POSITIONING BASED ON TRACKING OF SIGNAL PARAMETERS IN A SINGLE BASE STATION WIMAX NETWORK USING FINGERPRINTING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT MİRAN KÖKSAL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

AUGUST 2010

Approval of the thesis:

POSITIONING BASED ON TRACKING OF SIGNAL PARAMETERS IN A SINGLE BASE STATION WIMAX NETWORK USING FINGERPRINTING

submitted by **MURAT MİRAN KÖKSAL** in partial fulfillment of the requirements for the degree of
Master of Science in Computer Engineering Department, Middle East Technical University by,

ABSTRACT

POSITIONING BASED ON TRACKING OF SIGNAL PARAMETERS IN A SINGLE BASE STATION WIMAX NETWORK USING FINGERPRINTING

Köksal, Murat Miran
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. Fethi Payidar Genç

August 2010, 155 pages

IEEE 802.16 is a point to multipoint broadband wireless access standard, designed from ground up for fast and reliable mobile networking. Several location-related MAC layer fields specified in the standard indicate that WiMAX networks can be convenient backbones for future positioning systems. Information encapsulated in MAC headers is especially important for single base station positioning systems which require fewer network resources than multiple reference station location systems, but need more location-related input data. In this thesis, an algorithm for positioning mobile stations in a single base station network is presented to investigate location capability of WiMAX systems. The algorithm makes use of fingerprinting to create a training database and seeks to find locations of mobile stations by tracking them according to their signal parameters. Experimental results give an idea about how a single base station positioning system performs in the absence of sufficient location-related data, and suggest that better results can be obtained if MAC headers specified in IEEE 802.16 standard can be accessed.

Keywords: positioning, WiMAX, fingerprinting, tracking

ÖZ

TEK BAZ İSTASYONLU BİR WIMAX AĞINDA SAHNE ANALİZİ YÖNTEMİ İLE SİNYAL PARAMETRELERİNİN TAKİBİNE DAYALI KONUMLANDIRMA

Köksal, Murat Miran
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Fethi Payidar Genç

Ağustos 2010, 155 sayfa

IEEE 802.16, hızlı ve güvenilir mobil ağların kurulması için tasarlanmış, tek noktadan çok noktaya hizmet vermeye yönelik geniş menzilli bir kablosuz erişim standartıdır. Standartta tanımlanan ve konum bilgisi taşıyan bazı Ortama Erişim Kontrolü katmanı alanları, WiMAX ağlarının özellikle tek baz istasyonlu konumlardırma sistemleri için uygun birer omurga olacağına işaret etmektedir. Bu tezde, WiMAX sistemlerinin konumlandırma kapasitelerinin incelenmesi için tek baz istasyonlu ağlarda bulunan mobil istasyonların yerlerini tahmin etmeye yarayan bir algoritma tanıtılmaktadır. Algoritma, temel eğitim veritabanını oluşturmak için sahne analizi tekniğinden faydalanmakta ve sinyal parametrelerine göre mobil istasyonları izleyerek yerlerini belirlemeye çalışmaktadır. Deney sonuçları, konum bilgisi içeren yeterli miktardaki verinin elde bulunmaması durumunda tek baz istasyonlu bir konumlandırma sisteminden ne kadar verim alınabileceği hakkında bir fikir vermekle beraber IEEE 802.16 standartında tanımlanan Ortama Erişim Kontrolü başlıklarına erişilmesi durumunda daha iyi sonuçların elde edilebileceğini öne sürmektedir.

Anahtar Kelimeler: konumlandırma, WiMAX, sahne analizi, izleme

Dedicated to Assoc. Prof. Dr. Ahmet Hasan Koltuksuz and Burak Galip Aslan.

ACKNOWLEDGMENTS

The author wishes to express his gratitude to his advisor Prof. Dr. Fethi Payidar Genç, Asst. Prof. Dr. Sinan Gezici, Prof. Dr. Volkan Atalay, Assoc. Prof. Dr. Şebnem Düzgün, Mustafa Kemal Emil, Özgür Norman, Alper Kılıç, Assoc. Prof. Dr. Ferda Nur Alpaslan, Assoc. Prof. Dr. Halit Oğuztüzün, Dr. Attila Özgit, Assoc. Prof. Dr. Soner Önder, Asst. Prof. Dr. İbrahim Körpeoğlu, Bahar Şengün, Şebnem Atun, Efe Cem Kocabaş, Işıl Goler, Beril Erdoğan, Gizem Erdoğan, Beyza Erdoğan, Sena Erdoğan, Erdal Erdoğan, Sinan Cate, Sevi Gökçen Mamaç, Haydar Ali Şeker, Gökçe Atalay, Banu Erdoğan Çelik, Hüseyin Birtan Kalelioğlu, Hale Altınova, Derya Kasacı, Oya Sertel, Hande Tiryaki and to his family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Position estimation is the process of finding the location of a mobile (or stationary) node in a wireless network by evaluating the signals that are transmitted between that node and a number of fixed nodes [18, 31]. Since the introduction of wireless networks like GSM, fast development of which made mobile communication a reality, positioning systems has gained a significant amount of interest. It is expected that positioning techniques based on WiMAX networks which implement IEEE 802.16 standard will receive high attention in a similar way; because, WiMAX provides high speed network access and mobility at the same time which is tomorrow's trend in wireless communications [8]. This affects deployment of WiMAX in a positive way. It is a natural result of availability that positioning with these systems is getting more attention.

In this thesis project, location capability of a real world single base station WiMAX network is investigated through development and evaluation of a position estimation algorithm. The purpose is to realize position estimation by only making use of the information encapsulated in transmitted network packets as specified in IEEE 802.16 standard, and not modifying the available WiMAX hardware at all. Existence of positioning related data in MAC headers, like Timing Adjust which can provide fine grained information about distance of a mobile station from a base station, encouraged research in this area.

The algorithm proposed in this work is a two dimensional, physical positioning technique which provides absolute locations to users. It utilizes fingerprinting, that is, it collects signal parameters at known locations, to create a training database. When a mobile station requests its location to be estimated, a training database is queried with signal parameters reported by that station, and a position estimate is computed using the location information contained in resulting records. Normally, in case of a base station that is installed in the middle of an absolutely straight area, fingerprinting can only help to determine a circle (centered at the base station) on which a mobile station to be located can be found. However, in a real world scenario it is known that the service area of a base station has a unique topography, and signal impairments come into play because of environmental variability. Therefore, instead of a whole circle, more specific results about a mobile station's location can be obtained via fingerprinting. The algorithm takes advantage of this. Since the lack of sufficient information caused by utilization of one base station can not be compensated by fingerprinting alone, a tracking method is developed and integrated into the algorithm. Tracking is a crucial part of this work, and inherently requires mobile stations to be in motion during positioning.

The remainder of this thesis is organized as follows. Chapter 2 provides background information about IEEE 802.16 standard and WiMAX. Signal impairments that reduce transmission quality, fundamental position estimation techniques, and parameters that are used to locate mobile stations are also introduced in Chapter 2. Chapter 3 reviews related work about single base station positioning. In Chapter 4, hardware and software employed in this work are enumerated, and equipment problems which fundamentally influenced the proposed algorithm's development process are explained. Chapter 5 presents the studied area in the Middle East Technical University campus and gives details about how signal measurements and fingerprinting were carried out. In Chapter 6, a two dimensional visualization of collected fingerprints is described. Chapter 7 explains both the proposed position estimation algorithm and k Nearest Neighbor algorithm in detail. Chapter 8 describes the experiments, discusses the results of the algorithms, and compares them. Finally, Chapter 9 concludes the thesis and presents suggestions for future work.

CHAPTER 2

BACKGROUND INFORMATION

This chapter covers background information about

- wireless broadband technologies utilized in this thesis,
- signal impairments that can be encountered in wireless networks,
- techniques for estimating positions of mobile stations in wireless networks, and
- parameters that can be used by position estimation techniques.

2.1 IEEE 802.16 and WiMAX

IEEE 802.16 WirelessMAN Standard for Wireless Metropolitan Area Networks is a point to multipoint broadband wireless access standard developed by the IEEE 802.16 Working Group on Broadband Wireless Access Standards which is a unit of IEEE 802 LAN/MAN Standards Committee [2]. The standard describes the air interface and focuses on establishing requirements for physical layer (PHY) and Medium Access Control layer (MAC) [43]. Main purpose of the standard is to define a technology that will provide a cost effective and easy to deploy wireless alternative to wired broadband systems, allowing high speed fixed and mobile network access established with stability and security in mind.

Work on IEEE 802.16 standard was first started in 1998 with the formation of the IEEE 802.16 Working Group. The goal was to come up with a wireless broadband system with an emphasis on fixed, point to multipoint communication in line of sight (LOS) environments. The first deliverable of the group was the original 802.16 standard released in December

2001. Operation frequencies defined in the standard ranged from 10GHz to 66GHz, and Time Division Multiplexing (TDD) was employed as the multiplexing scheme. Following the first standard, a number of amendments were released, each one of which improved or extended the standard in a different way [4]. Support for non line of sight (NLOS) propagation and new frequency ranges (2GHz to 11GHz) was added with introduction of Orthogonal Frequency Division Multiplexing (OFDM) in PHY, which is a modulation scheme known for its resistance to harsh channel conditions like multipath [4, 44]. Further revisions, including addition of Orthogonal Frequency Division Multiple Access (OFDMA) to available multiplexing schemes, resulted in releasing an updated version of the standard in 2004, named IEEE 802.16-2004. Later in 2005, with completion of IEEE 802.16e-2005 amendment, which specified means for roaming and seamless handover, the standard earned the ability to support mobility [4].

IEEE 802.16e-2005 combines efficient transmission and multiplexing schemes with advanced techniques that are used to achieve stable data transmission. Availability of multiple channel bandwidths, frequency bands, transmission schemes, etc. means that it is flexible enough to be adapted to different needs. As mentioned above, IEEE 802.16e-2005 supports both single carrier and multi carrier transmission schemes (OFDM.) Time Division Multiplexing (TDM,) Time Division Multiple Access (TDMA,) and more importantly OFDMA are specified as available multiplexing schemes (OFDMA makes it possible to dynamically adjust data rate according to channel conditions which is why it is of big importance for the standard.) Also explained in the standard are modulation schemes of Quadrature Phase Shift Keying (QPSK,) Quadrature Amplitude Modulation (QAM,) and duplexing schemes of Time Division Duplexing (TDD,) Frequency Division Duplexing (FDD.) The standard is designed to operate on 2GHz-11GHz frequency range for fixed applications, and 2GHz-6GHz frequency range for mobile applications with channel bandwidths ranging from 1.75Mhz to 15MHz [4]. Typical cell radius is stated to vary between 3-5 miles for fixed applications, and 1-3 miles for mobile applications according to [6].

Switching from technical details of the standard to a more abstract look, IEEE 802.16 protocol architecture consists of four sub layers [36]. Physical layer, the lowest layer in the architecture, deals with specification of transmission medium and identification of frequency bands over which signals can be transmitted. Transmission layer, found above physical layer, takes on the tasks of signal encoding, preamble generation, bit transmission on the transmitting side, and signal decoding, preamble removal, and bit reception on the receiving side. Together, these two sub layers are equivalent to the lowest layer of Open Systems Interconnection (OSI) model (physical layer) [36].

MAC layer, which stands above transmission layer, manages base station's (BS) and subscriber stations' (SS) access to the communication medium, that is, the radio channel. On the transmitting end it appends an address and error detection bits to the data which was received from upper layers and destined for a receiver, while on the receiving end it acknowledges a received frame by its address, extracts actual data, and checks integrity of the data using error detection bits. Steps one should take to access medium in downlink direction (from base station to subscriber stations) are easier compared to the ones concerned with access in uplink direction (subscriber stations to base station) since in uplink direction multiple subscriber stations have to compete for access to the medium [36].

On top of the IEEE 802.16 protocol architecture is convergence layer whose task is to differentiate between different kinds of packets received from upper layers and process them appropriately according to their class [23, 24, 36]. MAC layer and convergence layer together are equivalent to layer two of OSI model (data link layer) [36].

It is a good idea to learn about basics of OFDM, OFDMA, and Adaptive Modulation and Coding (AMC) to get a better understanding of IEEE 802.16e-2005. OFDM is a transmission scheme which makes use of multiple carriers to transmit a data stream. A single frequency channel which would normally carry a single data stream is divided into multiple smaller channels called subcarriers. The idea in OFDM is to split the main data stream into multiple lower bit streams and carry each one of them over one subcarrier [4]. "Each subcarrier is modulated with a conventional modulation scheme" such as QAM or QPSK [44]. Unlike regular frequency division multiplexing, OFDM does not need non overlapping subcarriers but it requires subcarriers to be orthogonal, meaning that they can overlap but not interfere with each other [9]. An advantage of orthogonal subcarriers is that they require less bandwidth compared to non overlapping subcarriers. The most important advantage of having multiple subcarriers is that they show high resistance against Inter-symbol Interference (ISI) which is a problem caused by multipath propagation. When a transmitter sends a signal, it is likely that multiple copies of the signal, each taking a different path, arrive at the receiving end at different times. That means while a transmitted signal is being received, a component of a previously transmitted signal may also be received which can interfere with the actual signal. OFDM can cope with this problem, because by dividing a frequency channel into multiple subcarriers it increases the symbol time, that is, the period of time during which receiver waits for a signal. Therefore, it reduces the likelihood of receiving a component of a previously sent signal during transmission of a subsequent one. Further prevention against Inter-symbol Interference is provided by using guard intervals between symbols since it adds up to the symbol time. However, it requires more power and bandwidth to be consumed [4].

OFDM deals with multi carrier modulation of one bit stream over one communication channel [44]. So, it is not a channel access method for multiple users. OFDMA extends OFDM to make it usable in multi user environments by grouping subcarriers into multiple subchannels which are assigned to different users [4]. The ability to assign subchannels with different number of subcarriers to different users allows serving each user with a different Quality of Service (QOS) [44]. Such subchannelization is obtained by Adaptive Modulation and Coding which dynamically changes modulation and coding schemes according to channel conditions. First, quality of a communication link that was set up between a base station and a subscriber station is evaluated. If downlink is being investigated, subscriber station measures signal-to-noise and interference ratio and informs base station about it. In case of uplink, base station can handle channel quality estimation itself. Information about channel quality is processed by base station to find the highest modulation and coding schemes for the channel to provide highest possible data rate [4]. However, note that robustness is more desirable than high throughput, which is why a base station changes modulation and coding schemes of a channel to lower ones when stability of the channel is questionable. For example, for a subscriber station that is close to its associated base station, 64 QAM can be used in the beginning for high throughput. But, as the subscriber station moves away from the base station, to protect reliability of the link, the base station can dictate using a more robust modulation scheme like 16 QAM or QPSK, increasing effective range and decreasing data rate at the same time [6]. Therefore, there is a “real-time trade-off between throughput and robustness on each link” [4].

In IEEE 802.16e-2005, connection reliability is further ensured by the error control mechanism called Automatic Retransmission Request (ARQ) which requires an acknowledgement packet to be sent from a receiver to a transmitter when a transmitted packet successfully arrives at the receiver. If an expected acknowledgement packet is not received on the transmitting side, the transmitter assumes that the packet was not delivered correctly, and resends it. The standard also supports Hybrid-ARQ (H-ARQ) which combines regular ARQ with Forward Error Correction (FEC) [4]. In FEC, a receiver tries to identify and correct possible bit errors in a received data block using error correction information transmitted with the data. Before sending data to a receiver, a transmitter evaluates a function which takes the data to be sent as its input, and outputs a bit string called "error-correcting code." The transmitter creates a data block by appending that bit string to the actual data, and transmits it to the receiver. Upon receiving the data block, the receiver computes an error-correcting code using the received data to compare it with the received code. If codes are the same, then it is assumed that there are no errors in the data. If they are different, then it is understood that there are bit errors. In the case of bit errors being identified as recoverable depending on code length and algorithm being used, erroneous bits can be found and corrected in the receiving end [36]. Hybrid-ARQ basically works by passing a data block through a FEC coder before it is transmitted. When the data is received the process is reversed using a FEC decoder, and if the decoding fails a retransmission is requested. When retransmitted data is received, it is combined with the incorrect one to extract the actual data correctly [4].

In addition to the techniques mentioned above, IEEE 802.16 based networks can deliver enhanced performance by utilizing multiple antenna elements. Advanced Antenna Systems is the optional feature which increases range, system capacity, spectral efficiency and signal-to-noise ratio (SNR) gain. Simply put, a base station can improve reliability by adapting its antenna array to focus on an individual subscriber station and transmitting powerful signals that combine multiple regular signals [25].

IEEE 802.16 has a connection oriented MAC layer which was designed with Quality of Service in mind. The standard supports a number of different traffic types, like voice or web, and allows efficient allocation of radio resources to users according to their needs. In the case of multiple connections between a base station and a subscriber station, each connection is handled separately, meaning that each one can be assigned to a different traffic class [4]. In IEEE 802.16e-2005 there are five scheduling services called Quality of Service classes, defined [25]. Unsolicited Grant Service (UGS) is used for uplink connections over which fixed size data packets are transmitted periodically. Voice over Internet Protocol (VoIP) without silence suppression is an application generating constant bit rate (CBR) traffic for which UGS can be used. Real-Time Polling Service (rtPS) is suitable for real time streams that require periodic transmission of variable size data packets in uplink direction. It can be used for carrying variable bit rate (VBR) streams like Moving Pictures Experts Group (MPEG) videos [25]. Extended rtPS combines UGS and rtPS, guaranteeing certain data rate while transmitting real time VBR streams. Extended rtPS is a convenient choice for applications like VoIP with silence suppression [4, 25]. Non-Real-Time Polling Service (nrtPS) is designed for delay tolerant VBR streams that require certain minimum data rate to be ensured. nrtPS is adequate for applications like FTP. Finally, Best Effort (BE) provides efficient service for uplink traffic that requires no specific data rate or delay guarantees, e.g. HTTP [4].

Overall, IEEE 802.16 puts all schemes, methods and features that are mentioned above into a standard, and defines means in which they work together in harmony to form the foundations of last generation broadband wireless networks. The standard's broad scope and numerous options come in handy to satisfy different needs and realize various network applications. However, high customizability is an obstacle for deployment of interoperable networks. IEEE 802.16 Working Group developed the standard but did not deal with narrowing down the scope of it, leaving that task to the industry. This lead to formation of a non profit organization which defined the details of "interoperable implementations [45]" of IEEE 802.16 based networks called WiMAX, short for Worldwide Interoperability for Microwave Access. The organization itself is called WiMAX Forum [4].

WiMAX Forum consists of equipment and component suppliers from the industry [6]. Main purpose of the forum is to make agreed decisions about options of IEEE 802.16 standards that a WiMAX device should implement. For example; IEEE 802.16e-2005 supports frequencies from 2GHz to 11GHz, but WiMAX Forum pays attention to 2.5GHz and 3.5GHz for actual implementation. Forum defines a number of 'system profiles' for compatibility and interoperability among IEEE 802.16 based devices. These profiles are sets of physical and MAC layer features some of which are mandatory while others are optional [4]. Apart from ensuring compatibility and interoperability among devices with system profiles, WiMAX Forum promotes adoption of WiMAX products; acts as a catalyzer in deployment process of WiMAX systems, and speeds up the penetration of those systems into the market [6, 46].

Figure 2.1: WiMAX enabled HTC EVO 4G cellular phone and Overdrive 3G/4G Mobile Hotspot by Sierra Wireless [13, 11].

There are two versions of WiMAX that are being mentioned frequently. Fixed WiMAX is the version based on IEEE 802.16-2004 standard which is specialized for wireless communication between stationary nodes while Mobile WiMAX is based on IEEE 802.16e-2005 standard which extends the former with support for mobility. One of the important properties of both is that they have IP based reference network architectures. That means an IP architecture utilizing IP based protocols is responsible for carrying services between end nodes [4].

In general, there are two types of end nodes in a WiMAX network; base station and subscriber station. Base stations can be installed on top of buildings or towers while subscriber stations may come in a variety of forms. A fixed subscriber station can be in form of an outdoor antenna which provides better signal reception and network performance, or an all in one indoor modem (like an ADSL modem) which has the advantage of easier installation. A subscriber station can also be a mobile device like a notebook, cellular phone, mobile hotspot (Figure 2.1,) portable media player, etc. allowing access to ubiquitous networks on the go. An example of WiMAX topology can be seen in Figure 2.2.

Advantages of WiMAX networks make them a good competitor against other wireless

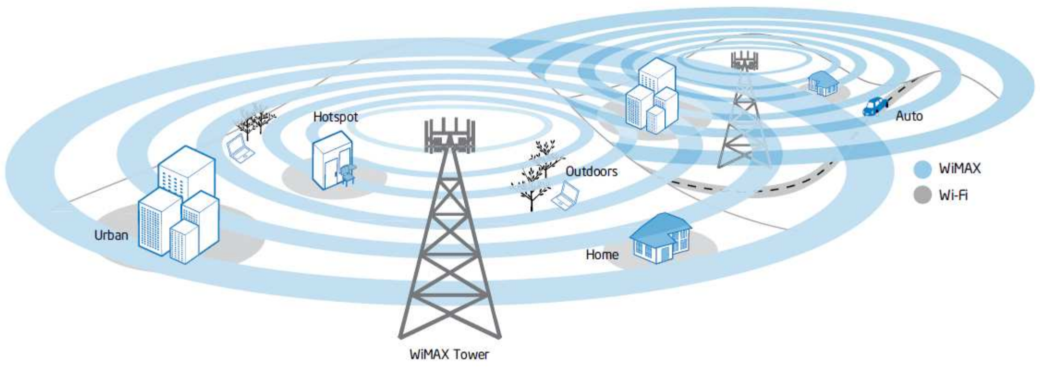


Figure 2.2: An example WiMAX topology [27].

broadband and wired network technologies. Painless base station installation and total cost of deployment within reasonable limits make WiMAX networks a good choice especially for areas where it is not practical to build a wired infrastructure because of inconvenient terrain. Installing base stations is a less time consuming task compared to wiring a large service area. WiMAX networks can be used for data, Voice over Internet Protocol (VoIP) and Internet Protocol Television (IPTV) services, and may be employed to complement other communication technologies, e.g. WiMAX as backhaul for Wi-Fi networks [6]. Existing companies providing wired network services can make use of mobility features of WiMAX to extend their service areas and leap ahead of the competition. Mobile operators which did not make the transition to third generation (3G) systems can utilize WiMAX to fill the gap, and provide high performance mobile applications just like 3G operators.

According to WiMAX Forum's Industry Research Report [47] issued in April, 2010 there are 148 countries with WiMAX deployments, and a total of 588 WiMAX networks "that are either in service or planned/in deployment." As stated by Dr. Mohammad Shakouri, vice president of WiMAX Forum, "Today the initial WiMax system is designed to provide 30 to 40 megabit [per second] data rates" [40]. Real world speed tests carried out by subscribers of Vividwireless reveal an average data rate of 12.23 Mbps and a peak data rate of 36.78 Mbps [41].

2.2 Signal Impairments

In a communication system, either wired or wireless, a transmitted signal that is destined for a receiver always encounters some degradation, causing the received signal to be different from the original one. These modifications are caused by signal impairments which are explained in this chapter to gain a better understanding of wireless communication in general [36].

2.2.1 Attenuation

Attenuation is reduction in signal strength due to distance between a transmitter and a receiver. In order for a receiver to notice and interpret a signal, strength of the signal must be above a certain level. Information carried with a signal can be extracted correctly only if the received signal is more powerful than perceived noise. At certain distances, attenuation can cause communication to get interrupted or be completely halted. Existence of multiple receivers makes attenuation a bigger concern since each receiver can be at a different distance from a transmitter. It is not hard to see that mobility adds up to the problem. To overcome these factors amplifiers can be used. One property of attenuation is that it increases as higher frequencies are used for communication. That causes relative strengths of frequency components of received signals to be different than that of transmitted signals [36].

2.2.2 Free space loss

Free space loss is the reduction in signal power caused by dispersion of signals in wireless communication medium. Because of that a transmitted signal disperses with distance, a receiver that is coupled with a fixed antenna receives less powerful signals as it moves away from the transmitter. A signal experiences attenuation caused by free space loss even when other types of signal impairments are assumed not to be present. Normally, free space loss increases as signals are transmitted at higher frequencies. However, introduction of antenna gain changes this relation such that free space loss decreases as higher frequencies are used for transmission [36].

2.2.3 Noise

As mentioned previously, a received signal is not the same as its transmitted version since transmitted signal gets modified by distortions related to the transmission system. However, this is not the only reason why transmitted and received signals are different. There are other signals in the transmission system and communication medium that get included in the reception process without the receiver knowing anything about it. Those undesirable signals are called as noise and affect reception of actual signals destructively. There are four different types of noise. Thermal noise is the one that is caused by thermal agitation of electrons. Every electronic device suffers from thermal noise, and there is no way of getting rid of it. That is why thermal noise sets an upper bound on communication performance. Second noise type is intermodulation noise. If there are two signal transmissions at different frequencies that are being carried out in the same medium at the same time, a new signal can be produced at a frequency that is equal to the sum or difference of those two frequencies. That newly produced signal is called intermodulation noise and is an unwanted signal since it can interfere with transmissions being modulated at that frequency. Third noise type is crosstalk which is caused by uncontrollable coupling of two signal paths. In wireless communication this happens when unexpected signals are sensed by receiving antennas unintentionally (because of that signals disperse, as mentioned previously.) In general, crosstalk and thermal noise disturb a transmission at a similar rate. However, for transmissions at the unlicensed industrial, scientific and medical (ISI) radio bands crosstalk is a more damaging. The last noise type is impulse noise which is the most problematic type for digital transmissions. It is an unexpectedly produced noise with a short duration and high amplitude. It can be caused by electromagnetic disturbances or communication system defects [36].

2.2.4 Atmospheric absorption

Atmospheric factors like water vapor and oxygen can add up to loss in signal strength in wireless communication. Transmissions at around 22GHz suffer the most from water vapor while oxygen disturbs transmissions at around 60GHz the most. These factors are less dangerous for transmissions occurring at lower frequencies, especially below 15 GHz and 30 GHz for water vapor and oxygen, respectively. A transmission can also be affected by water droplets hanging in the air (caused by rain and fog) since droplets cause signals to scatter and result in attenuation [36].

2.2.5 Refraction

Refraction is another atmospheric impairment in wireless systems and is the bending of a transmitted signal. It emerges as a result of change in propagation speed of a signal depending on the change in that signal's height above sea level. In the case of a fixed, line of sight (LOS) communication, refraction can obstruct reception of a transmitted signal partially or completely [36].

2.2.6 Multipath

In a wireless environment, a transmitter and a receiver can be located such that their communication is not hindered by existence of obstacles in between them, that is, there can be a line of sight path between them. However, with the introduction of mobility, numerous obstacles like trees, buildings and living beings in motion are brought into place depending on transposition of the moving node. Because of these obstacles, transmitted signals may experience reflection, diffraction or scattering, causing multiple copies of the same signal to be produced and propagated in the medium. Inherently, these copies, which follow different paths, may arrive at the receiving end at different times and actively influence reception of the actual signal either constructively or destructively. Variation of received signal power due to such multipath propagation and changes in transmission medium is called fading, and it is a crucial subject to cover in wireless communication systems design. As mentioned above, multipath propagation can be in form of reflection, diffraction or scattering.

Reflection happens when one component of a signal hits a surface that is larger than the wavelength of the signal and gets reflected in another direction. Therefore, the signal component follows a path that is different from the actual signal path, and may or may not be received by its destined receiver. If that new path leads the signal component to its destination, depending on the length of the path it may be received approximately at the same time as the actual signal or later than that (in which case it is highly likely that the signal component interferes with a subsequent signal.)

Diffraction is similar to reflection, but it happens when a signal hits the edge of an object that is larger than the wavelength of the signal. Such an encounter causes the signal to continue its propagation in another direction. Diffraction may aid reception of signals in non line of sight (NLOS) cases [36]. Nevertheless, because of that every time a signal is diffracted it loses some of its power; the received signal may be totally unintelligible. Such loss in signal power is called as shadowing [4].

Scattering is the spreading of multiple weaker copies of a signal in different directions as a result of the original signal experiencing an obstacle with a size close to or less than the wavelength of the signal. Scattering and diffraction are important propagation mechanisms especially for mobile wireless communications since they can influence signal reception positively [36].

2.2.7 Doppler shift

The last impairment affecting mobile communication is the Doppler shift. When there is a relative motion between a transmitter and a receiver, frequency of the transmitted signal observed by the receiver becomes different than the actual one. The amount of difference depends on vehicular speed and the carrier frequency that is being used. Doppler shift causes reduction in signal-to-noise ratio (SNR) [4].

2.3 Position Estimation

Position estimation is the process of finding the location of a mobile station (MS) or a stationary node in a wireless network by evaluating signals that are exchanged between that node and a number of fixed nodes, that is, base stations (BSs) [18, 31]. There are three techniques in general that can be used to estimate the position of a node in a wireless network: *dead reckoning*, *proximity location*, and *radiolocation* [30].

Dead reckoning is a basic location technique in which starting position of a node that will be located is known in the beginning. As the node moves around its acceleration, velocity and direction of travel are measured to calculate how much and in which direction it moves from the starting point. Each new position estimate is computed based on a previous one. Downside of such a chain of computations is that positioning errors in early estimations get stacked together, causing subsequent estimations to suffer from larger errors. Still, it is possible to reduce such errors by utilizing more reliable positioning techniques from time to time to obtain accurate position estimates, and then using dead reckoning to compute subsequent positions [30].

In proximity systems, position of a node is calculated by collecting data about proximity of the node to a number of fixed detection devices and processing them. These systems can be suitable for positioning applications covering small areas; however they are not practical for locating devices in larger areas arguably because of the cost associated with installing lots of fixed detection devices. Proximity systems support both *self* and *remote positioning* [30].

In self positioning, which is also known as mobile centric positioning [20]; a mobile station gathers signals from a number of reference points, e.g. base stations, and makes some measurements based on those signals to estimates its location itself [12]. Naturally, signals transmitted by reference points must contain information about their position [30]. The most popular example of self positioning (note that it is not a proximity system) is Global Positioning System (GPS) which consists of satellites revolving around the Earth and broadcasting positioning information which allow GPS receivers to determine their own location [12].

The opposite of self positioning is remote positioning, also known as network centric positioning, in which location of a node is determined by the network [20]. First, the node to be positioned transmits signals which are sensed and measured by one or more receivers (reference nodes) [18]. Measurement results are forwarded to a central unit by the receivers where they are combined together to come up with a position estimation. If it is required, the node can be informed about its location [12]. An example of proximity systems utilizing remote positioning is primitive positioning systems that are currently being used by cellular networks where location of a cellular phone is roughly estimated by identifying the sector or cell that the phone is currently residing in. Advantage of remote positioning is that it does not require any modifications on mobile stations [30].

There is a third positioning technique called *indirect positioning* which is investigated in two groups. In *indirect remote positioning*, a mobile station determines its own location just like a regular self positioning node after which it sends the location data to a remote site. Similarly in *indirect self positioning*, a remote positioning system computes the location of a mobile station and transmits the location data to the node [12].

Going back to positioning techniques, the last one is radiolocation. Radiolocation deals with finding position of a mobile station by measuring and processing signals exchanged between the station and multiple base stations. Just like proximity systems, radiolocation systems can utilize self or remote positioning. GPS, which was mentioned as a self positioning system previously, is also a radiolocation system. Radiolocation with self positioning requires establishing forward links (from base station to mobile station) which means that a mobile station owns antennas receiving signals broadcasted by base stations to determine its location. Radiolocation with remote positioning requires establishing reverse links (from mobile station to base station,) meaning that a mobile station must have a transmitter to send signals to base stations that will calculate its location [30].

In this thesis, radiolocation is considered as the primary positioning technique and everything from this point on will be explained for radiolocation. Also, the positioning algorithm proposed in this work is an example of a two step algorithm as explained by [18]. In the first step of a two step positioning algorithm, certain signal parameters are extracted from transmitted signals. These parameters can be Time of Arrival (TOA,) Angle of Arrival (AOA,) Received Signal Strength (RSS,) etc. which will be described in next section. In the second step, according to the parameters that are obtained, location estimation is carried out using methods like fingerprinting, geometric methods, etc. In a two step remote positioning algorithm, reference nodes can obtain parameters from collected signals and forward them to a central unit where actual positioning will take place. It is also possible for reference nodes to directly forward the signals they collect to a central unit; however in that case the central unit will be left with more work to do including parameter extraction [18].

Each positioning technique that is described in here can be used for a different positioning application. Requirements of different applications have implications not only for the technique to be employed but also for the type of location information to be used. There are four types of location information which are *physical location*, *symbolic location*, *absolute location* and *relative location* [31].

Location information provided by a positioning system can be in form of physical or symbolic location [22]. Physical location defines an object's location accurately without leaving room for ambiguity. An example system is GPS which provides latitude, longitude and altitude of an object. Symbolic location, on the other hand, gives abstract, verbal explanations, e.g. ‘in room 107,’ ‘in meeting room,’ ‘next to bus station.’ Therefore, systems that are only based on symbolic location provide undetailed information. It is possible to convert a physical location to a symbolic location by querying a database where each record consists of a physical location and its corresponding symbolic location.

An absolute location system provides location information according to a shared reference grid. That means all nodes in the system agree on using the same reference grid, so that data corresponding to a location are interpreted in the same way by all nodes. For example, coordinates given by two GPS receivers which are located at the same point will be the same, and for any GPS equipment (receiver or satellites) those coordinates will point to the same place on the Earth. A positioning system can also make use of relative location, meaning that there is no shared reference grid and each node in the system can have a reference grid of its own. An example can be a robot swarm where each robot computes distance between itself and other robots in its vicinity to find out which direction it should be headed to. One can convert an absolute location to a relative location in existence of a reference point according to which relative location can be calculated. Conversion in reverse direction is also made possible by triangulation, but it requires knowledge about absolute positions of reference points [22].

2.4 Parameters for Positioning

In a two step positioning algorithm, parameters extracted from signals in the first step can be used by a number of location estimation schemes in the second step. Here, two of those schemes will be explained with the signal parameters that accompany them. Briefly, geometric techniques estimate the position of a node by exploiting geometric relationships depending on signal parameters while mapping (also known as fingerprinting or scene analysis) techniques rely on extracting location dependent signal patterns from in field signal observations, and compare signal parameters of the node to be located with those patterns [18].

2.4.1 Geometric Techniques

Geometric techniques estimate position by utilizing geometric properties of triangles [31]. It combines circular or hyperbolic uncertainty regions obtained by evaluating signal parameters to find their intersection point. An uncertainty region is the locus which a mobile node to be located resides on, meaning that if there are multiple such regions then the node must be at a location that is a part of all the uncertainty regions at hand. Geometric techniques deal with identifying that location which is the intersection point [18]. There are two types of geometric techniques: trilateration and triangulation [31].

2.4.1.1 Trilateration

Trilateration, also known as range measurement, uses parameters of signals transmitted by multiple reference stations (RSs) to determine distances between them and the mobile station (MS) to be located [31]. Each one of those distances represents a circle around one of the reference stations. The point where circles intersect gives the location of the mobile station [18].

Signal parameters which allow position estimation by trilatateration are

- Time of Arrival (TOA,)
- Roundtrip Time of Arrival (RTOA,)
- Time Difference of Arrival (TDOA) and
- Received Signal Strength (RSS.)

Time of Arrival In a wireless communication system, the distance between a transmitter and a receiver is proportional to the amount of time a transmitted signal spends to reach its destination (the receiver) [31, 12]. This amount of time is called as propagation time or flight time of a signal. In a TOA based technique, one way propagation time of a signal travelling between a reference station and a mobile station is measured to compute the distance between them by multiplying the propagation time with the radio signal velocity [31]. Since there is no knowledge about the direction of the mobile station (relative to the reference station,) a

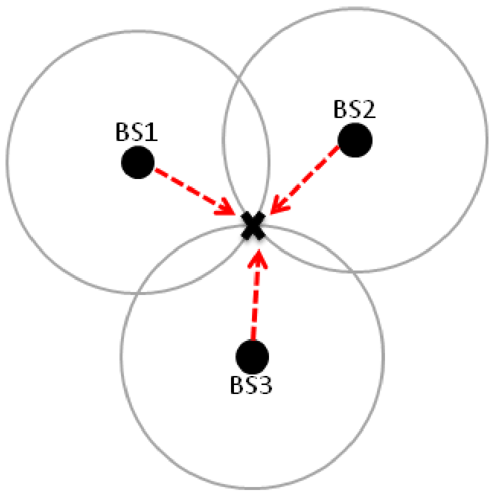


Figure 2.3: Positioning with TOA, RTOA, or RSS parameters.

computed distance only defines a circle centered at the reference station. The exact position of this mobile station on the circle can not be known until extra information is provided. A TOA measurement using a second reference station defines another circle, resulting in two circles intersecting at two points. Because of that there is still ambiguity about which one of those points to select as the location of the mobile station, a third reference station must be employed to do another TOA measurement [30, 12]. Therefore, for two dimensional positioning of a mobile station, at least three reference stations are required (Figure 2.3) to measure TOA parameters of signals transmitted to or received from the mobile station [31].

The problem with TOA is that a receiver has to know the exact time that a transmitter transmits a signal to it [12]. Both receiver and transmitters have to be precisely synchronized to prevent possible errors in position estimation process [31]. Synchronization can be provided by using a shared clock or exchanging timing information. Apart from imposing rigid timing constraints, signal to noise ratio or effective signal bandwidth can be increased to provide more accurate position estimates [18].

Roundtrip Time of Arrival RTOA is the time it takes for a signal to travel from a transmitter to a receiver, and then back to the transmitter. Positioning wise RTOA is used in the same way as Time of Arrival. If a reference station is to measure RTOA of a mobile station, then it transmits a signal to the mobile station and starts a timer. Upon receiving the transmitted signal, the mobile station responds by transmitting another signal back to the reference station. The reference station stops the timer when it receives that signal from the mobile station, and uses the elapsed time (which is approximately two times the one way measurement [12], Time of Arrival) in estimating position of the mobile station. The advantage of RTOA is that it does not require synchronization between communicating stations. However, because of that the time a responding station spends to process a received signal is included in RTOA and the transmitter does not know how long that process time is, there may be positioning errors that can not be ignored, especially in short range systems. For long range systems where transmission time is significantly higher than the responder's process time, this may not be a very important problem [31].

Timing Advance Timing Advance (TA) is a parameter defined in Global System for Mobile Communications (GSM) specification and can be considered as a real world implementation of Roundtrip Time of Arrival. The parameter was introduced to manage the transmissions originating from mobile stations and destined for base transceiver stations (BTSs.) GSM telecommunications standard employs Time Division Multiple Access (TDMA) as the channel access scheme, meaning that all transmissions from mobile stations to base transceiver stations must comply with certain timing rules. A base transceiver station has multiple mobile stations associated with it, each one of which seeks permission for channel access to transmit signals. Permission is required, because if two or more mobile stations transmit signals to the same base transceiver station at the same, they will cause interference at the receiving end. Therefore, each mobile station must be assigned to a time slot during which it is allowed to transmit signals. TA deals with those time slots [12].

A base transceiver station sets a different TA for each one its associated mobile stations "according to the perceived round trip propagation delay BTS-MS-BTS [14]." A mobile station receiving a TA adjusts its transmission timing accordingly, so that the signals it transmits arrive at the receiving end during the expected time slot. Since TA is a representation of propagation time, it can be used to estimate the distance between communicating stations. Just like other trilateration parameters, it defines a circular locus that contains position candidates for a mobile station [12]. TA can have a value from 0 to 63 [14] where each increment corresponds to a distance of 554 meters [12]. That means it can only be used for coarse position estimation.

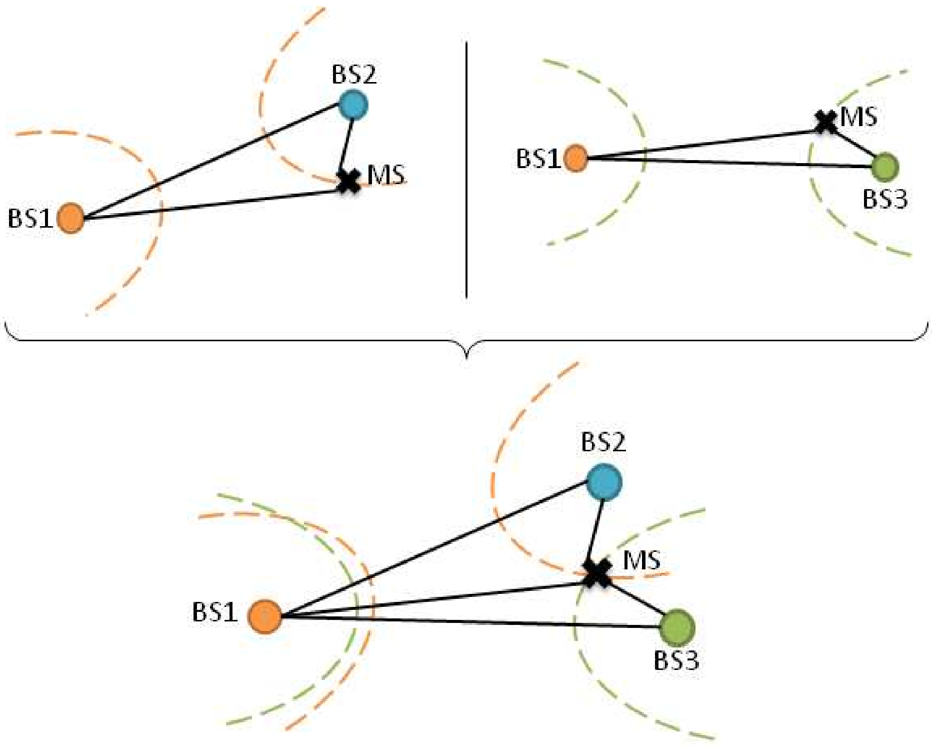


Figure 2.4: Positioning with TDOA parameter.

Time Difference of Arrival TDOA is the difference between arrival times of two signals transmitted from two different reference stations to a mobile station or vice versa. A TDOA parameter defines a hyperbola with its foci at the reference stations [18]. The mobile station is located somewhere on that hyperbola, meaning that there is an uncertainty curve, and to find out its exact location a second hyperbola (another TDOA measurement) is required. That means there must be a third reference station to do a new TDOA measurement. Two hyperbolas generally intersect at one point (Figure 2.4) which determines the position of the mobile station being investigated [31]. However, it is also possible for two hyperbolas to intersect at two points and cause ambiguity in position estimation. The solution to such a case is an extra TDOA measurement to draw a third hyperbola which intersects other two hyperbolas at one certain point [12].

TDOA measurements require reference stations to be synchronized, but the mobile station to be located does not need to be synchronized with reference stations. In self positioning case, reference stations must transmit signals to the mobile station at the same time, or they should provide time offset information with their signals, so that TDOA parameter can be measured correctly. Similarly, in remote positioning case, reference stations must be knowledgeable about the relationship between their clocks in order to correctly evaluate the signals they receive from a mobile station [12].

TDOA of a mobile station and two reference stations can be measured by subtracting the Time of Arrival parameter measured between the mobile station and one of the reference stations from the Time of Arrival parameter measured between the mobile station and the other reference station. One of those Time of Arrival parameters may include an unknown timing offset which is caused by the absence of synchronization between the mobile station and reference stations. This may sound like it can give birth to an erroneous TDOA computation; however because of that the reference stations are synchronized with each other, the same unknown timing offset will be included in all Time of Arrival measurements. During subtraction timing offsets of the two Time of Arrival parameters will cancel each other and the result will be the correct value of TDOA [18].

Received Signal Strength Multipath effects explained in previous sections cause problems in radio propagation which in turn has a negative influence on the accuracy of measured signal parameters. An alternative parameter is RSS which is an indicator of the attenuation that a transmitted signal experiences until it is received, that is, the difference between signal strengths of a transmitted signal and a received signal. RSS parameter of a signal transmitted between a mobile station and a reference station can be translated into a distance or range estimate [31]. A circle centered at the reference station is drawn using that range estimate, defining the area where the mobile station can be at. Therefore, like the cases in Time of Arrival and Roundtrip Time of Arrival, multiple circles (multiple RSS measurements with different reference stations) are needed to find an intersection point where the mobile station is actually located at [30].

Multipath fading and shadowing can have bad effects on RSS measurements, leading to insufficient accuracy in positioning [31]. Averaging RSS values measured "over a sufficiently long time interval [18]" can help reducing such effects to some degree. However, for a slow or stationary mobile station, averaging RSS values is likely to be useless since low mobility will cause most of the measurements to suffer from the same signal impairments [30].

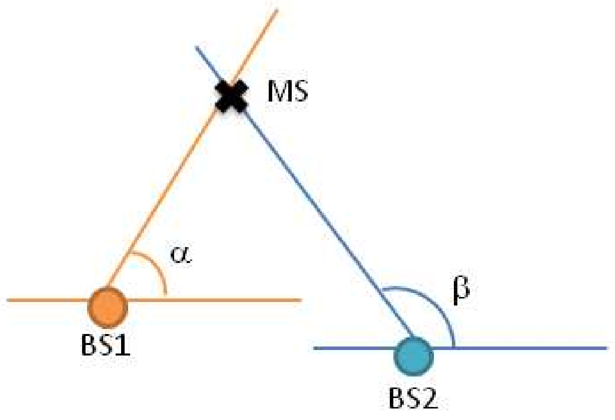


Figure 2.5: Positioning with AOA parameter.

2.4.1.2 Triangulation

Triangulation is the process of locating a mobile station by measuring its angles relative to multiple reference stations [31]. Each angle defines a straight line between the mobile station and one of the reference stations that intersects with another straight line defined by another angle. In absence of measurement errors, multiple lines are expected to intersect at one point. In case of intersection at multiple points (because of random errors,) one can not know which point to select as the location of the mobile station. Furthermore, introducing more parameters can increase the number of intersections; because, each new parameter may incorporate errors that occur during parameter estimation process, eventually resulting in erroneous lines intersecting at more than one point [18].

There is one parameter that allows triangulation which is called as Angle of Arrival (AOA) or Direction of Arrival (DOA.)

Angle of Arrival AOA is the angle "formed by the circular radius" from a reference station to a mobile station [31]. As mentioned in the previous paragraph, an AOA defines a straight line between two communicating stations. In order to locate a mobile station in two dimensions at least two intersecting lines (two AOA parameters) must be available (Figure 2.5.) A third AOA parameter allows positioning in three dimensions. Apart from requiring less number of reference stations (compared to trilateration methods,) an advantage of AOA based positioning is that it does not require reference stations to be synchronized [31].

Directional antennas or antenna arrays are essential parts of an AOA system [31]. A system incorporating an antenna array estimates this parameter by doing calculations based on arrival times of a signal at different array elements, and antenna geometry [18]. These indispensable components of AOA systems constitute a disadvantage because of their size and inherent complexity. Another downside is that positioning performance gets degraded as an observed mobile station moves away from its corresponding reference station. For example, if AOA of a mobile station that is 200 meters away from a measuring reference station is estimated as 46 degrees instead of 45 degrees, the error in position estimation will be around 4 meters (distance will be computed as approximately 204 meters.) However, when the mobile station moves 1000 meters away from the reference station in the same direction, an AOA error of 1 degree will cause 18 meters of positioning error (computed distance will be 1018 meters) [31]. There is no such problem in Time of Arrival, Roundtrip Time of Arrival or Time Difference of Arrival based systems [30].

AOA measurements are susceptible to multipath reflections; because, components of a signal which follow paths other than the actual one can mislead the antenna array into assuming that the signal is transmitted from another direction [31]. Most of the time, this is the case with wireless networks which suffer from the absence of line of sight (LOS) paths between communicating stations [30].

It is stated in [18] that increasing signal-to-noise ratio (SNR,) effective bandwidth, inter element spacing or the number of antenna elements improves AOA estimation accuracy. Also, [30] states that achieving accurate time estimations like Time of Arrival, Roundtrip Time of Arrival or Time Difference of Arrival is considerably easier than achieving accurate AOA estimations.

A positioning system can make use of multiple signal parameters that are explained so far to locate mobile stations in a wireless network. The motivation behind such hybrid schemes is that different signal parameters extracted from signals that are transmitted to or received from a mobile station provide more information about its location and allow better position estimation performance. For example; Time of Arrival and Angle of Arrival, or Time Difference of Arrival and Angle of Arrival can be used together by a system [18].

2.4.2 Mapping / Fingerprinting / Scene Analysis

Fingerprinting techniques estimate position of a mobile station by making use of previously collected information about signals. Before performing position estimations, in field investigations about certain signal parameters are done and stored into a database with corresponding location information [18]. Each piece of gathered data belonging to a known location is called as a fingerprint. Location of a mobile station is determined by comparing its signal parameters with records (fingerprints) in the database. Fingerprinting is currently used in most positioning systems and a common implementation is Received Signal Strength (RSS) based fingerprinting [31].

Fingerprinting techniques are developed to overcome measurement errors to some degree [31]. It is easy for geometric techniques to get affected by signal impairments negatively and deliver incorrect results. Fingerprinting strives to solve this issue by requiring a mobile station to measure signals at known locations before actually estimating position of any mobile station. The idea is that at the same location in an environment, all mobile stations will experience signal degradation at approximately the same amount because of that they all suffer from the same signal impairments taking place in that region. Therefore, investigations previously done in the coverage area of a wireless network give valuable information about what kind of signals to expect at different points, and the information is reliable since it takes the effects of signal impairments into account.

Fingerprinting based positioning consists of two stages which are offline and online (run time, real time, or training) stage. Offline stage is the phase where in field signal measurements are carried out to collect training data that will be used in online stage. As mentioned earlier, data about signals received from one or more reference stations are coupled with the investigating mobile station's location coordinates at that time instance, and stored into a database as a fingerprint record. After sufficient training data are gathered, actual position estimation process can be started which corresponds to the online stage. The database containing training data is queried with signal parameters of an observed mobile station, and fingerprints containing similar parameter values are evaluated to come up with a location estimate [31].

Training data can be expressed as

$$\mathcal{T} = \{(\mathbf{m}_1, \mathbf{l}_1), (\mathbf{m}_1, \mathbf{l}_1), \ldots, (\mathbf{m}_{N_T}, \mathbf{l}_{N_T})\}, \tag{2.1}$$

where $\mathbf{l}_i$ is the coordinates of a location where i^{th} signal measurement is done, $\mathbf{m}_i$ is the parameter array of the signals received at location $\mathbf{l}_i$, and N_T is the total number of fingerprints collected. If a mobile station is exploring an environment to create an RSS map for fingerprinting purposes, each $\mathbf{m}_i$ element contains RSS parameters of signals it receives from a number of reference stations, at location $\mathbf{l}_i$. In online stage, RSS parameters belonging to a mobile station to be located are interpreted according to certain position estimation rules, e.g. pattern matching algorithms, and a location $\mathbf{l}$ is produced as a result [18]. There is a variety of rules that can be used in fingerprinting systems. In this thesis project, k Nearest Neighbor (kNN) algorithm is employed.

Disadvantage of fingerprinting based positioning systems is that they require sufficiently large training databases which correctly map accurately estimated signal parameters to the investigated environments. Training databases may need to be updated periodically to take into account the variations in signal characteristics caused by environmental changes. This is a costly operation in terms of the time and effort involved, especially for wireless networks covering large areas [18].

2.4.2.1 k Nearest Neighbor

The simplest version of kNN algorithm returns coordinates $\mathbf{l}_i$ corresponding to the parameter array $\mathbf{m}_i$ whose Euclidean distance to the parameter set of the mobile station to be located is the shortest among all distances $||\mathbf{m} - \mathbf{m}_i||$ as shown below.

$$j = \arg \min_{i \in \{1,\ldots,N_T\}} ||\mathbf{m} - \mathbf{m}_i|| \tag{2.2}$$

General kNN formula takes into consideration the coordinates of k training database records, $\mathbf{l}^{(1)}$, $\mathbf{l}^{(2)}$, ..., $\mathbf{l}^{(k)}$, whose parameter vectors, $\mathbf{m}^{(1)}$, $\mathbf{m}^{(2)}$, ..., $\mathbf{m}^{(k)}$, have the smallest distance to the parameter vector of the tracked mobile station, $\mathbf{m}$. The position estimate, $\hat{\mathbf{l}}$, is computed as the weighted sum of those coordinates,

$$\hat{\mathbf{l}} = \sum_{i=1}^{k} w_i(\mathbf{m})\mathbf{l}^{(i)}, \tag{2.3}$$

where $w_i(\mathbf{m})$ represents the weighting factor for $\mathbf{l}^{(i)}$, and is calculated using parameter vectors $\mathbf{m}$ and $\mathbf{m}^{(i)}$. There are various weighting functions that can be utilized. For example, a kNN algorithm that makes use of the uniform weighting scheme basically computes the arithmetic mean of k coordinate vectors [18]:

$$\hat{\mathbf{l}} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{l}^{(i)}. \tag{2.4}$$

CHAPTER 3

RELATED WORK

In this chapter, position estimation techniques designed for single base station mobile networks are reviewed.

Porretta et al. [34] presented “a novel algorithm that makes use of a single-BS [single base station] antenna array to locate MTs [mobile terminals] in cellular networks.” The proposed location technique required a base station that allowed estimation of multipath channel (MPC) parameters including Time of Arrival (TOA) and Angle of Arrival (AOA). There was no need to modify mobile terminals for the technique to work; modifications were only to be applied to base stations. Location determination was performed according to the decision of whether the mobile terminal was in line of sight (LOS) of the base station or not. If the terminal was in LOS, then simple trigonometric computations were done in order to determine the location. If the terminal was in non line of sight (NLOS), then a cost function minimization was carried out. Time of Arrival and Angle of Arrival information were essential for both situations. In non line of sight case, some information about the environment in the BS neighborhood was also required which was called as “sentinel function.” As explained on the paper, a sentinel function “is defined as the Euclidean distance between the BS and the nearest [fixed] obstacle found along the azimuth direction identified by the angle α” (Figure 3.1.) Samples for the function were measured at small steps like 0.5 or 1 degrees, and all area covered by the base station was scanned.

The decision of line of sight or non line of sight was made according to a simple procedure. First, a distance measurement was done using Time of Arrival information of an MPC sent from a mobile terminal which was received at a base station. The distance was compared with the sentinel function defined by Angle of Arrival of the received MPC, and in case that the distance was not bigger than the sentinel function the mobile terminal was assumed to be in

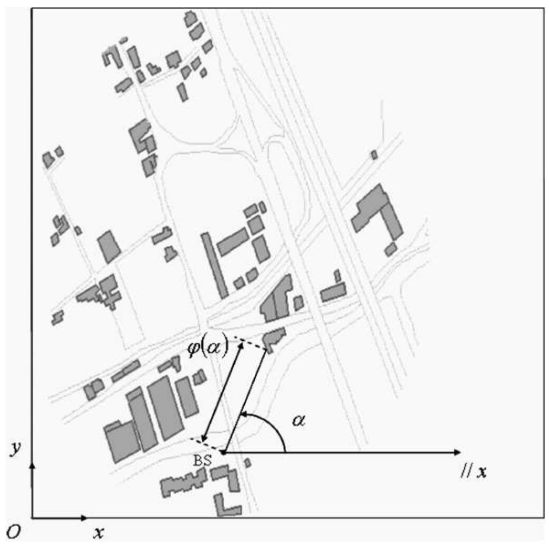


Figure 3.1: “Definition of the sentinel function $\varphi(\alpha)$ for a real urban scenario (a district of the town of Viareggio, Italy).” [34]

line of sight condition. In other words, if

$$d_1 = \sqrt{(c\tau_1)^2 - (h_{BS} - h_{MT})^2} \tag{3.1}$$

(where c is the speed of light, τ_1 is the absolute propagation delay, h_{BS} and h_{MT} are heights of the base station and the mobile terminal, respectively) was smaller than or equal to the sentinel function for α (Angle of Arrival) then line of sight was assumed and coordinates of the mobile terminal were calculated as

$$\hat{x}_{MT} = x_{BS} + d_1 \cdot \cos(\alpha_1) \tag{3.2}$$

$$\hat{y}_{MT} = y_{BS} + d_1 \cdot \sin(\alpha_1) \tag{3.3}$$

where $\hat{x}_{MT}$ and $\hat{y}_{MT}$ define coordinates of the base station.

Non line of sight condition was assumed if d_1 was bigger than the sentinel function. That meant a cost function was minimized in order to estimate the coordinates. In this case, multiple subsequent MPCs were taken into account. Scatterers along the Angle of Arrival, objects that caused MPCs to “reflect or diffract for the last time before reaching the BS”, were found for N MPCs, and their coordinates were computed using following formulas since their distance to the base station (sentinel function,) and Angle of Arrivals were known.

$$x_{Si} = x_{BS} + \varphi(\alpha_i) \cdot \cos(\alpha_i) \tag{3.4}$$

$$y_{Si} = y_{BS} + \varphi(\alpha_i) \cdot \sin(\alpha_i) \tag{3.5}$$

where x_{BS} and y_{BS} define coordinates of base station, $\varphi(\alpha_i)$ is sentinel function, α_i is Angle of Arrival, and $i = 1, \ldots, N$. Propagation delays between the mobile terminal and the scatterers were also computed using the formula.

$$\tau_{Ri} = \tau_i - \frac{\varphi(\alpha_i)}{c} \tag{3.6}$$

where τ_i is Time of Arrival. The cost function to be minimized for location estimation was defined as

$$F(x,y) = \sum_{i=1}^{N} p_i^2 f_i^2(x,y) \tag{3.7}$$

where p is a weighting factor, with

$$f_i(x,y) = c\tau_{Ri} - \sqrt{(x - x_{Si})^2 + (y - y_{Si})^2}. \tag{3.8}$$

Subsequently, coordinates of the mobile terminal were chosen as the (x,y) pair which minimized $F(x,y)$.

$$(\hat{x}_{MT}, \hat{y}_{MT}) = \arg \min_{(x,y)\in D} \{F(x,y)\} \tag{3.9}$$

where D is defined as

$$D = \left\{ (x,y) \mid \sqrt{(x - x_{BS})^2 + (y - y_{BS})^2} \le c\tau_1 \right\} \tag{3.10}$$

since the mobile terminal had to be at most $c\tau_1$ away from the base station.

As seen above, the defined cost function took N MPCs into account for each possible (x,y) pair. The pair that ensured the cost function to yield the smallest value was assumed to be the coordinates of the mobile terminal. What attracts attention here is that the algorithm made use of a Time of Arrival method without using multiple base stations. Time of Arrival information that should have been retrieved from multiple base stations was replaced with the information retrieved from multiple scatterers using only one base station.

One downside of the proposed technique was that performance degradation was observed in cases where multipath components reached base station after experiencing multiple reflections, diffractions, etc. The reason for this degradation was that the algorithm took into account only the last scatterers visited before reaching base station.

The proposed technique showed good performance in both line of sight and non line of sight environments according to the performance tests and complied with the United States

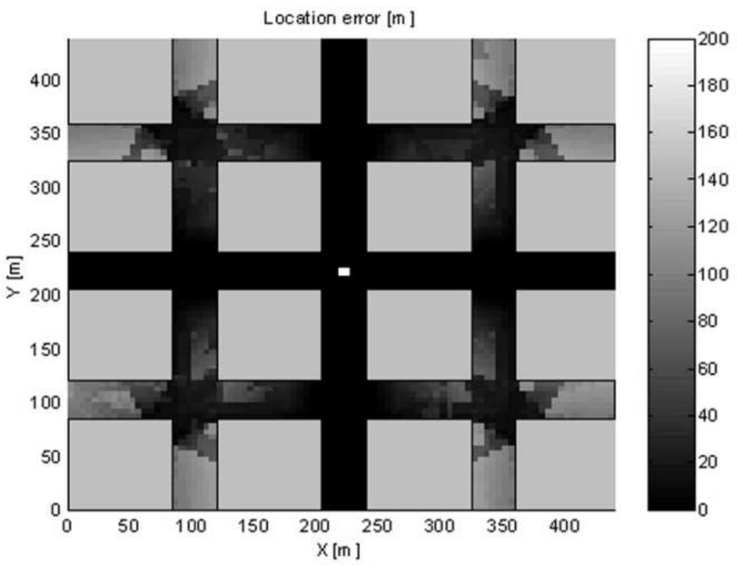


Figure 3.2: “16-building Manhattan environment with the BS located at the center of the scenario.” [34]

Federal Communications Commission (FCC) regulations of Enhanced 9-1-1 service [15] (which required an accuracy of 125 meters in 67% of location estimation trials at the time of writing.) In a 16 building Manhattan Environment (Figure 3.2) with a base station in the middle of the scenario environment, location error was recorded as 31 meters, and standard deviation was 40 meters. With the base station on a corner of a building which was in the middle of the scenario environment, location error increased to 54 meters, and standard deviation increased to 68 meters. In a 64 building Manhattan Environment with the base station in the middle of the scenario environment, location error was 33 meters, and standard deviation was 45 meters. Finally, in a district of the town of Viareggio, Italy, location error was recorded as 34 meters, and standard deviation was 93 meters.

Zhaounia et al. [49] used Time of Arrival (TOA,) Angle of Arrival (AOA,) and scatterer information like Porretta et al. [34] in order to locate a mobile terminal (MT.) The scattering model that was used is shown in Figure 3.3.

While in reference [34] initially it was decided whether the observed mobile terminal was in line of sight (LOS) or non line of sight (NLOS) with its associated base station, in reference [49] line of sight was considered for the beginning and following iterations handled accuracy issues for non line of sight cases.

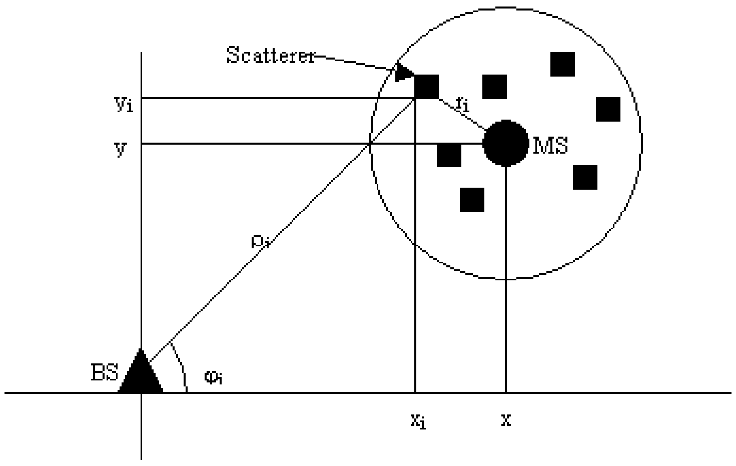


Figure 3.3: Scattering model used in reference [49]. Image courtesy of Mohamed Zhaounia.

The proposed method was initialized with the estimation of location using the following formulas (similar to Equation 3.2 and Equation 3.3) which were applicable for line of sight situations.

$$x = \frac{1}{N}\sum_{i=1}^{N} \delta_i \cos(\varphi_i) \tag{3.11}$$

$$y = \frac{1}{N}\sum_{i=1}^{N} \delta_i \sin(\varphi_i) \tag{3.12}$$

where δ_i is the i^{th} non line of sight range parameter calculated from the i^{th} Time of Arrival measurement, N is the number of resolved multipaths, φ_i is the Angle of Arrival of the signal impinging on the base station (BS) belonging to the i^{th} scatterer, and x and y are coordinates of the mobile terminal.

For estimation of location in non line of sight situations, more information was needed to be incorporated into the estimation process. Therefore, information about scatterers around the mobile terminal was also taken into consideration which would be used for improving accuracy. One thing to note here was that some offline processing was required in order to obtain “a reasonable accurate characterization of the scattering environment”, which may remind the reader of fingerprinting technique.

After computation of the initial coordinates, second step was to estimate the scatterers’ coordinates around the mobile terminal using an equation that made use of the initial coordinates, δ_i, and φ_i.

"Joint pdf [probability density function] of the vector consisting of all the distances r_i" between the MT and scatterers "given $\theta = [x, y]^{\mathrm{T}}$," $f\left([r_1 \ldots r_N]^{\mathrm{T}} | \theta\right)$, was used as the base of the third step. "Maximum Likelihood estimate" was explained as "the solution that minimizes the quantity" $J = \sum_{i=1}^{N} r_i$.

Above quantity simply stated that closest scatterers to the mobile terminal were sought. Following this, a system of equations was obtained by "setting the gradient of J with respect to θ to zero." By applying a Least Squares estimator to this system of equations, the mobile terminal location estimate was deduced in the third step. After third step, second step was revisited, therefore above procedure was executed in an iterated manner.

Experimental studies showed that the proposed technique provided "a performance enhancement of about 40%" compared to line of sight localization technique explained in Equation 3.11 and Equation 3.12.

Bishop et al. [7] proposed a localization technique that made use of "sensor fusion," that is, both Direction of Arrival (DOA) estimations and Received Signal Strength Indicator (RSSI) measurements.

Direction of Arrival estimations required utilization of estimation algorithms using antenna arrays, and provided "angle of the target relative to the base station." However, for Direction of Arrival-only localization, estimations from multiple base stations were needed to be collected. On the other hand, methods which only relied on RSSI measurements were found to be "prone to high uncertainties." It was thought that a single base station method which used RSSI measurements could be improved with Direction of Arrival estimations since it could help reducing "uncertainty in the movement of the mobile target." Therefore, sensor fusion was decided to be exploited.

A Robust Extended Kalman Filter (REKF) was employed for localization purposes as well as for tracking mobile terminals. The proposed technique relied on measurements taken at a single base station as with previously mentioned techniques, and was based on online state estimations derived via Robust Extended Kalman Filter.

To test the performance of the "sensor fusion based localization estimator using the REKF," a simulation environment was prepared which included a base station and a moving mobile terminal. RSSI measurements and Direction of Arrival estimations were provided by the base station, and there was no information about “maneuverability” of the mobile terminal. Experiments which were performed in a 10x10 km suburban area showed that mobile terminal location was “estimated to a relatively high accuracy.” Through observations of “error in location estimation,” it was also shown that sensor fusion based localization yielded better results than RSSI-only localization. It was stated that “presence of high uncertainties in the measurement noise” caused “localization to within approximately 500 m” to take 10 minutes “over a tracking range of approximately 12 km” with the proposed technique.

Apart from [34], Porretta et al. proposed another method [33] which concentrates on tracking mobile terminals (MT) using subsequent position estimations produced by an already existing “Single Base Station Positioning (SBSP) method designed for microcellular and third generation (3G) wireless communications systems.” The method also provided velocity estimations of mobile terminals. Tracking and velocity estimation process basically started with receiving position estimations of a mobile terminal from an “accurate location technique” “at fixed time intervals.” After collection of “raw data,” “a linear regression setup” [21] was utilized for processing the data in an “adaptive” fashion. Using a number of previous position estimations, it was decided whether the mobile terminal was moving “fast” or “slow.” In case of a “slow” mobile terminal, “the inertia of the proposed algorithm” was increased, leading to accurate position estimation. In case of a “fast” mobile terminal, the inertia was decreased, meaning that the mobile terminal could be “suitably tracked.” The algorithm was repeated each time a new position estimate was made available by the chosen estimation technique.

“A well known, linear regression based, smoothing algorithm [21]” formed the base of the presented method. While $\underline{x}(t_i) = [x_1(t_i), x_2(t_i)]^T \in \mathbb{R}^2$ denoted the actual location of a mobile terminal $\underline{\hat{x}}(t_i) = [\hat{x}_1(t_i), \hat{x}_2(t_i)]^T$ denoted the estimated location that was provided by a chosen positioning algorithm, where $i \in \mathbb{N}$ and $t_i = i \cdot \Delta t$. It was deemed suitable to apply smoothing to $\underline{\hat{x}}(t_i)$ in order to discard negative effects of erroneous estimations. It was explained that “the actual track followed by the user can be approximated more closely by smoothing the data $\underline{\hat{x}}(t_i)$.” The position of the tracked mobile terminal was approximated by

$$\underline{\tilde{x}}(t_i) = \left[t_i \cdot \underline{\hat{a}}(t_i)\right] + \underline{\hat{b}}(t_i) \tag{3.13}$$

“where $\underline{\hat{a}}(t_i)$ is the estimated constant speed vector and $\underline{\hat{b}}(t_i)$ is the estimated MT position at time $t_0 = 0$.” Here, last k position estimations were used to compute the time dependent coefficients, $\underline{\hat{a}}(t_i)$ and $\underline{\hat{b}}(t_i)$, according to a “linear regression setup” [21]. As explained by Porretta et al. [33] the sequence given by Equation 3.13 “yields a smoothed estimation for the track followed by the user.” Differential of the 3.13 also yielded time dependent velocity of the mobile terminal.

$$\tilde{v}(t_i) = \left\| \underline{\hat{a}}(t_i) \right\| \tag{3.14}$$

Further smoothing of mobile terminal position was provided by utilizing its velocity. It was known that the terminal could not move more than $\Delta t \cdot v_{max}$ within a time interval of Δt, where v_{max} is the maximum possible velocity. It was also known that the value of v_{max} could be different depending on how mobile the terminal was. For example, v_{max} of a walking man and v_{max} of a driving man could not be the same. Therefore, to provide better positioning accuracy for different types of mobility schemes, an adaptive approach was employed. v_{max} was reduced for mobile terminals labeled as “slow” and increased for mobile terminals labeled as “fast.” In order to decide whether the terminal was moving “fast” or “slow,” following criteria was specified.

$$v_{max}(t_i) = \begin{cases} v_{max}^{fast} & \text{if } \bar{\tilde{v}}(t_i) > v^* \\ v_{max}^{slow} & \text{if } \bar{\tilde{v}}(t_i) \leq v^* \end{cases} \tag{3.15}$$

where $\bar{\tilde{v}}(t_i) = (1/k) \cdot \sum_{j=i-k+1}^{i} \tilde{v}(t_j)$ and v^* is a threshold value. The criteria basically checked whether the average of “the last k MT velocity estimations” was bigger than the given threshold or not. If the average velocity was bigger than the threshold, the mobile terminal was assumed to be “fast” and assigned to a maximum velocity of v_{max}^{fast}. If the average velocity was smaller than the threshold, the terminal was assumed to be “slow” and assigned to a maximum velocity of v_{max}^{slow}. Following this, below equation was obtained.

$$\underline{\bar{x}}(t_{i+1}) = \begin{cases} \underline{\hat{x}}(t_{i+1}) & \text{if } \left\| \underline{p} \right\| \leq r_{max}(t_i) \\ \underline{\tilde{x}}(t_i) - \frac{r_{max}(t_i)}{\left\| \underline{p} \right\|} \cdot \underline{p} & \text{otherwise} \end{cases} \tag{3.16}$$

where $\underline{p} = [\underline{\tilde{x}}(t_i) - \underline{\hat{x}}(t_{i+1})]$, and $r_{max}(t_i) = v_{max}(t_i) \cdot \Delta t$ is the radius of the circle centered at the approximated position, $\underline{\tilde{x}}(t_i)$. This equation implied that the next position of the mobile terminal had to be at most $r_{max}(t_i)$ away from the current position. “The position of the MT at time t_{i+1}” then could be “estimated through the regression approach” explained in the paper “using the last k values” obtained from Equation 3.16. As seen from the above equation, $v_{max}(t_i)$ was the value that managed the inertia of the algorithm. When $v_{max}(t_i) = v_{max}^{fast}$ the inertia was decreased which allowed the algorithm to “suitably” track “fast” users while the inertia was increased when $v_{max}(t_i) = v_{max}^{slow}$ so that “slow” users could be tracked “accurately.”

A disadvantage of the presented method was that it was “adapted to, at least locally, straight motion” since it was “based on a linear regression setup.” The method was analyzed using “two different microcellular scenarios.” First, a simulation was carried out using “data from a real urban scenario, namely a district of the town of Viareggio, Italy” where position estimation performance was degraded initially because of non line of sight (NLOS) propagation of signals in the beginning of the path followed by the mobile terminal. This non line of sight situation was stated to represent “the worst case for performance assessment of a tracking algorithm.” On the other hand, the performance was improved when the mobile station was in line of sight (LOS) with its associated base station (BS.) “A mean value of 18 m and a standard deviation of 29 m” were obtained for the “location error.” Performance of velocity estimation was also evaluated, and showed “an error with a mean value of 2.16 m/s and a standard deviation of 2.57 m/s.”

Another simulation was performed in a “16-building Manhattan-like environment” (similar to the one in Figure 3.2.) In this scenario, non line of sight condition was experienced in most of the followed path which caused performance degradation in position estimation as expected. Mean value of the location error was measured as 30 m with a standard deviation of 15 m. Evaluation of velocity estimation revealed “an error with a mean value of 2.15 m/s and a standard deviation of 2.36 m/s.” It should be noted that performance of the proposed method partially relied on the performance of its underlying position estimation algorithm.

Yang and Liang [39] proposed a method that makes use of scatters around a mobile terminal (MT) like [34] and [49]. The method presented a solution to the problem of single base station (BS) positioning by utilizing a “maneuverable BS” unlike other methods which assume stationary base stations. This allowed to measure parameters, which were used in positioning, at multiple reference points (base station positions) as the base station moved which resembled measurements carried out with multiple stations. Therefore, the method turned the single base station location into multiple base station problem, and eased the process of estimation.

The scatter distribution model assumed that scatters were distributed in a circular area with a radius of R_s around the mobile terminal from which they were emitted (Figure 3.4.) Signals sent from the terminal to its associated base station were reflected by the scatterers. Angle of Arrival (AOA, α_i) and Time of Arrival (TOA, τ_i), as well as position of the base station, were known. Distance travelled by a scattering signal was equal to sum of the distance between the

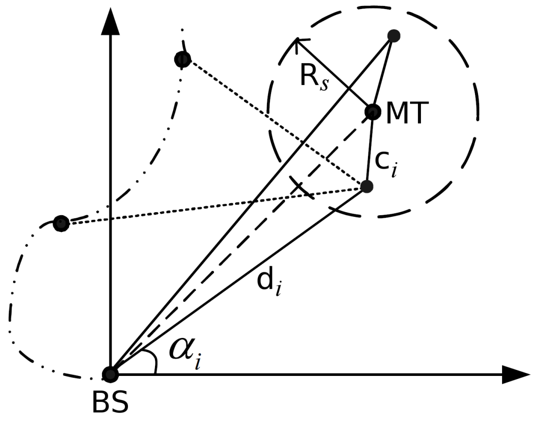


Figure 3.4: "The single reflection model for the SSL [single station location]." [39]

mobile terminal and its i^{th} scatterer, c_i, and the distance between between the scatterer and the base station, d_i, that is

$$r_i = C \cdot \tau_i = c_i + d_i \tag{3.17}$$

where C is the speed of light. Position of a scatterer was computed according to a "circle fitting algorithm" using known parameters, r_i, α_i, and position of the base station. That allowed calculation of d_i. Because of that the base station was moving, information about multiple scatterers could be collected. Once scatterer positions were known, it was easy to estimate the mobile terminal's position. As seen in Figure 3.5, scatterers labeled with S_1, S_2, and S_3 could be thought of as base stations of their own, that is, as "virtual base stations." Since distances between the terminal and "virtual base stations" could be calculated using the formula $\hat{c}_i = r_i - \hat{d}_i$ the setup in Figure 3.5 could be treated as a multiple base station case.

A simulation of the presented method was conducted with 3 scatters and a scattering radius of R_s=300 meters. Measurements revealed a root mean square error (RMSE) of above 200 meters when 40 measurement points were used which fell below 50 meters with 80 and more measurement points. Another simulation investigated the impact of different number of scatters and R_s. The results showed a decrease in RMSE with increasing number of scatters. RMSE was also decreased when smaller R_s was used.

Although it allowed conversion of single base station problem to multiple base station

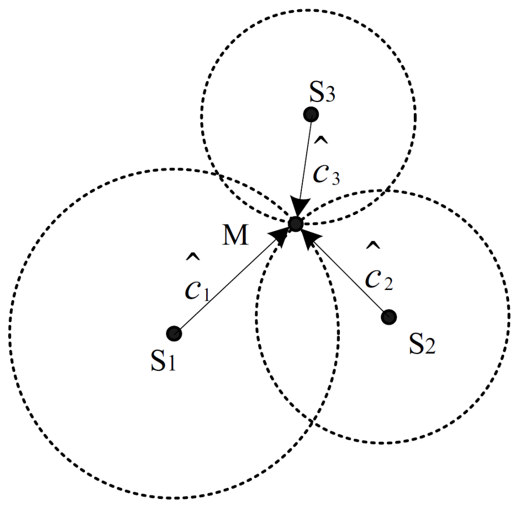


Figure 3.5: “Mobile station location in the multi station location system.” [39]

problem and simplified the estimation method, the maneuverability requirement of base stations may be counted as a downside since those are generally stationary equipments.

CHAPTER 4

SYSTEM MODEL

This chapter gives information about the WiMAX network infrastructure, portable WiMAX hardware and DGPS equipment employed in this thesis. Problems that were directly related to the WiMAX equipment are described, and their effects on development of the proposed position estimation algorithm are explained.

4.1 Equipment and Software

Measurement equipment for this research project consisted of

- WiMAX infrastructure including one base station at the Department of Computer Engineering in Middle East Technical University which was granted by Intel, provided and installed by ZTE,
- notebooks with Intel WiMAX adapters granted by Intel, and
- professional DGPS equipment borrowed from Geodetic and Geographic Information Technologies Division in Middle East Technical University.

WiMAX infrastructure consists of two cabinets as depicted in Figure 4.1. The one on the left contains indoor baseband unit of the base station (to which cables coming from antennas on top of the Computer Engineering building are connected) and wireless access gateway; the one on the right contains Authentication Authorization Accounting (AAA) servers and firewall. From multiple profiles for Access Service Network (ASN), each calling for a different decomposition of functions within ASN (to base station and ASN gateways),

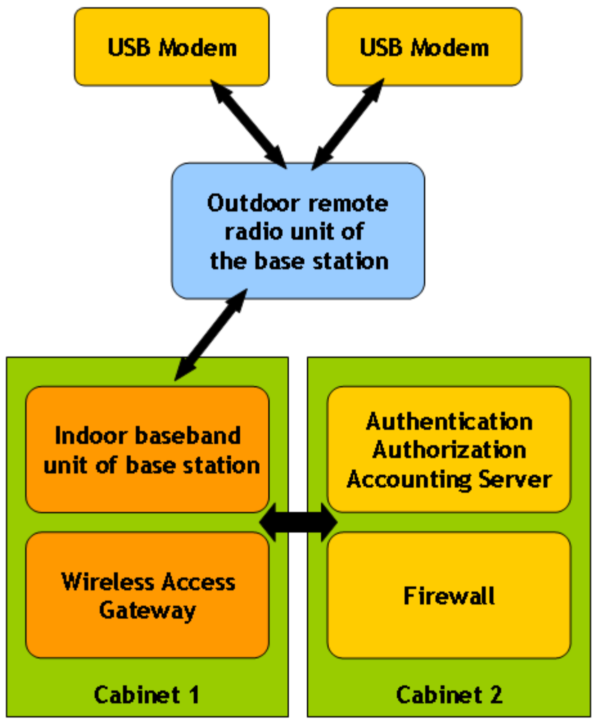

USB Modem
USB Modem
Outdoor remote radio unit of the base station
Indoor baseband unit of base station
Wireless Access Gateway
Cabinet 1
Authentication Authorization Accounting Server
Firewall
Cabinet 2

Figure 4.1: General view of existing WiMAX system at the Department of Computer Engineering in Middle East Technical University

infrastructure in the department makes use of Profile C which splits the functions between the base station (where radio resource management is implemented) and the ASN gateway.

There are two sector antennas one of which operates at 3.5GHz and covers a 90 degree range in the direction of Food Engineering Department while the other one operates at 2.5GHz and covers a 90 degree range in the direction of Culture and Convention Center. The antennas are inclined downwards in order to receive signal inside the building. The infrastructure complies with IEEE 802.16e specification, providing support for handoff and roaming as well as fixed services. WiMAX signal can be received line of sight (LOS) from 8th dormitory although it is not in the exact range of the antenna pointing to that direction. According to ZTE's tests, coverage range of the system should be more than 1.1 km at 2.5GHz and 2.23 km at 3.5GHz [51]. Also, indoor test results show that at 2.5GHz, data rate inside various rooms of building A was 10.3 Mbps (with the exception of two rooms out of fifteen, where data rate fell down to 5 and 6 Mbps), and inside various rooms of building B it ranged from 5 Mbps to 15 Mbps. Using outdoor CPE, data rates of 3.5 Mbps in first floor hall of rectoria and 5 Mbps in third floor meeting room were achieved at 3.5 Ghz [50].

Two people are in charge of the infrastructure. Research Assistant Alper Kılıç is in charge of operation of the infrastructure, and Dr. Cevat Şener is in charge of managerial work.

Notebooks granted for research purposes were coupled with Intel WiMAX/Wi-Fi Link 5150 adapters, supporting IEEE 802.16e and 802.11a/b/g/Draft-N standards while operating at 2.5GHz spectrum for WiMAX and 2.4GHz and 5.0GHz spectra for Wi-Fi [28]. Official Linux drivers developed by Intel [29] were installed on the notebooks. DGPS equipment which consisted of a main unit, an external antenna, an external battery and an external Bluetooth transceiver was said to be achieving up to 10 cm accuracy levels. During operation, the equipment transferred location information to a Pocket PC over Bluetooth which was saved into a file using a mobile Geographic Information System (GIS) application called MakroPAD developed by Mehmet İlvan.

Apart from the equipment mentioned above, a couple of desktop applications were utilized throughout the experiments. MakroMap Pro 3.2 is the desktop companion of MakroPAD which was used to visualize and export coordinate data produced by the DGPS equipment. Quantum GIS [3], which is an open source GIS, was also used to visualize coordinate data with corresponding signal measurements in detail to identify erroneous DGPS readings.

For the programming part, the Microsoft .NET Framework (version 2) [32] was used to develop simulation of the proposed positioning system in C# language. In collaboration with the framework, MATLAB R2007b was employed to make use of available clustering code which was needed for position estimation, and the link between the framework and MATLAB was established by MATLAB Builder for .NET component [38].

4.2 Equipment Problems

Before doing any actual measurements, data collected by both WiMAX adapter and Differential Global Positioning System (DGPS) were examined in order to see what kind of information could be received. As expected, DGPS equipment provided all information about position including coordinates and GPS time.

On WiMAX side, initially it was expected to see IEEE 802.16e Medium Access Control

(MAC) layer headers of incoming and outgoing packets so that a number of fields that would come handy in position estimation process could be extracted.

On Microsoft Windows operating system, it was possible to see Received Signal Strength Indicator (RSSI,) Carrier to Interference-plus-Noise Ratio (CINR,) and Average Transmit Power (AvgTxPwr) from Intel's own WiMAX software [26]; however, parameter values were only printed to screen. Therefore, the data could not be written into a text file or database for later use.

For analyzing IEEE 802.16e MAC headers, Wireshark [48], a network protocol analyzer supporting WiMAX packet demodulation, was used to capture packets flowing through the adapter in Windows. The program successfully captured incoming and outgoing packets; however those were identified as Ethernet II packets instead of IEEE 802.16e packets. An Ethernet II packet only contains two MAC addresses (one for destination, one for source) and a field called EtherType which defines the protocol used in payload of a MAC frame [1, 35]. That meant there were seemingly no IEEE 802.16e specific data in the data packets sent and received by the WiMAX adapter.

After seeing that information could not be gathered in a useful form in Windows, attention was turned to Linux. Official Linux drivers of Intel WiMAX devices provided from [29] were installed and investigated in order to find some information that would be utilized in positioning. In fact, same signal parameters as in Windows (RSSI, CINR, AvgTxPwr) were shown to user when the command used for connecting to a WiMAX network was invoked with certain options. Although these parameters would be useful, it was still thought that information contained in IEEE 802.16e MAC headers were necessary for better positioning. Therefore, packet capture was repeated using Wireshark in Linux. Unfortunately, all captured packets were identified as Ethernet II packets again.

Normally, IEEE 802.11 (Wi-Fi) packet capture resulted the same way WiMAX did. However, in Linux, Wi-Fi adapters could be set to work in "monitor mode" which allowed capturing of "management packets." It was thought that if WiMAX adapter could also be set to work in monitor mode, then it might capture management packets which contained data that would be used for positioning. An e-mail was sent to official Linux WiMAX Mailing List[1] and it was

[1] wimax@linuxwimax.org

asked how to set the adapter to work in monitor mode.[2] A reply to the e-mail was sent by Inaky Perez-Gonzalez[3], who was a responsible for Linux drivers of Intel's WiMAX adapters, and it was stated that "The Intel device's firmware stripes the MAC headers and only delivers to the host IP packets, so there is no way to access them." Since reason of the problem was the adapter's firmware, only solution could be loading the adapter with a new firmware which would not cut off MAC headers before handing packets on to the operating system. An e-mail was sent to Inaky Perez-Gonzalez and it was asked whether Intel could provide another firmware for research purposes if an individual request was made. In reply to that mail, it was explained "I very much doubt it. The firmware team is extremely busy; they have a hard time servicing our own requests, so servicing external requests is even more difficult." In another reply posted by Mr. Jayant[4] it was stated "There are no definite/known plans to have the firmware do that." The reason for such a firmware restriction was explained as "a resource problem" by Inaky Perez-Gonzalez. It was stated that there were no adapters which provided MAC headers to his knowledge.

The problem was discussed with chair of Department of Computer Engineering, and it was decided to contact Intel and ZTE, companies who granted and installed the WiMAX infrastructure in the department, respectively. Proper firmware and drivers for the granted adapters were requested from Intel, and a WiMAX Universal Serial Bus (USB) adapter which could dump packets and signal information into a file was requested from ZTE.

Companies, most of which had products certified by Wimax Forum [46], were also contacted to find out whether their products could provide detailed information about packets, including Airspan Networks[5], Beceem[6], Fujitsu[7], Mitsumi[8], Modacom[9], Motorola[10], NEC[11], Samsung[12], WiMax Network Solutions, Inc.[13], and Zyxel[14]. Replies were received from Fujitsu, Modacom, Motorola, and Zyxel all of which were negative. The most

[2] The e-mail that was sent can be accessed from `http://www.mail-archive.com/wimax@linuxwimax.org/msg00639.html`
[3] inaky@linux.intel.com
[4] jayant@linux.intel.com
[5] aweiner@airspan.com
[6] info@beceem.com, aagrawal@beceem.com
[7] kuroda.takahiro@jp.fujitsu.com
[8] otani.koji@mitsumi.co.jp
[9] thkim@modacom.co.kr
[10] greg.mcgee@motorola.com
[11] sugimoto@necat.nec.co.jp
[12] asiwish@samsung.com
[13] usa@wimaxns.com
[14] Rachel.Chen@zyxel.cn

satisfactory reply came from Teny Kim[9] (Modacom) who actually summarized the situation: “In my opinion, all WiMAX products could support your needs. Because development engineers need debugging messages for IOT, device should support or have this function. But, these messages have some critical information of manufactures (technical know how and their own skill, etc.) Then this functionality cannot be supported for end user.”

Eventually, no adapter was provided to the Department by Intel, ZTE, and individual contacts. Accordingly, only RSSI, CINR, and AvgTxPwr were left to be used for positioning.

4.3 Non accessible Parameters

In WiMAX networks, in addition to parameters like Time of Arrival (TOA,) Angle of Arrival (AOA,) Received Signal Strength (RSS,) etc. a MAC layer field called Timing Adjust (TA) can be used for positioning [8]. Timing Adjust is similar to Timing Advance (previously mentioned in Section 2.4.1.1) in GSM networks which is determined by base stations (BSs) and sent to mobile stations (MSs) for adjusting timing of transmissions in the uplink direction (from mobile station to base station.)

Timing Adjust is a 32 bit signed integer which defines “The amount of the time required to adjust [subscriber station] SS transmission so the burst will arrive at the expected time instance at the BS.” [25] Sign of a Timing Adjust value commands a mobile station to either advance or delay its transmission time. Timing Adjust units depend on the employed physical layer. For single carrier physical layers (PHY) of WirelessMAN-SC and WirelessMAN-SCa, Timing Adjust units are set as $\frac{1}{4}$ and $\frac{1}{32}$ modulation symbols, respectively. For Orthogonal Frequency Division Multiplexing (OFDM) or Orthogonal Frequency Division Multiple Access (OFDMA) based PHY (WirelessMAN-OFDM and WirelessMAN-OFDMA) units are set as $\frac{1}{F_s}$ where F_s is sampling frequency [23]. Timing Adjust provides more fine grained range measurements compared to Timing Advance, each increment of which corresponds to a long distance of 554 meters [12].

A mobile station is informed about its Timing Adjust via a ranging response packet which is one of the management packets defined in IEEE 802.16 specifications. Since ZTE branded WiMAX adapters were able to capture and log management packets with their MAC headers, it would be possible to investigate and utilize encapsulated Timing Adjust information for positioning purposes. In addition to that, MAC headers could reveal more information about parameters that were dynamically controlled, e.g. modulation technique in use, as a mobile station moved around in coverage area. Availability of more parameters was likely to increase the location ability of a single base station; however, lack of necessary network equipment disallowed to extract such information from received packets.

CHAPTER 5

SIGNAL MEASUREMENTS

The position estimation algorithm proposed in this thesis requires a training database to be created in the first place. This database should consist of signal parameters measured at known locations in the coverage area of the employed WiMAX base station. A technique called fingerprinting is utilized in order to collect such signal parameters. In this chapter, investigated part of the coverage area is presented, and details about signal measurements and fingerprinting process are given.

Measurements were carried out between July 2nd, 2009 and July 26th, 2009 in Middle East Technical University campus (Figure 5.1). The difference between the smallest and the biggest latitudes visited during measurements were approximately computed as 904 meters (between green and red circles,) and the difference between the smallest and the biggest longitudes were approximately computed as 1185 meters (between yellow and blue circles.) In Figure 5.1 end points of the measured area can be seen. Green circle is where Civil Engineering building is, blue circle is where A4 Gate is, and red circle is where Faculty of Economic Administrative Sciences I building is. Yellow circle shows the location of WiMAX base station (BS) which is built on top of Computer Engineering building.

The points shown in 5.1 were not defined arbitrarily. Full circles actually show the places beyond which it was not possible to receive signals from the base station (pink one is where bus station in dormitory area is.) Considering the fact that base station antenna was originally a 90 degree sector antenna and was pointed towards the stadium, it was interesting to see that the signal could be received practically in a 180 degree area.

The 1185×904 m^2 area was not examined completely as it would take a certain amount of time. Timberlands were hard, insecure and unnecessary to examine; because terrain was

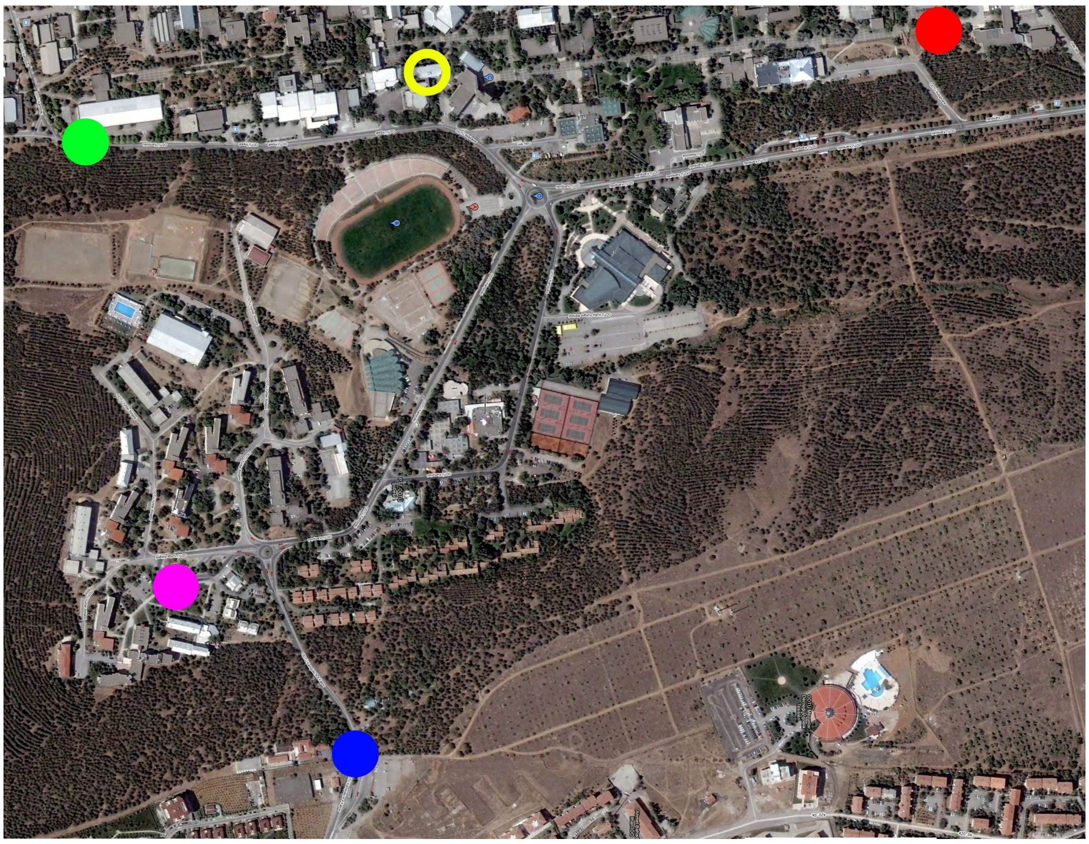


Figure 5.1: A part of Middle East Technical University campus which is in coverage of WiMAX. ©2010 Google - Imagery ©2010 DigitalGlobe, GeoEye, Map data ©2010 Başarsoft [19, 10, 17, 5].

rough, practically people were not going there, and GPS measurements would not yield accurate information since trees degraded or completely blocked signal reception. Doing measurements in diverse areas where residents of the campus were mostly passing through was thought to be a better idea since it would take less time, represent real life situations (which is important because a real life positioning case is being studied,) and still allow experimentation, demonstration and understanding of ideas expressed in the thesis. Areas where measurements were done are shown with transparent green in Figure 5.2.

As seen in Figure 5.2, main roads which are dispersed nicely and extend both horizontally and vertically in the area were almost completely investigated (Green line in left of the red circle is only used by people.) Roads were important; because, it would be possible to investigate behavior of signals over wide areas by doing measurements on roads. On the other hand, places like stadium and quasi rectangular regions in its vicinity were examined in order to

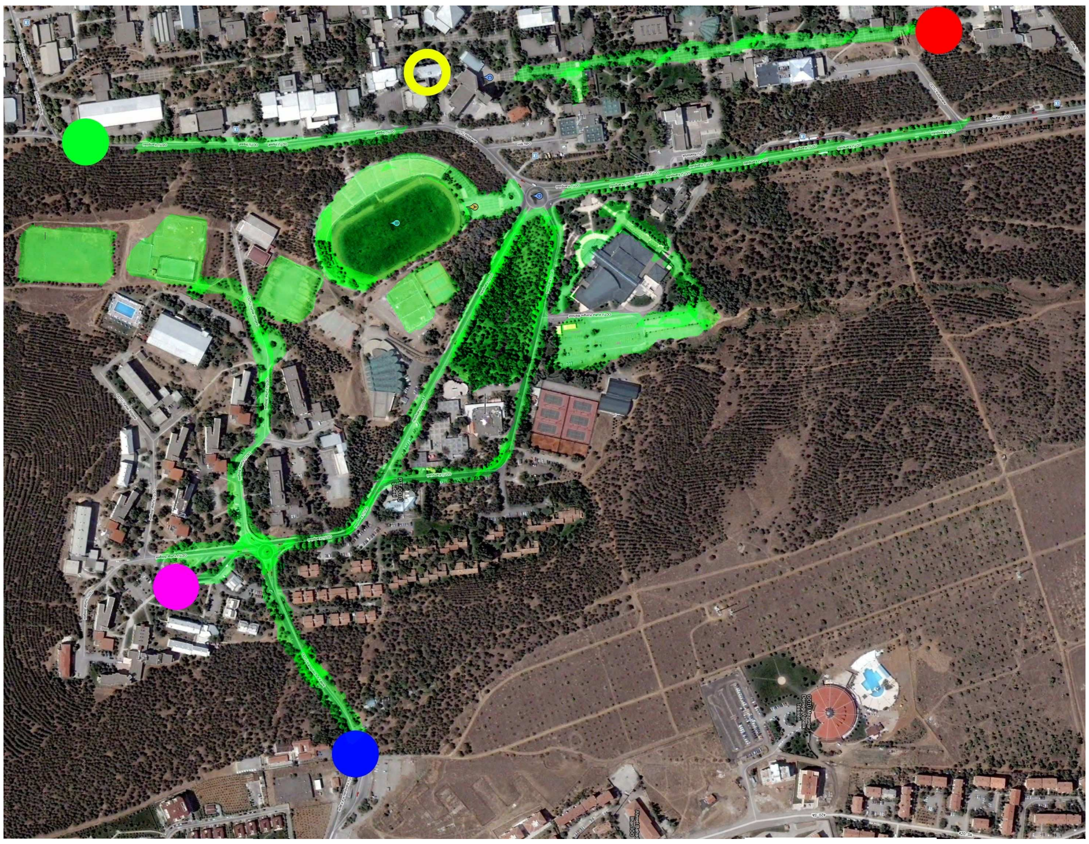


Figure 5.2: Areas where WiMAX and GPS measurements were done. ©2010 Google - Imagery ©2010 DigitalGlobe, GeoEye, Map data ©2010 Başarsoft [19, 10, 17, 5].

investigate signal behavior in a single region which did not extend over a wide area, e.g. the whole campus. Note that green lines generally show open areas where DGPS could provide accurate location information although problems still occurred in woody region in the middle of the map as well as on roadsides overshadowed by trees.

A notebook with a WiMAX adapter, and wearable DGPS equipment were used for measurements. The notebook was carried on a back pack. Since there were no connection between the DGPS unit and the notebook, two measurements were done separately at the same time. Therefore, it was necessary to combine collected WiMAX and GPS data for later use. In both WiMAX and GPS, time was stored each time a new measurement entry (a fingerprint) was made. So, time could be used as an attribute according to which entries from the two measurements could be matched. One thing to consider here was that GPS time was different from the local time. Because of that, before each session, difference between the notebook's time and GPS time was recorded. The difference was taken into account as an

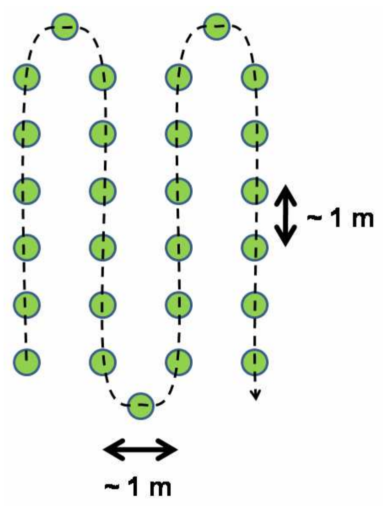


Figure 5.3: Measurement areas were completely scanned by walking in parallel lines.

offset value throughout the matching process. It was seen that the difference was changing from exactly 3 hours to 3 hours and 13 seconds.

For WiMAX, it was initially thought that fingerprints taken every 5 to 10 meters would be enough for positioning. However, after it was reconsidered that wireless signals were affected by signal impairments like attenuation, multipath, refraction, fading, etc. [36] which caused erroneous measurements [31], it was decided fingerprints could be taken approximately every 1-2 meters. Such dense measurements would allow reducing undesirable effects of erroneous fingerprints to a certain degree. That was going to be realized by averaging fingerprints [18] in small areas, e.g. 5×5 m^2 squares, and using the resulting values in positioning. That would surely decrease the number of fingerprints in offline database (considering the number of fingerprints collected every 1-2 meters); but, it would still provide enough data, close to the number of entries that would be yielded via measurements done using 5-10 meter intervals. The tradeoff was doing more time consuming and exhausting work for the sake of having more accurate WiMAX signal measurements, which was worthwhile. Measurement areas were completely scanned by walking in parallel lines as illustrated in Figure 5.3, where green dots correspond to fingerprints.

As mentioned before, there could also be errors in GPS positions because of trees and buildings. To handle these errors, after each session GPS data were checked with Quantum GIS [3], Geographical Information System software. Collected data were used as input for

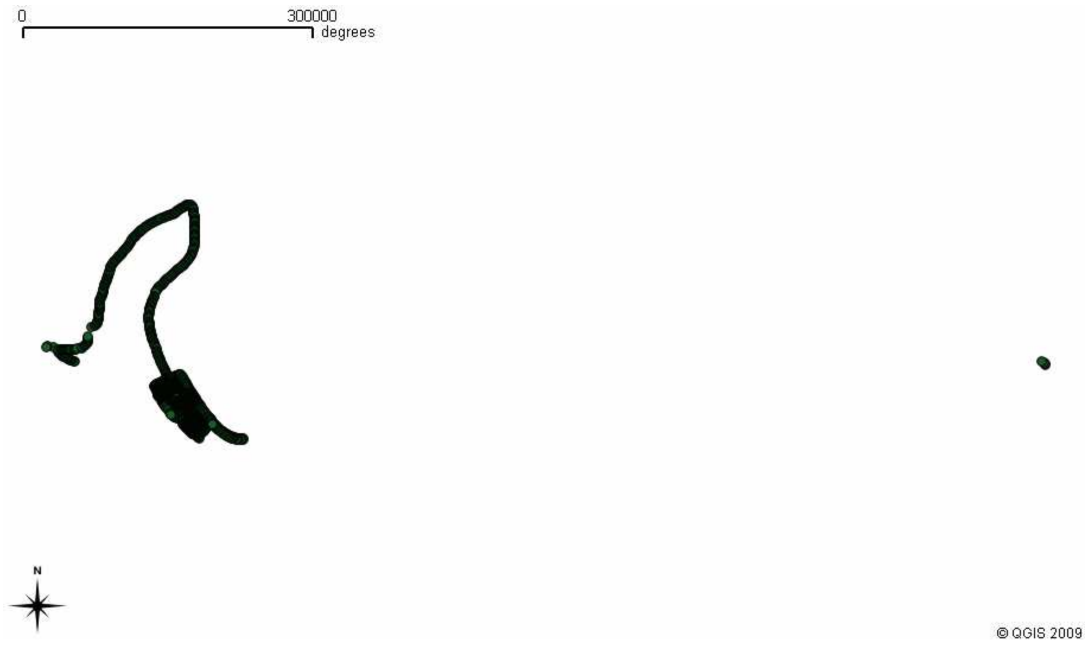


Figure 5.4: An example to erroneous GPS measurements. Points in the left show correct GPS readings and points in the right, which are far away from correct measurements, show erroneous measurements.

Quantum GIS using “Delimited Text” plug in, and visited points were shown in a graphical map. Considering the way the points were grouped in the map, and knowing what kind of a map to expect; it was easy to see whether there were errors or not. An example is given in Figure 5.4 where points in the right side of the map are results of erroneous measurements.

A part of erroneous GPS points were removed during combination process of WiMAX and GPS data using a C program. After that, a map was created again, and it was checked if there were still erroneous GPS data left. In order to remove those entries Microsoft Excel was chosen; because, it was required to examine data by eye. When an erroneous point was found, a simple macro was written to find other points similar to that one, and line containing that point was erased. Erroneous points consisted of latitude, longitude, and altitude values which were out of expected ranges.

As mentioned before combination of WiMAX and GPS data were handled using a program written in C which used time of entries to synchronize and match two measurements. Resulting file was written into a text file. An example of combination process is given in Tables 5.1, 5.2, and 5.3. Table 5.1 contains sample GPS data which includes row ID, coordinate values, GPS time, and normalized GPS time. Normalized GPS time is calculated

Table 5.1: GPS Measurement Sample

ID	Coordinates			GPS Time	GPS Time + Time Difference
1	4415635.321003	482214.963558	940.050000	03:07:37	06:07:37
2	4415634.601803	482214.007175	939.750000	03:07:38	06:07:38
3	4415633.956661	482213.022464	939.650000	03:07:39	06:07:39
4	4415633.126837	482211.909079	939.450000	03:07:40	06:07:40
5	4415631.688470	482209.982061	939.150000	03:07:42	06:07:42
6	4415631.098790	482209.011729	939.150000	03:07:43	06:07:43
7	4415630.490844	482207.941608	938.950000	03:07:44	06:07:44
8	4415629.753511	482206.828437	938.750000	03:07:45	06:07:45
9	4415629.163931	482205.815356	938.650000	03:07:46	06:07:46
10	4415628.574184	482204.873522	938.450000	03:07:47	06:07:47
11	4415627.947575	482203.874604	938.250000	03:07:48	06:07:48

Table 5.2: WiMAX Measurement Sample

ID	Notebook Time	RSSI	CINR	AvgTxPwr
1	06:07:40	-85 dBm	6 dB	5 dBm
2	06:07:41	-84 dBm	7 dB	4 dBm
3	06:07:43	-84 dBm	7 dB	5 dBm
4	06:07:44	-85 dBm	6 dB	0 dBm
5	06:07:45	-80 dBm	10 dB	-5 dBm
6	06:07:47	-83 dBm	8 dB	-3 dBm
7	06:07:48	-82 dBm	9 dB	-3 dBm
8	06:07:49	-78 dBm	12 dB	-7 dBm
9	06:07:50	-81 dBm	10 dB	-4 dBm

by adding time difference between GPS and notebook, which is exactly 3 hours in this case, to GPS time. Table 5.2 contains sample WiMAX data which includes row ID, notebook time, and signal parameters (RSSI, CINR, AvgTxPwr.) Table 5.3 contains data from Table 5.1 and Table 5.2 combined together. It can be seen from Table 5.3 that GPS entry with ID 1 is used twice, that is, it is matched with two WiMAX entries. Reason for such matching is that it is allowed to match WiMAX and GPS entries with a time difference smaller than or equal to 1 second. This is a necessarily provided flexibility; because, GPS time and notebook time does not match exactly all the time. As a worst case scenario, consider that a new WiMAX entry was recorded at each odd second, and a GPS entry was recorded at each even second. In this situation none of the entries could be matched if only entries with same time were allowed to be matched. This flexibility allows WiMAX entry with ID 2 to be utilized instead of to be wasted. Rows in italic in Table 5.1 and Table 5.2, for which there are no matching entries, are not used in Table 5.3.

Table 5.3: Combined WiMAX and GPS Measurements

GPS ID	WiMAX ID	GPS Coordinates			GPS Time + Time Diff.	Computer Time	WiMAX Signal Parameters		
1	1	4415633.126837	482211.909079	939.450000	06:07:40	06:07:40	-85 dBm	6 dB	5 dBm
1	2	4415633.126837	482211.909079	939.450000	06:07:40	06:07:41	-84 dBm	7 dB	4 dBm
6	3	4415631.098790	482209.011729	939.150000	06:07:43	06:07:43	-84 dBm	7 dB	5 dBm
7	4	4415630.490844	482207.941608	938.950000	06:07:44	06:07:44	-85 dBm	6 dB	0 dBm
8	5	4415629.753511	482206.828437	938.750000	06:07:45	06:07:45	-80 dBm	10 dB	-5 dBm
10	6	4415628.574184	482204.873522	938.450000	06:07:47	06:07:47	-83 dBm	8 dB	-3 dBm
11	7	4415627.947575	482203.874604	938.250000	06:07:48	06:07:48	-82 dBm	9 dB	-3 dBm
11	8	4415627.947575	482203.874604	938.250000	06:07:48	06:07:49	-78 dBm	12 dB	-7 dBm

CHAPTER 6

PREPARING THE ENVIRONMENT

In this chapter, it is explained how collected fingerprints were visualized by drawing a signal map and how a grid map was extracted from that initial signal map in order to gain more accuracy in positioning.

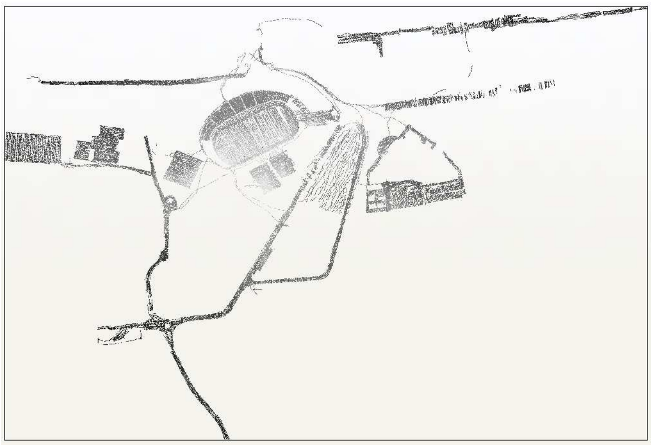
6.1 Drawing Initial Signal Map

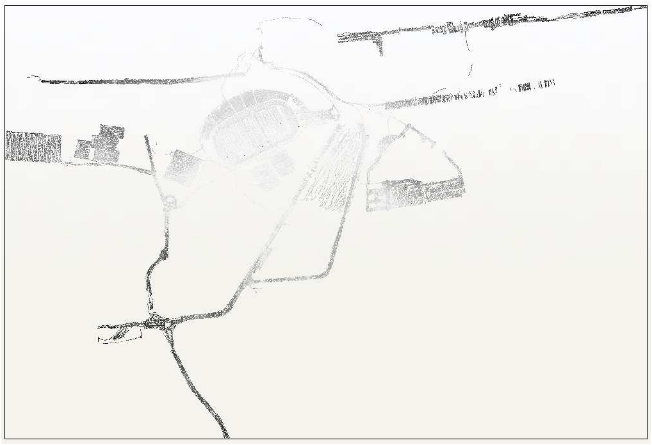
After signal measurements and data combination was completed, it was time to create signal maps which would demonstrate how signal parameters changed over WiMAX coverage area. A program was written in C# for plotting the maps, which later would be improved to do position estimation. First, a basic map was plotted which gave no clue about signal variation. The map was duplicated because there were three signal parameters, Received Signal Strength Indicator (RSSI,) Carrier to Interference-plus-Noise Ratio (CINR,) and Average Transmit Power (AvgTxPwr.) For each parameter a different map was to be plotted since behavior of each parameter was desired to be investigated on its own. There would also be another map which would combine all three parameters' information together, as the base of position estimation later. Each map belonging to a parameter was colored according to that parameter's value. Coloring process was handled in the following way.

Each pixel in screen has a RGB (red, green, blue) value defining the color to be displayed. Colors are identified by giving a number from 0 to 255 for each one of red, green, and blue values. For example; while (255, 0, 0) identifies pure red, (255, 255, 255) identifies black. For each parameter, the biggest and the smallest values were found from measurement data, and a variation interval was calculated. Red, green, and blue values corresponding to one unit of variance in each parameter was found, and pixels were colored accordingly. For example; RSSI value changed from -28 dBm to -92 dBm which meant that its variation interval was 64 units. Therefore, 1 unit of change in RSSI could be represented by $\frac{255}{64} \approx 4$ units of change in color. Because of that -92 dBm was the smallest value of RSSI, it was represented by (0, 0, 0). Similarly, -28 dBm was represented by (255, 255, 255). If it was required to find RGB value of -60 dBm RSSI, its difference from the smallest value RSSI could take would be computed, $-60-(-92) = 32$, and the resulting value would be converted to an RGB value, $32 \times \frac{255}{64} = 127$. A pixel with a RSSI value of -60 dBm would then be represented by (127, 127, 127). Since the same value was given for red, green, and blue, maps were plotted in grayscale.

It should be noted here that multiple real points (GPS coordinates, therefore fingerprints) could correspond to one pixel in plotted maps because of scaling. For this reason, average of RSSI, CINR, and AvgTxPwr of all fingerprints corresponding to each pixel was calculated and saved. While maps were colored, those average values were considered. RSSI, CINR, AvgTxPwr maps can be seen in Figure 6.1(a), Figure 6.1(b), and Figure 6.1(c). In RSSI and CINR maps (Figure 6.1(a) and Figure 6.1(b)) lighter colors imply higher parameter values which are desirable since high RSSI and CINR are found in good communication links. In AvgTxPwr map darker places imply usage of lower transmit power which is desirable for longer battery life.

As mentioned before, there was a fourth map which was colored according to all three signal parameters at the same time. That map was the most important one; because, it formed the base of position estimation process which would utilize all parameters. Coloring of that map was done in the same way explained above with one difference. Value of red was set according to RSSI, value of green was set according to CINR, and value of blue was set according to AvgTxPwr. Therefore, for each parameter a scale of 255 units was used. Each pixel in resulting map was colored according to the pixel's average RSSI, CINR, and AvgTxPwr values, and the map was colorful unlike previous grayscale ones. As can be seen in Figure 6.2(a), colorfulness of the map made it more detailed and informative than other maps. Variation of signal is easier to observe here. Yellow places imply better signal because RSSI is high (signal strength is good,) CINR is high (signal is effective because carrier signal is intelligible) and AvgTxPwr is low (mobile terminal (MT) battery will last longer because power used for sending signals to base station (BS) is low.) Greener areas imply worse


(a)


(b)

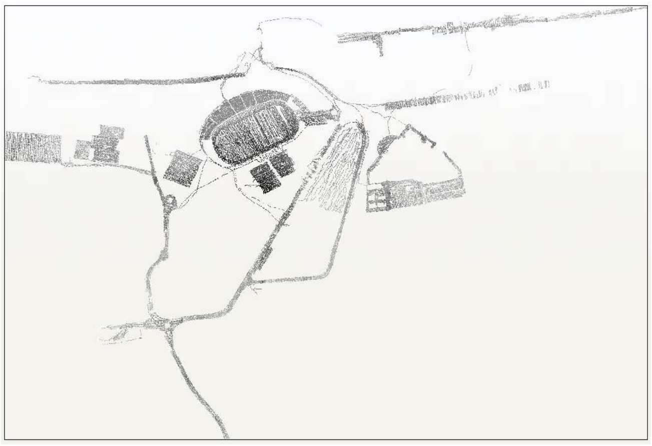

(c)

Figure 6.1: (a) RSSI map. (b) CINR map. (c) AvgTxPwr map.

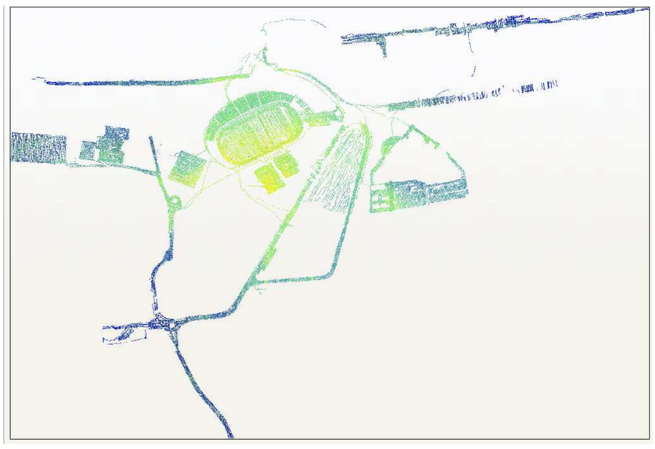
signal compared to yellow areas. Blue areas show places where signal is bad.

6.2 Splitting Signal Map into Grid Cells

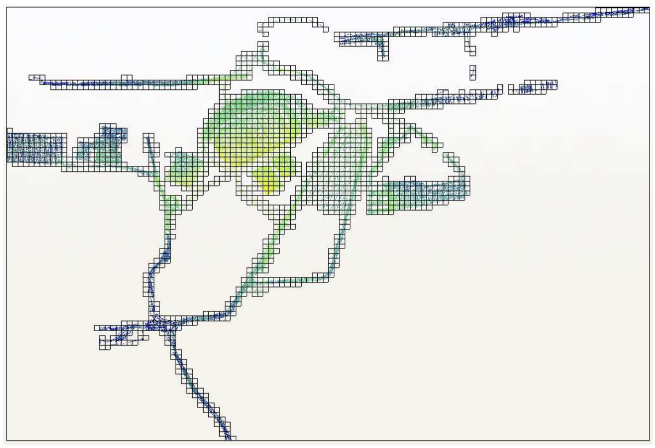
In Chapter 5 it was briefly explained that dense measurements were carried out in order to obtain more accuracy. The desired accuracy was provided by extracting a grid map from the signal map. The process was fairly straightforward. The signal map was simply divided into square regions (*grid cells* or shortly *cells*) with same dimensions. Each cell was assigned an RSSI value, a CINR value, and an AvgTxPwr value by computing average of parameter values of individual fingerprints recorded in that cell. Expected signal of a mobile terminal (MT) moving in the area of a cell was identified by that grid's assigned parameter values.

The advantage of taking average of multiple fingerprints in cells was reducing undesirable effects of erroneous measurements which would cause misleading results in position estimation. Therefore, it was decided to do positioning according to grid cells only. On the downside, cells would provide coarse estimations; because, a mobile terminal that was found to be moving in a cell could be anywhere in the area of that cell. However, that downside was not a significant one for mainly two reasons. First, it was already known there was a low probability of reducing the number of possible cells where a mobile terminal might reside to one (or just a few cells.) In other words, there would be an obscurity problem in estimations to some degree, anyway, that would prevent relatively less coarse estimations. The problem was caused by lack of information, interconnected to utilization of only one base station. Having a coarse estimation (estimating a cell instead of exact coordinates) was insignificant compared to that problem. Second, cells covered a reasonably small area, approximately 10×10 m^2 throughout studies. Uncertainty caused by the area covered by a cell was not considered as a problem, especially realizing the fact that a single base station positioning technique was being studied.

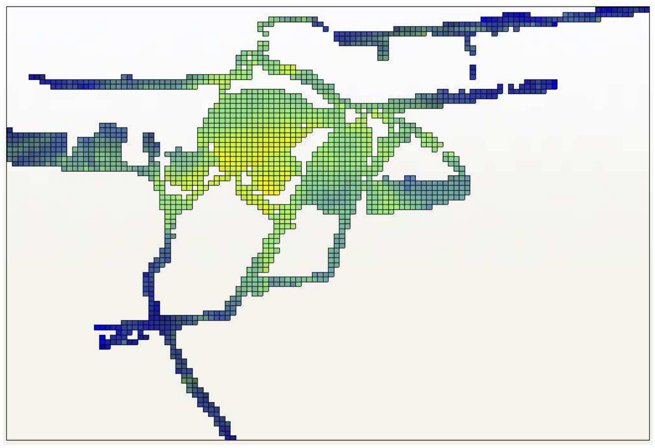
In initial grid map only grid borders were plotted over the signal map which utilized all parameters as seen in Figure 6.2(b). The map was needed to be improved since individual fingerprints were not considered anymore. So, cells were colored by applying the coloring process of individual fingerprints to cells. Each cell was colored by giving it an RGB value according to its average RSSI, CINR, and AvgTxPwr values. The resulting map is shown in



(a)



(b)



(c)

Figure 6.2: (a) Signal map plotted using RSSI, CINR, and AvgTxPwr. (b) Initial version of the grid map (118×90 cells.) (c) Grid map colored according to averages of RSSI, CINR, and AvgTxPwr (118×90 cells.)

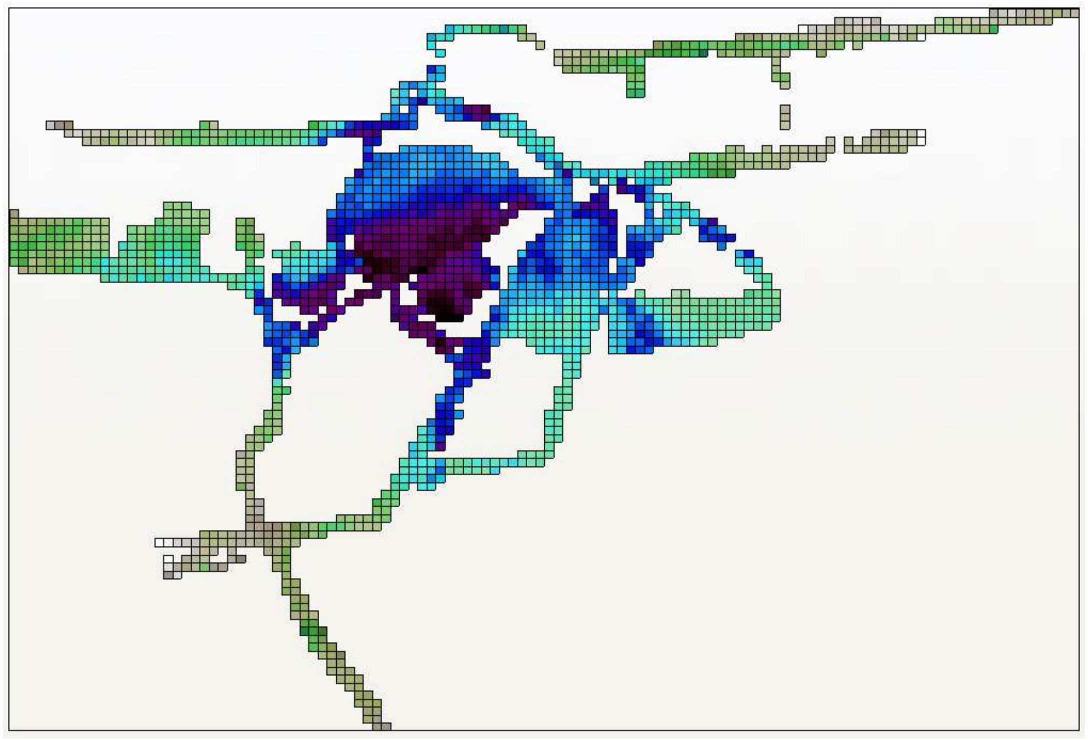


Figure 6.3: Grid map colored using HSL color space (118×90 cells.)

Figure 6.2(c).

Figure 6.2(c) showed in grid form how signal changed, but it was not very satisfactory. The reason was that, although it nicely demonstrated variation of the signal over coverage area, it failed to show color differences between neighboring cells. For example; in stadium area it seemed that the signal changed smoothly from up to down like there were two regions with different signal prints, and a soft transition in between. However, when a close look was taken at stadium it could be seen that there were small variations between neighboring cells which implied that the signal might not be changing smoothly. Since the primary purpose of drawing map was to get visual aid in understanding the problem that was dealt with, it was thought that the map could be improved. Upon recommendation, Hue Saturation Lightness (HSL) color spaced was employed instead of RGB. It is explained that the model “attempt to describe perceptual color relationships more accurately than RGB” [42]. Coloring of cells was done in the same way as before: signal parameters, RSSI, CINR, and AvgTxPwr were assigned to elements of HSL color space, hue, saturation, and lightness, respectively. The algorithm in [42] was implemented in C# to convert HSL value of a cell to its corresponding

RGB value so that it could be represented in the grid map. Latest version of the grid map was like Figure 6.3.

The resulting map gave better idea about signal variation than the previous map. It could clearly be seen from the HSL map that the color transition in stadium was not that smooth, and even though neighboring cells were of the same color there was noticeable shading between them. It was a good progress to see variations existed between cells because, as will be seen in following chapters, position estimation would rely on those variations. Actually, it was not important to know what kind of signal was implied by which color. The important thing to consider was that simply there were parameters assigned to cells, and changes in these parameters could give information about mobility between cells. The grid map allowed comprehending that.

CHAPTER 7

POSITION ESTIMATION

In this chapter, both the proposed position estimation algorithm and k Nearest Neighbor algorithm are described in detail. It is explained how tracking was carried out in the proposed algorithm and why clustering was used.

The proposed position estimation algorithm is designed to provide two dimensional, physical, and absolute location information for mobile stations (MSs) in a WiMAX network with a single base station (BS.) The algorithm utilizes fingerprinting to make up for the absence of multiple base stations and the shortage of signal parameters required for position estimation. As mentioned in previous chapter, positioning is based on comparisons made between signal parameters of tracked mobile stations and training data (fingerprints) corresponding to grid cells (or shortly *cells*.) Signal parameters of a cell were computed by averaging signal parameters of individual points falling into the cell. There are a number of positioning parameters that must be optimized for the network environment where positioning will take place. This implies that positioning performance not only depends on robustness of the algorithm, but also is significantly affected by selection of parameter values. The algorithm can return one or, more likely, multiple locations as a result of the estimation process and it can be forced to return one result by setting the related parameter. Input section of Figure 7.1 lists the parameters used in positioning algorithm.

7.1 Proposed Position Estimation Algorithm

Figure 7.1 and Figure 7.2 form backbone of the proposed algorithm. Position estimation starts with collecting signal parameters, that is, Received Signal Strength Indicator (RSSI,)

```
input    : cells is an array containing data of all cells in grid map.
           maxRssiDiff is the maximum RSSI difference allowed.
           maxCinrDiff is the maximum CINR difference allowed.
           maxAvgTxPwrDiff is the maximum AvgTxPwr difference allowed.
           maxUncontinuity is an integer.
           radiusForNeighbors is an integer.
           expansionFactor is an integer.
           increment is an integer.
           n is an integer.
           maxK is an integer.
output   : clusterCenters is a storage array for multiple cells' data.
variable: window is a FIFO array with a length of n.
           stream is a stream for incoming fingerprints.
           incoming is a storage for various fingerprint data.
           toBeFound is a storage for various fingerprint data.
           previousResult is an array containing (cell, uncontinuity) pairs.
           possibleCells is an array for multiple cells' data.
           filteredResult is an array containing (cell, uncontinuity) pairs.

           cell is a storage for various cell data.
           uncontinuity is an integer.
```

Figure 7.1: Main position estimation algorithm. See Figure 7.2 for the rest.

Carrier to Interference-plus- Noise Ratio (CINR) and Average Transmit Power (AvgTxPwr,) of a tracked mobile station (line 1 in Figure 7.2.) Signal parameters are received from a stream (line 2) which is created between the mobile station and a station taking on the task of position estimation (either a base station with which the mobile station is associated or the mobile station itself.) The reason for having the mobile station provide its own signal parameters in the algorithm was that the only network element which both measured signal parameters and provided corresponding data in a useable format was a notebook coupled with a WiMAX adapter, and that all training and scenario data were collected with that machine.

Whenever a new fingerprint (a signal parameter set coming from the mobile station) is available, it is pushed into a First In, First Out (FIFO) array (window) with a maximum length of n (lines 2 and 3,) which is the first parameter of the positioning algorithm. In the beginning, the array is empty, meaning that its number count (Count(window)) is zero. As fingerprints (incoming) are received from the tracked mobile station, the array is filled up. When it reaches its maximum capacity (line 4) the average of signal parameters encapsulated as fingerprints that are stored in window is computed (line 5.) They will be used throughout

```
// Part 1: Receiving incoming fingerprints.
while there is incoming fingerprint do
    incoming ← new fingerprint from stream;
    Push incoming into window;
    if Count (window )=n then
        toBeFound ← Average (window);
        // Core position estimation starts.
        // Part 2: Parameter comparison with grid map (Figure 7.4.)
        possibleCells ← ParameterComparison (toBeFound, maxRssiDiff,
        maxCinrDiff, maxAvgTxPwrDiff, cells);
        // Part 3: Tracking (Figure 7.5.)
        filteredResult ← Tracking (cells, radiusForNeighbors, expansionFactor,
        increment, previousResult, possibleCells, maxUncontinuity);
        // Part 4: Clustering.
        clusterCenters ← KMeansClustering (filteredResult, maxK);
        // Part 5: Preparation for next iteration.
        previousResult ← filteredResult;
        Pop from window;
    end
end
```

Figure 7.2: Main position estimation algorithm, continued.

the positioning step. What this means is that the core position estimation mechanism (innermost code portion in Figure 7.2) is provided with an input that is the average of multiple actual inputs. As for why such an input method was applied, individual fingerprints may be erroneous because of signal impairments or temporary measurement problems the mobile station experiences. Averaging values of multiple subsequent fingerprints help normalizing such deviations which, if not handled properly, can lead the algorithm yield incorrect estimations. Here, parameter n defines the number of individual fingerprints to be collected and averaged. It may be assumed that using window reduces the number of inputs fed into the core estimation mechanism at the rate of n, e.g. if total number of individual fingerprints received throughout the execution of a position estimation task is t then because of window, actual number of computed inputs is $\frac{t}{n}$. That would be true if at the end of an iteration of core position estimation window was completely emptied. However, only the oldest entry in array is removed, and next time a new fingerprint is received in the next iteration, it is pushed into the array (Figure 7.3.) Core position estimation is started again

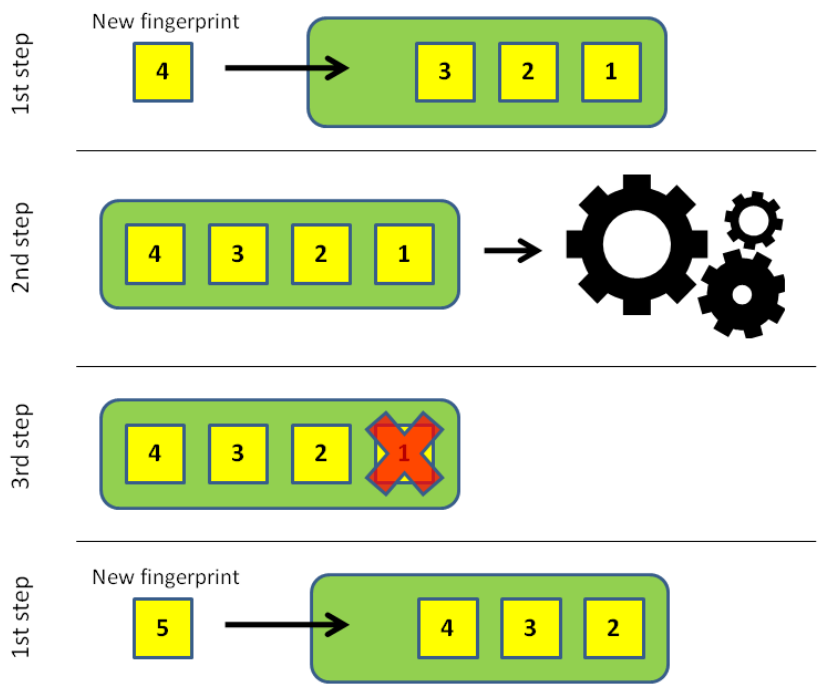


Figure 7.3: Received signal parameter sets are pushed into a FIFO array (1st step.) When the array is full, average of signal parameters stored in the array is computed and used throughout one algorithmic iteration (2nd step.) Before a subsequent iteration starts, the oldest signal parameter set is removed from the array (3rd step) and a new one is presented (back to 1st step.)

with the average of fingerprints in the array, resulting in $t - \mathsf{n}$ inputs which is close to t since n is mostly a one digit number.

7.1.1 Parameter Comparison and Initial Grid Cell Selection

First step of core position estimation is comparing signal parameters of all cells in the map (cells) with signal parameters of the newly computed input fingerprint (toBeFound.) The purpose is to identify the cells whose signal parameters are in certain proximity to those of the input fingerprint. As stated in Figure 7.4 depicting the ParameterComparison function, proximity is specified by three algorithmic parameters called maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff. These algorithmic parameters define the maximum difference that a cell's signal parameters are allowed to have from signal parameters of the input fingerprint. In order for a cell to have the opportunity to get examined later in the position estimation

```
input    : toBeFound is a storage for various fingerprint data including rssi, cinr,
           avgTxPwr.
           maxRssiDiff is the maximum RSSI difference allowed.
           maxCinrDiff is the maximum CINR difference allowed.
           maxAvgTxPwrDiff is the maximum AvgTxPwr difference allowed.
           cells is an array containing data of all cells in grid map.

           rssi is an integer storing RSSI.
           cinr is an integer storing CINR.
           avgTxPwr is an integer storing AvgTxPwr.
output   : possibleCells is an array for multiple cells' data.
variable : element is a storage for various cell data including rssi, cinr, avgTxPwr.
           rssiDiff is an integer.
           cinrDiff is an integer.
           avgTxPwrDiff is an integer.

// Part 2:  Parameter comparison with grid map.
1 foreach element in cells do
2     rssiDiff ← Absolute (toBeFound.rssi – element.rssi);
3     cinrDiff ← Absolute (toBeFound.cinr – element.cinr);
4     avgTxPwrDiff ← Absolute (toBeFound.avgTxPwr – element.avgTxPwr);
5     if rssiDiff ≤ maxRssiDiff and cinrDiff ≤ maxCinrDiff
6     and avgTxPwrDiff ≤ maxAvgTxPwrDiff then
7         Push element into possibleCells;
8     end
9 end
```

Figure 7.4: ParameterComparison function.

process, so that it may be elected as one of the cells where a tracked mobile station can be, the cell should pass the parameter test in this step. Every time ParameterComparison function is invoked, all cells in the grid map are examined (line 1 in Figure 7.4.) For each cell (element,) absolute differences (Absolute function) between cell parameters and the input fingerprint are calculated (lines 2, 3 and 4 in Figure 7.4 showing operations related to RSSI, CINR and AvgTxPwr, respectively.) Since each one of these parameter differences has to be smaller than or equal to the maximum amount specified by the algorithmic parameters, the algorithm must have corresponding conditional constructs (lines 5 and 6 in Figure 7.4.) Cells whose parameters ensure the condition are identified as the ones where a tracked mobile station can be. Therefore, they are pushed into an array (possibleCells) and passed on to the main algorithm for further investigation.

7.1.2 Tracking

Switching back to the main algorithm (Figure 7.2,) now there is a list of cells (possibleCells) that will be used for tracking the mobile station to be located as it moves around in the coverage area of the network. During parameter comparison, all cells ensuring a certain condition were collected. If the algorithm was designed to estimate position in one step without using previously obtained data, then some properties of the cells in possibleCells would be evaluated to come up with a final position estimate. However, that estimate would be far from accurate since those cells would be selected as the result of a simple computation based on just a few signal parameters (because of unavailability of more parameters,) meaning that most probably there would be a lot of selected cells which is an undesirable situation. In addition to that, those cells would likely be located at opposite directions of the coverage area with similar signal patterns because of the fact that mobile stations which are situated at different locations and which have the same distance to a base station can have very similar signal parameters. In such a case, if a simple positioning scheme was developed, e.g. returning the average of coordinates of those cells as the position estimate, final estimate would mostly be somewhere around the middle of the map.

It was known that with such a small number of available parameters simple straightforward positioning like the one explained above would be very troublesome. Accordingly, it was thought that if extra meaningful information could be extracted from previous positioning steps and used in the current step; there could be a chance to improve performance of the algorithm. Tracking exactly does this. Simply put, it checks whether the cells in possibleCells are neighbors of the cells that were selected as final position estimates in one previous step of the main algorithm. The motivation behind such a control is that if a mobile station located at a certain point is moving around at a certain speed, then after a short time interval it can not be at a very distant point, but somewhere around its initial point. That idea was implemented in tracking portion of the algorithm. The cells which are neighbors of the results of a previous iteration of the algorithm keep their ‘possible *result* cell’ title for this iteration. The ones which are not neighbors of previous result cells are eliminated since the idea was that a mobile station could not move somewhere other than close proximity of a point at which it was located a short time ago. The intention was to track possible paths the mobile station could be following and reduce those paths, therefore the number of possible result cells, with each iteration of the algorithm. Tracking function makes use of a number of algorithmic parameters to allow the algorithm to be optimized for different scenarios, e.g. tracking of humans, tracking of cars, etc.

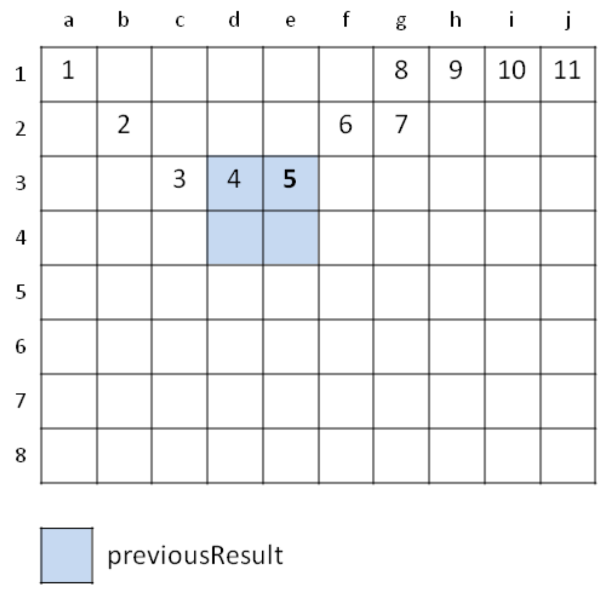
If Tracking is invoked for the first time (line 1 in Figure 7.5) it means that this is the first iteration of the main algorithm and as a matter of course the array containing cells resulted from a previous iteration (previousResult) is empty. However, even for the first iteration, the algorithm requires a non empty array of cells to start tracking process. Since in the beginning there is no information available about the location of a mobile station (before receiving any signal parameters from it) the only thing that can be said without doubt is that the mobile station is somewhere in the coverage area of the network. In other words, it can be located at any one of the cells in the map. So, each (element) and every cell in the map (cells) are saved into the array of previousResult (as results of 0th iteration) only when it's the first time the function is executed.

Before explaining rest of the Tracking function, here is an example which shows the problem with basic tracking and the solution to it that is employed in this algorithm. In Figure 7.6(a) there is a map of a network environment with 8 rows and 10 columns, consisting of a total of 80 cells. There is a mobile station in the environment which follows a path starting from the cell labeled 1 (the cell with index a1) and moving towards the cell labeled 11 (the cell with index j1), visiting the cells in increasing label order (a1, b2, c3, d3, e3, f2, g2, g1, h1, i1, j1.) The cell whose label is bold (5th cell or e3) is where the mobile station was in the last iteration of basic tracking algorithm, and cells colored in light blue are the position estimates resulting from that iteration. Imagine that, the mobile station moves to the next cell as depicted in Figure 7.6(b), and is still being tracked. Light blue cells are now what were previously mentioned as previousResult for this iteration. Also imagine that according to the signal parameters received from the mobile station, the algorithm identified possibleCells as the yellow ones. As mentioned in the definition of tracking previously, the algorithm will check whether the yellow cells are among neighbors of cells from a previous iteration, (neighbors of the light blue cells.) Since this is a simple positioning example, assume that the neighbors (neighbors) of previousResult are the dark blue cells in Figure 7.7(a). What the current picture tells is that location of the mobile station was estimated as one of the four light blue cells in previous iteration, and now it is expected to be somewhere in blue (dark or light) region; however, according to the signal parameters received from the mobile station in the

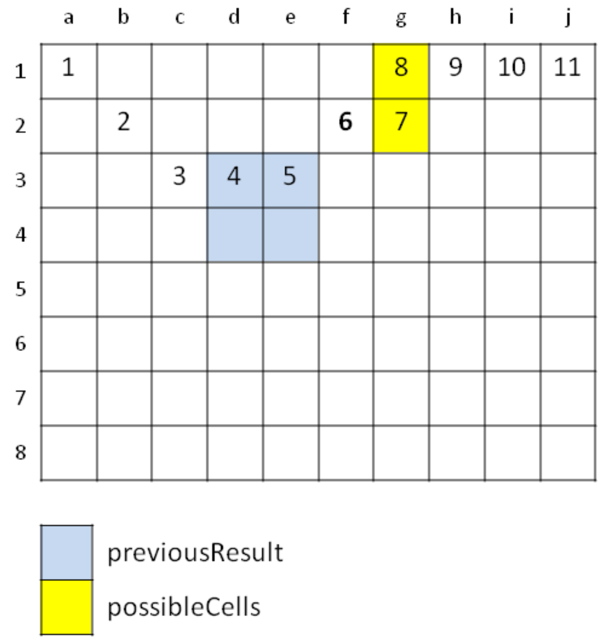
```
input    : cells is an array containing data of all cells in grid map.
           radiusForNeighbors is an integer.
           expansionFactor is an integer.
           increment is an integer.
           previousResult is an array containing (cell, uncontinuity) pairs.
           possibleCells is an array for multiple cells' data.
           maxUncontinuity is an integer.

           cell is a storage for various cell data.
           uncontinuity is an integer.
output   : filteredResult is an array containing (cell, uncontinuity) pairs.
variable: element is a storage for various cell data.
           cellUncontinuityPair is a storage for (cell, uncontinuity) pair.
           neighbors is a storage array for multiple cells' data.
           count is an integer.
// Part 3:  Tracking.
if this is the first time Tracking function is invoked then
    foreach element in cells do
        Push (element, 0) into previousResult;
    end
end
foreach cellUncontinuityPair in previousResult do
    // GetNeighboringCells is defined in Figure 7.10.
    neighbors ← GetNeighboringCells (cellUncontinuityPair, increment,
    radiusForNeighbors, expansionFactor);
    foreach element in possibleCells do
        if element is in neighbors then
            Push (element, 0) into filteredResult;
            count ← count + 1;
        end
        if count = 0 and cellUncontinuityPair.uncontinuity < maxUncontinuity then
            Push (cellUncontinuityPair.cell, cellUncontinuityPair.uncontinuity + 1)
            into filteredResult;
        end
    end
end
```

Figure 7.5: Tracking function.

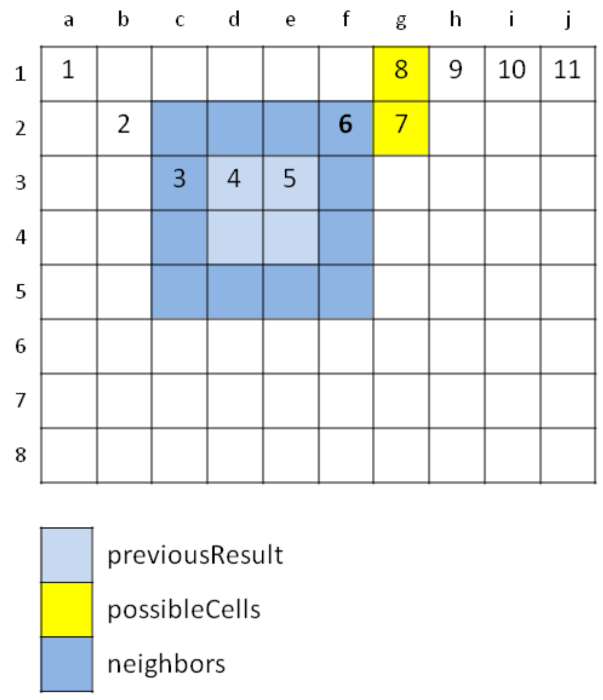


(a)

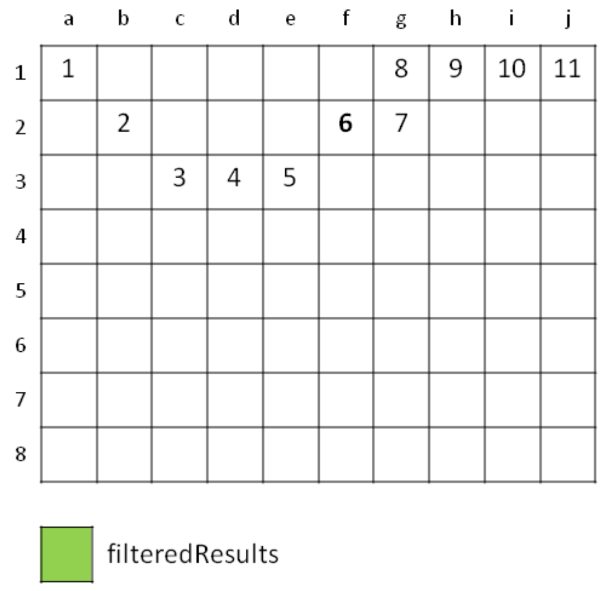


(b)

Figure 7.6: Tracking example, 1st (a) and 2nd (b) part.



(a)

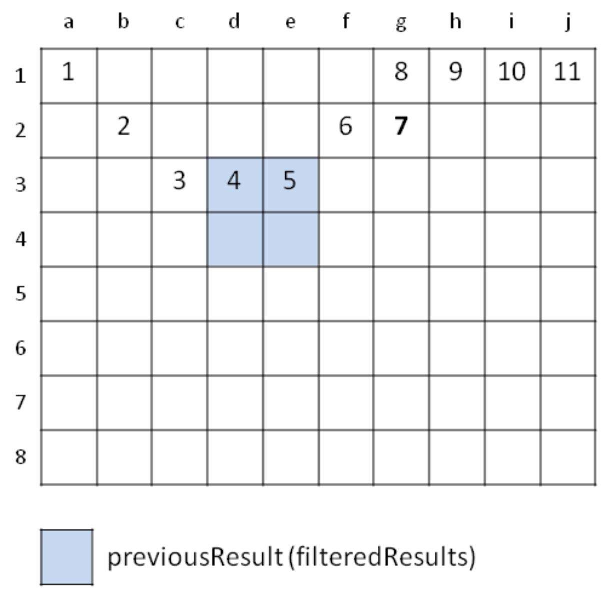


(b)

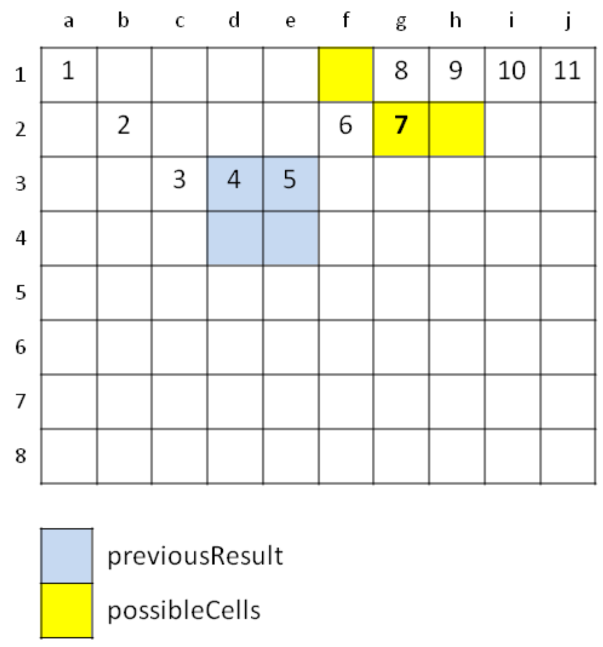
Figure 7.7: Tracking example, 3rd (a) and 4th (b) part.

current iteration, it must also be in one of the two yellow cells. So, cells that are colored both in yellow and blue will be final position estimates of this iteration, and other cells will not be considered as locations where the mobile station can be. Clearly, in Figure 7.7(a) there is no cell with both yellow and blue colors, and the algorithm ends up in a situation as depicted in Figure 7.7(b). There are no colored cells at all which implies that the path the mobile station was following is lost. This problem may occur frequently in position estimation because of various reasons like a mobile station providing incorrect signal parameters due to signal impairments or temporary hardware problems. For example, if signal impairments did not affect parameter measurements of the mobile station, cells f1 and f2 might be identified as possibleCells instead of g1 and g2, and the algorithm would not lose the path as f2 would be a successful position estimate.

It is a known fact that signal impairments and similar unwelcome factors influencing position estimation can not be controlled or modified according to requirements of the position estimation algorithm. Therefore, the algorithm must be ready for such problems and have a rescue plan in case of their occurrence. In Figure 7.7(b), all yellow (possibleCells) and blue (previousResult and neighbors) cells were removed (identified as not being final estimations) because of that there were no cells colored in both yellow and blue, that is, there were no cells belonging to both possibleCells, and previousResult or neighbors. As an alternative approach, consider giving a second chance to the cells colored in light blue before immediately making a judgment about them. Put differently; consider that final estimations of a previous iteration are accepted as final estimations of the current iteration. Figure 7.8(a) depicts initial state of the alternative case. The mobile station moves to 7th cell (g2) and provides newly measured signal parameters to the algorithm, meaning that a new iteration is started. Light blue cells will be used as previousResult one more time for this iteration. As usual, parameter comparisons are carried out between all cells in the map and the received signal parameters to find a coarse list of cells where the mobile station can be, which are shown in Figure 7.8(b) (possibleCells in yellow.) The algorithm will continue executing just like it did in previous iterations. To refine possibleCells, neighbors of previousResult will be controlled to see whether there are cells that belong to both possibleCells, and previousResult and its neighbors. However, in this iteration there will be a slight change in neighbor identification process. Normally, neighboring cells are identified as the ones which are one cell away from the elements of previousResult, e.g. as in Figure 7.7(a), since it was

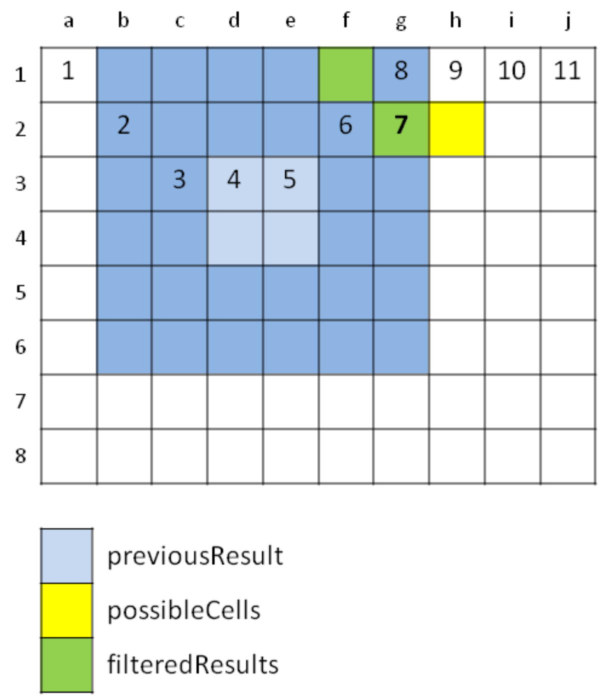


(a)

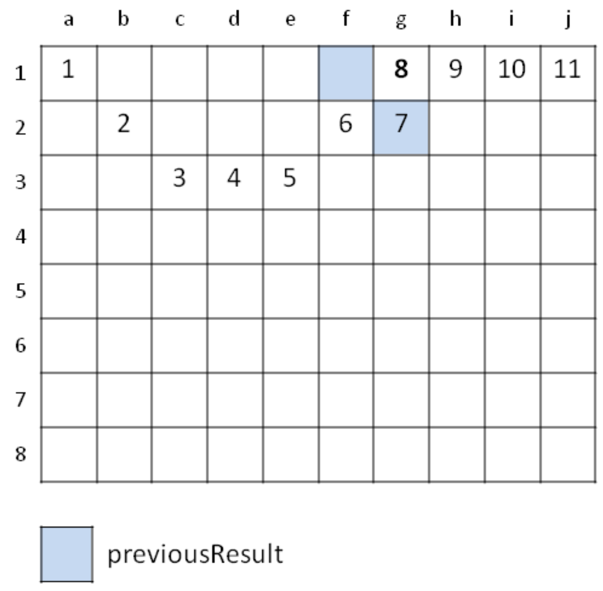


(b)

Figure 7.8: Tracking example, 5th (a) and 6th (b) part.



(a)



(b)

Figure 7.9: Tracking example, 7th (a) and 8th (b) part.

thought that the mobile station could move at most one cell away from its previous location in one unit of time. In Figure 7.8(b) light blue cells (previousResult) are actually estimation results of two previous iterations which means that selecting locations that are one cell away from previousResult elements as neighboring cells will not be sufficient to catch up with the mobile station. That's why for this iteration neighbors will be selected as locations that are two cells away from previousResult elements as depicted in Figure 7.9(a). Evidently there are cells belonging to both possibleCells and previousResult in that case, and they are shown in green color. These two green ones are the final position estimates of the current iteration, and they seem to catch up with the mobile station this time even though there has been a problem with following its path in previous iteration. The algorithm will continue estimating position as long as it receives new signal parameters. Figure 7.9(b) shows the initial state of a subsequent iteration.

The problem mentioned above was encountered in practice during development and testing of the algorithm, so the solution above was proposed and finally employed in Tracking (Figure 7.5) function. Implementation of the alternative approach above was realized with the introduction of a parameter for the cells stored in previousResult called uncontinuity. uncontinuity can be thought as an age attribute for cells. Normally, it has a value of 0 for a cell that is a member of both possibleCells, and previousResult and its neighbors. Existence of such a cell implies that the path identified by that cell is being tracked successfully as one of the possible paths which a mobile station to be located may be following. When there is a problematic situation like the one explained in the above example, uncontinuity is incremented one by one in each subsequent iteration to point out that those cells in previousResult are given a second (or more) chance to be evaluated. In problematic case of losing a tracked path, cells have limited opportunities to be reevaluated in multiple iterations, and the maximum amount of uncontinuity is set by the algorithmic parameter called maxUncontinuity. Tracking function returns the list of cells that are identified as final position estimates to the main algorithm using filteredResult array.

Note that it is actually highly desirable that the total number of paths decreases, because this is how the algorithm obtains better estimates. It works its way through many possible paths (coarse position estimates) to fewer paths (better estimates.) The reason for proposing such an approach to prevent removal of paths immediately when there is a disconnection between previous estimates and candidates for current estimates was to be more flexible in tracking

```
input   : cellUncontinuityPair is a storage for (cell, uncontinuity) pair.
          increment is an integer.
          radiusForNeighbors is an integer.
          expansionFactor is an integer.

          cell is a storage for various cell data.
          uncontinuity is an integer.

output  : neighbors is a storage array for multiple cells' data.
variable: rangeExpansion is an integer.
1 if cellUncontinuityPair.uncontinuity > 0 then
2     rangeExpansion ← cellUncontinuityPair.uncontinuity − 1;
3     rangeExpansion ← 1 + (rangeExpansion / expansionFactor);
4     rangeExpansion ← rangeExpansion × increment;
5 else
6     rangeExpansion ← 0;
7 end
8 neighbors ← Cells at most radiusForNeighbors + rangeExpansion cells away from
  cellUncontinuityPair.cell;
```

Figure 7.10: GetNeighboringCells function.

paths so that the path a mobile station actually follows is not eliminated easily because of temporary disturbances.

Rest of the Tracking function works as follows: first, for each cell cellUncontinuityPair in previousResult (line 6 in Figure 7.5,) neighboring cells neighbors are determined (line 7.) Neighbor determination is handled by GetNeighboringCells function (Figure 7.10) which will be explained later. Following that, each element of possibleCells (the array which contains cells whose Euclidean distances of signal parameters to those sent by the tracked mobile station are in a certain range defined by algorithmic parameters) is checked to see whether it is a neighbor of the currently examined cellUncontinuityPair (lines 8 and 9.) In other words, cells in previousResult are handled one by one such that all elements in possibleCells are compared with one cell (cellUncontinuityPair) in previousResult and its neighbors at a time. If an element in possibleCells is a neighbor of cellUncontinuityPair then that element is elected as one of the final estimates and stored into the corresponding array (filteredResult) meaning that path tracking proceeds without interruption.

In the event that none of the possible cells are members of a cellUncontinuityPair's neighbors, there are two actions that can be taken. One, cellUncontinuityPair cell is given another chance to be a final estimate if its uncontinuity is smaller than the maximum value allowed (lines 13

Table 7.1: An example showing how range expansion changes with uncontinuity when expansion factor is 2 and increment is 3.

uncontinuity	rangeExpansion
1	3
2	3
3	6
4	6
5	9
6	9

and 14,) two, cellUncontinuityPair cell is no longer considered as a location where the tracked mobile station can be if its uncontinuity has reached the maximum value. While the first action increases robustness, the second one reduces the number of possible position estimates which is the key to better location.

As mentioned in previous paragraphs, neighbors of a cell are determined by GetNeighboringCells function which computes the distance that a neighboring cell must have from another cell (cellUncontinuityPair) whose neighbors are being identified. Unit of distance is cells. Apart from uncontinuity, three algorithmic parameters called increment, radiusForNeighbors and expansionFactor are used to calculate the distance value and identify neighbors. Normally, when neighbors of a cell with an uncontinuity value of 0 are being identified, neighbor distance is set as the default value defined by radiusForNeighbors. For example, in the previously mentioned example (Figure 7.7(a)) radiusForNeighbors was 1. If uncontinuity of a cellUncontinuityPair is bigger than 0, then a rangeExpansion is calculated which is the additional distance added to the original one in order to expand the neighboring area. Formula used for rangeExpansion is

$$\mathsf{rangeExpansion} = \lfloor 1 + [(\mathsf{uncontinuity} - 1)/\,\mathsf{expansionFactor}] \times \mathsf{increment} \rfloor. \tag{7.1}$$

The formula expresses that for every expansionFactor amount of increment in uncontinuity, neighboring area will be expanded by a factor of increment. Accordingly, neighbor distance is set as radiusForNeighbors+ rangeExpansion. Table 7.1 shows how much the distance is expanded for different uncontinuity values when expansionFactor is 2 and increment is 3. In the example depicted in Figure 7.9(a), both expansionFactor and increment are 1.

7.1.3 Clustering

After tracking process is completed, an array of cells (filteredResult) containing position estimates is returned to the main positioning algorithm in Figure 7.2. Early experiments showed that the array contained numerous cells which rendered it inconvenient to return the cells to a user who is being tracked. Visual investigations of experimental positioning results revealed that those cells were actually grouped together in a few regions in the map, e.g. Figure 7.11. Therefore, instead of returning all those cells, central points of cell groups could be returned. However, to apply that idea to the proposed positioning mechanism, an algorithmic way of grouping cells together must have been found. It was easy to see that the problem at hand was actually the data organization problem called clustering. Upon recommendation, k-means algorithm was examined to see whether it was a viable clustering technique for the case at hand. MATLAB's [38] k-means implementation was tested with input from actual position estimation experiments, and it was seen that generated clusters represented actual groups of cells correctly which lead to the conclusion that k-means was an appropriate choice.

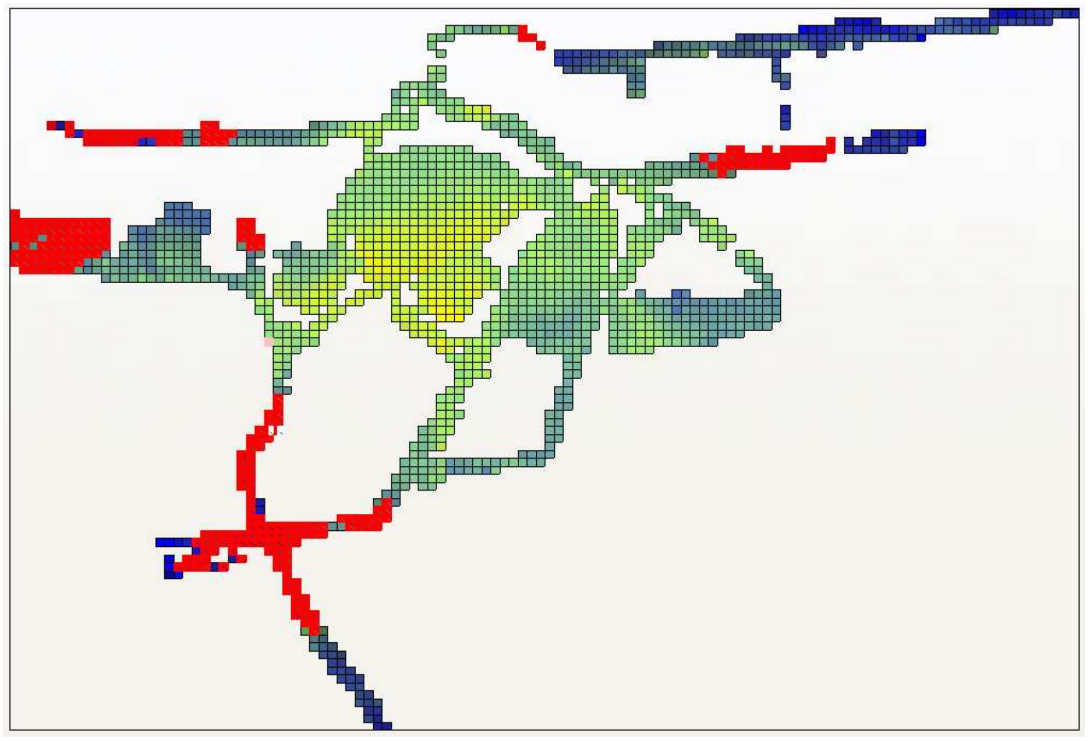


Figure 7.11: A brief visual investigation reveals that red cells form three clusters; one in upper right part, one in upper left part and one in the crossroad in lower left part.

K-means was to be integrated into the proposed positioning algorithm, but there was an obstacle. *k* in k-means determines the number of clusters to be generated, and it must be known before clustering process is started. During testing of k-means, the value of *k* was given explicitly and *k* clusters were generated eventually. However, during execution of the positioning algorithm there was no information about clusters at all including how many clusters should be generated. In fact, when it was thought that clustering was required for the positioning algorithm it was implied that the clustering technique to be used would provide a solution similar to what would be perceived as clusters by a human being with availability of no a priori information. Like a human who can say that there are three big clusters in Figure 7.11 without being given any hints in advance, the clustering algorithm to be employed was expected to come up with the most appropriate number and formation of clusters. Therefore, it was necessary to find a way to determine the number of clusters first. Investigation about the problem revealed that there were methods called validity indices which dealt with searching for a proper number of clusters.

A number of validity indices were examined to see how they performed for the case of clustering estimation results. A MATLAB tool developed by Kaijun Wang for estimating the number of clusters [37] was employed for the examinations. The tool first uses either k-means or Partitioning Around Medoids (PAM) algorithm to divide a data set into *k* clusters for different values of *k*, and then it runs various validity indices to evaluate the quality of results. Validity indices included in the package are Rand index, Adjusted Rand index, Mirkin index, Hubert index, Silhouette, Davies-Bouldin, Calinski-Harabasz, Krzanowski-Lai, Hartigan, weighted inter- to intra-cluster ratio, Homogeneity, and Separation. Position estimates (an array of cells) obtained by running the proposed algorithm on different data sets were extracted and reformatted for the tool to process them properly. At the end of execution, computed indices were plotted on screen which allowed identifying the validity index that returned the closest result to what was perceived by the eye to be the most appropriate number of clusters for the provided data set. Homogeneity was selected as the method of identifying *k* value of k-means algorithm because of that it generated clusters for various input data as desired. The tool was integrated into the .NET framework based simulation of the proposed positioning algorithm using the MATLAB component called MATLAB Builder for .NET [38].

Clustering is the last important step of the main positioning algorithm in Figure 7.2. Cells

```
input   : k is an integer.
          cells is an array containing data of all cells in grid map.
          n is an integer.
output  : meanResult is a storage for various cell data.
          rbfResult is a storage for various cell data.
variable: window is a FIFO array with a length of n.
          stream is a stream for incoming fingerprints.
          incoming is a storage for various fingerprint data.
          toBeFound is a storage for various fingerprint data.
          possibleCells is an array for multiple cells' data.

// Part 1:  Receiving incoming fingerprints.
1 while (there is incoming fingerprint) and (Count (window ) < n ) do
2     incoming ← new fingerprint from stream;
3     Push incoming into window;
4 end
5 toBeFound ← Average (window);

// Part 2:  Get k cells with smallest distances
// (Figure 7.13.)
6 possibleCells ← CellsWithSmallestDistance (toBeFound, cells, k);

// Part 3:  Estimate location using two weighting functions
// (Figure 7.15 and Figure 7.16.)
7 meanResult ← Mean (possibleCells, cells);
8 rbfResult ← RBF (possibleCells, toBeFound, cells);
```

Figure 7.12: kNN position estimation algorithm.

identified by Tracking function are fed into the k-means clustering algorithm to come up with just a few estimations for the current location of a mobile station that is being tracked (line 8.) Here, maxK value, which is the last algorithmic parameter, defines the maximum number of clusters that k-means is permitted to generate, implying that for any input data set consisting of cells k-means can form maxK or less number of clusters (whichever number is more appropriate.) Ultimately, cluster centers are make up the final position estimates of the tracked mobile station.

7.2 k Nearest Neighbor Algorithm

k Nearest Neighbor (kNN) algorithm, which was previously explained in Section 2.4.2.1, was also implemented in this thesis project upon recommendation. The purpose was to form an

```
input    : toBeFound is a storage for various fingerprint data.
           cells is an array containing data of all cells in grid map.
           k is an integer.
output   : possibleCells is an array for multiple cells' data.
variable : element is a storage for various cell data.
           distance is a floating point number.
           temp is an array containing (element, distance) pairs.
// Part 2: Find k cells with smallest distances.
1 foreach element in cells do
      // Euclidean distance between element and toBeFound
         (Figure 7.14.)
2     distance ← EuclideanDistanceOfParameters(toBeFound, element);
3     Push (element, distance) into temp;
4 end
5 temp ← SortByDistance (temp);
6 Push first k elements of temp into possibleCells;
```

Figure 7.13: kNN CellsWithSmallestDistance function.

idea about the performance of a simple position estimation mechanism utilizing one base station (BS.) It was thought that results collected from kNN experiments could help make judgments about the performance of the proposed positioning algorithm. It was not possible to compare the two algorithms directly since the proposed algorithm yielded one or more final position estimates while kNN always returned exactly one estimate. Nevertheless, to be able to do at least a partial comparison between the two, several performance evaluations were carried out on simulation results of both algorithms which are explained in detail in the next chapter.

Backbone of the employed kNN algorithm can be seen in Figure 7.12. kNN algorithm starts with the same operations as the proposed positioning algorithm explained in previous section: incoming fingerprints (incoming) sent by a mobile station over a data stream (stream) are stored in a FIFO array (window) whose length is n, and average of the signal parameters encapsulated in those fingerprints (toBeFound) are fed into rest of the algorithm as input (lines 1 to 5.) Following the initial parameter computation, a search is started in grid map of the environment for k number of cells (not to be confused with k in $k - means$) with the smallest Euclidean distances of signal parameters to those of toBeFound (line 6.) k is the only algorithmic parameter in here. Figure 7.13 shows the CellsWithSmallestDistance function in detail which undertakes the task of identifying those k cells. For each cell (element) in

```
input   : toBeFound is a storage for various fingerprint data including rssi, cinr,
          avgTxPwr.
          element is a storage for various cell data including rssi, cinr, avgTxPwr.
          rssi is an integer storing RSSI.
          cinr is an integer storing CINR.
          avgTxPwr is an integer storing AvgTxPwr.
output  : distance is a floating point number.
variable: rssiDiff is a floating point number.
          cinrDiff is a floating point number.
          avgTxPwrDiff is a floating point number.
1 rssiDiff ← toBeFound.rssi – element.rssi;
2 cinrDiff ← toBeFound.cinr – element.cinr;
3 avgTxPwrDiff ← toBeFound.avgTxPwr – element.avgTxPwr;
4 distance ← Sqrt (Square (rssiDiff)+Square (cinrDiff)+Square (avgTxPwrDiff));
```

Figure 7.14: kNN EuclideanDistanceOfParameters function.

the map (cells) (line 1 in Figure 7.13) Euclidean distance of signal parameters is calculated (line 2) and pushed into an array (temp) (line 3.) After distance computation is finished, cells (temp) are sorted by their distances in increasing order (line 5) and first k elements (possibleCells) are returned to the main kNN algorithm. The function calculating Euclidean distance between two signal parameter sets (EuclideanDistanceOfParameters) is described in Figure 7.14. It is very similar to the ParameterComparison function in Figure 7.4 written for the proposed positioning algorithm in previous section. Only actual difference is that at the end of EuclideanDistanceOfParameters, squares of parameter differences are summed up, and square root of that value is calculated which gives the Euclidean distance.

After identification of k cells, main kNN position estimation algorithm (Figure 7.12) continues its execution with determination of final position estimates for the mobile station that is being tracked using two different weighting functions, mean and radial basis function (RBF) (lines 7 and 8.) Mean function which is depicted in Figure 7.15 implements the uniform weighting scheme in Equation 2.4. First, arithmetic mean of coordinates (coordMean) of possibleCells is found (line 1,) then the cell that contains the point represented by coordMean is identified. In other words, Euclidean distance of coordinates of each cell (element) in the map (cells) to coordMean is calculated (lines 2 and 3,) and the cell with the smallest distance is returned to the main kNN algorithm as the final position estimate that is obtained using the uniform weighting scheme (lines 3 to 5.)

```
input    : possibleCells is an array for multiple cells' data.
           cells is an array containing data of all cells in grid map.
output   : meanResult is a storage for various cell data.
variable : element is a storage for various cell data including coordinates.
           coordMean is a storage for coordinates data.
           minEuclDist is a floating point number.
           coordinates is a storage for coordinates data.
// Part 3:  Estimate location using Mean weighting function.
coordMean ← MeanOfCoordinates (possibleCells);
foreach element in cells do
    if EuclideanDistance (element.coordinates, coordMean ) < minEuclDist then
        minEuclDist ← EuclideanDistance (element.coordinates, coordMean);
        meanResult ← element;
    end
end
```

Figure 7.15: kNN Mean function.

Radial basis function weighting function is shown in Figure 7.16 which implements the following equation:

$$\hat{\mathbf{l}} = \frac{\sum_{i=1}^{k} w_i(\mathbf{m})\mathbf{l}^{(i)}}{\sum_{i=1}^{k} w_i(\mathbf{m})}, \tag{7.2}$$

where $\hat{\mathbf{l}}$ is the position estimate, $\mathbf{l}^{(i)}$ is coordinate vector of i^{th} cell , and $w_i(\mathbf{m})$ is the weighting factor which is

$$w_i(\mathbf{m}) = e^{-(\text{EuclideanDistanceOfSignalParameters}(p^{(i)},f))^2} \tag{7.3}$$

where $p^{(i)}$ is the signal parameter of i^{th} cell and f is the input fingerprint. The implementation starts with computing the numerator (numerator) of Equation 7.2. For each cell (element) in possibleCells (line 1) a weighting factor (weight) according to Equation 7.3 is calculated (line 2) by making use of EuclideanDistanceOfParameters function described in Figure 7.14. Rest of the first loop deals with computation of two summations (numerator and denumerator) that make up Equation 7.2 (lines 3 and 4) whose result (coordRbf) is later set by dividing numerator by denumerator (line 6.) Although coordRbf is the actual radial basis function estimate, the function returns the cell that contains the point represented by coordRbf as the

```
input    : possibleCells is an array for multiple cells' data.
           toBeFound is a storage for various fingerprint data.
           cells is an array containing data of all cells in grid map.
output   : rbfResult is a storage for various cell data.
variable: element is a storage for various cell data including coordinates.
           weight is a floating point number.
           numerator is a floating point number.
           denominator is a floating point number.
           coordRbf is a storage for coordinates data.
           MinEuclDist is a floating point number.
           coordinates is a storage for coordinates data.
   // Part 3:  Estimate location using RBF weighting function.
 1 foreach element in possibleCells do
       // EuclideanDistanceOfParameters is defined in Figure 7.14.
 2     weight ← $e^{-\mathtt{EuclideanDistanceOfParameters}\,(\text{toBeFound, element})^2}$
 3     numerator ← numerator + (weight × element.coordinates);
 4     denominator ← denominator + weight;
 5 end
 6 coordRbf ← numerator/denominator;
 7 foreach element in cells do
 8     if EuclideanDistance (element.coordinates, coordRbf ) < MinEuclDist then
 9         MinEuclDist ← EuclideanDistance (element.coordinates, coordRbf);
10         rbfResult ← element;
11     end
12 end
```

Figure 7.16: kNN location estimation using radial basis function.

final estimate (to the main kNN algorithm) since grid based positioning is being carried out. Identification of that cell (lines 7 to 11) is done in the same way as in Mean function (lines 2 to 6 in Figure 7.15.)

Both the proposed position estimation algorithm and kNN algorithm return coordinates of estimated cells' central points to the user when their execution is completed.

CHAPTER 8

EXPERIMENTS

In this chapter, experiments of the proposed position estimation algorithm and k Nearest Neighbor algorithm are explained. Effects of algorithmic parameters on estimation results are investigated for each algorithm. Finally, experimental results of the algorithms are compared and factors that are thought to affect the results are stated.

Primary fingerprint data, which provided a priori information for position estimation, was collected in July, and first version of the proposed position estimation algorithm was developed to interpret those fingerprints and track mobile stations (MSs.) At the end of August, it was decided that in order to test performance of the algorithm and make improvements accordingly, additional scenario data was required, that is, signal measurements had to be done in the coverage area, again. Instead of scanning the area as previously shown in Figure 5.3, signal parameters would be measured while walking around in campus like ordinary campus residents. Normal behavior of a person in motion would be represented; because, scenario data needed to exemplify the case of a person on the go. Since state of the environment was mostly unchanged (weather, buildings, etc.) compared to how it was back in July, same signal impairments were expected to be experienced. Therefore, there were no concerns about capturing fingerprints in a certain location that were evidently different from the ones in the primary training database corresponding to the same location. All WiMAX and DGPS equipment on the mobile station side were ready for data collection. However, measurements could not be started because of a serious problem with the WiMAX infrastructure at the Department of Computer Engineering. Although machines composing the infrastructure seemed to be working, no WiMAX network was detected by the WiMAX enabled notebook that previously connected to the network without a problem. Rebooting the system multiple times did not solve the issue. The company which installed the infrastructure was contacted, the problem was explained, and urgency of the situation was emphasized. Unfortunately, it took some time for a technician to be assigned, and the person who handled the problem was apparently not knowledgeable about the infrastructure. The problem was still persistent more than two months after the company was contacted. It was not a good idea to wait more for the system to be fixed since scenario data had to be collected as soon as possible because of time constraints. An algorithm was developed to generate scenario data based on existing fingerprints in the training database. It allowed drawing routes on the signal map (imitating a mobile station walking on a path) and producing sequences of measured signal parameters which were randomly selected from fingerprint sets corresponding to the grid cells (or shortly *cells*) that made up those routes. Simulations were carried out using numerous scenarios created by that algorithm. Three months after the problem was reported, and just before those simulations were finished, the technician working on the infrastructure told that the network was back online. Despite the fact that the system was not fully functional as it should have been, the available WiMAX enabled notebook was able to detect and connect to the network. This much delayed progress caused to reconsider the current situation. Eventually, scenario data generated by the algorithm were decided to be discarded to be replaced by actual signal measurements. Nevertheless, results of the simulations based on algorithmically generated scenario data helped a lot in improving the proposed position estimation algorithm, investigating effects of changing algorithmic parameters on positioning accuracy, and defining parameter sets that were thought to yield better position estimates.

In field measurements were carried out between December 4th, 2009 and December 7th, 2009 to capture real world scenario data similar to fingerprints stored in the training database. Many experiments were performed with the data, results were examined, algorithmic parameters were changed accordingly, and this sequence of processes was reiterated multiple times. Results mentioned in here and listed in Appendix B were obtained by executing the latest version of the proposed position estimation algorithm in a limited time period. Note that it was not possible to test the algorithm with every possible parameter set. It might be possible to obtain better estimates if more experiments were done with more parameter sets. Still, results shown in this chapter are thought to be good indicators of the proposed algorithm's performance since the experiments that were realized were evaluated in detail to come up with the most efficient parameter sets.

Experiments were also conducted to see how k Nearest Neighbor (kNN) algorithm would perform with the same data set, so that a number of comparisons would be drawn between the two algorithms. kNN experiments are explained later in this chapter.

8.1 Proposed Position Estimation Algorithm

An example of simulation results is shown in Table 8.1. First thirteen columns indicate algorithmic parameters that were used in that experiment while rest of the columns shows the results. To make it easier to explain each column, column numbers are written in the first row. Information contained in each column is listed as follows:

- 1st and 2nd columns: Horizontal and vertical cell counts, that is, the number of cells that were contained in each row of the map used in positioning and the number of cells that were contained in each column of the same map.

- 3rd, 4th and 5th columns: Maximum difference that a cell's signal parameters, Received Signal Strength Indicator (RSSI,) Carrier to Interference-plus-Noise Ratio (CINR) and Average Transmit Power of Mobile Station (AvgTxPwr,) were allowed to have from signal parameters of a tracked mobile station. As explained in Section 7.1.1, maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff are algorithmic parameters used in ParameterComparison function (Figure 7.4) to find an initial set of cells which may later be identified as final position estimates.

- 6th column: Maximum uncontinuity that a cell was allowed to have. maxUncontinuity is used in Tracking function (Figure 7.5) to limit the additional number of times that a cell is considered as a possible candidate for the location of a tracked mobile station.

- 7th, 8th and 9th columns: Algorithmic parameters that were provided to GetNeighboringCells function (Figure 7.10.) As described previously the function is invoked during tracking of a mobile station to determine a given cell's neighbors as explained in Section 7.1.2.

- 10th column: Capacity of the array (window in Figure 7.2) which stored fingerprints received from a tracked mobile station. Recall that the proposed algorithm computes

Table 8.1: An example of the proposed algorithm's simulation results.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster
237	182	7	5	7	10	1	3	1	3	1	90	100	188	0	188	0
18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	
number of 3 cluster	number of <100	% of <100 (1)	% of <100 (2)	number of <300	% of <300 (1)	% of <300 (2)	prob. of <100 (1)	prob. of <100 (2)	prob. of <300 (1)	prob. of <300 (2)	rmse	rmse, all points	rmse, edges	total time	avg. est. time	
0	44	23%	23%	150	79%	79%	23,4%	23,4%	79,79%	79,79%	248,8	248,8	248,8	10688 sec	57 sec	

average of signal parameters that are stored in the array to provide reliable input to the positioning mechanism.

- 11^{th} column: Maximum number of clusters the algorithm was allowed to generate at the end of an iteration.
- 13^{th} column: Number of subsequent fingerprints that made up a scenario. Scenario data were collected in multiple fingerprinting sessions each one of which lasted for a few hours. It meant that during a single session an arbitrarily selected path was followed for a long time and signal parameters were continuously measured. Since it did not make sense to have a few experiments based on such long scenarios (it would imply tracking a user for hours which was not the case in practice) those large data were divided into smaller streams of m subsequent fingerprints. The algorithm was modified to finish an ongoing positioning experiment after it processed m fingerprints from a source of scenario data, and to start a new experiment with the next m fingerprints. Therefore, m fingerprints represented one actual scenario used for one experiment. The value shown in 13^{th} column in Table 8.1 defines the value of m.
- 12^{th} column: Number of *average* inputs that were processed before results of an experiment were written into an output file. For example; in the example in Table 8.1 each path (or scenario) consisted of 100 fingerprints, and the experiment was concerned with position estimates obtained after processing the first 90 fingerprints.

Columns 14^{th} to 33^{rd} contain results of the experiment defined by above parameters.

- 14^{th} column: Total number of estimations that were carried out. In the example shown, it is stated that 188 estimations were recorded, meaning that there were 188 scenarios (consisting of 100 fingerprints) evaluated in that experiment.
- 15^{th}, 16^{th}, 17^{th} and 14^{th} columns: Number of estimations that yielded a certain number of clusters. 15^{th} column shows the number of estimations that yielded no clusters, reflecting that position of the tracked mobile station could not be estimated; 16^{th}, 17^{th}, 18^{th} columns show the number of estimations that yielded one, two, three clusters, respectively. It seems that all estimations in the example generated one cluster which is reasonable because of that the maximum number of clusters was set to one as shown in 11^{th} column.

- 19th column: Number of scenarios, for each one of which the proposed algorithm returned a number of estimates (cells) that included at least one cell which was at most 100 meters away from the actual location of a tracked mobile station. For example; if the algorithm returned three cells as final position estimates for one scenario, and one of those three cells was less than 100 meters away from the actual location, then the number in 19th column is incremented by one for that scenario.

- 20th column: Percentage of the number of estimations which returned cells that were at most 100 meters away from actual locations (19th column) to the number of estimations that yielded one or more position estimates, that is, 19th column divided by the summation of 16th, 17th and 18th columns.

- 21st column: Percentage of the number of estimations which returned cells that were at most 100 meters away from actual locations (19th column) to the total number of estimations, that is, 19th column divided by 14th column.

- 22nd, 23rd and 24th columns: Same as 19th, 20th, 21st columns, respectively, except that the inspected distance is 300 meters instead of 100 meters.

- 25th column: Average probability of selecting a cell from the result set of an estimation, which is less than 100 meters away from the actual location of a tracked mobile station.

  As stated multiple times previously, the proposed algorithm can return more than one position estimate as the result of a position estimation operation. That means a user who is being tracked can be given a number of locations (instead of exactly one,) at one of which the user may be situated. If k cells are returned to a user after a positioning operation is finished, the probability that the user randomly selects a cell which is at most 100 meters away from its actual location is computed by the formula

$$p_e = \begin{cases} \frac{n_e}{\mathsf{k}} & \text{if } \mathsf{k} > 0 \\ 0 & \text{if } \mathsf{k} = 0 \end{cases} \tag{8.1}$$

  where n_e is the number of estimated cells that are less than 100 meters away from the actual location of a tracked user (out of k cells.) p_e is calculated for every position estimation operation (For example, in the example shown in Table 8.1, p_e is calculated 188 times for 188 estimations.) These probabilities are used to compute the average probability in 25th column by the formula

$$\frac{\sum_{i=1}^{t} p_e^{(i)}}{n_e^{1+}} \tag{8.2}$$

where t is the total number of estimations (14th column,) $p_e^{(i)}$ is p_e of ith estimation and n_e^{1+} is the number of estimations with one or more results. This formula defines the average probability of randomly selecting a position estimate, which is in 100 meter proximity of a tracked mobile station's actual location, from a set of results returned by a positioning operation that is known to yield at least one estimate.

- 26th column: Average probability of randomly selecting a position estimate, which is in 100 meter proximity of a tracked mobile station's actual location, from a set of results returned by a positioning operation. It is computed by a slightly modified version of Equation 8.2.

$$\frac{\sum_{i=1}^{t} p_e^{(i)}}{n_a} \tag{8.3}$$

where n_a is the total number of estimations. Note that position estimations yielding no results are taken into account in this equation because of that the sum of individual probabilities is divided by the total number of estimations.

- 27th and 28th columns: Same as 25th and 26th columns, respectively, except that the inspected distance between cells and tracked mobile station's actual location is 300 meters instead of 100 meters.

- 29th column: Root mean square error of the results obtained from position estimation operations each one of which returned at least one position estimate. It is computed by utilizing mean square errors of individual estimations' results. Mean square error belonging to an individual estimation is calculated by the formula

$$mse(i) = \frac{1}{r}\sum_{j=1}^{r}(p_e^{(i,j)} - p_a^{(i)})^2 \tag{8.4}$$

where $mse(i)$ is mean square error of ith estimation operation, r is the number of cells returned as results (same as the number of clusters generated,) $p_e^{(i,j)}$ is coordinate vector of j^{th} cell's central point, and $p_a^{(i)}$ is coordinate vector of a tracked mobile station's actual location. Following that, root mean square error of one experiment consisting of multiple estimations is computed by the formula

$$\sqrt{\frac{1}{t}\sum_{i=1}^{t} mse(i)} \tag{8.5}$$

where t is the number of estimations that return at least one position estimate.

- 30th column: Root mean square error of the results obtained from all position estimation operations including the ones that yielded no results. Main formula is the same as the one shown in Equation 8.5 with one difference; t is the total number of estimations (14th column.) The function calculating mean square errors, $mse(i)$, is modified to provide error values for estimations with no results. If one or more final position estimates are available, mean square error is calculated as shown in Equation 8.4. On the other hand, if no position estimates are returned, mean square error is calculated as the maximum distance between a cell in the map and actual location of the mobile station tracked during that positioning operation. In other words, distance between the actual location and each cell in the map is found, and the biggest one is identified as mean square error of that estimation operation.

$$mse(i) = \begin{cases} \frac{1}{r}\sum_{j=1}^{r}(p_e^{(i,j)} - p_a^{(i)})^2 & \text{if } r > 0 \\ \max\left((p_g^l - p_a)^2\right) & \text{if } r = 0 \end{cases} \tag{8.6}$$

Second part of Equation 8.6 handles the computation explained above where p_g^l is coordinate vector of lth cell's center (lth cell being one of many cells in the map.)

- 31st column: Root mean square error of the results obtained from all position estimation operations including the ones that yielded no results. This is different from the previous root mean square error. For an estimation which returns no results, mean square error is defined as the maximum distance between a corner of the map and actual location of a tracked mobile station. While in the previous computation all cells are investigated to find an error value, here only four corners of the map are examined.

$$mse(i) = \begin{cases} \frac{1}{r}\sum_{j=1}^{r}(p_e^{(i,j)} - p_a^{(i)})^2 & \text{if } r > 0 \\ \max\begin{pmatrix} (p_c^{ur} - p_a)^2 \\ (p_c^{ul} - p_a)^2 \\ (p_c^{lr} - p_a)^2 \\ (p_c^{ll} - p_a)^2 \end{pmatrix} & \text{if } r = 0 \end{cases} \tag{8.7}$$

p_c^{ur}, p_c^{ul}, p_c^{lr}, p_c^{ll} in Equation 8.7 are coordinate vectors of the map's upper right, upper left, lower right and lower left corners, respectively.

Difference between root mean square errors in 30th and 31st columns can be easily understood by looking at a grid map, e.g. Figure 7.11. Each row in that map contains 118 cells while each column contains 90 cells, implying a total 10620 cells. However, there are 1701 cells shown in the map; because, cells are created only for the places where fingerprinting was done. The algorithm always returns none or some of those 1701 cells as position estimates which is why root mean square error in 30th was decided to be computed by examining the available cells in the map. Root mean square error in 31st was also decided to be computed to provide extra information. The latter one always generates higher error values.

If all estimation operations related to an experiment yield at least one result, outputs of all three root mean square error formulas will be the same.

- 32nd column: Total time spent for the experiment on a computer powered by a 1.6GHz Intel Pentium M processor.

- 33rd column: Average time spent for one position estimation operation on a computer powered by a 1.6GHz Intel Pentium M processor.

8.2 Algorithmic Parameters' Effects on Estimation Results of Proposed Position Estimation Algorithm

Before comparing results of the proposed position estimation algorithm to those of k Nearest Neighbor (kNN) algorithm, results of the former will be discussed on their own. Algorithmic parameters' effects on estimation results will be expressed by comparing a number of key performance indicators (explained in previous section) belonging to different experiments. The main indicators to be mentioned are average probabilities of randomly selecting position estimates, which are in certain proximity to the actual location of a tracked mobile station, from a set of results returned by a positioning operation (26th and 28th columns in previous section.) These two indicators are employed since they (along with root mean square errors and total estimation time) integrate all of the information regarding to an experiment's results.

Table 8.2: A set of experiments belonging to the proposed algorithm, demonstrating effects of cell count on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	118	90	6	4	6	3	1	1	1	3	3	100	100	188	24	6	72	86	71	145	18,88%	49,47%	466	671 sec	4 sec
	237	182	6	4	6	3	1	1	1	3	3	100	100	188	12	6	74	96	83	164	21,19%	55,32%	413,1	5157 sec	28 sec
2	60	46	8	6	8	4	1	1	1	3	3	80	100	188	0	2	14	172	94	181	22,07%	62,23%	352,2	945 sec	6 sec
	118	90	8	6	8	4	1	1	1	3	3	80	100	188	0	1	16	171	106	180	25,18%	61,7%	360,9	1758 sec	10 sec
3	60	46	8	6	8	7	1	2	1	3	3	80	100	188	0	2	16	170	93	180	21,72%	62,23%	353,8	432 sec	3 sec
	118	90	8	6	8	7	1	2	1	3	3	80	100	188	0	1	15	172	107	181	25,09%	62,68%	359,9	1879 sec	10 sec
4	60	46	10	8	10	7	1	2	1	3	2	80	100	188	0	1	187	0	72	172	22,61%	66,49%	339,9	489 sec	3 sec
	118	90	10	8	10	7	1	2	1	3	2	80	100	188	0	0	188	0	78	169	25,27%	65,69%	343,9	2930 sec	16 sec

8.2.1 Horizontal and Vertical Grid Cell Counts

Table 8.2 shows algorithmic parameters and results of eight experiments which are divided into four pairs. Experiments in each pair consist of the same algorithmic parameter set except for horizontal and vertical cell counts. Therefore, differences between results of experiments belonging to the same pair are caused by differences between cell count parameters (Effects of other parameters will also be investigated according to such experiment pairs.)

As demonstrated by all experiment pairs in Table 8.2, when cell count is increased, probability of obtaining a position estimate which deviates at most 100 meters from the actual location is increased. It is understandable that such performance improvement is obtained; because, increase in cell count is achieved by decreasing size of cells, and cells representing smaller areas bring more accuracy to estimations (Dividing the coverage area into 60×46, 118×90 and 237×182 pieces results in square cells with approximately 20, 10 and 5 meter sides, respectively.) On the other hand, smaller cells' average signal parameters will be calculated according to fewer fingerprints which may mean that undesired effects of signal impairments on training data can not be removed to a sufficient degree. Accordingly, the tendency to select irrelevant cells during parameter comparison (Figure 7.4) may increase, eventually causing final estimates to contain significant errors. This may be the reason why probability of obtaining a position estimate which deviates at most 300 meters from the actual location is decreased when cell count is increased in 2nd and 4th experiment pairs (root mean square error is also increased in 2nd, 3rd and 4th pairs.)

To make it easier to understand the possible downside of smaller cells, extreme case can be considered; every single fingerprint on the map can be identified as a cell. In that situation erroneous fingerprints will not be taken care of and incoming signal parameters from a tracked mobile station will be compared to those. This will make it harder to track mobile stations since two neighboring fingerprints in the training database can have huge parameter differences. That is why cells were decided to be utilized in the first place; to overcome such fluctuations by calculating average of multiple fingerprints (That was also why average of a number of incoming signal parameters from a tracked mobile station was fed into the proposed position estimation algorithm.)

One last thing to mention is the change in execution time of experiments. All experiment pairs clearly show that as the number of cells is increased, total time spent on experiments increases simply because of that there are more cells to process. Switching from 118×90 cells to 237×182 cells causes the execution time to increase more than switching from 60×46 cells to 118×90 cells. Horizontal and vertical grid cell counts are the algorithmic parameters which have the greatest impact on total execution time.

8.2.2 Maximum Received Signal Strength Indicator, Carrier to Interference-plus-Noise Ratio and Average Transmit Power Differences

maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff are maximum Received Signal Strength Indicator (RSSI,) Carrier to Interference-plus-Noise Ratio (CINR) and Average Transmit Power of Mobile Station (AvgTxPwr) differences that a cell's average signal parameters are allowed to have from incoming signal parameters of a tracked mobile station in order for the cell to be identified as a possible position estimate during parameter comparison phase of the proposed algorithm (Figure 7.4.) In Table 8.3 there are four groups, three of which contain algorithmic parameters and results of two experiments while the last one contains information related to three experiments.

It can be seen from the first and fourth experiment groups in Table 8.3 that increasing values of maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff provide higher probabilities of obtaining position estimates that are at most 100 meters away from actual locations. However, arbitrarily increasing those values can degrade positioning performance as demonstrated by second, third and fourth experiment groups. Results suggest that there is an upper limit to these three algorithmic parameters for different numbers of cells. For example, consider a map consisting of many small cells. In that case, neighboring cells' average signal parameters will be close to each other, implying that smaller values of maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff will likely be more suitable for tracking. If bigger values are assigned to these parameters, more than adequate number of cells will be identified as possible position estimates during parameter comparison and accordingly *accuracy* of final estimates will decrease. This is why probability of obtaining a position estimate that is at most 100 meters away from the actual location in fourth experiment group decreases (last two experiments in Table 8.3) even though probability of obtaining a position estimate that is at most 300 meters away from the actual location increases modestly. In fact, this holds for

Table 8.3: A set of experiments belonging to the proposed algorithm, demonstrating effects of maximum RSSI difference, maximum CINR difference and maximum AvgTxPwr difference on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	4	3	4	4	1	1	1	3	3	80	100	188	62	11	79	36	34	89	7,89%	29,26%	574,67	119 sec	1 sec
	60	46	8	6	8	4	1	1	1	3	3	80	100	188	0	2	14	172	94	181	22,07%	62,23%	352,17	945 sec	6 sec
2	60	46	8	6	8	7	1	2	1	3	2	80	100	188	0	2	186	0	91	163	27,66%	65,16%	345,13	318 sec	2 sec
	60	46	10	8	10	7	1	2	1	3	2	80	100	188	0	1	187	0	72	172	22,61%	66,49%	339,9	489 sec	3 sec
3	118	90	8	6	8	6	1	1	1	3	2	80	100	188	0	1	187	0	92	164	28,72%	64,63%	354,65	1738 sec	10 sec
	118	90	10	8	10	6	1	1	1	3	2	80	100	188	0	0	188	0	80	170	25,8%	65,96%	344,53	2923 sec	16 sec
4	237	182	6	4	6	4	1	1	1	3	2	90	100	188	6	7	175	0	81	150	25,53%	56,12%	391,89	5051 sec	27 sec
	237	182	7	5	7	4	1	1	1	3	2	90	100	188	0	1	187	0	98	166	30,85%	62,77%	360,36	8771 sec	47 sec
	237	182	10	8	10	4	1	1	1	3	2	90	100	188	0	0	188	0	92	168	29,26%	64,89%	346,93	25028 sec	134 sec

results in second and third experiment groups, too. Note that as these probabilities increase, root mean square errors decrease. This is a desirable improvement.

Similar to what is explained above, if a map is divided into larger squares, neighboring cells' average signal parameters will not be very close to each other since each cell will be composed of more fingerprints distributed over a wider area (pointing to more variability in signal parameters.) That means parameter difference between large neighboring cells will be greater than parameter difference between smaller neighboring cells. Therefore maxRssiDiff, maxCinrDiff and maxAvgTxPwrDiff values for positioning based on large cells should be bigger than those values for positioning based on small cells. Of course, combined effects of other algorithmic parameters can cause altering of above explanations.

Increasing the algorithmic parameters mentioned in this section causes the proposed algorithm to require more execution time; because, more cells are selected during parameter comparison phase to be processed later.

8.2.3 Maximum Uncontinuity

Maximum uncontinuity is used to bring flexibility to tracking of a mobile station. The algorithmic parameter prevents immediately acknowledging a cell (which is likely to represent a location where a tracked mobile station is) as an incorrect position estimate in presence of temporary disturbances in signal measurements. Table 8.4, which contains algorithmic parameters and results of twelve experiments divided into five groups, shows how different values of maximum uncontinuity affect positioning performance.

First three experiment groups clearly show that when maximum uncontinuity is set (when a value greater than zero is assigned to it) probability of obtaining a position estimate which deviates at most 100 meters from the actual location of a tracked mobile station increases evidently. For example, probability value of the last experiment in third group, which uses a maximum uncontinuity value of 4, is more than two times the probability value of the first experiment in the same group. Probability of obtaining a position estimate less than 300 meters away from the actual location also increases noticeably as value of the algorithmic parameter is increased. However, there are cases where increasing the parameter's value does not bring performance improvement as shown in fourth and fifth experiment groups.

Table 8.4: A set of experiments belonging to the proposed algorithm, demonstrating effects of ‘maximum uncontinuity’ on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	6	4	6	0	1	1	1	3	2	40	100	188	51	3	134	0	49	107	13,56%	38,83%	529,7	119 sec	1 sec
	60	46	6	4	6	2	1	1	1	3	2	40	100	188	33	2	153	0	60	125	16,76%	46,28%	467,92	148 sec	1 sec
2	118	90	6	4	6	0	1	1	1	3	2	50	100	188	41	6	141	0	49	118	14,36%	44,68%	513,16	410 sec	3 sec
	118	90	6	4	6	4	1	1	1	3	2	50	100	188	7	2	179	0	74	147	23,94%	57,18%	392,1	753 sec	5 sec
3	237	182	6	4	6	0	1	1	1	3	2	100	100	188	57	3	128	0	37	105	11,17%	38,56%	567,13	2529 sec	14 sec
	237	182	6	4	6	2	1	1	1	3	2	100	100	188	24	4	160	0	64	138	20,48%	51,86%	460,93	4055 sec	22 sec
	237	182	6	4	6	3	1	1	1	3	2	100	100	188	12	6	170	0	73	150	22,61%	55,85%	417,16	4742 sec	26 sec
	237	182	6	4	6	4	1	1	1	3	2	100	100	188	7	7	174	0	84	156	26,86%	58,78%	396,01	4955 sec	27 sec
4	118	90	10	8	10	4	1	1	1	3	2	80	100	188	0	0	188	0	84	170	27,13%	66,49%	343,78	2788 sec	15 sec
	118	90	10	8	10	6	1	1	1	3	2	80	100	188	0	0	188	0	80	170	25,8%	65,96%	344,53	2923 sec	16 sec
5	237	182	7	5	7	4	1	1	1	3	2	90	100	188	0	1	187	0	98	166	30,85%	62,77%	360,36	8771 sec	47 sec
	237	182	7	5	7	6	1	1	1	3	2	90	100	188	0	1	187	0	95	165	30,05%	62,5%	358,87	10173 sec	55 sec

The reason for obtaining decreased positioning performance with a large value of maximum uncontinuity is that cells which should be eliminated in early iterations of the proposed positioning algorithm are not removed because of high flexibility introduced by the parameter, and presence of those cells causes different clustering of final estimations in following stages. Such unwelcome results hint at an upper limit for this algorithmic parameter (which changes according to values of other parameters.)

As expected, root mean square error decreases when positioning performance improves due to utilization of a suitable uncontinuity value. One small downside is that total positioning time required by the algorithm increases since the parameter causes more cells to be processed throughout the execution. But this increase in total estimation time is not as significant as those caused by previously explained algorithmic parameters.

8.2.4 Radius for Neighbors, Expansion Factor and Increment

Radius for neighbors, expansion factor and increment are the algorithmic parameters used by GetNeigboringCells (Figure 7.10) function which identifies a number of neighbors for a given cell. The function is an essential component of the proposed algorithm's tracking phase. Table 8.5 contains eleven experiments in five groups, showing how radius for neighbors, expansion factor and increment parameters affect positioning results.

As it can be seen from the experiments in Table 8.5, having a parameter trio other than (1, 1, 1) increases both probabilities of obtaining position estimates that are in certain proximity to the actual location of a tracked mobile station, and decreases root mean square errors of final estimates. Just like previously explained algorithmic parameters, values of the ones mentioned here should be adjusted for the particular positioning case at hand. For example, arbitrarily assigning a large number to the parameter 'increment' may not provide higher positioning performance as partially demonstrated by last two experiments of the fifth group in Table 8.5. As the value of 'increment' increases, computed neighborhood range of a given cell increases as well, indicating availability of more neighboring cells. Since tracking is based on processing neighbors of previous position estimates, the more neighbors there are to handle the more paths there are to track. Therefore, a lot of cells will be labeled as possible estimates in current iteration of the proposed algorithm, and all of them will be taken into consideration during clustering, eventually degrading accuracy.

Table 8.5: A set of experiments belonging to the proposed algorithm, demonstrating effects of ‘radius for neighbors,’ ‘expansion factor’ and ‘increment’ on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	4	3	4	4	1	1	1	3	1	80	100	188	62	126	0	0	10	66	5,32%	35,11%	530,98	63 sec	1 sec
	60	46	4	3	4	4	2	1	1	3	1	80	100	188	62	126	0	0	10	67	5,32%	35,64%	530,72	80 sec	1 sec
2	118	90	4	3	4	3	1	1	1	3	1	100	100	188	67	121	0	0	13	64	6,91%	34,04%	560,53	213 sec	2 sec
	118	90	4	3	4	3	1	1	3	3	1	100	100	188	60	128	0	0	14	74	7,45%	39,36%	532,8	296 sec	2 sec
3	118	90	4	3	4	3	1	1	1	3	3	100	100	188	67	12	75	34	31	89	7,09%	29,61%	592,46	271 sec	2 sec
	118	90	4	3	4	3	1	1	3	3	3	100	100	188	60	7	75	46	34	99	7,98%	31,12%	576,81	462 sec	3 sec
4	118	90	6	4	6	3	1	1	1	3	3	100	100	188	24	6	72	86	71	145	18,88%	49,47%	466	671 sec	4 sec
	118	90	6	4	6	3	1	2	2	3	3	100	100	188	20	7	67	94	75	151	19,68%	51,95%	451,78	696 sec	4 sec
5	237	182	6	4	6	4	1	1	1	3	2	90	100	188	6	7	175	0	81	150	25,53%	56,12%	391,89	5051 sec	27 sec
	237	182	6	4	6	4	1	3	3	3	2	90	100	188	4	4	180	0	82	156	26,06%	58,78%	380,97	5658 sec	31 sec
	237	182	6	4	6	4	1	3	5	3	2	90	100	188	2	8	178	0	82	160	25,8%	59,57%	374,39	6759 sec	36 sec

Earlier observations regarding total estimation time hold for the algorithmic parameters mentioned here. When larger parameter values are used, extra execution time will be required; because, there will be additional cells to deal with. Absence of various experiments investigating the effects of radius for neighbors, expansion factor and increment parameters on estimation results prevents making better inferences about these parameters.

8.2.5 N

n defines the number of incoming fingerprints from a tracked mobile station, the average of which is fed into the proposed position estimation algorithm as an input. Table 8.6 includes eighteen experiments assembled into nine pairs, and shows how positioning performance is affected by changes in value of n. There is an important thing to note here about the data in the table. Because of that the experiments which utilized real world data concentrated on analyzing other parameters, there was no useful information provided by them that would allow making observations about the parameter n. For this reason, results related to earlier experiments, which utilized manually generated scenario data, are presented in Table 8.6 to make interpretation of the parameter's effects possible. Root mean square error values are missing from the results since it was not being computed at that time.

Although results of earlier experiments give confusing information about n, a general inference can be drawn based on data in Table 8.6. All experiment pairs show the effect of switching from an n value of 3 to 6. Probability of obtaining a position estimate which deviates at most 100 meters from the actual location of a tracked mobile station increases in first six pairs (the ones with 60×46 and 118×90 cells) and decreases in last three pairs (the ones with 237×182 cells) as value of the algorithmic parameter is increased. Also, increases in probability values of fourth, fifth and sixth pairs (based on 118×90 cells) are less than those of first three pairs (based on 60×46 cells.) Combining these observations together reveals that there is a relationship between the size of cells (or the number of cells) and n. As cell size gets smaller, higher values of the algorithmic parameter have either less positive effects (e.g. when switching from 20x20 square meter cells to 10x10 square meter cells) or mostly negative effects (e.g. when switching from 10x10 square meter cells to 5x5 square meter cells) on estimation results.

As demonstrated by all experiment pairs in Table 8.6 (except the fifth one,) higher value

Table 8.6: A set of experiments belonging to the proposed algorithm, demonstrating effects of the algorithmic parameter ‘n’ on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	6	4	6	0	1	1	1	3	3	80	100	246	31	6	49	160	134	199	24,93%	60,7%	-	181 sec	1 sec
	60	46	6	4	6	0	1	1	1	6	3	80	100	246	11	1	53	181	147	221	25,54%	63,41%	-	201 sec	1 sec
2	60	46	6	4	6	0	1	1	1	3	3	90	100	246	34	0	62	150	118	194	22,56%	59,82%	-	183 sec	1 sec
	60	46	6	4	6	0	1	1	1	6	3	90	100	246	12	3	52	179	144	222	25,07%	64,84%	-	201 sec	1 sec
3	60	46	6	4	6	0	1	1	1	3	3	100	100	246	40	7	43	156	116	187	20,53%	57,86%	-	177 sec	1 sec
	60	46	6	4	6	0	1	1	1	6	3	100	100	246	13	5	47	181	148	221	25,61%	64,3%	-	196 sec	1 sec
4	118	90	6	4	6	3	1	1	1	3	3	80	100	246	3	1	59	183	147	237	26,22%	66,8%	-	760 sec	3 sec
	118	90	6	4	6	3	1	1	1	6	3	80	100	246	2	0	56	188	154	241	28,05%	67,28%	-	738 sec	3 sec
5	118	90	6	4	6	3	1	1	1	3	3	90	100	246	3	1	56	186	154	241	27,44%	68,02%	-	757 sec	3 sec
	118	90	6	4	6	3	1	1	1	6	3	90	100	246	2	1	57	186	164	240	30,15%	67,48%	-	740 sec	3 sec
6	118	90	6	4	6	3	1	1	1	3	3	100	100	246	4	0	60	182	161	239	29,13%	66,94%	-	749 sec	3 sec
	118	90	6	4	6	3	1	1	1	6	3	100	100	246	2	0	60	184	165	243	29,74%	67,55%	-	729 sec	3 sec
7	237	182	6	4	6	0	1	1	1	3	3	80	100	246	7	2	55	182	173	234	32,72%	67,14%	-	3847 sec	16 sec
	237	182	6	4	6	0	1	1	1	6	3	80	100	246	2	0	48	196	166	242	31,5%	67,48%	-	4490 sec	18 sec
8	237	182	6	4	6	0	1	1	1	3	3	90	100	246	7	1	62	176	176	234	33,47%	67,62%	-	3846 sec	16 sec
	237	182	6	4	6	0	1	1	1	6	3	90	100	246	2	0	49	195	179	243	32,38%	68,56%	-	4525 sec	18 sec
9	237	182	6	4	6	0	1	1	1	3	3	100	100	246	8	1	64	173	184	232	34,89%	67,62%	-	3807 sec	15 sec
	237	182	6	4	6	0	1	1	1	6	3	100	100	246	3	2	45	196	181	243	32,72%	68,09%	-	4457 sec	18 sec

of n provides higher probability of obtaining a position estimate that is at most 300 meters away from the actual location of a tracked mobile station. Performance improvements in first three pairs (related to this probability) are more noticeable than those in other pairs, partially supporting the relationship explained above. However, there are cases where such an increase in n degrades positioning performance, even when large cells are used. Selection of other algorithmic parameters is likely the cause for such situations.

8.2.6 Maximum Cluster Count

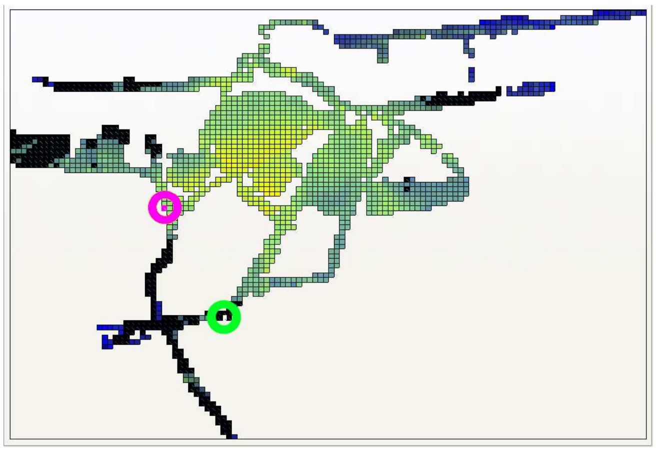
Maximum cluster count parameter defines the maximum number of clusters that can be generated by k-means clustering algorithm at the end of each iteration of the proposed position estimation algorithm. Table 8.7 shows how changing the algorithmic parameter's value impacts estimation performance.

Probability of obtaining a position estimate that is at most 100 meters away from the actual location of a tracked mobile station increases in all experiments when maximum cluster count is increased from one to two. However, lower probability is obtained when the parameter is set to a higher value as shown in last experiments of the second and third groups in Table 8.7. This performance reduction hints at an upper limit for the value of maximum cluster count.

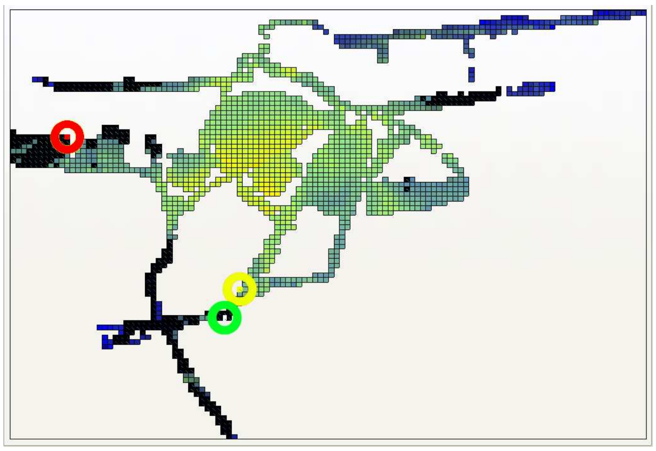
Probability of obtaining a position estimate that is at most 300 meters away from the actual location of a tracked mobile station decreases in all experiments which have maximum cluster count values greater than one. Clearly, behavior of this probability is different from that of the former one, the reason of which can be explained based on screenshots shown in Figure 8.1. The screenshots depict position estimates of a mobile station computed by the proposed algorithm when maximum cluster counts of one, two and three are used, respectively. Green circles show actual location of the mobile station. Yellow circles show estimates that are less than 100 meters away from the green circles, pink circles show estimates that are more than 100 meters and less than 300 meters away from the green circles, and red circles show estimates that are more than 300 meters away from the green circles. Probability of selecting an estimate (from returned estimation results) which has a positioning error less than 100 meters is 0, $\frac{1}{2}$ and $\frac{1}{3}$ for the first, second and third screenshots in Figure 8.1. That means, switching from one cluster to two clusters provides higher probability, while switching from two clusters to three clusters causes a decrease in value of

Table 8.7: A set of experiments belonging to the proposed algorithm, demonstrating effects of maximum cluster count on results.

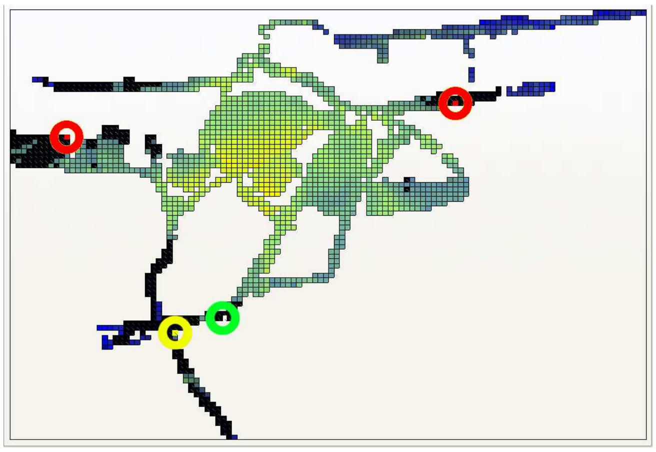
	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	8	6	8	7	1	2	1	3	1	80	100	188	0	188	0	0	42	146	22,34%	77,66%	242,32	237 sec	2 sec
	60	46	8	6	8	7	1	2	1	3	2	80	100	188	0	2	186	0	91	163	27,66%	65,16%	345,13	318 sec	2 sec
2	118	90	8	6	8	7	1	2	1	3	1	80	100	188	0	188	0	0	43	142	22,87%	75,53%	247,2	1595 sec	9 sec
	118	90	8	6	8	7	1	2	1	3	2	80	100	188	0	1	187	0	90	164	28,46%	64,89%	354,02	1743 sec	10 sec
	118	90	8	6	8	7	1	2	1	3	3	80	100	188	0	1	15	172	107	181	25,09%	62,68%	359,89	1879 sec	10 sec
3	237	182	7	5	7	6	1	1	1	3	1	90	100	188	0	188	0	0	44	148	23,4%	78,72%	250,74	9952 sec	53 sec
	237	182	7	5	7	6	1	1	1	3	2	90	100	188	0	1	187	0	95	165	30,05%	62,5%	358,87	10173 sec	55 sec
	237	182	7	5	7	6	1	1	1	3	3	90	100	188	0	1	42	145	106	184	27,39%	61,17%	357,81	10441 sec	56 sec



(a)



(b)



(c)

Figure 8.1: Position estimates of a mobile station computed by the proposed algorithm when maximum cluster counts of one (a,) two (b) and three (c) are used. Green circles show actual location of the mobile station and others show position estimates.

the probability. This supports the results given in Table 8.7. Probability of selecting an estimate which has a positioning error less than 300 meters is 1, $\frac{1}{2}$ and $\frac{1}{3}$ for the first, second and third screenshots, respectively. Therefore, as the number of clusters increases, the latter probability decreases, which also supports the results mentioned before.

One last thing to note is that root mean square error belonging to an experiment increases most of the time if its maximum cluster count value is increased.

8.2.7 Experiment Length

Experiment length is the number of inputs to be processed (or the number of algorithmic iterations to be executed) before an experiment is finished. Results of earlier experiments, which were carried out using manually generated scenario data, show that positioning performance fluctuates as the number of iterations increases. Performance generally gets better when 40, 50, 90 or 100 iterations are executed.

Table 8.8 presents experiment results obtained with real world data. Fluctuations in performance can partially be seen here, too. Probability of obtaining a position estimate that is at most 100 meters away from the actual location of a tracked mobile station noticeably decreases in first and second experiment groups when experiment length is increased. In third group, performance reduction is less noticeable, and in fourth group performance increases as value of the algorithmic parameter is increased. Probability of obtaining a position estimate that is at most 300 meters away from the actual location of a tracked mobile station also decreases in first and second experiment groups as the parameter's value is increased; but it increases in third and fourth groups. Lastly, root mean square error increases every time the parameter's value is increased; however, the amount of increase in error in last two experiment groups is less than that in first two experiment groups. Therefore, it looks like 90 or 100 iterations tend to increase (or slightly decrease) positioning performance in general. It is for sure that selection of other algorithmic parameters impacts the value of experiment length to be used.

Table 8.8: A set of experiments belonging to the proposed algorithm, demonstrating effects of experiment length on results.

	Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
1	60	46	4	3	4	1	2	1	1	3	1	10	100	188	40	148	0	0	18	91	9,57%	48,4%	451,92	64 sec	1 sec
	60	46	4	3	4	1	2	1	1	3	1	40	100	188	70	118	0	0	12	70	6,38%	37,23%	548,34	56 sec	1 sec
2	60	46	6	4	6	2	1	1	1	3	2	40	100	188	33	2	153	0	60	125	16,76%	46,28%	467,92	148 sec	1 sec
	60	46	6	4	6	2	1	1	1	3	2	80	100	188	44	6	138	0	44	109	11,97%	40,69%	514,61	146 sec	1 sec
3	237	182	6	4	6	0	1	1	1	3	2	90	100	188	52	2	134	0	38	102	11,44%	36,44%	551,07	3735 sec	20 sec
	237	182	6	4	6	0	1	1	1	3	2	100	100	188	57	3	128	0	37	105	11,17%	38,56%	567,13	2529 sec	14 sec
4	237	182	6	4	6	4	1	1	1	3	2	90	100	188	6	7	175	0	81	150	25,53%	56,12%	391,89	5051 sec	27 sec
	237	182	6	4	6	4	1	1	1	3	2	100	100	188	7	7	174	0	84	156	26,86%	58,78%	396,01	4955 sec	27 sec

8.3 k Nearest Neighbor Algorithm

Simulation results of k Nearest Neighbor (kNN) algorithm are in the form shown in Table 8.9. First five columns indicate algorithmic parameters that were used in that experiment while rest of the columns shows the results. Information contained in each column is listed as follows:

- 1st and 2nd columns: Horizontal and vertical grid cell counts of the map used.
- 3rd column: Number of cells (k in Figure 7.13.) that were identified by the kNN algorithm to determine a final position estimate.
- 4th column: Capacity of the array (window in Figure 7.12) which stored fingerprints received from a tracked mobile station.
- 5th column: Number of subsequent fingerprints that made up a scenario. This parameter is the same as the one used by the proposed position estimation algorithm.
- 6th column: Total number of estimations that were carried out. Note that in kNN algorithm, every time an incoming signal parameter set is received from a mobile station, a final position estimate is computed without using information related to previous iterations, unlike the proposed algorithm. This is why in the example shown in Table 8.9, it is stated that 18300 estimations were recorded. One may think that there should have been 18800 estimations since there were 188 scenarios, each containing 100 fingerprints. But, because of that a few input fingerprints make up one input for the kNN algorithm (window explained in 4th column) the actual number is less than that.

Columns 7th to 18th contain results of the experiment defined by above parameters.

- 7th column: Number of position estimates, computed by mean weighting function, that were at most 100 meters away from the actual location of a tracked mobile station.
- 8th column: Percentage of the number of position estimates, which were at most 100 meters away from the actual location of a tracked mobile station, to the total number of estimates. It is possible to compare this percentage value to the average probability of selecting a cell, which is less than 100 meters away from the actual location of a

Table 8.9: An example of the k Nearest Neighbor algorithm's simulation results.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	1200	8	100	18300	4642	25%	15050	82%	223,46	4472	24%	13124	71%	279,16	169 sec	0 sec

tracked mobile station, from the result set of an estimation operation belonging to the proposed positioning algorithm. Because of that $p_e^{(i)}$ in Equation 8.2 (average probability formula) will be either zero or one in kNN, numerator of the equation will be equal to the number position estimates that were at most 100 meters away from the actual location. Since kNN returns exactly one estimate after each positioning operation, 'number of estimations with one or more results' in the denominator part of the equation will be equal to the total number of kNN estimations. Therefore, computing Equation 8.2 for kNN gives the value of this percentage.

- 9th column: Number of position estimates, computed by mean weighting function, that were at most 300 meters away from the actual location of a tracked mobile station.

- 10th column: Percentage of the number of position estimates, which were at most 300 meters away from the actual location of a tracked mobile station, to the total number of estimates. Explanations in 8th column are valid for this one, too. The percentage value is comparable to the average probability of selecting a cell, which is less than 300 meters away from the actual location of a tracked mobile station, from the result set of an estimation operation belonging to the proposed position estimation algorithm.

- 11th column: Root mean square error of estimation results that were computed according to mean weighting function. This is an ordinary root mean square error computation since kNN returns exactly one result at the end of each position estimation operation (unlike the proposed algorithm.)

- 12th column: Number of position estimates, computed by radial basis function, that were at most 100 meters away from the actual location of a tracked mobile station.

- 13th column: Percentage of the number of position estimates computed by radial basis function, which were at most 100 meters away from the actual location of a tracked mobile station, to the total number of estimates. This percentage value is comparable to the average probability of selecting a cell, which is less than 100 meters away from the actual location of a tracked mobile station, from the result set of an estimation operation belonging to the proposed positioning algorithm.

- 14th column: Number of position estimates, computed by radial basis function, that were at most 300 meters away from the actual location of a tracked mobile station.

- 15th column: Percentage of the number of position estimates computed by radial basis function, which were at most 300 meters away from the actual location of a tracked mobile station, to the total number of estimates.
- 16th column: Root mean square error of estimation results that were computed according to radial basis function.
- 17th column: Total time spent for the experiment on a computer powered by a 1.6GHz Intel Pentium M processor.
- 18th column: Average time spent for one position estimation operation on a computer powered by a 1.6GHz Intel Pentium M processor.

8.4 Algorithmic Parameters' Effects on Estimation Results of k Nearest Neighbor Algorithm

In this section, results of k Nearest Neighbor (kNN) algorithm will be discussed on their own. Algorithmic parameters' effects on estimation results will be expressed as in Section 8.2. The main indicators to be mentioned are

- percentages of the number of position estimates, which are in certain proximity to the actual location of a tracked mobile station, to the total number of estimates, and
- root mean square error of estimation results that were computed according to mean weighting function.

Results related to mean weighting function are employed as indicators; because, that function provided more accurate position estimates than radial basis function in almost all of the experiments. Average estimation time will not be mentioned as an indicator since it was less than one seconds for each experiment.

8.4.1 Horizontal and Vertical Grid Cell Counts

Table 8.10 shows algorithmic parameters and results of seven experiments assembled into three groups. Experiments in each group consist of the same algorithmic parameter set except

Table 8.10: A set of experiments belonging to the k Nearest Neighbor algorithm, demonstrating effects of grid cell count on results.

	Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
	60	46	40	1	100	18800	4742	25%	13898	73%	264,90	3681	19%	11912	63%	321,04	19 sec	0 sec
1	118	90	40	1	100	18800	5059	26%	13496	71%	276,65	3820	20%	12100	64%	314,28	52 sec	0 sec
	237	182	40	1	100	18800	5150	27%	13478	71%	274,07	4587	24%	13152	69%	284,55	158 sec	0 sec
2	60	46	120	1	100	18800	4813	25%	14420	76%	244,18	3681	19%	11912	63%	321,03	20 sec	0 sec
	237	182	120	1	100	18800	5079	27%	13754	73%	269,02	4554	24%	13182	70%	283	157 sec	0 sec
3	118	90	200	4	100	18800	4552	24%	14214	75%	253,94	3892	20%	12278	65%	310,68	54 sec	0 sec
	237	182	200	4	100	18800	4973	26%	13943	74%	265,93	4560	24%	13227	70%	284,68	156 sec	0 sec

for horizontal and vertical cell counts.

As demonstrated by the examples in Table 8.10, higher horizontal and vertical cell counts increases percentage of the number of position estimates, which are at most 100 meters away from the actual location of a tracked mobile station, to the total number of estimates. On the other hand, such an increase in algorithmic parameters causes a decrease in percentage of position estimates which are at most 300 meters away from the actual location of a tracked mobile station. Root mean square error of estimates also increases in most cases, which is an undesirable situation as mentioned before.

8.4.2 Neighbor Count

Table 8.11 lists a number of experiments that demonstrate how positioning performance of kNN algorithm changes with different values of neighbor count, the number of cells used by the algorithm to compute a position estimate.

As seen in experiments belonging to the first group in Table 8.11 (experiments with 60x46 cells,) number of position estimates that are at most 100 meters away from the actual location of a tracked mobile station increases little by little as neighbor count is increased from 40 to 200. Therefore, value of the corresponding percentage increases slightly. In the second group, where 118×90 cells are used, the same percentage decreases when the algorithmic parameter's value is increased from 40 to 200. However, increasing it further may provide a small amount of increase in the percentage value. In the last group, where 237×182 cells are used, value of the percentage decreases as neighbor count is increased from 40 up to 4000, regardless of fluctuations in the number of position estimates that have a maximum positioning error of 100 meters.

Percentage value related to the position estimates which are at most 300 meters away from the actual location of a tracked mobile station is affected differently by the change in neighbor count parameter. In the first group in Table 8.11, value of that percentage increases as the algorithmic parameter is increased. In cases where 118×90 cells are used, increased percentage is obtained as the parameter's value is increased (provided that the value assigned to the parameter does not exceed 500.) In the third group, value of the percentage peaks when a neighbor count between 1200 and 1600 is used.

Table 8.11: A set of experiments belonging to the k Nearest Neighbor algorithm, demonstrating effects of neighbor count on results.

	Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
	60	46	40	4	100	18800	4754	25%	13900	73%	264,95	3760	20%	12059	64%	319,28	19 sec	0 sec
1	60	46	80	4	100	18800	4770	25%	14276	75%	254,88	3760	20%	12059	64%	319,28	19 sec	0 sec
	60	46	120	4	100	18800	4884	25%	14397	76%	244,43	3760	20%	12059	64%	319,28	20 sec	0 sec
	60	46	200	4	100	18800	4910	26%	14894	79%	231,85	3760	20%	12059	64%	319,28	21 sec	0 sec
	118	90	40	1	100	18800	5059	26%	13496	71%	276,65	3820	20%	12100	64%	314,28	52 sec	0 sec
	118	90	200	1	100	18800	4525	24%	14234	75%	253,39	3824	20%	12119	64%	314,07	54 sec	0 sec
2	118	90	300	1	100	18800	4537	24%	14401	76%	242,96	3824	20%	12119	64%	314,07	54 sec	0 sec
	118	90	400	1	100	18800	4596	24%	14776	78%	235,05	3824	20%	12119	64%	314,07	56 sec	0 sec
	118	90	600	1	100	18800	4635	24%	14865	79%	231,72	3824	20%	12119	64%	314,07	60 sec	0 sec
	237	182	40	1	100	18800	5150	27%	13478	71%	274,07	4587	24%	13152	69%	284,55	158 sec	0 sec
	237	182	400	1	100	18800	4744	25%	14173	75%	256,76	4542	24%	13185	70%	282,81	159 sec	0 sec
	237	182	800	1	100	18800	4551	24%	14499	77%	242	4542	24%	13185	70%	282,81	162 sec	0 sec
3	237	182	1200	1	100	18800	4606	24%	15146	80%	232,09	4542	24%	13185	70%	282,81	165 sec	0 sec
	237	182	1600	1	100	18800	4636	24%	15239	81%	229,48	4542	24%	13185	70%	282,81	168 sec	0 sec
	237	182	2000	1	100	18800	4595	24%	14873	79%	231,02	4542	24%	13185	70%	282,81	173 sec	0 sec
	237	182	4000	1	100	18800	4245	22%	14445	76%	234,7	4542	24%	13185	70%	282,81	194 sec	0 sec

Finally, as demonstrated by almost all of the experiments in Table 8.11, root mean square error is inversely proportional to percentage of position estimates which are in 300 meter proximity of the actual location of a tracked mobile station. That means, as value of that percentage increases, root mean square error decreases.

8.4.3 N

n is the number of incoming fingerprints from a tracked mobile station, the average of which is fed into the proposed position estimation algorithm as an input. According to the results of experiments that were carried out, n does not have a noticeable effect on positioning performance most of the time. But, for some values of n it is possible to see a little improvement on percentages of position estimates that are in certain proximity to the actual location of a tracked mobile station. Table 8.12 lists a number of experiments that gave such results. As it can be seen in both experiment groups in the table, values of 1 and 4 for the algorithmic parameter do not cause any significant changes in results. Slight improvements are achieved (both in percentages and root mean square error) when greater values are assigned to n.

8.5 Comparison of Proposed Position Estimation Algorithm and k Nearest Neighbor Algorithm

In this section, experimental results of the proposed position estimation algorithm and k Nearest Neighbor (kNN) algorithm will be compared to come up with a discussion about the former's positioning performance. Table 8.13 lists algorithmic parameters and results of eleven experiments belonging to the proposed positioning algorithm. Table 8.14 lists information related to ten kNN experiments. Experiments in the tables are selected in the following way: for each experiment of the proposed algorithm, a performance criterion was computed by giving 65% weight to the probability of obtaining a position estimate with less than 100 meter error and 35% weight to the probability of obtaining a position estimate with less than 300 meter error. Same computation was repeated for all experiments of the kNN algorithm. Experiments belonging to each algorithms were sorted on their own by the computed performance criterion in decreasing order, and top experiments were presented in

Table 8.12: A set of experiments belonging to the k Nearest Neighbor algorithm, demonstrating effects of the algorithmic parameter ‘n’ on results.

	Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
1	118	90	60	1	100	18800	4977	26%	13682	72%	272,12	3822	20%	12116	64%	314,13	53 sec	0 sec
	118	90	60	4	100	18800	5007	26%	13695	72%	272,83	3895	20%	12278	65%	310,72	52 sec	0 sec
	118	90	60	6	100	18600	4983	26%	13599	73%	272,24	3900	20%	12147	65%	308,35	52 sec	0 sec
	118	90	60	8	100	18300	4960	27%	13582	74%	265,76	3896	21%	12145	66%	301,33	52 sec	0 sec
2	237	182	4000	1	100	18800	4245	22%	14445	76%	234,70	4542	24%	13185	70%	282,81	194 sec	0 sec
	237	182	4000	4	100	18800	4231	22%	14429	76%	235,02	4555	24%	13229	70%	284,63	193 sec	0 sec
	237	182	4000	6	100	18600	4230	22%	14421	77%	233,34	4461	23%	13080	70%	285,74	197 sec	0 sec
	237	182	4000	8	100	18300	4227	23%	14411	78%	226,49	4472	24%	13124	71%	279,16	197 sec	0 sec

Table 8.13: Algorithmic parameters and results of eleven experiments belonging to the proposed position estimation algorithm.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	avg. est. time
237	182	7	5	7	10	1	3	1	3	1	90	100	188	0	188	0	23,4%	79,79%	248,81	57 sec
237	182	7	5	7	6	1	1	1	3	1	90	100	188	0	188	0	23,4%	78,72%	250,74	53 sec
237	182	10	8	10	8	1	3	1	3	2	90	100	188	0	0	188	29,79%	65,16%	344,18	150 sec
118	90	10	8	10	4	1	1	1	3	1	80	100	188	0	188	0	21,81%	79,79%	242,08	14 sec
237	182	7	5	7	4	1	1	1	3	2	90	100	188	0	1	187	30,85%	62,77%	360,36	47 sec
237	182	10	8	10	7	1	2	1	3	2	90	100	188	0	0	188	29,52%	65,16%	345,79	149 sec
237	182	10	8	10	8	1	3	1	3	1	90	100	188	0	188	0	20,74%	80,85%	242,55	134 sec
237	182	7	5	7	7	1	2	1	3	2	90	100	188	0	1	187	30,32%	63,03%	358,66	54 sec
237	182	10	8	10	4	1	1	1	3	2	90	100	188	0	0	188	29,26%	64,89%	346,93	134 sec
60	46	8	6	8	7	1	2	1	3	1	80	100	188	0	188	0	22,34%	77,66%	242,32	2 sec
237	182	7	5	7	6	1	1	1	3	2	90	100	188	0	1	187	30,05%	62,5%	358,87	55 sec

Table 8.14: Algorithmic parameters and results of ten experiments belonging to the k Nearest Neighbor algorithm.

1	2	3	4	5	6	7	8	9	10	11	12
Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	avg. est. time
237	182	1200	8	100	18300	4642	25%	15050	82%	223,46	0 sec
237	182	1600	8	100	18300	4691	25%	15117	82%	220,63	0 sec
118	90	500	8	100	18300	4695	25%	14971	81%	222,59	0 sec
237	182	1200	6	100	18600	4664	25%	15135	81%	230,51	0 sec
60	46	200	3	100	18800	4902	26%	14905	79%	231,92	0 sec
60	46	200	4	100	18800	4910	26%	14894	79%	231,85	0 sec
118	90	400	8	100	18300	4642	25%	14744	80%	226,99	0 sec
118	90	500	6	100	18600	4691	25%	15018	80%	229,67	0 sec
118	90	600	8	100	18300	4636	25%	14775	80%	223,26	0 sec
237	182	2000	8	100	18300	4622	25%	14767	80%	222,24	0 sec

Table 8.13 and Table 8.14.

Data to be compared are listed as the following:

- Average probability of obtaining a cell, which is within certain proximity to the actual location of a tracked mobile station, from the result set of an estimation (18th and 19th columns in Table 8.13,) and percentage of the number of position estimates, which have certain distance to the actual location of a tracked mobile station, to the total number of estimates (8th and 10th column in Table 8.14.) Recall that the percentage in kNN results is computed the same way as the average probability in the proposed algorithm's results as explained in Section 8.3.

- Root mean square errors (20th column in Table 8.13 and 11th column in Table 8.14)

- Average estimation times (21th column in Table 8.13 and 12th column in Table 8.14)

A general first look at the tables reveals that experiments of the kNN algorithm yielded very similar results, no matter how different their algorithmic parameters were. Experimental results of the proposed algorithm show more variability. While probability of obtaining a position estimate with less than 100 meter error does not exceed 26% in Table 8.14, it reaches up to almost 31% in Table 8.13. On the other hand, probability of obtaining a position estimate with less than 300 meter error is around 65% or less in half of the experiments in Table 8.13 and around 80% in the rest while the same probability is always around 80% in Table 8.14. There is one thing that attracts attention about the experiments in Table 8.13, latter probabilities of which are around 80%. Maximum cluster count values of those experiments (11th column) are one. Taking a look at the results of those experiments makes it clear that when the proposed positioning algorithm is dictated to generate at most one cluster, it produces results similar to those yielded by the kNN algorithm. Because of that, results of experiments which were carried out using a maximum cluster count value of two (or more) exhibit the actual positioning performance of the proposed algorithm. Therefore, it is a better idea to compare results of only those experiments in Table 8.13 with kNN results in Table 8.14.

As mentioned above, proposed algorithm's probability of obtaining a position estimate with less than 100 meter error is higher than that of the kNN algorithm, and the proposed algorithm's probability of obtaining a position estimate with less than 300 meter error is lower than that of the kNN algorithm. The reason for the proposed algorithm's latter probability to be lower can be understood by looking at an example like the one in Figure 8.2. In the figure, green circle shows the actual location of a tracked mobile station, orange circles show position estimates calculated by the proposed algorithm, and red circle shows estimate of the kNN algorithm. Orange estimate on lower left side of the map is the only one that is less than 100 meters away from the green circle. So, the proposed algorithm's former probability is $\frac{1}{3}$ and the kNN algorithm's former probability is 0. This explains why the proposed algorithm's former probability is higher. The same orange circle and the red circle are the two estimates which are less than 300 meters away from the green circle. That means the kNN algorithm's latter probability is 1 while the same probability of the proposed algorithm is $\frac{1}{3}$. Apparently, such situations occur many times, causing latter probability value of the proposed algorithm to be less than that of the kNN algorithm.

Root mean square errors in Table 8.13 and Table 8.14 exhibit a relationship that is consistent with the one between probabilities explained above. Root mean square errors belonging to the experiments in Table 8.13 that have a maximum cluster count value of two are greater than those in Table 8.14, implying that positioning errors in the proposed algorithm are worse than

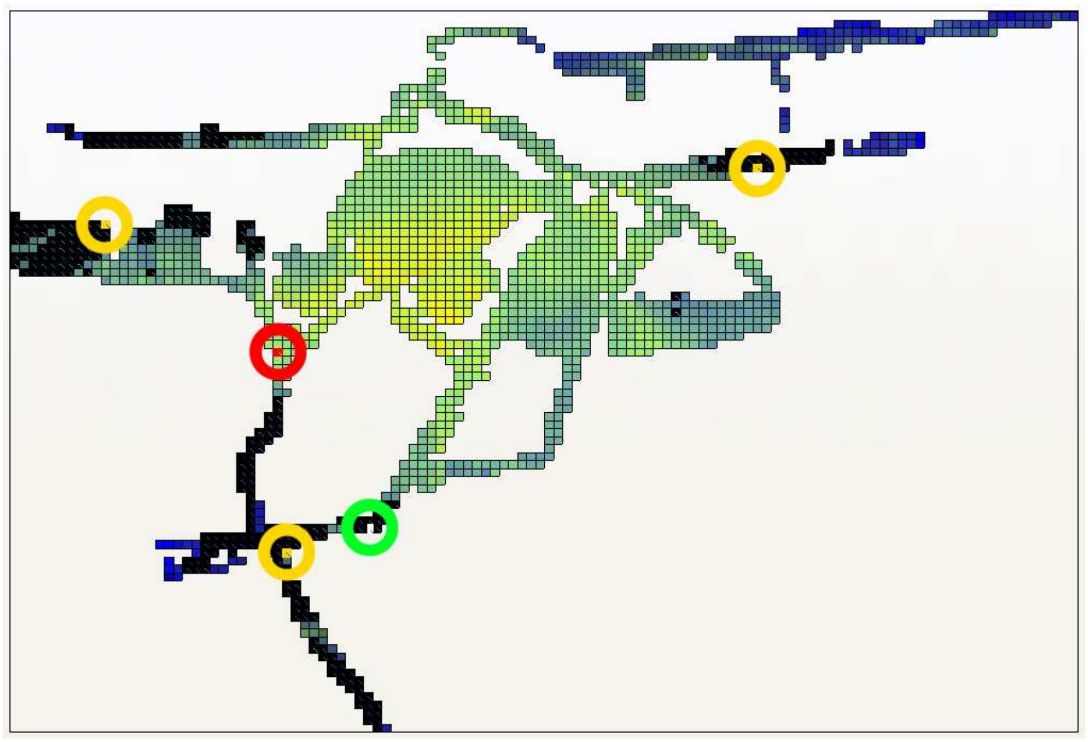


Figure 8.2: A position estimation example. Green circle shows the actual location of a tracked mobile station. Orange circles show position estimates calculated by the proposed algorithm, and red circle shows estimate of the k Nearest Neighbor algorithm.

those in the kNN algorithm. The reason for such big error values in the proposed algorithm can be comprehended by looking at Figure 8.3, a different view of the case in Figure 8.2. In the figure, positioning error of the kNN algorithm's estimate (red circle) is simply the distance between the estimate and the actual location of the tracked mobile station (green circle,) which is less than 300 meters. When it comes to calculating positioning error of the proposed algorithm, all three estimates represented by orange circles are included in the computation. Because of that second and fourth distances in Figure 8.3 are around 500 and 600 meters away from the green circle, respectively, proposed algorithm's root mean square error is higher than 300 meters (higher than kNN algorithm's error.)

The last comparison will be made between average estimation times of the two algorithms. There is an obvious difference between the average estimation times listed in Table 8.13 and Table 8.14. The kNN algorithm seems to finish an estimation operation in less than one second. However, in real world application, it will take a few seconds because the algorithm has to wait for a number of fingerprints that will be sent by a tracked mobile station. Still,

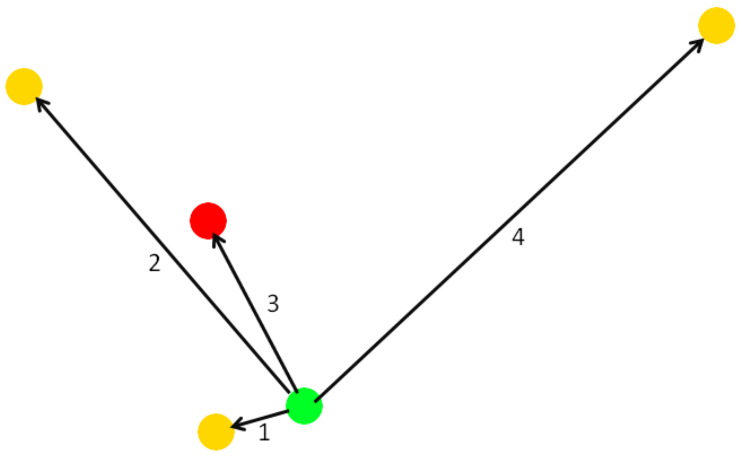


Figure 8.3: A different view of the case in Figure 8.2

average estimation time of the kNN algorithm will be less than that of the proposed algorithm since the former does not require multiple iterations and more incoming fingerprints like the latter does. Estimation time of the proposed algorithm mainly depends on the algorithmic parameter called experiment length which defines the number of iterations to be executed. If higher values are assigned to the parameter (as is the case in experiments in Table 8.13) estimation times in real world applications will be as long as what is shown in Table 8.13. Actually, it is normal to have such values; because, if experiment length is set to 90, that means 90 fingerprints are expected to be received from the tracked mobile station in order to come up with a set of final position estimates. Since reception of an incoming fingerprint fires an algorithmic iteration and there is available processing time until reception of the next fingerprint, it is likely that a few seconds after the last fingerprint is obtained final estimates will be ready. During collection of signal parameters for the training database and for the scenarios, every time the measurement equipment moved one meter, one fingerprint was recorded. Similarly, if a tracked mobile station sends a fingerprint every time it travels one meter, it takes more than one minute to send 90 fingerprints. Therefore, even if the proposed algorithm is able to process that many fingerprints in one second, it has to wait more than one minute to receive the last incoming fingerprint and process it just like previously received ones.

One thing that had a huge impact on average estimation time was the hardware utilized for simulations. The computer which ran simulations to produce data in Table 8.13 and Table 8.14

used an Intel Pentium M 725 1.6GHz processor that was released in 2004. Proposed position estimation algorithm worked two times faster when simulations were repeated using one core of a dual core processor. Considering that dual core processors are the standard now and that the proposed algorithm can be parallelized easily (because numerous independent fingerprint comparisons are made,) position of a mobile station can be estimated at least four times faster using an ordinary up-to-date computer. In that case, execution time drops to much more acceptable levels. The proposed algorithm will still execute longer than the kNN algorithm by its nature, but it is possible to further decrease the execution time by keeping the experiment length parameter small, moderately risking accuracy of final estimates.

8.5.1 Total Number of Estimations

Apart from what is explained above, there are several other factors that worth mentioning like the total number of estimation operations carried out during an experiment. Scenario data used in experiments consisted of 188 paths, each one of which included around 100 fingerprints. As seen in Table 8.13, in one experiment of the proposed algorithm, 188 estimations are completed (one estimation for each path in the scenario data.) However, in Table 8.14 more than 18000 estimations are handled in one experiment (one estimation operation for each fingerprint in the scenario data.) This huge difference of the total number of estimations may imply some inequality between results of the two algorithms. It might be possible for the proposed algorithm to yield better and more accurate results if a lot more estimations were executed.

8.5.2 k Nearest Neighbor Estimates

Experiments revealed that estimated location of a mobile station that is computed by the k Nearest Neighbor (kNN) algorithm points to somewhere around the central region of a base station's coverage area most of the time. When a mobile station moves around the central region of a base station's coverage area and sends fingerprints to the kNN algorithm, a number of cells in the central region are identified as possible places where the mobile station can be. Processing those cells yields an estimate that points to somewhere around the central region, naturally, and the estimate's root mean square error is likely to be less than a certain value. The situation changes a little when the mobile station moves towards other regions of the area.

Consider that it is moving to the crossroad on lower left side of the map in Figure 7.11. The algorithm marks a predefined number of cells according to the incoming fingerprints sent by the mobile station (red cells in the figure.) Processing those cells yields a position estimate that is in the middle of the center and left border of the figure (pink cell,) which corresponds to a place not far away from the central region. Positioning error of this estimate does not exceed a certain value like the one explained above. Therefore, it can be said that as long as the mobile station does not move to outer regions of the examined area, position estimates will point to locations around the central region, and errors will be in acceptable levels. These two positioning cases seem to occur a lot when small maps are used, like the one in Figure 7.11.

Scenario data used for simulations mostly consisted of fingerprints collected around the central region of the coverage area and its neighboring regions. Fewer fingerprints were collected in outer regions because of issues related to topology and measurement equipments. This might have a positive effect on the results of kNN estimations.

8.5.3 Selection of Parameter Values in the Proposed Algorithm

As mentioned before, selection of well matched algorithmic parameters is important for the proposed algorithm to track and locate mobile stations as accurately as possible. For example, small maximum signal parameter differences (Section 8.2.2) coupled with a small maximum uncontinuity (Section 8.2.3) can very easily cause problems in tracking, like completely being unable to track a mobile station. The reason is that signal parameters of subsequently received fingerprints tend to fluctuate frequently. Because of such fluctuations, in subsequent iterations of the algorithm, different cells will be identified as possible locations where the mobile station can be, and there will be no or few common cells that are marked in those iterations. Since common cells are the key to tracking, it will fail in such a case. Similarly, selecting very high values for those parameters does not ensure obtaining the best possible positioning performance from the algorithm; because, too many cells will be identified as possible locations, leading the algorithm to provide inaccurate estimates. Consider a maximum uncontinuity value of 10. That means cells which are normally marked for removal (from being possible cells) will be kept available for ten more algorithmic iterations. It is not wise to give such a great flexibility to the algorithm; because, in cases where uncontinuity is not required, that is, when signal transmission is robust and reliable, cells that should be removed will be given extra chances unnecessarily for multiple iterations. A mobile station may move through a few cells and provide ten fingerprints to the algorithm during its short trip. If that mobile station is tracked successfully, then it will be totally redundant to save the cells, which should be removed previously, for ten more algorithmic steps. The actual downside is that those cells will cause the algorithm to track extra paths and generate different clusters than what should normally be obtained. These examples are strong indicators that it is important to optimize the algorithmic parameters for the positioning case at hand.

8.5.4 Training and Scenario Data

The problem with training and scenario data was that they consisted of fingerprints that were collected in different seasons. Training data was collected in June, and WiMAX signals could be received from all of the colored regions in the map depicted in Figure 6.2(a). Scenario data, on the other hand, was collected in December, and WiMAX signals could not be received in certain regions of the map, e.g. in the blue road on lower left side of the figure. That was why it was not possible to do experiments related to some outer regions of the map. Experimental results of the proposed position estimation algorithm and k Nearest Neighbor (kNN) algorithm might be different if scenario data were collected during summer time.

It is thought that because of environmental changes due to seasonal variation, there might have been differences in fingerprints corresponding to the same location in training and scenario data. These differences might have affected experiment results negatively; however, maximum signal parameter differences (Section 8.2.2) defined in the proposed algorithm probably compensated for estimation problems related to such differences.

CHAPTER 9

CONCLUSION

In this thesis project, a position estimation algorithm for single base station WiMAX networks is proposed. Before the algorithm was developed, WiMAX equipment at hand was investigated to determine what kind of information about transmitted signals and network packets could be obtained. It was found out that only Received Signal Strength Indicator (RSSI,) Carrier to Interference-plus-Noise Ratio (CINR) and Average Transmit Power (AvgTxPwr) were available. In order to compensate for the lack of sufficient information (to be used in positioning) originating from utilization of a single base station, fingerprinting technique was decided to be used. The technique allowed analyzing coverage area of the employed WiMAX base station. Parameters of received signals in downlink direction (from base station to mobile station) were measured in different places using a WiMAX-enabled mobile station and Differential GPS (DGPS) equipment. These measurements made it possible to tell what kind of signal parameters were expected at certain points in the coverage area. Data collected via fingerprinting formed a training database which the proposed algorithm was based on. A signal map was created using that training database and it was divided into a (variable) number of square grid cells (or shortly *cells*.) Cells were preferred to be used in position estimation; because, averaging signal parameters of points belonging to individual cells could remove possible measurement errors in individual fingerprints. Therefore, average signal parameters of each cell were computed to generate a grid map from the signal map, and only that map was employed by the proposed algorithm.

The main idea behind the proposed algorithm was to receive signal measurements from a mobile station to be located, query a training database for records that contain signal parameters similar to the ones reported by that mobile station, and return coordinates stored in resulting records as possible position estimates. This idea remained the same during development phase of the algorithm; but, some modifications had to be done (e.g. using a grid map as the training database as mentioned above) and a number of problems had to be solved to realize it. First, it was known that one base station would not provide adequate data to accurately locate a mobile station. It seemed unlikely to find the location of a mobile station correctly in a single step using a single signal measurement (a single parameter set) received from the mobile; because, it would not be reliable to use only one incoming parameter set since WiMAX signals tended to fluctuate, and most of the time there were many cells which had the same signal parameters as the incoming ones. Solution to the former was to gather multiple incoming signal parameters and calculate their average to smoothen out possible errors in measurements. Solution to the latter was tracking mobile stations, meaning that the algorithm would consist of multiple iterations. Basically, in each iteration some cells were marked according to their Euclidean distance of signal parameters to those sent by a mobile station. It was checked whether those cells were among the ones which were identified as possible position estimates in a previous iteration (and among their neighboring cells.) Existence of such cells implied that the paths the mobile station might be following could really be tracked.

In early versions of the proposed position estimation algorithm, it was observed that the number of estimated cells at the end of each iteration was more than a few, which made it impractical to return all of them as results. On the other hand, as revealed by visual examinations, those cells were organized into a few groups, implying that one estimate per group could be returned as results. Therefore, k-means clustering algorithm, coupled with Homogeneity validity index (a method that helps to search for a proper number of clusters as explained in Section 7.1.3,) was integrated into the algorithm to provide an acceptable number of final position estimates to users. The algorithm generally generated more than one position estimates.

In order to investigate how the proposed algorithm performed against a simple position estimation solution, k Nearest Neighbor (kNN) algorithm was implemented. Similar to what was done in the proposed algorithm; averages of multiple incoming signal parameters sent from a mobile station were fed into the kNN algorithm as input. kNN did not incorporate any kind of tracking methods. It processed a given input to identify a number of cells, put those cells through a weighting function and came up with exactly one position estimate in one step. Note that, the proposed algorithm returned no estimates when algorithmic parameters were selected randomly without considering their effects on positioning performance. Experiments were carried out on both algorithms and the following results were obtained.

- With the proposed algorithm, it was possible to get a higher average probability of selecting a cell, which was less than 100 meters away from the actual location of a tracked mobile station, from the result set of an estimation operation.
- With the kNN algorithm, it was possible to get a higher average probability of selecting a cell, which was less than 300 meters away from the actual location of a tracked mobile station, from the result set of an estimation operation.
- There was not a case in either the proposed or the kNN algorithm where both the former and the latter probabilities had their highest possible values. In experiments of the proposed algorithm where the former probability was high (around 30%,) the latter probability was low (around 65%,) and in experiments of the kNN algorithm where the latter probability was high (around 80%,) the former probability was low (around 25%.)

"Accuracy and reliability requirements" for Enhanced 9-1-1, a service that is used to determine location of a cellular phone user in an emergency situation, are specified by the Federal Communications Commission (FCC) as the following [16].

- "For network-based solutions: 100 meters for 67 percent of calls, 300 meters for 95 percent of calls;
- For handset-based solutions: 50 meters for 67 percent of calls, 150 meters for 95 percent of calls."

Experimental results suggest that the proposed algorithm does not comply with above requirements and it can be used for non-emergency situations where low accuracy is acceptable.

Advantages of the proposed algorithm can be listed as following.

- There is no hardware modification required for any network equipment (base stations and mobile stations.)

- Although the algorithm was developed as a remote positioning solution in the first place, it can easily be implemented as a self positioning solution. In remote positioning case, a tracked mobile station measures various WiMAX signal parameters and sends them to its associated base station which executes the algorithm. Remote positioning can also be handled by a dedicated server. In self positioning case, not only a mobile station measures parameters of signals that it receives from a WiMAX base station, but also computes its own position by executing the algorithm itself. Self positioning requires mobile stations to have an up-to-date version of the training database.
- For remote positioning, a lightweight application is required for mobile stations to send measured signal parameters to base stations, and a main position estimation application is required for base stations. For self positioning, mobile stations need to have the main application. Even though the main application is processing power intensive, any modern computer can handle it well.

The proposed algorithm also has some disadvantages.

- A mobile station to be located should be in motion. Because, the algorithm is based on tracking, and tracking can be realized by processing multiple signal parameters measured at different places by the monitored mobile station. If a stationary user is located with the algorithm, most probably very coarse position estimates will be obtained.
- A disadvantage caused by utilization of fingerprinting is that it should be repeated periodically in order to update a training database. These updates are necessary to take effects of seasonal and environmental changes into account. If a network faces more significant changes like relocation of antennas that are connected to a base station, training database should be completely renewed. The more fingerprinting sessions are performed, the more time and effort are consumed.
- A lot of experiments should be carried out to optimize algorithmic parameters for better positioning performance.
- Lack of sufficient location related information prevents the algorithm from returning exactly one position estimate.

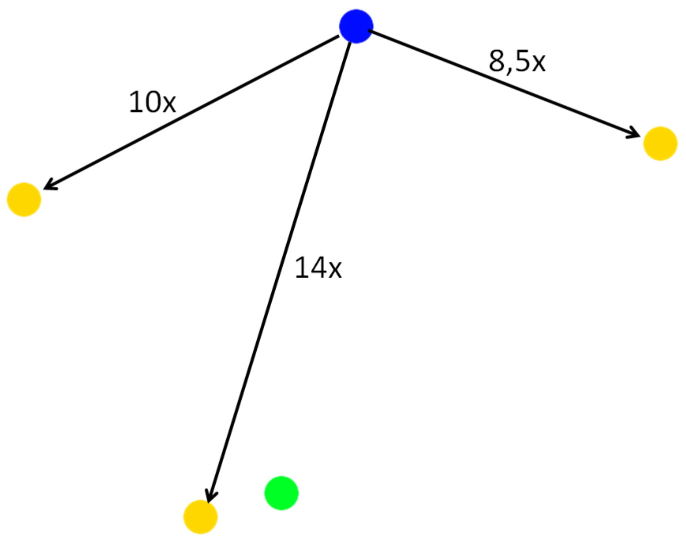


Figure 9.1: A position estimation example. Blue circle shows location of an employed base station. Green circle shows actual location of a mobile station. Orange circles show position estimates computed by the proposed algorithm.

According to experimental results, root mean square error of the proposed algorithm's position estimates was around 350 meters in the best case where more than one clusters were generated. It was worse than the k Nearest Neighbor algorithm's positioning error which was around 230 meters in the best case. These errors do not necessarily mean that the proposed algorithm's results were worse than those of the k Nearest Neighbor algorithm. In fact, the proposed algorithm can come up with estimates that are quite close to the actual location of a mobile station. For example, in Figure 8.3, the estimate which has the smallest distance to the actual location of a mobile station, depicted by the green circle, is the orange circle in lower left part of the map which is generated by the proposed algorithm. Nevertheless, because of that other position estimates produced by the proposed algorithm (two orange circles in upper part of the figure) are far away from the actual location, its root mean square error is high.

The reason for having multiple position estimates was a shortcoming which originated from being unable to access location-related data provided by IEEE 802.16 MAC headers. If location-related MAC fields could be accessed, number of final position estimates could be reduced, meaning that root mean square error could be minimized. The most important data required for achieving such an improvement is the MAC layer field called Timing Adjust which allows doing fine grained range measurements. It gives valuable information about the distance between a mobile station and a base station.

Consider the case in Figure 9.1 where blue circle shows location of an employed base station, green circle shows actual location of a mobile station, and orange circles show position estimates computed by the proposed algorithm. As seen in the figure, distance of each estimate to the base station is different. A range measurement obtained by utilizing Timing Adjust can help to eliminate some of the estimates or directly identify one estimate as the final positioning result. For example, if the distance between the base station and the mobile station is computed to be around 14 units (14x) using Timing Adjust, estimates in upper left and upper right part of the figure can be eliminated instantly, leaving only the estimate in lower part of the figure. As demonstrated by this example, the most obvious way of improving the algorithm to provide fewer and more accurate estimates is to utilize Timing Adjust.

During collection of fingerprints for scenarios, equipment that measured signal parameters moved at the walking speed of a human. For this reason, results of experiments represented the algorithm's performance of positioning a walking person. As a future work, new scenario data can be collected using vehicles of different types to investigate how accurately the algorithm can locate mobile stations moving at different speeds. The investigation process includes doing experiments with algorithmic parameter sets that are specially selected for tracking mobile stations with varying speeds.

Another future work can be development of a strategy for dynamic selection of algorithmic parameters based on analysis of signal parameters and position estimates obtained over time. For example, when the number of cells that are identified as possible estimates in one iteration falls below a certain level (possibly implying that tracking may fail soon,) the algorithm can switch to more flexible algorithmic parameters to catalyze the tracking process. After it is made sure that the algorithm returns to a more stable state, it can switch back to a previously used set of algorithmic parameters.

REFERENCES

[1] "The ethernet version II frame format," http://www.firewall.cx/ethernet-frames-ethernet-ii.php, last visited on October 26, 2009.

[2] "The IEEE 802.16 working group on broadband wireless access standards," http://www.ieee802.org/16/, last visited on June 11, 2010.

[3] "Welcome to the Quantum GIS project," http://www.qgis.org/, last visited on October 27, 2009.

[4] J. G. Andrews, A. Ghosh, and R. Muhamed, *Fundamentals of WiMAX: Understanding Broadband Wireless Networking*, ser. Prentice Hall Communications Engineering and Emerging Technologies Series. Upper Saddle River, New Jersey, USA: Prentice Hall PTR, 2007.

[5] Başarsoft, "Başarsoft," http://www.basarsoft.com.tr/, last visited on June 14, 2010.

[6] P. Bedell, *Wireless Crash Course*, 2nd ed. New York: McGraw-Hill/Osborne, 2005, previous ed.: 2001.

[7] A. N. Bishop and P. N. Pathirana, "Sensor fusion based localization of a mobile user in a wireless network," *TENCON 2005 2005 IEEE Region 10*, pp. 1–6, 21–24 November 2005, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4085296&isnumber=4084860.

[8] M. Bshara, N. Deblauwe, and L. V. Biesen, "Localization in WiMAX networks based on signal strength observations," *IEEE GLOBECOM 2008, Global Communications Conference, Exhibition & Industry Forum Co-Located With WTC*, 30 November–4 December 2008, http://wwwtw.vub.ac.be/elec/Papers%20on%20web/Papers/Mussa/Mussa.Bshara_IEEEGlobalCom08.pdf.

[9] P. Coulon, "Principles of modulation in wireless communications," http://www.tml.tkk.fi/Studies/Tik-110.300/1999/Wireless/modulation_3.html, 11 October 1999, last visited on June 11, 2010.

[10] DigitalGlobe, Inc., "DigitalGlobe | an imagery and information company," http://www.digitalglobe.com/, last visited on June 14, 2010.

[11] DIT Technologies, "Sprint Overdrive 3G/4G Mobile Hotspot by Sierra Wireless – Steve Ballmers' cameo at Sprint event," http://www.ditii.com/2010/01/07/sprint-overdrive-3g4g-mobile-hotspot-by-sierra-wireless-steve-ballmers-cameo-at-sprint-event/, 7 January 2010, last visited on June 11, 2010.

[12] C. Drane, M. Macnaughtan, and C. Scott, "Positioning GSM telephones," *Communications Magazine, IEEE*, vol. 36, no. 4, pp. 46–54, 59, April 1998, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=667413&isnumber=14678&tag=1.

[13] Engadget, “HTC EVO 4G press shots – Engadget galleries,” http://www.engadget.com/photos/htc-evo-4g-press-shots/, 23 March 2010, last visited on June 11, 2010.

[14] European Telecommunications Standards Institute, “ETSI TS 145 010 V9.0.0 (2010-02) technical specification, digital cellular telecommunications system (phase 2+); radio subsystem synchronization (3GPP TS 45.010 version 9.0.0 release 9),” http://www.3gpp.org/ftp/Specs/html-info/45010.htm, 2010.

[15] Federal Communication Commission, “Enhanced 9-1-1 – wireless services,” http://www.fcc.gov/pshs/services/911-services/enhanced911/, last visited on October 13, 2009.

[16] ——, “OET bulletin no. 71 – guidelines for testing and verifying the accuracy of wireless E911 location systems,” http://www.fcc.gov/Bureaus/Engineering_Technology/Documents/bulletins/oet71/oet71.doc, 12 April 2000.

[17] GeoEye, Inc., “GeoEye | high resolution imagery, earth imagery & geospatial services,” http://www.geoeye.com/, last visited on June 14, 2010.

[18] S. Gezici, “A survey on wireless position estimation,” *Wireless Personal Communications*, vol. 44, no. 3, pp. 263–282, February 2008, http://www.springerlink.com/content/w1481x761266m417.

[19] Google, “Google maps,” http://maps.google.com/, last visited on June 14, 2010.

[20] F. Gustafsson and F. Gunnarsson, “Mobile positioning using wireless networks: possibilities and fundamental limitations based on available wireless network measurements,” *Signal Processing Magazine, IEEE*, vol. 22, no. 4, pp. 41–53, July 2005, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1458284&isnumber=31384.

[21] M. Hellebrandt, R. Mathar, and M. Scheibenbogen, “Estimating position and velocity of mobiles in a cellular radio network,” *IEEE Transactions on Vehicular Technology*, vol. 46, no. 1, pp. 65–71, February 1997, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=554738&isnumber=12085.

[22] J. Hightower and G. Borriello, “Location systems for ubiquitous computing,” *Computer*, vol. 34, no. 8, pp. 57–66, August 2001, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=940014.

[23] IEEE 802.16 Working Group on Broadband Wireless Access Standards, “IEEE standard for local and metropolitan area networks part 16: Air interface for fixed broadband wireless access systems,” *IEEE Std 802.16-2004 (Revision of IEEE Std 802.16-2001)*, pp. 0_1–857, 2004, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1350465.

[24] ——, “A generic packet convergence sublayer (GPCS) for supporting multiple protocols over 802.16 air interface,” http://www.ieee802.org/16/netman/contrib/C80216g-05_025r1.pdf, 9 November 2005.

[25] ——, “IEEE standard for local and metropolitan area networks part 16: Air interface for fixed and mobile broadband wireless access systems amendment 2: Physical and medium access control layers for combined fixed and mobile operation in licensed bands

and corrigendum 1," *IEEE Std 802.16e-2005 and IEEE Std 802.16-2004/Cor 1-2005 (Amendment and Corrigendum to IEEE Std 802.16-2004)*, pp. 0_1–822, 2006, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1603394.

[26] Intel Corporation, "Intel® WiMAX/WiFi link 5350 and Intel® WiMAX/WiFi link 5150 products," http://www.intel.com/support/wireless/wmax/5350_5150/index.htm, last visited on October 26, 2009.

[27] ——, "Welcome to your internet future – mobile broadband brought to you by Intel," http://download.intel.com/network/connectivity/products/wireless/welcome-to-your-internet-future.pdf, 2007.

[28] ——, "Intel® WiMAX/WiFi link 5150 product description," http://download.intel.com/network/connectivity/products/wireless/320663_5150.pdf, 2009.

[29] Intel Corporation and others, "Linux WiMAX," http://www.linuxwimax.org/, last visited on October 26, 2009.

[30] J. James J. Caffery, *Wireless Location in CDMA Cellular Radio Systems*, ser. The Kluwer International Series in Engineering and Computer Science. Kluwer Academic Publishers, 2002.

[31] H. Liu, H. Darabi, P. Banerjee, and J. 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, pp. 1067–1080, November 2007, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4343996.

[32] Microsoft, "Microsoft .NET framework," http://www.microsoft.com/net/, last visited on June 13, 2010.

[33] M. Porretta, P. Nepa, G. Manara, F. Giannetti, M. Dohler, B. Allen, and A. Aghvami, "Estimating position and velocity of mobile terminals in a microcellular network using an adaptive linear regression setup," *15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004.*, vol. 1, pp. 561–565, 5–8 September 2004, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1370933.

[34] ——, "A novel single base station location technique for microcellular wireless networks: description and validation by a deterministic propagation model," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1502–1514, September 2004, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1337328.

[35] D. Reynders and E. Wright, *Practical TCP/IP and Ethernet Networking for Industry*. Newnes, August 2003.

[36] W. Stallings, *Wireless Communications & Networks*, 2nd ed. Upper Saddle River, New Jersey, USA: Prentice Hall, 2005.

[37] The MathWorks, Inc., "(simple) Tool for estimating the number of clusters," http://www.mathworks.com/matlabcentral/fileexchange/13916, last visited on June 21, 2010.

[38] ——, "What's new in release 2007b," http://www.mathworks.com/products/new_products/release2007b.html, last visited on June 13, 2010.

[39] Y. Tianchi and J. Liang, “The maneuverable single based station position method in the non-line-of-sight cellular environment,” *International Conference on Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC ’09.*, vol. 2, pp. 133–135, 25–26 April 2009, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4908423.

[40] C. Weinschenk, “Speeding up WiMax,” http://www.itbusinessedge.com/cm/community/features/interviews/blog/speeding-up-wimax/?cs=40726, 16 April 2010, last visited on June 11, 2010.

[41] Whirlpool, “Vividwireless (4G) - speed test thread,” http://forums.whirlpool.net.au/forum-replies-archive.cfm/1418240.html, last visited on June 11, 2010.

[42] Wikipedia contributors, “HSL and HSV – Wikipedia, the free encyclopedia,” http://en.wikipedia.org/w/index.php?title=HSL_and_HSV&oldid=321950904, 2009, last visited on October 28, 2009.

[43] ——, “IEEE 802.16 – Wikipedia, the free encyclopedia,” http://en.wikipedia.org/wiki/IEEE_802.16, 2010, last visited on June 11, 2010.

[44] ——, “Orthogonal frequency-division multiplexing – Wikipedia, the free encyclopedia,” http://en.wikipedia.org/wiki/Orthogonal_frequency-division_multiplexing, 2010, last visited on June 11, 2010.

[45] ——, “WiMAX – Wikipedia, the free encyclopedia,” http://en.wikipedia.org/wiki/WiMAX, 2010, last visited on June 11, 2010.

[46] WiMAX Forum, “WiMAX Forum,” http://www.wimaxforum.org/, last visited on June 11, 2010.

[47] ——, “WiMAX Forum® industry research report - april, 2010,” http://www.wimaxforum.org/sites/wimaxforum.org/files/page/2009/12/wf_monthly_report_04-2010.pdf, April 2010.

[48] Wireshark Foundation, “Wireshark :: Go deep,” http://www.wireshark.org/, last visited on October 26, 2009.

[49] M. Zhaounia, M. A. Landolsi, and R. Bouallegue, “Approximate maximum likelihood mobile localization using scatterer information,” *IEEE 69th Vehicular Technology Conference, 2009. VTC Spring 2009.*, pp. 1–3, 26–29 April 2009, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5073405.

[50] ZTE Corporation, “Indoor coverage test result,” 2008.

[51] ——, “METU WiMAX project test specification,” 2008.

APPENDIX A

LIST OF ABBREVIATIONS

3G	Third Generation
AAA	Authentication Authorization Accounting
ADSL	Asymmetric Digital Subscriber Line
AMC	Adaptive Modulation and Coding
AOA	Angle of Arrival
ARQ	Automatic Retransmission Request
ASN	Access Service Network
AvgTxPwr	Average transmit power
BE	Best Effort
BS	Base station
BTS	Base Transceiver Stations
CBR	Constant Bit Rate
CINR	Carrier to Interference-Plus-Noise Ratio
DGPS	Differential Global Positioning System
DOA	Direction of Arrival
ertPS	Extended Real-Time Polling Service
FCC	The Federal Communications Commission
FDD	Frequency Division Duplexing
FEC	Forward Error Correction
FIFO	First In, First Out
FTP	File Tranfer Protocol
GIS	Geographic Information System
GPS	Global Positioning System
GSM	Global System for Mobile Communications

H-ARQ	Hybrid-ARQ
HSL	Hue Saturation Lightness
HTTP	Hypertext Transfer Protocol
IEEE	The Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IPTV	Internet Protocol Television
ISI	Industrial, Scientific and Medical radio bands
ISI	Inter-symbol Interference
LOS	Line of Sight
kNN	k Nearest Neighbor
MAC	Medium Access Control layer
MPC	Multipath channel
MPEG	Moving Pictures Experts Group
MS	Mobile station
MT	Mobile terminal
NLOS	Non Line of Sight
nrtPS	Non-Real-Time Polling Service
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
OSI	Open Systems Interconnection
PAM	Partitioning Around Medoids
PHY	Physical layer
QAM	Quadrature Amplitude Modulation
QOS	Quality of Service
QPSK	Quadrature Phase-Shift Keying
RBF	Radial Basis Function
REKF	Robust Extended Kalman Filter
RGB	Red, Green, Blue color model
RMSE	Root Mean Square Error
RS	Reference Station
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator

RTOA	Roundtrip Time of Arrival
rtPS	Real-time Polling Service
SBSP	Single base station positioning
SNR	Signal-to-Noise Ratio
SS	Subscriber Station
TA	Timing Adjust
TA	Timing Advance
TDD	Time Division Multiplexing
TDMA	Time Division Multiple Access
TDOA	Time Difference of Arrival
TOA	Time of Arrival
UGS	Unsolicited Grant Service
USB	Universal Serial Bus
VBR	Variable Bit Rate
VoIP	Voice over Internet Protocol
Wi-Fi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access

APPENDIX B

EXPERIMENTAL RESULTS OF PROPOSED POSITION ESTIMATION ALGORITHM

Table B.1: Experimental results of the proposed position estimation algorithm (1st part.)

Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
237	182	7	5	7	10	1	3	1	3	1	90	100	188	0	188	0	0	44	150	23,40%	79,79%	248,81	10688 sec	57 sec
237	182	7	5	7	6	1	1	1	3	1	90	100	188	0	188	0	0	44	148	23,40%	78,72%	250,74	9952 sec	53 sec
237	182	10	8	10	8	1	3	1	3	2	90	100	188	0	0	188	0	93	169	29,79%	65,16%	344,18	28115 sec	150 sec
118	90	10	8	10	4	1	1	1	3	1	80	100	188	0	188	0	0	41	150	21,81%	79,79%	242,08	2586 sec	14 sec
237	182	7	5	7	4	1	1	1	3	2	90	100	188	0	1	187	0	98	166	30,85%	62,77%	360,36	8771 sec	47 sec
237	182	10	8	10	7	1	2	1	3	2	90	100	188	0	0	188	0	92	169	29,52%	65,16%	345,79	27845 sec	149 sec
237	182	10	8	10	8	1	3	1	3	1	90	100	188	0	188	0	0	39	152	20,74%	80,85%	242,55	25118 sec	134 sec
237	182	7	5	7	7	1	2	1	3	2	90	100	188	0	1	187	0	97	166	30,32%	63,03%	358,66	9967 sec	54 sec
237	182	10	8	10	4	1	1	1	3	2	90	100	188	0	0	188	0	92	168	29,26%	64,89%	346,93	25028 sec	134 sec
60	46	8	6	8	7	1	2	1	3	1	80	100	188	0	188	0	0	42	146	22,34%	77,66%	242,32	237 sec	2 sec
237	182	7	5	7	6	1	1	1	3	2	90	100	188	0	1	187	0	95	165	30,05%	62,50%	358,87	10173 sec	55 sec
237	182	7	5	7	10	1	3	1	3	2	90	100	188	0	1	187	0	97	166	30,05%	62,50%	357,99	11265 sec	60 sec
118	90	8	6	8	7	1	2	1	3	1	80	100	188	0	188	0	0	43	142	22,87%	75,53%	247,20	1595 sec	9 sec
118	90	8	6	8	6	1	1	1	3	2	80	100	188	0	1	187	0	92	164	28,72%	64,63%	354,65	1738 sec	10 sec
118	90	8	6	8	7	1	2	1	3	2	80	100	188	0	1	187	0	90	164	28,46%	64,89%	354,02	1743 sec	10 sec
118	90	8	6	8	4	1	1	1	3	2	80	100	188	0	1	187	0	92	162	28,46%	64,10%	355,68	1619 sec	9 sec
118	90	10	8	10	4	1	1	1	3	2	80	100	188	0	0	188	0	84	170	27,13%	66,49%	343,78	2788 sec	15 sec
60	46	8	6	8	7	1	2	1	3	2	80	100	188	0	2	186	0	91	163	27,66%	65,16%	345,13	318 sec	2 sec

Table B.2: Experimental results of the proposed position estimation algorithm (2nd part.)

Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
118	90	10	8	10	6	1	1	1	3	2	80	100	188	0	0	188	0	80	170	25,80%	65,96%	344,53	2923 sec	16 sec
237	182	6	4	6	4	1	1	1	6	3	50	100	186	4	0	59	123	105	171	28,41%	59,95%	367,35	5267 sec	29 sec
118	90	10	8	10	7	1	2	1	3	2	80	100	188	0	0	188	0	78	169	25,27%	65,69%	343,86	2930 sec	16 sec
237	182	7	5	7	4	1	1	1	3	3	90	100	188	0	1	44	143	105	182	27,30%	61,88%	359,88	8968 sec	48 sec
237	182	7	5	7	6	1	1	1	3	3	90	100	188	0	1	42	145	106	184	27,39%	61,17%	357,81	10441 sec	56 sec
237	182	8	6	8	7	1	2	1	3	3	90	100	188	0	0	26	162	110	185	26,33%	62,77%	354,89	15140 sec	81 sec
237	182	7	5	7	7	1	2	1	3	3	90	100	188	0	1	38	149	104	184	26,51%	61,97%	357,26	10200 sec	55 sec
237	182	6	4	6	7	1	2	1	3	2	90	100	188	1	5	182	0	89	159	28,19%	58,78%	370,03	5975 sec	32 sec
237	182	8	6	8	4	1	1	1	3	3	90	100	188	0	0	32	156	109	183	26,77%	61,26%	357,07	13744 sec	74 sec
237	182	10	8	10	8	1	3	1	3	3	90	100	188	0	0	14	174	111	186	26,15%	61,97%	351,31	26297 sec	140 sec
237	182	10	8	10	7	1	2	1	3	3	90	100	188	0	0	13	175	110	187	25,89%	62,32%	351,51	26165 sec	140 sec
237	182	7	5	7	10	1	3	1	3	3	90	100	188	0	1	41	146	104	184	25,98%	62,15%	356,96	11212 sec	60 sec
118	90	8	6	8	6	1	1	1	3	3	80	100	188	0	1	15	172	108	181	25,44%	62,59%	359,79	1882 sec	11 sec
237	182	10	8	10	4	1	1	1	3	3	90	100	188	0	0	14	174	109	186	25,53%	62,32%	351,92	24472 sec	131 sec
118	90	8	6	8	7	1	2	1	3	3	80	100	188	0	1	15	172	107	181	25,09%	62,68%	359,89	1879 sec	10 sec
237	182	6	4	6	4	1	1	1	3	2	100	100	188	7	7	174	0	84	156	26,86%	58,78%	396,01	4955 sec	27 sec
60	46	10	8	10	7	1	2	1	3	2	80	100	188	0	1	187	0	72	172	22,61%	66,49%	339,89	489 sec	3 sec
118	90	8	6	8	4	1	1	1	3	3	80	100	188	0	1	16	171	106	180	25,18%	61,70%	360,91	1758 sec	10 sec

Table B.3: Experimental results of the proposed position estimation algorithm (3rd part.)

Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	mse, all points	total time	avg. est. time
237	182	6	4	6	4	1	3	5	3	2	90	100	188	2	8	178	0	82	160	25,80%	59,57%	374,39	6759 sec	36 sec
237	182	6	4	6	4	1	3	3	3	2	90	100	188	4	4	180	0	82	156	26,06%	58,78%	380,97	5658 sec	31 sec
60	46	8	6	8	2	1	1	1	3	3	60	100	188	2	2	12	172	105	177	24,29%	61,17%	358,61	413 sec	3 sec
118	90	10	8	10	4	1	1	1	3	3	80	100	188	0	0	15	173	95	184	22,16%	64,36%	351,01	3000 sec	16 sec
237	182	6	4	6	4	1	1	1	3	2	90	100	188	6	7	175	0	81	150	25,53%	56,12%	391,89	5051 sec	27 sec
60	46	8	6	8	4	1	1	1	3	3	80	100	188	0	2	14	172	94	181	22,07%	62,23%	352,17	945 sec	6 sec
118	90	6	4	6	5	1	2	1	3	2	50	100	188	6	3	179	0	74	150	23,94%	58,24%	385,76	773 sec	5 sec
60	46	8	6	8	7	1	2	1	3	3	80	100	188	0	2	16	170	93	180	21,72%	62,23%	353,80	432 sec	3 sec
118	90	6	4	6	4	1	1	1	3	2	50	100	188	7	2	179	0	74	147	23,94%	57,18%	392,09	753 sec	5 sec
237	182	6	4	6	2	2	1	1	3	2	100	100	188	16	4	168	0	75	149	23,14%	55,05%	435,39	9269 sec	50 sec
237	182	6	4	6	3	1	1	1	3	2	100	100	188	12	6	170	0	73	150	22,61%	55,85%	417,16	4742 sec	26 sec
237	182	6	4	6	3	1	1	1	3	3	100	100	188	12	6	74	96	83	164	21,19%	55,32%	413,08	5157 sec	28 sec
237	182	6	4	6	2	1	1	1	6	3	100	100	186	16	4	66	100	79	161	20,79%	54,66%	427,98	4095 sec	23 sec
118	90	6	4	6	3	1	1	1	3	2	100	100	188	24	6	158	0	68	137	21,54%	50,53%	466,23	595 sec	4 sec
237	182	6	4	6	2	1	1	1	3	2	100	100	188	24	4	160	0	64	138	20,48%	51,86%	460,93	4055 sec	22 sec
118	90	6	4	6	3	1	2	2	3	3	100	100	188	20	7	67	94	75	151	19,68%	51,95%	451,78	696 sec	4 sec
118	90	6	4	6	3	1	1	1	3	3	100	100	188	24	6	72	86	71	145	18,88%	49,47%	466,00	671 sec	4 sec
60	46	6	4	6	2	1	1	1	3	2	40	100	188	33	2	153	0	60	125	16,76%	46,28%	467,92	148 sec	1 sec

Table B.4: Experimental results of the proposed position estimation algorithm (4th part.)

Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
118	90	6	4	6	0	1	1	1	3	2	50	100	188	41	6	141	0	49	118	14,36%	44,68%	513,16	410 sec	3 sec
237	182	4	3	4	8	1	7	3	3	1	80	100	188	37	151	0	0	18	93	9,57%	49,47%	438,71	3191 sec	17 sec
60	46	4	3	4	1	2	1	1	3	1	10	100	188	40	148	0	0	18	91	9,57%	48,40%	451,92	64 sec	1 sec
60	46	6	4	6	0	1	1	1	3	2	40	100	188	51	3	134	0	49	107	13,56%	38,83%	529,70	119 sec	1 sec
60	46	6	4	6	2	1	1	1	3	2	80	100	188	44	6	138	0	44	109	11,97%	40,69%	514,61	146 sec	1 sec
237	182	4	3	4	6	1	5	2	3	1	80	100	188	43	145	0	0	15	86	7,98%	45,74%	468,35	2370 sec	13 sec
237	182	4	3	4	6	1	5	2	3	2	80	100	188	43	8	137	0	43	107	12,23%	37,77%	525,56	2448 sec	14 sec
237	182	6	4	6	0	1	1	1	3	2	100	100	188	57	3	128	0	37	105	11,17%	38,56%	567,13	2529 sec	14 sec
237	182	4	3	4	8	1	7	3	3	3	80	100	188	37	3	77	71	42	128	9,84%	40,87%	502,19	3340 sec	18 sec
118	90	4	3	4	1	3	1	1	3	1	30	100	188	50	138	0	0	15	83	7,98%	44,15%	488,85	377 sec	3 sec
237	182	6	4	6	0	1	1	1	3	2	90	100	188	52	2	134	0	38	102	11,44%	36,44%	551,07	3735 sec	20 sec
118	90	4	3	4	6	1	1	1	3	1	100	100	188	51	137	0	0	14	79	7,45%	42,02%	509,42	331 sec	2 sec
237	182	4	3	4	3	1	1	1	3	1	80	100	188	58	130	0	0	15	77	7,98%	40,96%	518,27	1954 sec	11 sec
60	46	4	3	4	2	1	1	1	3	1	40	100	188	62	126	0	0	13	79	6,91%	42,02%	530,77	49 sec	1 sec
60	46	4	3	4	2	1	1	1	3	1	40	100	188	62	126	0	0	13	79	6,91%	42,02%	530,77	49 sec	1 sec
60	46	4	3	4	2	1	1	2	3	1	30	100	188	54	134	0	0	12	79	6,38%	42,02%	503,27	55 sec	1 sec
118	90	4	3	4	3	1	1	3	3	1	100	100	188	60	128	0	0	14	74	7,45%	39,36%	532,81	296 sec	2 sec
118	90	4	3	4	3	3	1	1	3	3	60	100	188	51	11	68	58	37	108	8,69%	34,84%	546,39	793 sec	5 sec

Table B.5: Experimental results of the proposed position estimation algorithm (5th part.)

Horizontal cell count	Vertical cell count	maxRssiDiff	maxCinrDiff	maxAvgTxPwrDiff	maxUncontinuity	radiusForNeighbors	expansionFactor	increment	n	maxK	Experiment length	Scenario length	number of total est.	number of 0 cluster	number of 1 cluster	number of 2 cluster	number of 3 cluster	number of <100	number of <300	prob. of <100 (2)	prob. of <300 (2)	rmse, all points	total time	avg. est. time
118	90	4	3	4	6	1	1	1	3	3	100	100	188	51	15	76	46	36	109	8,69%	34,75%	551,02	425 sec	3 sec
60	46	4	3	4	1	2	1	1	3	1	40	100	188	70	118	0	0	12	70	6,38%	37,23%	548,34	56 sec	1 sec
118	90	4	3	4	3	1	1	3	3	2	100	100	188	60	7	121	0	31	97	8,51%	32,71%	575,81	394 sec	3 sec
60	46	4	3	4	3	2	1	2	3	3	60	100	188	55	11	77	45	35	100	7,98%	32,71%	546,79	149 sec	1 sec
118	90	4	3	4	3	1	1	1	3	1	100	100	188	67	121	0	0	13	64	6,91%	34,04%	560,53	213 sec	2 sec
118	90	4	3	4	3	1	1	3	3	3	100	100	188	60	7	75	46	34	99	7,98%	31,12%	576,81	462 sec	3 sec
118	90	4	3	4	0	3	1	1	3	3	40	100	188	63	4	73	48	41	93	8,87%	29,43%	567,17	393 sec	3 sec
60	46	4	3	4	4	2	1	1	3	1	80	100	188	62	126	0	0	10	67	5,32%	35,64%	530,72	80 sec	1 sec
237	182	4	3	4	3	1	1	1	3	3	90	100	188	60	2	76	50	26	106	6,47%	33,07%	569,01	1669 sec	9 sec
60	46	4	3	4	4	1	1	1	3	1	80	100	188	62	126	0	0	10	66	5,32%	35,11%	530,98	63 sec	1 sec
60	46	4	3	4	3	2	1	1	3	3	80	100	188	66	11	78	33	30	86	7,62%	30,59%	579,14	131 sec	1 sec
60	46	4	3	4	4	2	1	1	3	2	80	100	188	62	9	117	0	26	86	7,18%	31,38%	572,90	110 sec	1 sec
60	46	4	3	4	3	1	1	1	3	2	80	100	188	67	13	108	0	26	81	7,45%	30,32%	580,87	83 sec	1 sec
60	46	4	3	4	4	1	1	1	3	3	80	100	188	62	11	79	36	34	89	7,89%	29,26%	574,67	119 sec	1 sec
118	90	4	3	4	3	1	1	1	3	3	100	100	188	67	12	75	34	31	89	7,09%	29,61%	592,46	271 sec	2 sec
118	90	4	3	4	1	3	1	1	3	3	80	100	188	73	11	66	38	27	82	6,29%	25,98%	603,08	555 sec	3 sec

APPENDIX C

EXPERIMENTAL RESULTS OF K NEAREST NEIGHBOR ALGORITHM

Table C.1: Experimental results of the k Nearest Neighbor algorithm (1st part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	1200	8	100	18300	4642	25%	15050	82%	223,47	4472	24%	13124	71%	279,16	169 sec	0 sec
237	182	1600	8	100	18300	4691	25%	15117	82%	220,63	4472	24%	13124	71%	279,16	166 sec	0 sec
118	90	500	8	100	18300	4695	25%	14971	81%	222,59	3905	21%	12146	66%	301,27	56 sec	0 sec
237	182	1200	6	100	18600	4664	25%	15135	81%	230,52	4461	23%	13080	70%	285,74	170 sec	0 sec
60	46	200	3	100	18800	4902	26%	14905	79%	231,92	3702	19%	11924	63%	321,23	21 sec	0 sec
60	46	200	4	100	18800	4910	26%	14894	79%	231,85	3760	20%	12059	64%	319,28	21 sec	0 sec
118	90	400	8	100	18300	4642	25%	14744	80%	226,99	3905	21%	12146	66%	301,27	61 sec	0 sec
118	90	500	6	100	18600	4691	25%	15018	80%	229,67	3904	20%	12152	65%	308,29	59 sec	0 sec
118	90	600	8	100	18300	4636	25%	14775	80%	223,26	3905	21%	12146	66%	301,27	57 sec	0 sec
237	182	2000	8	100	18300	4622	25%	14767	80%	222,24	4472	24%	13124	71%	279,16	174 sec	0 sec
237	182	1600	1	100	18800	4636	24%	15239	81%	229,49	4542	24%	13185	70%	282,81	168 sec	0 sec
237	182	1600	2	100	18800	4656	24%	15258	81%	229,63	4580	24%	13178	70%	284,60	167 sec	0 sec

Table C.2: Experimental results of the k Nearest Neighbor algorithm (2nd part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	1600	3	100	18800	4658	24%	15275	81%	229,68	4604	24%	13276	70%	284,40	168 sec	0 sec
237	182	1600	4	100	18800	4674	24%	15275	81%	229,76	4555	24%	13229	70%	284,63	167 sec	0 sec
60	46	200	1	100	18800	4884	25%	14912	79%	231,95	3681	19%	11912	63%	321,03	21 sec	0 sec
60	46	200	2	100	18800	4881	25%	14899	79%	231,98	3690	19%	12001	63%	322,59	21 sec	0 sec
118	90	600	6	100	18600	4654	25%	14802	79%	230,26	3904	20%	12152	65%	308,29	57 sec	0 sec
237	182	40	8	100	18300	5126	28%	13396	73%	271,12	4534	24%	13032	71%	281,61	158 sec	0 sec
118	90	500	1	100	18800	4658	24%	15059	80%	231,22	3824	20%	12119	64%	314,07	57 sec	0 sec
118	90	500	2	100	18800	4671	24%	15078	80%	231,40	3859	20%	12119	64%	314,01	62 sec	0 sec
118	90	500	3	100	18800	4687	24%	15083	80%	231,41	3828	20%	12232	65%	311,11	56 sec	0 sec
118	90	500	4	100	18800	4681	24%	15104	80%	231,48	3892	20%	12278	65%	310,68	58 sec	0 sec
237	182	1200	1	100	18800	4606	24%	15146	80%	232,09	4542	24%	13185	70%	282,81	165 sec	0 sec
237	182	1200	2	100	18800	4619	24%	15172	80%	232,23	4580	24%	13178	70%	284,60	171 sec	0 sec

Table C.3: Experimental results of the k Nearest Neighbor algorithm (3rd part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	1200	3	100	18800	4637	24%	15191	80%	232,23	4604	24%	13276	70%	284,40	171 sec	0 sec
237	182	1200	4	100	18800	4635	24%	15204	80%	232,30	4555	24%	13229	70%	284,63	165 sec	0 sec
60	46	120	2	100	18800	4901	26%	14408	76%	244,22	3690	19%	12001	63%	322,59	20 sec	0 sec
60	46	120	3	100	18800	4896	26%	14440	76%	244,26	3702	19%	11924	63%	321,23	20 sec	0 sec
118	90	60	8	100	18300	4960	27%	13582	74%	265,76	3896	21%	12145	66%	301,33	52 sec	0 sec
237	182	80	8	100	18300	5053	27%	13617	74%	266,28	4486	24%	13110	71%	279,92	157 sec	0 sec
237	182	120	8	100	18300	5037	27%	13714	74%	263,00	4475	24%	13114	71%	279,45	159 sec	0 sec
118	90	400	6	100	18600	4623	24%	14722	79%	233,68	3904	20%	12152	65%	308,29	55 sec	0 sec
118	90	600	1	100	18800	4635	24%	14865	79%	231,72	3824	20%	12119	64%	314,07	60 sec	0 sec
118	90	600	2	100	18800	4636	24%	14855	79%	231,88	3859	20%	12119	64%	314,01	58 sec	0 sec
118	90	600	3	100	18800	4653	24%	14872	79%	231,96	3828	20%	12232	65%	311,11	57 sec	0 sec
118	90	600	4	100	18800	4644	24%	14873	79%	232,00	3892	20%	12278	65%	310,68	61 sec	0 sec

Table C.4: Experimental results of the k Nearest Neighbor algorithm (4th part.)

Horizontal cell count	Vertical cell count	Neighbor count (*k*)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	800	8	100	18300	4501	24%	14510	79%	234,47	4472	24%	13124	71%	279,16	163 sec	0 sec
237	182	2000	1	100	18800	4595	24%	14873	79%	231,02	4542	24%	13185	70%	282,81	173 sec	0 sec
237	182	2000	2	100	18800	4594	24%	14889	79%	231,11	4580	24%	13178	70%	284,60	178 sec	0 sec
237	182	2000	3	100	18800	4593	24%	14880	79%	231,18	4604	24%	13276	70%	284,40	176 sec	0 sec
237	182	2000	4	100	18800	4611	24%	14876	79%	231,24	4555	24%	13229	70%	284,63	172 sec	0 sec
237	182	2000	6	100	18600	4616	24%	14810	79%	229,34	4461	23%	13080	70%	285,74	177 sec	0 sec
118	90	100	8	100	18300	4815	26%	13805	75%	259,69	3904	21%	12146	66%	301,27	54 sec	0 sec
237	182	200	8	100	18300	4927	26%	13848	75%	259,02	4469	24%	13122	71%	279,22	159 sec	0 sec
118	90	40	8	100	18300	5059	27%	13449	73%	269,54	3897	21%	12142	66%	301,50	52 sec	0 sec
237	182	80	6	100	18600	5066	27%	13597	73%	272,97	4465	24%	13062	70%	286,50	160 sec	0 sec
237	182	120	1	100	18800	5079	27%	13754	73%	269,02	4554	24%	13182	70%	283,00	157 sec	0 sec
237	182	120	6	100	18600	5071	27%	13707	73%	269,41	4458	23%	13069	70%	285,99	160 sec	0 sec

Table C.5: Experimental results of the k Nearest Neighbor algorithm (5th part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
118	90	300	8	100	18300	4473	24%	14332	78%	235,68	3905	21%	12146	66%	301,27	58 sec	0 sec
118	90	400	1	100	18800	4596	24%	14776	78%	235,05	3824	20%	12119	64%	314,07	56 sec	0 sec
118	90	400	2	100	18800	4613	24%	14806	78%	235,06	3859	20%	12119	64%	314,01	55 sec	0 sec
118	90	400	3	100	18800	4621	24%	14806	78%	235,24	3828	20%	12232	65%	311,11	62 sec	0 sec
118	90	400	4	100	18800	4639	24%	14824	78%	235,27	3892	20%	12278	65%	310,68	56 sec	0 sec
237	182	3000	8	100	18300	4436	24%	14401	78%	225,20	4472	24%	13124	71%	279,16	186 sec	0 sec
60	46	120	1	100	18800	4813	25%	14420	76%	244,18	3681	19%	11912	63%	321,03	20 sec	0 sec
60	46	120	4	100	18800	4884	25%	14397	76%	244,43	3760	20%	12059	64%	319,28	20 sec	0 sec
237	182	400	8	100	18300	4689	25%	14075	76%	249,98	4471	24%	13124	71%	279,16	161 sec	0 sec
237	182	200	2	100	18800	4952	26%	13915	74%	265,72	4586	24%	13175	70%	284,65	162 sec	0 sec
237	182	200	3	100	18800	4974	26%	13925	74%	265,95	4610	24%	13276	70%	284,46	160 sec	0 sec
237	182	200	4	100	18800	4973	26%	13943	74%	265,93	4560	24%	13227	70%	284,68	156 sec	0 sec

Table C.6: Experimental results of the k Nearest Neighbor algorithm (6th part.)

Horizontal cell count	Vertical cell count	Neighbor count (*k*)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	200	6	100	18600	4951	26%	13833	74%	265,22	4457	23%	13080	70%	285,78	161 sec	0 sec
118	90	40	6	100	18600	5097	27%	13406	72%	276,35	3908	21%	12142	65%	308,54	52 sec	0 sec
118	90	200	8	100	18300	4552	24%	14153	77%	246,89	3905	21%	12146	66%	301,27	57 sec	0 sec
118	90	300	6	100	18600	4512	24%	14335	77%	242,07	3904	20%	12152	65%	308,29	54 sec	0 sec
237	182	800	1	100	18800	4551	24%	14499	77%	242,00	4542	24%	13185	70%	282,81	162 sec	0 sec
237	182	800	2	100	18800	4521	24%	14515	77%	242,15	4580	24%	13178	70%	284,60	167 sec	0 sec
237	182	800	3	100	18800	4514	24%	14517	77%	242,12	4604	24%	13276	70%	284,40	164 sec	0 sec
237	182	800	6	100	18600	4508	24%	14462	77%	240,72	4461	23%	13080	70%	285,74	166 sec	0 sec
60	46	80	1	100	18800	4716	25%	14263	75%	254,67	3681	19%	11912	63%	321,03	19 sec	0 sec
60	46	80	2	100	18800	4781	25%	14254	75%	254,68	3690	19%	12001	63%	322,59	20 sec	0 sec
60	46	80	3	100	18800	4750	25%	14270	75%	254,82	3702	19%	11924	63%	321,23	19 sec	0 sec
60	46	80	4	100	18800	4770	25%	14276	75%	254,89	3760	20%	12059	64%	319,28	19 sec	0 sec

Table C.7: Experimental results of the k Nearest Neighbor algorithm (7th part.)

Horizontal cell count	Vertical cell count	Neighbor count (*k*)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	400	1	100	18800	4744	25%	14173	75%	256,76	4542	24%	13185	70%	282,81	159 sec	0 sec
237	182	400	2	100	18800	4702	25%	14176	75%	256,88	4580	24%	13178	70%	284,60	163 sec	0 sec
237	182	400	3	100	18800	4723	25%	14159	75%	256,95	4604	24%	13276	70%	284,40	162 sec	0 sec
237	182	400	4	100	18800	4720	25%	14152	75%	257,00	4555	24%	13229	70%	284,63	158 sec	0 sec
237	182	400	6	100	18600	4688	25%	14061	75%	256,20	4461	23%	13080	70%	285,74	163 sec	0 sec
118	90	60	6	100	18600	4983	26%	13599	73%	272,24	3900	20%	12147	65%	308,35	52 sec	0 sec
118	90	100	1	100	18800	4897	26%	13866	73%	265,73	3821	20%	12119	64%	314,07	53 sec	0 sec
237	182	120	2	100	18800	5053	26%	13816	73%	269,36	4599	24%	13173	70%	284,82	162 sec	0 sec
237	182	120	3	100	18800	5074	26%	13777	73%	270,06	4609	24%	13267	70%	284,62	160 sec	0 sec
237	182	120	4	100	18800	5047	26%	13780	73%	269,94	4557	24%	13221	70%	284,88	156 sec	0 sec
237	182	200	1	100	18800	4971	26%	13894	73%	265,44	4546	24%	13184	70%	282,85	157 sec	0 sec
237	182	40	1	100	18800	5150	27%	13478	71%	274,07	4587	24%	13152	69%	284,55	158 sec	0 sec

Table C.8: Experimental results of the k Nearest Neighbor algorithm (8th part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
118	90	40	2	100	18800	5091	27%	13496	71%	277,30	3861	20%	12109	64%	314,22	52 sec	0 sec
237	182	40	2	100	18800	5138	27%	13486	71%	275,87	4627	24%	13098	69%	286,45	159 sec	0 sec
118	90	40	3	100	18800	5087	27%	13520	71%	276,93	3830	20%	12225	65%	311,31	52 sec	0 sec
237	182	40	3	100	18800	5156	27%	13456	71%	276,67	4642	24%	13210	70%	286,57	159 sec	0 sec
118	90	40	4	100	18800	5119	27%	13500	71%	277,19	3895	20%	12260	65%	310,93	52 sec	0 sec
237	182	40	4	100	18800	5176	27%	13515	71%	276,79	4599	24%	13175	70%	286,75	158 sec	0 sec
237	182	40	6	100	18600	5116	27%	13369	71%	277,28	4512	24%	13029	70%	288,16	159 sec	0 sec
237	182	4000	8	100	18300	4227	23%	14411	78%	226,49	4472	24%	13124	71%	279,16	197 sec	0 sec
118	90	200	6	100	18600	4555	24%	14140	76%	253,08	3904	20%	12152	65%	308,29	53 sec	0 sec
118	90	300	1	100	18800	4537	24%	14401	76%	242,96	3824	20%	12119	64%	314,07	54 sec	0 sec
118	90	300	2	100	18800	4540	24%	14420	76%	243,28	3859	20%	12119	64%	314,01	54 sec	0 sec
118	90	300	3	100	18800	4526	24%	14411	76%	243,27	3828	20%	12232	65%	311,11	59 sec	0 sec

Table C.9: Experimental results of the k Nearest Neighbor algorithm (9th part.)

Horizontal cell count	Vertical cell count	Neighbor count (*k*)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
118	90	300	4	100	18800	4517	24%	14420	76%	243,40	3892	20%	12278	65%	310,68	54 sec	0 sec
60	46	40	3	100	18800	4731	25%	13918	74%	265,23	3702	19%	11924	63%	321,23	19 sec	0 sec
118	90	100	2	100	18800	4825	25%	13918	74%	266,11	3863	20%	12119	64%	314,00	52 sec	0 sec
118	90	100	4	100	18800	4843	25%	13922	74%	266,32	3894	20%	12278	65%	310,69	52 sec	0 sec
118	90	100	6	100	18600	4820	25%	13782	74%	265,85	3903	20%	12152	65%	308,29	52 sec	0 sec
118	90	60	1	100	18800	4977	26%	13682	72%	272,12	3822	20%	12116	64%	314,13	53 sec	0 sec
118	90	60	2	100	18800	4984	26%	13696	72%	272,62	3857	20%	12114	64%	314,06	52 sec	0 sec
118	90	60	3	100	18800	4989	26%	13691	72%	272,64	3836	20%	12230	65%	311,20	52 sec	0 sec
118	90	60	4	100	18800	5007	26%	13695	72%	272,83	3895	20%	12278	65%	310,72	52 sec	0 sec
237	182	80	1	100	18800	5043	26%	13722	72%	271,52	4558	24%	13167	70%	283,29	157 sec	0 sec
237	182	80	2	100	18800	5074	26%	13716	72%	272,64	4610	24%	13162	70%	285,22	161 sec	0 sec
237	182	80	3	100	18800	5072	26%	13699	72%	273,16	4604	24%	13257	70%	284,98	159 sec	0 sec

Table C.10: Experimental results of the k Nearest Neighbor algorithm (10th part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	80	4	100	18800	5041	26%	13698	72%	273,10	4571	24%	13211	70%	285,30	156 sec	0 sec
237	182	800	4	100	18800	4508	23%	14536	77%	242,20	4555	24%	13229	70%	284,63	162 sec	0 sec
237	182	3000	6	100	18600	4449	23%	14414	77%	232,10	4461	23%	13080	70%	285,74	187 sec	0 sec
118	90	200	1	100	18800	4525	24%	14234	75%	253,39	3824	20%	12119	64%	314,07	54 sec	0 sec
118	90	200	2	100	18800	4537	24%	14225	75%	253,75	3859	20%	12119	64%	314,01	53 sec	0 sec
118	90	200	3	100	18800	4533	24%	14210	75%	253,84	3828	20%	12232	65%	311,11	57 sec	0 sec
118	90	200	4	100	18800	4552	24%	14214	75%	253,94	3892	20%	12278	65%	310,68	54 sec	0 sec
60	46	40	1	100	18800	4742	25%	13898	73%	264,90	3681	19%	11912	63%	321,04	19 sec	0 sec
60	46	40	2	100	18800	4760	25%	13888	73%	264,77	3690	19%	12001	63%	322,60	19 sec	0 sec
60	46	40	4	100	18800	4754	25%	13900	73%	264,95	3760	20%	12059	64%	319,28	19 sec	0 sec
118	90	100	3	100	18800	4869	25%	13887	73%	266,31	3833	20%	12232	65%	311,11	54 sec	0 sec
118	90	40	1	100	18800	5059	26%	13496	71%	276,65	3820	20%	12100	64%	314,28	52 sec	0 sec

Table C.11: Experimental results of the k Nearest Neighbor algorithm (11th part.)

Horizontal cell count	Vertical cell count	Neighbor count (k)	n	Scenario length	total number of est.	mean <100	% of mean <100	mean <300	% of mean <300	rmse of mean	rbf <100	% of rbf <100	rbf <300	% of rbf <300	rmse of rbf	total time	avg. est. time
237	182	3000	1	100	18800	4457	23%	14451	76%	233,64	4542	24%	13185	70%	282,81	183 sec	0 sec
237	182	3000	2	100	18800	4452	23%	14450	76%	233,69	4580	24%	13178	70%	284,60	188 sec	0 sec
237	182	3000	3	100	18800	4461	23%	14442	76%	233,80	4604	24%	13276	70%	284,40	187 sec	0 sec
237	182	3000	4	100	18800	4454	23%	14433	76%	233,87	4555	24%	13229	70%	284,63	182 sec	0 sec
237	182	4000	6	100	18600	4230	22%	14421	77%	233,34	4461	23%	13080	70%	285,74	197 sec	0 sec
237	182	4000	1	100	18800	4245	22%	14445	76%	234,70	4542	24%	13185	70%	282,81	194 sec	0 sec
237	182	4000	2	100	18800	4244	22%	14443	76%	234,80	4580	24%	13178	70%	284,60	199 sec	0 sec
237	182	4000	3	100	18800	4243	22%	14439	76%	234,91	4604	24%	13276	70%	284,40	198 sec	0 sec
237	182	4000	4	100	18800	4231	22%	14429	76%	235,02	4555	24%	13229	70%	284,63	193 sec	0 sec