniques (GPS, WiFi and Cell-ID) for mobile devices. Chapter 3 shows an overview of several positioning systems. In Chapter 4, my WiFi-based movement detection methods and evaluation results are presented. Finally, Chapter 5 concludes this paper.
Chapter 2

FUNDAMENTAL POSITIONING TECHNIQUES FOR MOBILE DEVICES

Major positioning techniques for mobile devices are Global Positioning System, WiFi positioning and Cell-ID positioning. In this chapter, these three techniques and their advantages and disadvantages are introduced.

2.1 Global Positioning System

The Global Positioning System (GPS) is a navigation system designed by multiple U.S. government organizations to serve the purpose of three-dimensional position determination [12]. Although it was initially designed for military purpose, civilian users can use it for location tracking, scientific researching and other uses with some restrictions. According to [13], GPS has the following three segments:

1. space segment: consists of satellites in space
2. control segment: includes multiple ground infrastructures
3. user segment: consists of GPS receivers in various forms

Specifically, the space segment has more than 28 satellites inclined at 55 degrees to the equator orbiting at a height of 20180 Km on six different orbital planes [14]. The satellites are well distributed in space so that at least 4 of them are always visible from any point on earth. The control segment consists of many ground facilities such as the master control station, alternates and monitoring sites, which are located all over the world to accomplish complicated tasks include monitoring the satellites’ health status, controlling their position on orbit, and sending data to the constellation [15]. The core of the user segment is composed of GPS receivers. A receiver receives signals
broadcast by the satellites (4 or more) that are visible to it, then the exact three-
dimensional location (latitude, longitude and altitude) and the precise current time
can be calculated using triangulation. Lastly, although civilian used receivers usually
receive less accurate GPS signals than military ones, some augmentation systems and
ongoing projects are capable of boosting the accuracy of civilian GPS [16].

![Satellite constellation](image)

**Figure 2.1:** Satellite constellation

GPS is popular due to its high accuracy. Previous research shows that GPS can
offer a location accuracy of around 10m [4] while the data provided by Federal Aviation
Administration shows that the horizontal accuracy of the civilian GPS service (SPS)
can be better than 3m with a high quality receiver [16]. Thus, GPS is widely used and
preferred in order to build location-aware applications for mobile devices. However,
there are still some drawbacks that have to be considered by developers before they
decide to use GPS in the process of application development. The main drawbacks of
GPS are slow response time, high power consumption and limited availability in an
indoor environment [17]. Response time for standard GPS receivers is typically between one to two minutes [18]. The worst response time usually occurs when a receiver has no knowledge of the approximate current time and the last known position. Thus, it has to perform a cold start to get this information by searching for GPS satellite signals to download the almanac (time, position information about the satellites in the constellation) first [19]. After the download is completed, the receiver can cache it for a while so the response time of the later positioning process can be reduced. High power consumption is the second drawback of GPS. The experiments performed by [3, 4] show that around 10 hours of activation of GPS can deplete the battery in their devices. Thus, the developers and users need to make sure that continuous GPS sensing is only enabled when necessary. Lastly, GPS cannot perform well under thick coverage since the satellite signals are blocked. However, it usually functions properly outdoors, especially in clear view of the sky.

Assisted-GPS (AGPS)[18, 19, 20] can improve the performance of GPS through the wireless network. The wireless network infrastructure predicts the satellites that are visible to a GPS receiver and then gives the receiver a hint of which specific frequency to search so that the search space of the receiver is reduced. The reduced search space shortens the response time and increases the receiver’s sensitivity. However, AGPS only works when the network is available to the mobile device.

2.2 WiFi Positioning

WiFi positioning [10, 11, 21], as its name implies, can determine mobiles’ location using IEEE 802.11 wireless network access points (APs). This technique usually requires APs existence in the environment, a database (or ”radio map”) and a mobile device.

APs can be easily found in public facilities such as universities, libraries, international airports and coffee shops. Each AP is able to broadcast radio signals, which contain a unique identifier called Basic Service Set Identifiers (BSSID, same as the mac address) belongs to that AP. The received BSSID is involved in the positioning process
and other information in the signal is usually ignored. The database is used to store a huge number of APs and their corresponding physical location (not exact location), optionally as well as their signal strength. These data can be obtained by performing a costly offline training process called War driving. War driving collects APs in the way that distributing vehicles to travel through an area at normal road speed. Those vehicles are usually equipped with a radio antenna and GPS, which allow the public broadcast WiFi signals and GPS coordinates to be received on the fly. These data are then updated to the database. Lastly, the mobile has to be equipped with a WiFi sensor to ensure that it is able to scan the nearby APs information on demand.

The scan result of the mobile device finally can be matched with those data already stored in the database to estimate the mobile’s position based on various algorithms. The most popular positioning algorithm could be ”fingerprints” proposed in [11], which uses multiple APs’ BSSID and their signal strengths for positioning. This algorithm is able to locate a device within a few meters.

WiFi positioning only works when there are APs nearby. This should not be a problem in urban areas where the density of AP is relatively high. It is also a good alternative to GPS in an indoor environment. However, frequently updating the database is necessary since the location and availability of APs may change often. Lastly, in comparison with GPS, WiFi positioning is usually considered more energy-efficient but less accurate [22].

2.3 Cell-ID Positioning

Cell-ID positioning requires the existing cellular network to accomplish the task of location determination. According to [23, 24, 25], the abstraction of a cellular network and Cell-ID technique is described as follows. In the cellular network, a Base Transceiver Station (BTS) can communicate with a mobile station (MS), or simply, the user’s device like a mobile phone, under limited frequencies. Each BTS is able to cover various numbers of areas that are known as cells. There are usually many BTSs and cells in a big network to ensure that service capacity is sufficient for mobile phone users.
Figure 2.2: A simple abstraction of a cellular network

Each cell can be distinguished by a unique identifier Cell-ID while a group of cells can compose a Location Area (LA). MS updates its location information including Cell-ID to the core network based on predefined strategies as it moves between different cells or LAs. When a location service is required, the network starts to check if the Cell-ID of the target MS is available. If so, the current location of MS can be approximated using the geographical coordinates of the BTS (which serves the cell that the target MS is in); If not, the network needs to perform an operation called ”page” on all the cells within the LA to obtain MS’ Cell-ID first. As soon as the Cell-ID is available, this MS can be located in the same way.

[26] also summarized many other positioning methods that are based on the cellular network, such as: time of arrival (TOA), time difference of arrival, angle of arrival (AOA), amplitude of arrival (AMPOA) and others. In these methods, some information (time, angle, amplitude, etc.) of the signal broadcasted from MS is measured at a set of BTSs, then the network infrastructure can fuse those measurements correspondingly to calculate MS’s location.

Cell-ID positioning cannot provide an accurate location result since MS can be anywhere in the cell, and cell size varies from around 150m in radius to more
than several kilometers [23, 25]. However, it has many advantages such as low energy consumption, works both indoor and outdoor [27] and it is economic (no upgrades or modifications needed on handsets) [23, 26].

As a short summary, GPS, WiFi and Cell-ID have different levels of energy consumption, accuracy and availability. Any one of them is not able to completely replace another.
3.1 Methods Focusing on Reducing Energy Consumption

Mobile devices suffer from the problem of limited battery life. Continuously activating GPS will lead to the battery completely drained within few hours[3, 4]. Therefore, sensing location using an energy-efficiency method is very important for localization on mobile devices.

3.1.1 Some General Principles

Improving energy efficiency of location sensing on smartphones [28] provides four general principles to build a location sensing system that is less power intensive. The four principles are substitution, suppression, piggybacking and adaptation, respectively. In detail, substitution means to substitute the high power component (i.e., GPS) with a low power one (i.e., WiFi) if the low power component is available and it can provide accurate location information. It also has to switch back to the high power one when a low power location mechanism does not work well (such as no WiFi access points around). Suppression means to deactivate the location sensing mechanism when it is possible. For example, GPS should be turned off if the mobile is left on an office desk since it is not necessary to waste energy to track a stationary object. Piggybacking is an idea that synchronizes the location sensing between multiple running processes on the mobile device to reduce the power consumption. Lastly, the adaptation principle indicates that the frequency of sensing should be changed dynamically based on the current battery level: sense more when the battery level is high, sense less otherwise.
3.1.2 GAC

Periodically activating GPS is a common approach to track users’ location. However, there is no a single optimal duty-cycle period to use since a mobile device carrier could either move very far or stay still between two location updates. In addition, a long period results in obtaining less location information of the user while a short period brings high power consumption.

GAC [5] is a positioning method that focuses on reducing the activation of GPS to save energy. The basic idea is using low energy sensors, an accelerometer and a compass, to estimate users velocity and moving direction so it is possible to estimate users’ displacement based on Newton’s law of motion (initial location is given by GPS). This is also known as dead-reckoning. Specifically, GAC requires that the mobile device is fixed on the tracked object. The accelerometer and compass then can be periodically sampled using a sampling interval $T$. The distance moved by the tracked object is equal to:

$$l(n) = v(n)T + \frac{1}{2}a(n)T^2$$  \hspace{1cm} (3.1)

also, the new estimated velocity is calculated by:

$$v(n + 1) = v(n) + a(n)T$$  \hspace{1cm} (3.2)

Where $a(n)$ is the accelerometer reading that indicates the acceleration along the movement direction at time instance $n$. The movement direction can be obtained from the compass. The new location of the tracked object that on the surface of the earth can be calculated by Vincenty’s formula.

The accelerometer and compass are noisy sensors, and the error will be accumulated as time goes on. Therefore, GAC will activate GPS every $T_{gps}$ seconds to replace the estimated location with the GPS location. GAC also uses an average of four sensor readings (accelerometer and compass) to further reduce the impact of noise.
3.1.3 Enloc

Enloc [22] provides an offline dynamic programming (DP) idea that focusses on minimizing localization error (ALE) within a given energy budget. The DP takes a user’s trace (all the location readings) as input then returns an optimal schedule that reveals how to turn on the right sensors (GPS, WiFi and so on) at the right time such that ALE is minimum. ALE is shown in the equation below. T is different time points and $L_{actual}$ is the ”ground truth” return from GPS.

$$ALE = \sum_{i=1}^{T} \frac{L_{reported}(t_i) - L_{actual}(t_i)}{T}$$ (3.3)

Besides the offline DP, Enloc provides a real-time heuristics system to predict users’ location. The key of the heuristic is that it benefits from human mobility patterns, and it is able to deal with the deviations when a user is not on his habitual paths.

Intuitively, a person may follow a very regular space-time movement. This can be a common case for those people who work in an office and students who have a regular course schedule. Enloc collects users’ mobility traces then models them as a logical mobility tree (LMT). The vertices are called uncertainty points, that is, the points that lead to many different locations on a user’s actual mobility paths. The edges of LMT represent actual paths between two uncertainty points (the starting time that depart from an uncertainty point, the average velocity on the actual path and other information are also stored in edges). Figure 3.1 shows an example of LMT. The core idea is that Enloc focuses on activating the location sensing mechanism right after the uncertainty points so that the user is placed on the correct path deriving from the uncertainty point. The user’s location can then be predicted based on the average velocity history on this path. A new location update is required when next uncertainty point is encountered.

Deviation occurs when a user does not follow his regular routine. In this case, Enloc changes to deviation mode to cope with it. Specifically, deviation mode is on
Figure 3.1: An example LMT with four uncertainty points, the edges represent the time departing from the uncertainty point when a GPS reading shows that the user is in an unexpected place (not on LMT for example). GPS activations are then equality distributed for the rest of the day according to battery level (more battery more reading, vise versa). The user’s locations between two GPS readings are predicted based on the mobility of large populations (the information shows that how the majority move or turn at intersections) rather than individual mobility history. For each intersection, the mobility of large populations can be represented by a matrix. Each element $X_{ij}$ in the matrix indicates the probability that the user is entering the intersection from street i and leaving from street j. However, collecting the mobility of the majority can be very costly in terms of time and money.

In short summary, both Enloc and GAC use GPS less frequently to achieve the goal of energy saving. GAC activates GPS periodically and mainly use an accelerometer and compass to estimate users’ location while Enloc focuses on prediction. It is worth mentioning that less energy consumption usually results in low accuracy (due to fewer location readings to estimate location). One should find the balance point for a specific
application.

### 3.2 Ambience Fingerprinting for Logical Locations

According to AAMPL [29] and SurroundSense [7], there are two types of location that we care about, namely, physical location and logical location. Physical location is usually represented by latitude/longitude on physical coordinates (e.g. earth). On the other hand, logical location is a logical label that indicates where the tracked user is at. For example, "Starbucks", "Mall" and "Gym" are different logical locations. Due to error existing in the current localization system, it is hard to distinguish a user’s location from two physical locations that are separated by a dividing wall. Knowing logical location can be very useful in this situation.

#### 3.2.1 SurroundSense

![The structure of SurroundSense](image)

**Figure 3.2:** The structure of SurroundSense

SurroundSense [7] is a localization system that tries to put a user at the correct logical location that exists within a range of a given physical location. The basic
observations are: for logical locations near a given physical location: 1) the environment theme(floor color, background sound, etc.) can be different from each other, 2) people may have a certain mobility pattern at different logical locations, 3) environment theme and user mobility pattern can be unique enough to distinguish them. The environment theme and users’ mobility pattern are usually represented by multiple sensors’ data, which are known as ambience fingerprints. As shown in Figure 3.2, SurroundSense uses the built-in camera, microphone, WiFi sensor and accelerometer to collect the ambience fingerprints \(AF_c\) (formed by light/color, sound, WiFi APs and users’ mobility) on the fly. It then queries the geographic database and fingerprint database using the current physical location \(L_c\) (obtained by Cell-ID for example) to get a list of possible logical locations (e.g. “Starbucks”, ”McDonalds” within maybe 150m of \(L_c\)) and their corresponding ambience fingerprints \(AF_d\). Lastly, by comparing \(AF_c\) and \(AF_d\), some impossible logical locations are filtered out and a list of possible logical locations are returned to report the user’s logical location (the list is sorted, the most possible logical location is on the top).

However, the drawback is that it has to build the geographic database and fingerprint database in advance by war-driving or other methods. A logical location can be very large (e.g. a mall), therefore knowing the logical location of a user may not provide sufficient information in this case.

### 3.3 Methods Focusing on Indoor Localization

The main concerns and challenges of indoor positioning (mentioned in [6, 30]) are: 1) GPS is not available in the indoor environment; 2) hardware or data may need to be pre-set manually (e.g. war driving), which is very costly in terms of time and money; 3) some sensors may not work well indoors due to heavy magnetic interference; 4) the accuracy requirement usually is high. A good indoor positioning system usually needs to solve one or more challenges to be effective and practical.
3.3.1 UnLoc

UnLoc [6] combines the ideas of dead-reckoning and ambience fingerprinting, as well as a new concept called landmark to build a less labor-intensive/cost indoor location system. A landmark is a small geographical area with an identifiable ambience fingerprint. In other words, a landmark is a logical location with rough longitude/latitude coordinates. For example, a "stairway" around (1,1) is a landmark, which is different from the other "stairway" around (2,2). There can be many potential landmarks in a building. This could include an "escalator" or even a "corner". The landmarks may or may not be known in advance.

The key idea behind UnLoc is a recursive process, which dynamically recognizes and discovers landmarks in an indoor environment to improve the accuracy of dead-reckoning. The more accurate result of dead-reckoning will in turn help to recognize more landmarks (the location of landmarks also becomes more accurate). As this recursive process goes on, Unloc is able to improve itself automatically for the purpose of unsupervised indoor localization. However, an initial reference location (the entrance of the building, for example) is needed to start the recursive process. Figure 3.3 shows the structure of UnLoc.

Figure 3.3: The structure of UnLoc
Figure 3.4: The gyroscope’s bias $\theta$ can be achieved when two landmarks are encountered.

The recursive process is explained in detail as below:

1) Using ambience fingerprints and Dead-reckoning to find landmarks

In order to find a landmark, UnLoc tries to identify the logical locations using ambience fingerprinting and check if they are physically within a small area. If so, a landmark is found. Specifically, all mobile devices that have UnLoc installed keep collecting ambience fingerprints (using multiple sensors) as well as the time when the data is collected. A K-means clustering algorithm will be used to classify those ambience fingerprints into different clusters based on the similarity between them. Each cluster should have a low similarity (correlation) with other clusters, and an individual cluster will be regarded as a candidate for a landmark. The result of dead-reckoning will test if candidates of landmark are within a small physical area. If so, a new landmark is discovered. The location of the landmark is estimated by applying a simple centroid on the result of dead-reckoning (this makes sense since the error of dead-reckoning is independent).

It is important to know that Dead-reckoning in UnLoc is different from the one in GAC in two main ways. First, in order to calculate the users’ displacement, UnLoc uses an accelerometer to get the user’s step count instead of getting velocity (Equation 3.1 and 3.2). The displacement is simply obtained by multiplying step count and step size of the user. The accelerometer reading can imply the step count since...
walks are regular and repetitive by nature (see [6, 30] for details). The step size for an individual user can be inferred backward using the distance and step count between two known landmarks. Second, UnLoc achieves users’ relative movement direction using the built-in gyroscope in the carried mobile device rather than using the compass since the compass is considered vulnerable to the heavy magnetic interference in the indoor environment. However, relative direction cannot indicate users’ location. At least two known landmarks have to be encountered to estimate the user’s location. Figure 3.4 explains this problem. The ”Dead-Reckoned path” in the figure is the estimated path taken by a user. As we can see, a gyroscope is only able to tell us the relative movement direction of the user (e.g., walks straight and turns right). There can be an angle bias $\theta$ between the estimated path and the real path (”ground truth path” in the figure). As long as two landmarks are encountered, the bias can be known by rotating the estimated path. It is then possible to estimate the following path using a gyroscope.

2) New landmarks help to get better dead-reckoning

The error of dead-reckoning accumulates until a new landmark is encountered, at where the user is put at the location of a landmark to restart dead-reckoning. Therefore, the accuracy of dead-reckoning is improved.

The process 1) is able to improve the process 2), which in turn improves process 1) itself. This mechanism allows UnLoc to provide better accuracy over time automatically. However, an initial reference location needs to be learned once to start dead-reckoning. There may be a lack of landmarks in some buildings, and UnLoc cannot work well in this case.

3.3.2 Zee

Zee [30] is another indoor localization system that aims at eliminating the labor-intensive pre-set process of WiFi positioning. All of the mobiles with Zee installed are supposed to enrich the WiFi database by crowdsourcing. As time goes on, Zee turns into a pure WiFi positioning system that locates a user just by performing a WiFi scan(see Chapter 2). Intuitively, Zee uses the floor plan of the building as the
Figure 3.5: An intuitive example of Zee. The user takes the path ABCD. The dark shadow is the possible location of this user. As this user walks, the dark shadow is getting smaller, and eventually this user is located constraint and combines sensors’ data to filter out those impossible locations as a user walks, As a result, eventually only the true location is left. This estimated true location is stored for later WiFi positioning. It is also possible to backtrack the path history of the user from the estimated true location. Figure 3.5 shows an example scenario of how Zee locates a user as he walks from A to D.

The core components of Zee are the motion estimator and augmented particle filter. In a similar way to UnLoc, the motion estimator here is composed by accelerometer, compass and gyroscope to detect the steps taken by the device carrier as well as the rough heading offset (the angle between phone and user’s movement direction, knowing the heading offset means the user’s movement direction can be inferred). The step count and rough heading offset are passed to the second component, the augmented particle filter, which will concurrently estimate the user’s step length and refine the heading offset based on the floor plan (e.g. by looking at the floor plan, the incorrect step length and heading offset that indicate the user "go through a wall" are discarded.
as the users walks). Specifically, the estimated locations converge as each step taken by the user using the equations:

\[ x_i^k = x_i^{k-1} + (s_i + \delta_i)\cos(\alpha_i + \theta + \beta_i) \]  

(3.4)

\[ y_i^k = y_i^{k-1} + (s_i + \delta_i)\sin(\alpha_i + \theta + \beta_i) \]  

(3.5)

As mentioned in Zee [30], \( x_i, y_i \) are the location, \( s_i \) is the step length, \( \alpha_i \) is the heading offset. The equations indicate how the \( i^{th} \) particle is updated after \( k^{th} \) step (\( \theta \) is the compass reading, \( \delta \) and \( \beta \) are used to accommodate variation). The line that connects \( (x_i^{k-1}, y_i^{k-1}) \) and \( (x_i^k, y_i^k) \) is tested at each step to see if it is into a wall. If so, the particle is eliminated. It is very important to randomly choose a new particle from \( k-1 \) step and update it to replace the eliminated particle. Eventually, the estimated location and the corresponding WiFi APs and signal strength will be stored in a WiFi database for later lookup. Zee does not require an initial location and the phone can be placed anywhere on the user. But Zee does assume that the phone placement is unchanged for a single walk.
Chapter 4

WIFI-BASED MOVEMENT DETECTION

Detecting mobile carrier’s mobility with a built-in acceleration sensor to avoid unnecessary location updates has been suggested by many previous researchers. However, the acceleration sensor is sometimes too fine-grained as a good movement detector. False detection may occur if the device is not handled properly [8]. For example, an accelerometer will report "move" when a person is doing exercise on a running machine with his mobile phone in hand, but the expected result is "stay”. It motivates a more coarser-grained movement detection method. Since WiFi has been widely used to locate a mobile device, naturally, it should be capable of detecting movement. In this chapter, two WiFi-based movement detection methods are presented.

4.1 Observations of WiFi Signals

As a reminder, the unit of WiFi signal strength is dBm. Any signal has less than -90dBm can be considered to be weak, around -40dBm indicates a strong signal. A WiFi signature represents an AP’s MAC address and its strength. Two observations, stability and variability, are described as follows:

A: Stability

To demonstrate the stability of WiFi signal strength, I collected a group of WiFi signatures at a fixed location with an Android phone. The phone periodically scanned all the nearby APs for two hours. The time interval between two scans was set to 6 seconds. Figure 4.1 shows the fluctuation of three different signals that were observed during those two hours. Signal 1, 2 and 3 are chosen because they clearly have different strength levels from strong, to medium, to weak, respectively. It is not hard to tell
**Figure 4.1:** The fluctuation of three different signals over two hours. The strength level of signal 1 is strong while signal 2 and 3 are medium and weak respectively.
that all the signals only fluctuate around their own strength level, and any single signal
does not have a more noticeable change (in terms of strength) than another.

Figure 4.2: The difference of signal strength between continual scans

Figure 4.2 provides a detailed view of how signal 1 changes between two con-
tinual scans. Each measured strength is compared with the previous one to calculate
the difference between them. For this specific signal, the absolute value of \( \Delta \)strength
is constantly less than 6 dBm, and it is seldom larger than 3 dBm. In fact, signal 2
and signal 3 also show similar results from the data that I collected. Thus, all the
observations have a tendency to indicate that signal strengths are relatively stable at
a fixed location.

B: Variability

Previous research [11] indicated that received signal strength is related to the
distance between the source and receiver, specifically, a stronger signal can be received
if the user stays closer to the source and it becomes weaker as the user walks far away.
To further verify this observation, I did an experiment out of my apartment to look
at how the received signal strength varied as I walked along a straight line that far away from my router. The walk began right out of my apartment and terminated 60 meters away and the signal strength was collected at 11 evenly distributed points (the distance between two adjacent points is 6 meters) on the path. At each point, the mean of more than one hundred samples was used to estimate the received signal strength at that specific location. As can be seen in Figure 4.3, the strength of my router rapidly decreases by 10 dBm for the first 10 meters, then slowly declines and eventually stabilizes at around -90 dBm. It is also worthwhile to look at how other signals change without knowing their sources’ location. For the weak signal plotted in Figure 4.3, received strength does not show a strong relation with distance, and it always stays around -90 dBm. On the other hand, the strength of the strong signal is very sensitive to the location change. It even can drop by more than 20 dBm within 10 meters. All the observations indicate that received WiFi signal strength varies as the location of the receiver changes, and the change of strength can be very significant.

Figure 4.3: Received signal strength at different locations
4.2 Methods

In this section, two coarse-grained WiFi-based moving detection methods, Majority agreement and Top K, are presented. The basic idea is that a mobile device periodically collects nearby WiFi signatures, and the collected information is used to reflect the change of WiFi environment. A huge change represents a movement that has occurred while a small change indicates that the device carrier is stationary.

4.2.1 Majority Agreement

Majority voting [31] is a widely used idea in the field of hardware fault tolerance to improve the reliability of a system. An M-of-N system is a system that has N modules in total and requires at least M of them working properly to ensure that the whole system generates a correct result. Figure 4.4 below shows the simplest system in this type, which is known as a 2-of-3 system. Three identical modules perform the same operation independently and send their output to a voter to evaluate the real result. A typical voter can do a bit-by-bit comparison of those outputs to check if they are matched. If a majority (2 or 3, in this case) of the 3 outputs are identical, the voter generates the majority as the final result.

![Figure 4.4: A 2-of-3 system](image)

The first WiFi-based moving detection algorithm, Majority agreement, is implemented by adopting the majority voting idea as well as taking some observations mentioned before into account. The key observations are the received signal strength
that usually fluctuates within a certain range by nature, and the strength of a signal can vary dramatically as the distance between the source and receiver changes.

Majority agreement takes two lists of WiFi signatures collected by two continual scans as inputs, namely the recentAPs list and the previousAPs list. When a pair of AP appears in both of the lists, subtraction is performed on their signal strength to check if the result is within a threshold. If so, it is highly possible that the calculated difference is caused by the inherent fluctuation (i.e., fluctuation of signal strength) rather than the location change. Thus, it implies that the device carrier does not move, so a "stayCounter" is increased by 1. If not, a "moveCounter" gets increased to indicate that a movement is detected. The other pairs of APs are also used in the same way. Finally, it is the time to compare the value of both counters and the majority provides the final estimation.

Algorithm 1 Majority Agreement

Initialization:
1: threshold = t
2: moveCounter = 0
3: stayCounter = 0
4: previousAPs = the list of WiFi signatures discovered by first scan
5: recentAPs = the list of WiFi signatures discovered by second scan

Procedure:
6: for each WiFi signature pap in previousAPs do
7:     for each WiFi signature ap in recentAPs do
8:         if pap.mac == ap.mac then
9:             if |pap.strength - ap.strength| < t then
10:                stayCounter++
11:             else
12:                moveCounter++
13:         end if
14:     end for
15: end for
17: if moveCounter > stayCounter then
18:    return move
19: else
20:    return stay
21: end if
It is worthwhile to explain the algorithm using an example. Suppose that the threshold is set to 4 dBm and the first scan discovered four different APs, A, B, C and D. Their corresponding strengths can be represented by (-40, -20, -60, -90). The second scan gets three APs, A, B and C with strengths of (-43, -23, -10). The APs that appear in both scans are A, B and C. Thus, the difference of strength for those three APs can be written as (3, 3, 50), which means that A and B vote for "stay" and C votes for "move". Lastly, a "stay" will be returned since 2 of 3 APs vote for "stay".

In the field of hardware fault tolerance, reliability is used to evaluate if a system is reliable during a time interval. Specifically, according to [31], reliability of a system represents the probability that the system has been up continuously during a time interval between 0 and t, and the reliability of an M-of-N system is given by

\[
R_{M\rightarrow N}(t) = \sum_{i=M}^{N} \binom{N}{i} R^i(t)[1 - R(t)]^{N-i}
\]  

(4.1)

R(t) is the reliability of a single module, \( R_{M\rightarrow N}(t) \) is the probability that M or more modules work properly during \([0, t]\). Using the same idea and equation provided in [31], we can replace R(t) with P from the equation and then use it to approximate the accuracy of Majority agreement. Accuracy here means the probability to return a correct result. For simplicity, I assume that each pair of common AP has the same probability P to provide a proper guessing of users' state ("move" or "stay"). The probability of successful detection of the algorithm can be written as:

\[
P_{alg} = \sum_{i=M}^{N} \binom{N}{i} P^i(1 - P)^{N-i}
\]  

(4.2)

In Figure 4.5, accuracy is plotted for three different cases to show how the number of common APs discovered in both scans affects this algorithm. For a high P, we can expect a high accuracy, and a large number of common APs can provide a positive effect to the algorithm. As P decreases, the accuracy is reduced. The more common APs are going to become a disadvantage when P is less than 0.5.
The key of this algorithm is to find a threshold that works in most cases to increase the accuracy of an individual pair of common AP. A pre-study of the WiFi environment can be performed to find the threshold.

![Figure 4.5: Estimating the accuracy of Majority agreement](image)

**Figure 4.5:** Estimating the accuracy of Majority agreement

### 4.2.2 Top K

As I have mentioned in the previous chapter, WiFi positioning usually requires an offline training process, War driving, to collect WiFi signatures as well as the corresponding position where the WiFi scan is performed at. This information is stored in a database for later lookup. In order to determine the current location of a mobile device, the result of a WiFi scan that was performed at the user end needs to be compared with those data stored in the database. Multiple nearest neighbors [11], also known as k nearest neighbors in signal space [10] is a method that describes how the comparison and calculation can be done. The method mainly includes the following steps: 1) find sets of recorded APs (RAP), which are similar to the ones observed at
the user end (OB); 2) chose k sets (KS) from RAP that have the smallest distance (in signal space) to OB; and 3) average the latitude-longitude coordinates of KS to approximate the real location of the user. This method is able to locate a user within few meters.

The success of the method mentioned above implies the distance in signal space can reflect the physical location to some extent. Thus, it is very likely that the distance (in signal space) between two groups of observed WiFi signatures is able to reflect the physical distance between the two locations (where those scans are performed). My second movement detection method, Top k, is developed based on this assumption. In short, Top k computes the distance $d$ (in signal space) of the top $k$ most changed (the change of signal strength) APs that are obtained from the two scans and compares if $d$ is larger than a preset threshold $th$. A larger $d$ signifies that a more likely movement has occurred. Thus, the algorithm will return ”move” when $d > th$, otherwise, it will return ”stay”. An example of using Top k ($k=3$) is described below.

For the first scan, let (-40, -50, -60, -70) denote the received signal strength of four APs, A, B, C and D. (-50, -60, -70, -71) represents their corresponding strength obtained by the second scan. A, B and C will be used to calculate the distance while D is ignored since D is not the top 3 changed AP. The well known Euclidean distance equation is applied to calculate the distance (in signal space):

$$d = \sqrt{(S_{A1} - S_{A2})^2 + (S_{B1} - S_{B2})^2 + (S_{C1} - S_{C2})^2}$$

(4.3)

From the equation above, we can get $d = 17.3$. The preset threshold is also calculated in the same way, $th_3 = \sqrt{6^2 + 6^2 + 6^2} = 10.39$. I choose 6 here because the maximum strength change of the received signal is usually within 6 dBm at a fixed location (see Figure 4.2). Thus, we can expect that $d$ should be always below 10.39 if no significant movement takes place. In this example, $d$ is larger than 10.39, so it is likely that the person has moved.

It is possible to find a long list of APs shared in both scans and each AP has different ”sensitivity” to the location change (see the two unknown signals in Figure
4.3. Calculating distance $d$ using those APs with high sensitivity should be able to reflect a user’s movement. This is the reason Top k only picks those APs that changed most and ignores weak signals.

**Algorithm 2 Top K**

**Initialization:**

1: $d = 0$
2: $\text{threshold} = t$
3: $\text{tempList} = \emptyset$
4: $\text{previousAPs} =$ the list of WiFi signatures discovered by first scan
5: $\text{recentAPs} =$ the list of WiFi signatures discovered by second scan

**Procedure:**

6: for each WiFi signature $pap$ in previousAPs do
7: \hspace{1em} for each WiFi signature $ap$ in recentAPs do
8: \hspace{2em} if ($pap$.mac == $ap$.mac) and ($pap$.strength and $pa$.strength are strong) then
9: \hspace{3em} $\text{tempList}.\text{add}(|pap$.strength - $ap$.strength|)
10: \hspace{2em} end if
11: \hspace{1em} end for
12: end for
13: sort $\text{tempList}$ in descending order
14: $d = \sqrt{\text{tempList}[1]^2 + \text{tempList}[2]^2 \ldots + \text{tempList}[k]^2}$
15: if $d > t$ then
16: \hspace{1em} return move
17: else
18: \hspace{1em} return stay
19: end if

4.3 Results

I implemented Majority agreement and Top k on a Samsung Galaxy S smartphone to evaluate their accuracy. The phone is equipped with a WiFi sensor so that nearby WiFi signals can be periodically collected for later calculations, then the results were written to a file. One full WiFi scan usually takes around 3 seconds to be completed on the test device and the scan interval was set to 6 seconds.

Two phases of experiments have been done for each method at six different locations. Three of the test locations are indoors in Evans Hall, Morris Library and Christina Commons. The other three locations are outdoors around my apartment,
the main campus and Main Street. The first phase is to test if the methods can successfully detect movement while the second phase is used to test the accuracy for detecting stillness. Specifically, in the first test phase, I carried the device and kept moving around the testing locations at walking speed. The starting point of the walk was picked randomly, and the walk patterns depended on the size of the test location. For the in-building environments, I walked along the hallway in the largest available circle. The round trip distance is typically 40m - 100m. For outdoor environments, I either walked straight in a single direction or took a larger circle trip. After the first test phase is completed, the program is restarted, and the device is placed at a random point around the path that was just walked on to collect the result for the second phase.

Table 4.1 shows the accuracy of Majority agreement at those six locations under different thresholds (from 1dBm to 9dBm). The rows having a ”Move” in it are the results of the first phase while the rows with a ”stay” are the results of the second phase. Accuracy in this table is represented by the percentages of correct detection, and all the percentages are calculated over 120 returned results. Good results are those that show a high value in the row labeled with ”move” and ”stay”, which means high accuracy for detecting movement and detecting stillness is achieved at the same time. As can be seen in Table 4.1, the best overall accuracy occurs when the threshold is set to 3dBm: it is over 90% at all the locations except in Christina Commons where the lowest accuracy occurs (80% for detecting movement). This algorithm tends to return ”stay” as the threshold increases, and it is biased to ”move” as the threshold decreases.

The same testing strategy was applied to test Top k. Instead of recording the number of ”stay” and ”move” returned by this algorithm, the calculated distances (in signal space) were written to a file for further analysis. At least 100 results were collected in both test phases at different locations, separately. Figure 4.6 shows the average value of the calculated distances. It is not hard to see that there is a significant difference between ”move” and ”stay” at all locations. The average of calculated distances is constantly less than 10 when the device is put at fixed positions and it is
Table 4.1: Accuracy of majority agreement at different locations. The thresholds are configured from 1dBm to 9dBm.

<table>
<thead>
<tr>
<th>locations</th>
<th>state</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apartment</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>98%</td>
<td>92%</td>
<td>79%</td>
<td>62%</td>
<td>47%</td>
<td>35%</td>
<td>27%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>17%</td>
<td>72%</td>
<td>91%</td>
<td>95%</td>
<td>96%</td>
<td>97%</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
</tr>
<tr>
<td>Evans</td>
<td>Move</td>
<td>100%</td>
<td>99%</td>
<td>98%</td>
<td>92%</td>
<td>90%</td>
<td>85%</td>
<td>81%</td>
<td>76%</td>
<td>67%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>28%</td>
<td>78%</td>
<td>93%</td>
<td>98%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>Library</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>98%</td>
<td>94%</td>
<td>87%</td>
<td>77%</td>
<td>68%</td>
<td>57%</td>
<td>45%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>76%</td>
<td>98%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>C. Commons</td>
<td>Move</td>
<td>100%</td>
<td>89%</td>
<td>80%</td>
<td>64%</td>
<td>44%</td>
<td>27%</td>
<td>17%</td>
<td>11%</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>73%</td>
<td>97%</td>
<td>98%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>Campus</td>
<td>Move</td>
<td>100%</td>
<td>98%</td>
<td>93%</td>
<td>77%</td>
<td>67%</td>
<td>55%</td>
<td>48%</td>
<td>45%</td>
<td>38%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>32%</td>
<td>80%</td>
<td>91%</td>
<td>94%</td>
<td>97%</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
</tr>
<tr>
<td>M. Street</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>93%</td>
<td>97%</td>
<td>97%</td>
<td>98%</td>
<td>98%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>10%</td>
<td>66%</td>
<td>93%</td>
<td>97%</td>
<td>97%</td>
<td>98%</td>
<td>98%</td>
<td>99%</td>
<td>99%</td>
</tr>
</tbody>
</table>

Figure 4.6: The average value of the calculated distances (meter)
Table 4.2: Accuracy of Top k for k =5 using different thresholds

<table>
<thead>
<tr>
<th>locations</th>
<th>state</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apartment</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>98%</td>
<td>98%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>69%</td>
<td>77%</td>
<td>88%</td>
<td>94%</td>
<td>96%</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
</tr>
<tr>
<td>Evans</td>
<td>Move</td>
<td>97%</td>
<td>97%</td>
<td>96%</td>
<td>96%</td>
<td>96%</td>
<td>96%</td>
<td>96%</td>
<td>95%</td>
<td>93%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>86%</td>
<td>88%</td>
<td>89%</td>
<td>90%</td>
<td>91%</td>
<td>93%</td>
<td>93%</td>
<td>93%</td>
<td>93%</td>
</tr>
<tr>
<td>Library</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>88%</td>
<td>90%</td>
<td>94%</td>
<td>96%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>C. Commons</td>
<td>Move</td>
<td>95%</td>
<td>94%</td>
<td>93%</td>
<td>92%</td>
<td>89%</td>
<td>87%</td>
<td>79%</td>
<td>72%</td>
<td>65%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>85%</td>
<td>88%</td>
<td>88%</td>
<td>88%</td>
<td>90%</td>
<td>91%</td>
<td>91%</td>
<td>91%</td>
<td>91%</td>
</tr>
<tr>
<td>Campus</td>
<td>Move</td>
<td>97%</td>
<td>96%</td>
<td>94%</td>
<td>88%</td>
<td>79%</td>
<td>74%</td>
<td>63%</td>
<td>61%</td>
<td>55%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>85%</td>
<td>87%</td>
<td>90%</td>
<td>93%</td>
<td>95%</td>
<td>96%</td>
<td>97%</td>
<td>97%</td>
<td>98%</td>
</tr>
<tr>
<td>M. Street</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>99%</td>
<td>99%</td>
<td>98%</td>
<td>96%</td>
<td>95%</td>
<td>94%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>54%</td>
<td>67%</td>
<td>79%</td>
<td>86%</td>
<td>92%</td>
<td>94%</td>
<td>99%</td>
<td>99%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 4.3: Accuracy of Top K for k = 3, 4, 5, 6. The preset thresholds are 10.39, 12, 13.4 and 14.7

<table>
<thead>
<tr>
<th>locations</th>
<th>state</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apartment</td>
<td>Move</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>81%</td>
<td>92%</td>
<td>95%</td>
<td>97%</td>
</tr>
<tr>
<td>Evans</td>
<td>Move</td>
<td>98%</td>
<td>98%</td>
<td>96%</td>
<td>96%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>88%</td>
<td>90%</td>
<td>91%</td>
<td>92%</td>
</tr>
<tr>
<td>Library</td>
<td>Move</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>91%</td>
<td>95%</td>
<td>97%</td>
<td>99%</td>
</tr>
<tr>
<td>C. Commons</td>
<td>Move</td>
<td>93%</td>
<td>92%</td>
<td>91%</td>
<td>87%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>88%</td>
<td>88%</td>
<td>89%</td>
<td>90%</td>
</tr>
<tr>
<td>Campus</td>
<td>Move</td>
<td>96%</td>
<td>92%</td>
<td>88%</td>
<td>78%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>88%</td>
<td>93%</td>
<td>94%</td>
<td>94%</td>
</tr>
<tr>
<td>M. Street</td>
<td>Move</td>
<td>100%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td></td>
<td>Stay</td>
<td>72%</td>
<td>82%</td>
<td>87%</td>
<td>92%</td>
</tr>
</tbody>
</table>
usually larger than 20 when I am moving. It implies that multiple thresholds within
the range from 10 to 20 could be used to distinguish between movement and stillness.
Table 4.2 shows the accuracy of Top k with k = 5 under different thresholds. All of the
thresholds between 12 and 15 are able to provide decent accuracy. Table 4.3 displays
the accuracy of Top k with k = 3, 4, 5, 6 using preset thresholds. This algorithm
provides over 90% accuracy in most cases, and the change of k does not affect the over-
all accuracy significantly. It means that a few APs should be able to provide enough
information to detect if a user is moving at walking speed.

The majority agreement method requires a pre-study of the WiFi environment
first to find the optimal threshold. Then one can use the threshold that is found for the
later movement detection. For Top k, the pre-set threshold works within a tolerable
error rate and should fulfill the basic needs of movement detection. The availability of
WiFi-based movement detection methods obviously is lower have than those methods
that detect movement using an accelerometer. Specifically, the first one only works
when APs exist in the nearby environment while the second can function at almost all
the places.
In this paper, widely used fundamental positioning techniques for mobile phones and their advantages and disadvantages are introduced. Developers should always keep the features of GPS, WiFi and Cell-ID in mind and wisely choose which one/ones to use when building a specific location-aware application. For the optimized positioning systems described in Chapter 3, it is important to recognize that multiple built-in sensors can provide many opportunities to estimate a user’s location. Dead-reckoning and ambience fingerprinting can be the core idea provided by sensors. For Majority agreement and Top k, the evaluation around the UD area shows that both methods are able to act as a binary movement detector with good accuracy. In the future work, I will test these two methods under different speeds and create energy profiles to analyze the energy usage under different scan intervals.
BIBLIOGRAPHY


[21] Anthony LaMarca, Yatin Chawathe, Sunny Consolvo, Jeffrey Hightower, Ian Smith, James Scott, Tim Sohn, James Howard, Jeff Hughes, Fred Potter, Jason Tabert, Pauline Powledge, Gaetano Borriello, and Bill Schilit. Place lab:


