A Testbed for Real-Time Performance Evaluation of RSS-based Indoor Geolocation Systems in Laboratory Environment

A thesis submitted to the faculty of

Worcester Polytechnic Institute

in partial fulfillment of the requirements for the degree of

Masters of Science

in

Electrical and Computer Engineering

By

Mohammad Heidari

_____

April 21, 2005

Prof. Kaveh Pahlavan, Thesis Advisor

_____

Prof. Fred Looft, ECE Department Head,

_____

Prof. Wenjing Lou, Thesis Committee

_____

Prof. Allen Levesque, Thesis Committee

_____

To my wife

Abstract

Recently, there has been an enormous growth of interests in geolocation applications that demand an accurate estimation of the user's location in indoor areas. The traditional geolocation system, GPS, which was designed for being used in outdoor environments, does not perform well in indoor areas, causing frequent inaccuracies in location estimation. Therefore the need for more accurate positioning systems and even positioning techniques is a motivation for researchers to turn their attention into indoor positioning systems.

In this thesis we present a unique testbed for indoor geolocation system's real-time performance evaluation. Then we present a real-time performance evaluation of a sample indoor positioning system. We make a comparison between the simulated results of the performance evaluation of the positioning engine and the real-time performance evaluation of the positioning system. Finally, we perform a sensitivity analysis for Ekahau™ indoor positioning engine. We show that the simulation with the introduced testbed yields the same results as one would obtain by evaluating the performance of the positioning system by means of massive measurement campaigns. Running the testbed for several measurement campaigns for different scenarios enabled us to compare the results and study the effect of selected parameters on the performance of the positioning system. We also perform primitive error analysis in terms of distance error to verify the

validity of the result obtained with the testbed. We show that under the same configuration both real-time performance evaluation and simulated performance evaluation will yield same result with respect to position error. We also use error modeling to determine which error model is best matched to the observed indoor positioning error.

Amongst all of the possibilities of choosing methods of positioning, we focused on the Received Signal Strength (RSS) based method along with fingerprinting. Briefly said, profiles previously gathered by measurement or simulation will decide on the location of mobile terminal if a new profile comes in.

It is worth mentioning that previous work similar to this testbed has been done for outdoor areas [16]. Their work is mainly focused on outdoor environment, in which multipath does not exist. In this research effort we tried to analyze the effect of different parameters on sensitivity of indoor positioning systems who suffer from multipath. Different setups for simulating real-time radio channels have been studied as well [17, 18], but still not focused on indoor areas.

Acknowledgement

I am greatly thankful to Professor Kaveh Pahlavan, not only for his guidance in academics, research, and skills but also enlightening me with the true philosophy of life. He has been more than an advisor to me. He provided me the insight, knowledge, and advice through my work and accomplishments were only possible because of his help and encouragement.

I am also very grateful to my fellow WPI friends, Bardia Alavi, and Nayef Ali Alsindi for their continuous help, support, friendship, and nice working atmosphere they provided.

I would also like to thank Dr. Fred Looft as the department head of Electrical and Computer Engineering department in WPI who gave me the opportunity of continue my studies here in WPI as a TA.

Simply I could not have reached where I am today without my father, Mr. Ahmad Heidari, my mother, Ms. Vahedeh Houshivar, my sister Hannaneh Heidari and my brother Hamed Heidari. To them I dedicate this work. And … Words can not express my thankfulness to my wife whose love is the source of my motivation to live.

Table of contents

List of figures

List of tables

1 Introduction

1.1 Background and motivation

Recently indoor geolocation applications have attracted considerable attention in the field of telecommunication. Accurately predicting the location of an individual or an object definitely can be a difficult task producing ambiguous results because of the harsh wireless environment. Applications for indoor geolocation systems can be placed in three main categories; commercial, public safety and military applications. In commercial applications the need for locating patients in a hospital, guiding blind people or tracking small children or elderly individuals is of great importance as well as locating specific and important objects in warehouses or in-demand objects in hospitals. Public safety application includes locating inmates in a prison or firefighters in a burning building. In military applications the main interest is locating soldiers in combat. Recently software packages have become available in the market that can locate the predefined object almost precisely, but still there is a strong interest for even more accurate systems comparable to outdoor geolocation systems such as Global Positioning System (GPS).

The indoor radio propagation channel is characterized as site-specific, exhibiting severe multipath and low probability of a Line Of Sight (LOS) signal propagation path between the transmitter and receiver [1]. The two major sources of errors in the measurement of location metrics in indoor environments are multipath fading and no LOS (NLOS) conditions due to shadow fading [2].

Radio propagation channel models have been developed to provide a means to analyze the performance of wireless receivers. The performance criteria adopted for telecommunication and geolocation systems are quite different [2]. The standard performance criterion for evaluating telecommunication systems is the Bit Error Rate (BER) of the received data stream, while for geolocation systems the most useful performance criterion is the accuracy of the estimated location coordinates. The accuracy of location estimation is a function of the accuracy of location metrics and the complexity of the positioning algorithms. Since the metrics for geolocation applications are Angle of Arrival (AOA), Received Signal Strength (RSS), and Time of Arrival (TOA), models for geolocation application must reflect the effects of channel behavior on the estimated value of these metrics at the receiver. The existing narrowband indoor radio channel models designed for telecommunication applications [2] can be used to analyze the RSS for geolocation applications. The emerging 3D channel models developed for smart antenna applications [3, 4] might be used for modeling of the AOA for indoor geolocation applications. However, the existing wideband indoor multipath channel measurement and models [1] are not suitable for analysis of the behavior of TOA for geolocation applications.

Basically the indoor geolocation procedure begins with collecting metrics related to the position of the mobile terminal relative to the reference point. Almost any sort of metric which is used in telecommunication systems can also be used in Geolocation systems. AOA and RSS are the most popular ones but one can use TOA and Phase of Arrival (POA) as well. For example, these metrics are used widely in location estimation

systems. GPS, which is the most well-known positioning system, estimates the location of the desired object by using the TOA of received signal.

The second step is to process the gathered metrics and estimate the location of the desired person or object. This step usually requires signal processing knowledge unless the fingerprinting method is used. In using the fingerprinting method, before any location estimation is attempted, one should build a grid-network for the place which the system is going to be installed and collect the metric according to the location of each node in the grid. After building the database for a new location one can measure the new metric respective to the new location and compare it with the database to find the best node which could be referred to the desired point. Processing the received data is the most tedious task in other methods of positioning. Figure 1 shows a block diagram of the positioning process. The more reliable measured metrics we have, the less complex the algorithm of estimation would be. The final step is to display the estimated location in an

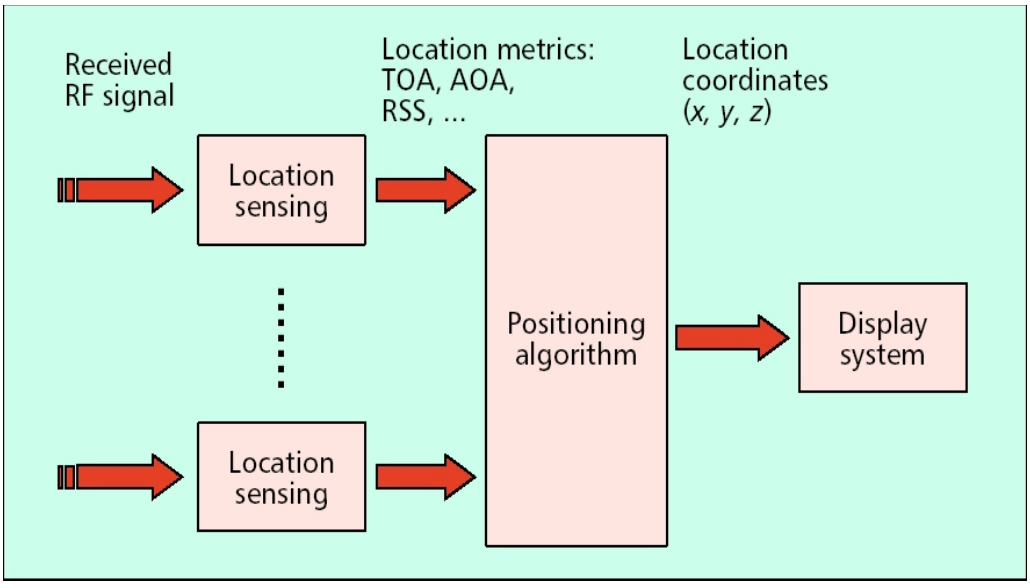

Figure 1: Outline of an indoor geolocation system

elegant method with a Graphic User Interface (GUI).

There are two approaches to building an infrastructure for positioning systems [5]. The first approach is to use an existing infrastructure that has already been designed for Wireless LAN (WLAN) communications and develop a signaling system that is compatible with the existing infrastructure. The second approach is to design a specific infrastructure for positioning. Each of these approaches has its own advantages and disadvantages. By using existing WLAN infrastructure the design of the signal processing part of the positioning system becomes of greater importance. The WLAN infrastructures have been developed specifically for telecommunication purposes while the design of positioning systems needs very different methods of modeling and signaling. On the other hand the cost for developing such an infrastructure for indoor positioning is minimal because the infrastructure already exists. Building a specific infrastructure for geolocation can be costly and installing the devices may be labor-intensive.

In this thesis we introduce a unique testbed for investigating the effects of different parameters on a sample positioning engine. We have performed numerous experiments have been done to show how parameters such as location of Access Points, distance between Access Points, or even number of Access Points can affect the accuracy of a geolocation system.

1.2 Contribution of the thesis

In this research effort we developed a real-time channel simulation environment for laboratory performance evaluation of indoor geolocation systems. The Ekahau™

(http://www.ekahau.com)[1] positioning software was used as the test positioning system for performance evaluation. The main objectives of this thesis are as follows:

- Developing a testbed for RSS-based indoor positioning systems
- Defining a framework for studying the performance evaluation of the indoor positioning system.
- Sensitivity analysis of a sample indoor positioning system to different parameters, *i.e.*, number of access points, number of training points, and location pattern of access points.

The core of the testbed hardware was the PROPSim™ real-time channel simulator originally developed for real time performance evaluation of wireless communication modems. We interfaced the hardware to 2D Ray tracing software to make it suitable for simulation of indoor location identification. We described the block diagram of the testbed and functionality of each block in addition to the details of hardware modification required to develop the testbed. Different measurement campaigns pertinent to each parameter are characterized. A specific experiment relevant to each parameter can be done to fully explore the effects of the parameter on the accuracy of the system. Results from different scenarios are discussed. Finally we discuss on the results and the relationship between the results and what has been expected.

---

[1] In this research effort, we used Ekahau Positioning Engine (EPE 3.0) product from Ekahau™. Related information can be found in Ekahau™ website : http://www.ekahau.com

1.3 Outline of the thesis

In this thesis we developed a testbed and a framework of laboratory performance evaluation of indoor geolocation systems. The remainder of the thesis is outlined as follows. Chapter 2 provides an overview of indoor geolocation systems. The system architecture and geolocation specific metrics are explained. In addition a classification methodology is introduced for RSS-based indoor channel measurements. The importance of the fingerprinting method is also examined. Chapter 3 describes the testbed development and discusses the details of each block in the block diagram and functionality of each block in the entire system. Chapter 4 introduces the various parameters that might affect accuracy of a positioning system. Chapter 5 provides the details of each measurement scenario[1] and discusses on the result and the relationship between the scenario and specific parameters. Finally, chapter 6 summarizes the results and discusses the future work which can be done through this testbed.

[1] The term "scenario" is used widely in this thesis. All of the experiments are done in the third floor of Atwater Kent Laboratories in WPI. However, these experiments are differentiated by the configuration of the access points and training points. Each of these experiments is referred to a scenario in this thesis.

2 Performance evaluation in indoor positioning

2.1 Background

The problem of accurately locating a user in an indoor environment has recently become both appealing and challenging to researchers. There are emerging civilian and military applications for indoor geolocation. In certain applications, the users have an RF tag that can be worn and while walking through a building they can be located accurately. This could be implemented in schools where the youngsters could be tagged so that the teachers knows exactly where they are at all times. In addition this technology can be used in hospitals to locate patients or in-demand equipment and medications. The harsh site-specific multipath environment in indoor areas introduces difficulties in accurately tracking the position of objects or people. The growing interest and demand for such applications dictates examining position estimation more carefully. The indoor channel poses a serious challenge to system designers due to the harsh multipath environment. The behavior of the channel changes from building to building and even within a single floor of a building. The channel may vary with added objects and people moving in the vicinity. As a result, considerable work is needed for modeling the indoor channel for geolocation applications.

Besides the channel modeling problem, another issue related to indoor positioning systems is their performance evaluation. We already know that in communication systems data-rate is usually the most important metric and every system is being evaluated by data-rate, but in positioning applications the most important metric is distance. Multipath would cause a problem in performance evaluation as the channels

might change from time of evaluation to the time of actual measurement and deployment. Another related problem is the inconsistency of the indoor channels. Since the channel changes fast, even in real-time indoor positioning systems the channel may change during calibration or evaluation and thus we may be forced to redo the evaluation. These are the related issues to performance evaluation.

Evaluating the performance of systems with the same functionality is the best way to compare them. This also gives us the statistics of how precisely the specific system works. Defining same scenarios for different systems and evaluating the performance of them gives us insights into the pros and cons of using each system. For example in telecommunication systems the standard performance criterion is data-rate. So for comparing two telecommunication systems we might observe which one provides higher data-rate. However, it is also likely that the system whose data-rate is lower has more stable response. Therefore in applications where we really do not need very high data-rate, but we need a more stable system which guarantees us that link between transmitter and receiver would work any time, we would rather use the system with more stable response. All of this information might be observed by defining the same positioning scenario and evaluating the performance of the systems. Of course the systems would be exactly in the same situation, *i.e.* same link between transmitter and receiver, same power transmitted, and same environmental situation. The idea of this kind of performance evaluation can be generalized to positioning systems. The criterion for comparing two positioning systems is different from the criterion in telecommunication systems. In positioning systems, since we are dealing with location and we are not concerned about the amount of data transmitted and received, it is understandable to define distance as our

criterion. If we define the performance evaluation criterion as the error in the difference of estimated location and actual location, then we would be able to compare systems under the same circumstances.

For outdoor positioning systems, such as GPS, performance evaluation has already been done. But anyhow this is a cumbersome and time consuming task. However, for performance evaluation of indoor positioning systems literature did not yield anything unambiguous. Running massive measurements is the simplest way to evaluate the performance of a positioning system. Although same situation exists for outdoor systems in terms of inconsistency of channel, they suffer less because in outdoor areas for a specific channel, the number of different paths between transmitter and receiver would not be as much as indoor areas. Hence evaluating the performance of an indoor positioning system is even more difficult and burdensome. The next issue is that since these are real-time measurements, it is obvious that the environment changes and the same situation do not exist for comparing systems. This sort of evaluation is also not repeatable since channel varies.

Section 2.2 takes a look at different elements that make up a typical indoor geolocation system. In addition, the different design approaches are described with complementary examples. Also section 2.2 introduces the different indoor geolocation metrics that deal with different aspects of the physical layer. More specifically the RSS-geolocation based metric is described in more detail since it is the focus of this thesis. Section 2.3 discusses in some detail the challenges of performance evaluation of indoor positioning systems.

2.2 Elements of indoor geolocation and various Indoor geolocation techniques

A typical functional block diagram of a wireless geolocation system is shown in Figure 2. It is composed of three elementary blocks. The first block is the location sensing block where the desired location metrics such as AOA, RSS, or TOA are extracted from the indoor propagation channel.

Second, with certain accuracy, these parameters are fed to the positioning algorithm block, which it produces the (x, y, and z) location coordinates. The algorithm receives the measured metrics from the indoor channel with a certain error and tries to improve the positioning accuracy. As a result when the metric collection procedure lacks accuracy then the positioning algorithm will have to be more complex. For each metric a different technical foundation exists. Regarding the metric which one uses the procedure and hardware equipment will be different and the accuracy changes. Even the

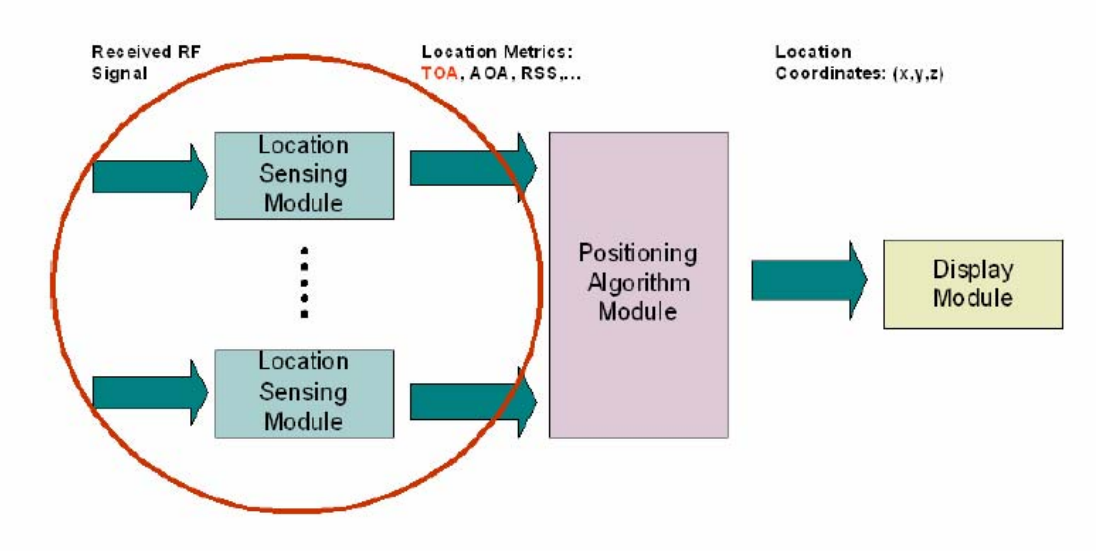

Figure 2: A functional block diagram of wireless indoor geolocation systems

requirement of the system would be different, i.e. bandwidth, time resolution, and etc. This thesis focuses on RSS-based indoor positioning. However it is worth mentioning the other metrics in order to have an overview of the different available positioning techniques. Finally the display system presents the location coordinates for the user. The system can provide the coordinates numerically or it can provide them in graphical user interfaces on a certain site-specific map. For example, the user can walk around with a PDA and can view his own location within a floor, or a worker tries to identify the location of a product in a warehouse, then he walks around with his display unit until he finds his desired object.

In general, there are two approaches for wireless indoor geolocation implementation. The first approach is to develop a signaling system and a network infrastructure of location sensors focused primarily on geolocation application [5]. The second is to use an existing wireless network infrastructure to locate a mobile terminal (MT) such as Wireless Local Area Networks (WLAN). The advantage of the first approach is that the system details are tailored towards the positioning application. The focus is on detecting the first path since it is very important in positioning and all the system architecture building blocks are designed accordingly. In addition the overall design is under the control of the system designer. As a result the system could be implemented as small wearable tags or stickers and the complexity and density of the locating infrastructure can be customized according to the degree of accuracy needed. The second approach has the advantage that it avoids expensive and time-consuming infrastructure deployment. On the other hand, more intelligent algorithms are needed in such systems to compensate for the low accuracy of the measured metrics. In terms of

system implementation when considering the first approach, the advantage is that the geolocation system is designed from the ground up. One way to approach it is the implementation of super-resolution algorithm for higher time-domain resolution. The system captures snapshots in the frequency domain and then through use of spectral estimation it is possible to obtain an accurate representation of the time domain. Another emerging approach that has better accuracy and potential is Ultra wideband (UWB) technology [5]. The large bandwidth provides high time-domain resolution which in return provides better ranging accuracy. For the second approach, the use of the network infrastructure in indoor geolocation is also feasible but more complex algorithms are needed in order to compensate for overall design. One current example is Ekahau positioning software, which uses existing WLAN infrastructure. Unlike the other positioning technologies, Ekahau does not apply propagation methods that suffer from multipath, scattering and attenuation effects. Instead Ekahau collects radio network sample points from different site locations. Each sample point contains Received Signal Strength Intensity (RSSI) and the related map coordinates, stored in an area-specific positioning model for accurate tracking. Ekahau provides average positioning accuracy approaching 1 meter. In this research we have analyzed this method even further for different measurements conducted for indoor geolocation. The software is compatible with industry-standard Wi-Fi (IEEE 802.11b) networks [6]. When it comes to system deployment, a positioning model is first created. Then the positioning model is calibrated where RSSI samples are collected from the different points on the map. Then the tracking or positioning can start once the system is calibrated. In other words, this positioning algorithms works with the WLAN infrastructure and no information about the access

point locations is required. Such technology depends on complex positioning algorithms and does not concentrate on the physical layer. In fact, it uses RSS as a metric instead of trying to extract the TOA or AOA which is more challenging task at the physical layer. Needless to say, when following the RSS method and bypassing the propagation issues the complexities lie in the software itself.

As mentioned earlier there are different metrics that can be used in indoor positioning. Although this research focuses on RSS-based indoor geolocation it is worth mentioning the other techniques. In AOA-based indoor geolocation direction-based triangulation is used, where two or more reference points (RPs) are used to determine the position of the MT. The AOA is usually measured with directional antennas or antenna arrays. This metric is not preferable in indoor environment because of the harsh multipath which introduces inaccuracies into the detection of the AOA in both LOS and OLOS conditions. In TOA-based geolocation systems the important parameter is the TOA of the Direct Line of Sight (DLOS) path since it is directly proportional to the physical distance between transmitting and receiving antennas. However since the system is not ideal – it has finite bandwidth, finite dynamic range, and introduces noise – the DLOS path can never be extracted perfectly from a measurement. The most reasonable approximation is the first detected path in the profile above a given noise floor. The other paths are also important since they can affect the TOA and amplitude of the first path [7]. The strength of the strongest path relative to the weakest path provides the dynamic range of the system and it is also important. The remaining paths are not very important in geolocation. Instead they are more important for telecommunication in terms of multipath

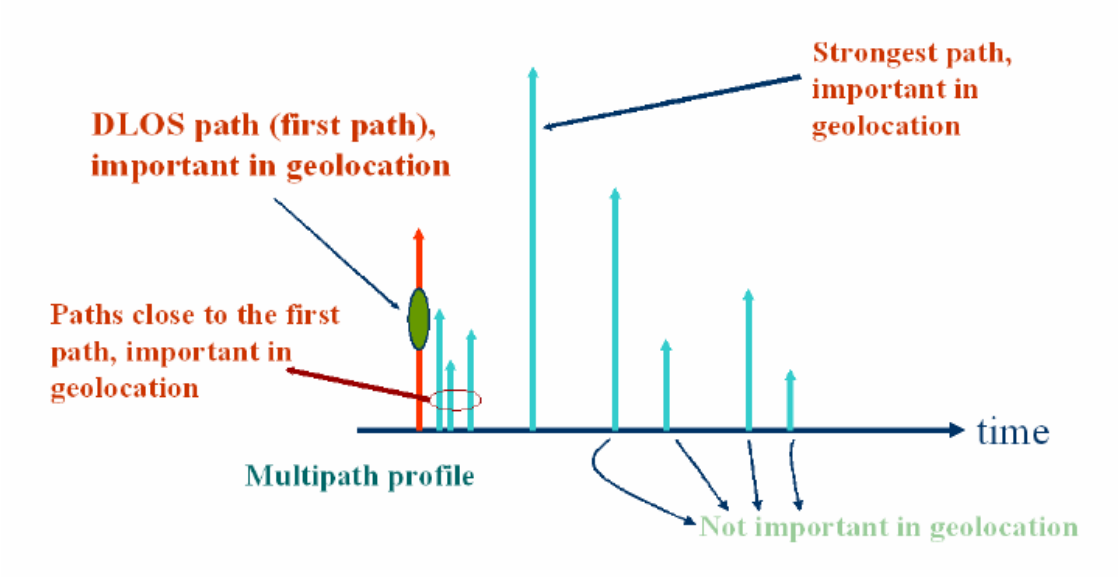

Figure 3: TOA example of categorizing paths

delay spread. An example of the indoor multipath and the geolocation specific parameters is shown in Figure 3.

In RSS-based indoor geolocation, as it is obvious from its name the received signal strength is measured at the receiver. The RSS is related to the distance between the transmitter and the receiver mathematically in the form of path loss models [5]. The path loss models portray the signal power attenuation as the signal travels through the indoor environment. If the path loss model is known in advance then the distance between the transmitter and receiver can be calculated by measuring the received signal strength and comparing to the known path loss model. A wide variety of path loss models have been developed for different environments, each with different values of model parameters or different parameters and mathematical function forms [1]. The path loss model in indoor environment is highly site-specific. For example, the value of power-distance gradient, which is a parameter of path loss models, varies in a wide range between 15-20 dB/decade and a value as high as 70 dB/decade[1]. As a result of the complex indoor

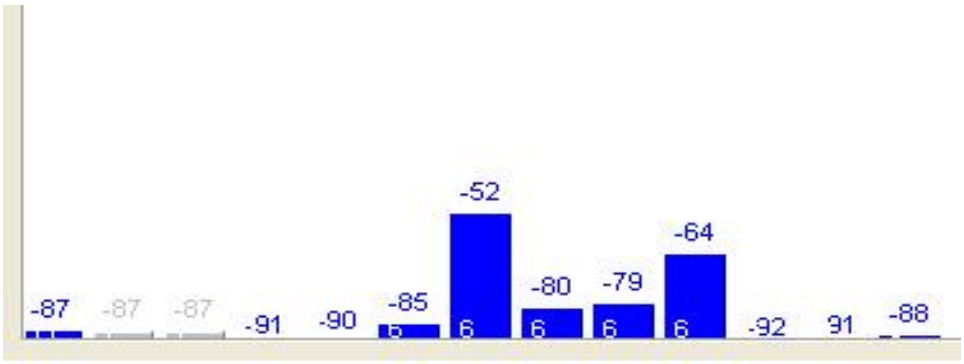

Figure 4: Power profile from Ekahau

radio propagation channel, in practice the RSS-based indoor geolocation technique can be accomplished by estimating the path loss model of a specific indoor environment during system installation. In addition, there has to be a frequent re-estimation of the path loss model of the indoor radio propagation channel in order to establish accurate positioning values. An example of an RSS-based geolocation system is Ekahau software which was described earlier. Figure 4 illustrates an example of the received power profile of a sample positioning system. The measured power is the average power within different paths that arrive to the receiver. In telecommunication models the relationship between this average power and power gradient has been investigated.

In spite of TOA-based systems since the metric in RSS-based systems is the average power and not instantaneous power, the relationship between received metric and error is ambiguous. The purpose of developing this testbed is mainly to find a method to analyze the error with what we had as result from positioning system.

2.3 Challenges for indoor geolocation performance evaluation

As it was mentioned before basically indoor geolocation systems be classified into two major categories; network based systems and handset based systems. As it is obvious from the names, in handset based systems, mobile station receives the respective metric from fixed terminals and computes its location. In spite of that in network based systems fixed terminals receive the respective metric from mobile station and then they send the information to control station in where the location of the mobile station is computed. The network based systems also referred to as network dependent systems, depend on the ability of the mobile terminal to receive signal from a mobile network covering its area of presence. Conversely handset based technologies, also referred to as network independent technologies, can provide location identification information even in the absence of mobile network coverage. The prevalent solution in this category is the Global Positioning System (GPS). GPS is the worldwide satellite-based radio navigation system, consisting of 24 satellites, equally spaced in six orbital planes 20,200 kilometers above the Earth, that transmit two specially coded carrier signals, one for civilian use and one or military and government use. The system's satellites transmit navigation messages, which a GPS receiver uses to determine its position. GPS receivers process the signals to compute position in 3D – latitude, longitude, and altitude – with an accuracy of 10 meters or less. To operate properly, GPS receivers need a clear view of the skies and signals from at least four satellites, requirements that exclude operation in indoor environments [8].

However, indoor GPS systems have also been developed, that can compute the position of a mobile device in a closed environment, such as a building, taking

information from pseudo-satellites. Such systems are discussed in the following section. A slightly different version of GPS, called Assisted Global Positioning System (A-GPS), combines features of both network-based and handset-based technologies, and hence it can be considered as a hybrid solution. A-GPS helps to overcome some of the drawbacks of pure GPS such as cost, power consumption, and speed to determine location, and the line-of-sight requirement, by shifting much of the processing burden from the handset to the mobile network. In A-GPS, the network keeps track of location, so that when satellites are obstructed, a good estimate of location can be obtained based on last reading. A-GPS is accurate within 50 meters when users are in indoor environments and 15 meters when they are in outdoor areas. Although A-GPS is less costly than GPS from a handset perspective, it requires additional investment on behalf of the network. Nevertheless, its performance in terms of speed, coverage and accuracy is considered to be superior. That is why that A-GPS, augmented with elements from other location technologies, is expected to become the solution to which most wireless systems will ultimately converge [8].

In contrast to the aforementioned technologies that are capable of identifying the location of an object or person in open areas, indoor positioning technologies set the constraint of a limited coverage range, such as a building or other confined spatial area, *i.e.,* a stadium or an exhibition. These technologies are therefore not dependent on any 'external' network. However, they are dependent on a set of technologies used for transmitting wireless data in closed environments, such as radios, infrared sensors, wireless local area networks (WLANs), and Bluetooth [9]. These technologies are briefly reviewed in this section.

The first indoor positioning systems that were developed used infrared sensors [10]. In these systems, several infrared transmitters, which can automatically send their own IDs, hang from various places on a building, such as walls, doors, rooms, and corridors. A computing device with an infrared receiver uses these signals to determine its current position. However, as intervening objects can easily block infrared signals, radio-based positioning has emerged as a more attractive alternative. Most recent research in location-based computing has emphasized on newer rapidly developing technologies, such as WLAN, Bluetooth, and their associated positioning methods for the identification of objects and persons in limited-range areas. As an example of WLAN-based implementation, RADAR, a building-wide tracking system developed by Microsoft Research Group, is based on the IEEE 802.11 wireless networking technology [11]. RADAR measures the signal strength and signal-to-noise ratio of signals sent by wireless devices, and computes their 2D position within a building. This approach has the advantages of requiring only a few base stations, and using the same wireless networking infrastructure of the building. Its disadvantages include the difficulty in applying this system to multi-floor buildings. Bluetooth, the short-range radio standard for connecting devices and enabling point-to-multipoint voice and data transfer, provides higher proximity accuracy than WLANs. Therefore, future state-of-the-art location-based systems are expected to make use of Bluetooth technology, and several companies have already announced Bluetooth location-based services and positioning technology [12].

Another relatively new technology used for identifying and tracking objects within a few square-meters is Radio Frequency Identification (RF-ID). An RFID system integrates an antenna with electronic circuitry to form a transponder that, when polled by

a remote interrogator, will echo back an identification number. The advantages of the wireless RF-ID system include identification at a distance, hands-free operation, versatile memory and processing requirements, and high accuracy due to the very short operating range [13]. RF-ID technology has been successfully used in a wide range of markets and is expected to play primary role in future mobile location applications since it enables the automated data collection and tracking of objects as they move across a limited geographical area.

Finally, the Indoor GPS location identification system focuses on exploiting the advantages of GPS for developing a location-sensing system for indoor environments. It is now well understood that the GPS signal does not typically work indoors because the signal strength is too low to penetrate a building. Nevertheless, Indoor GPS solutions can be applicable to wide space areas where no significant barriers exist. Indoor GPS takes into account the low power consumption and small size requirements of wireless access devices, such as mobile phones and handheld computers. The navigation signal is generated by a number of pseudolites (pseudo-satellites). These are devices that generate a GPS-like navigation signal. The signal is designed to be similar to the GPS signal in order to allow pseudolite-compatible receivers to be built with minimal modifications to existing GPS receivers. As in the GPS system, at least four pseudolites have to be visible for navigation, unless additional means, such as altitude aiding are used. The signal generated by the pseudolites is monitored by a number of reference receivers. Latest innovations in this area have delivered receivers to the size of a postage stamp. The small footprint combined with ultra-low power consumption and low cost make it feasible to apply indoor GPS positioning technology in mass-market applications for the first time.

The metrics that are typically used in positioning systems are RSS, AOA, TOA, and TDOA. Here we take a glance at the technologies using these metrics in indoor scenarios. In RSS-based systems the power the received signals from three different fixed stations together with the path loss model of the building form a vector consisting of three distances respective to each fixed station. These distances can be used by triangulation method to estimate the location of the mobile terminal. Each distance from fixed terminal gives us a circle which means the perimeter which the mobile terminal must lay. Combining the three circles, triangulation, gives us the exact location of the mobile terminal as it is shown in Figure 5[14].

Considering the error that we have in our measurement system, the result of the triangulation would be a region instead of exact location which the transmitter must be located.

In this method the error in location estimation can be caused by multipath, non line-of-sight conditions, and local shadow fading. Also the algorithm in solving nonlinear equations may cause some error in position estimation. So briefly we can conclude that error in RSS-based methods would be substantially related to the environment properties

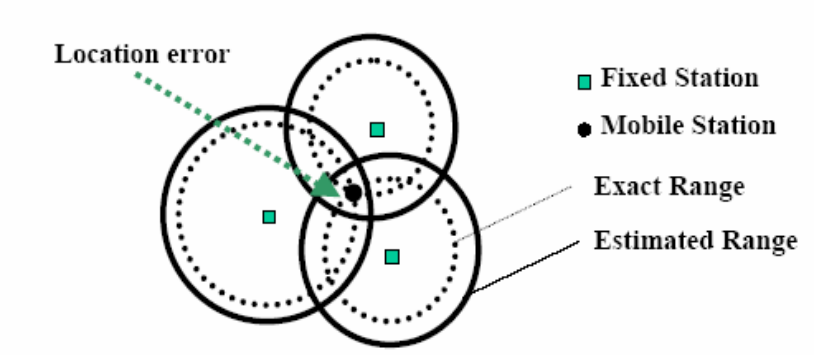

Figure 5: Triangulation

(multipath, NLOS, and shadow fading), path loss model used, number of fixed terminals uses, and the algorithm used to estimate the location.

AOA-based systems use the angle of the arriving signal to determine what are the logical areas which the mobile terminal can be located in. In these systems received signal from two fixed terminals would be enough to determine the location of the mobile terminal as it is shown in Figure 6[14].

Error in estimating direction of the arriving path is most likely due to multipath, NLOS, and the accuracy of the antenna array system been used. The algorithm been used to derive the location is also important. Similar to RSS-based systems the number of fixed terminals and relative location of them to mobile terminal is also important.

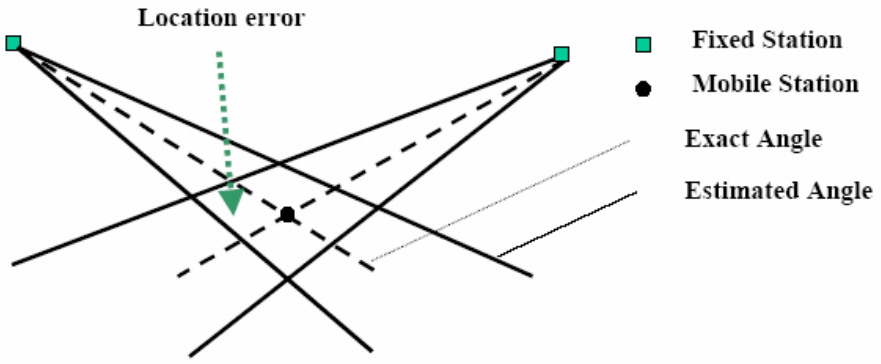

Figure 6: AOA-based range estimation

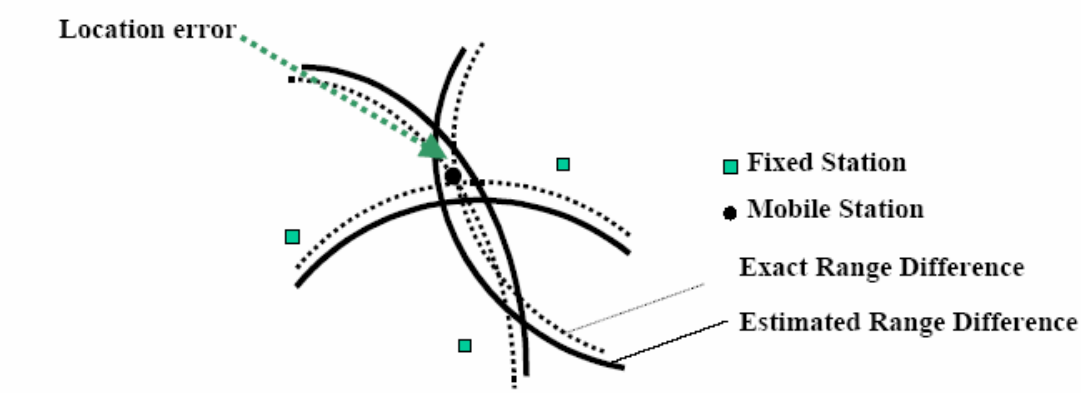

Figure 7: TDOA triangulation

TOA is another metric which have been used in many positioning systems. Since the speed of the signal is known the time that first path arrives in receiver is directly related to the distance between mobile terminal and fixed terminal. Triangulation is used to determine the exact location of the mobile terminal considering the time that first path has arrived. For this purpose the transmitter and receiver should be synchronous which is costly and needs more complicated circuitry both in transmitter and receiver. To solve this problem we may use time difference metric which is the relative times that signals arrive in receivers. Each metric in this system will give a hyperbola, with the foci on receivers, on which a transmitter must lie. Figure 7 shows us an example of such systems using TDOA. Intersection of the hyperbolas gives the region that transmitter must lie. In

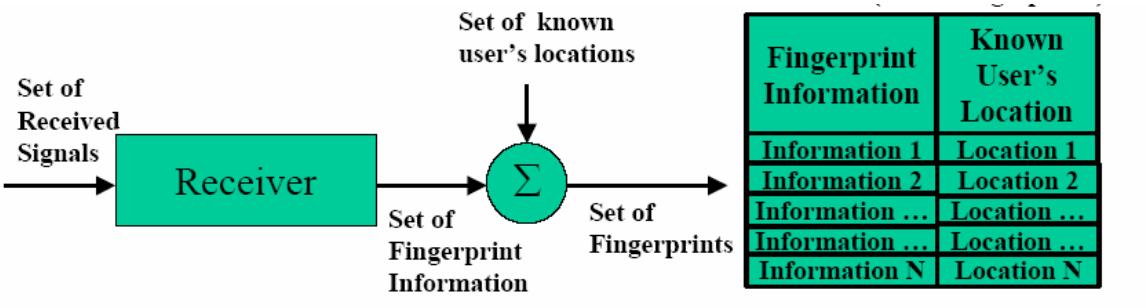

Figure 8: Offline Phase of fingerprinting

this case just receivers should be synchronous to each other [14].

Error would occur in case of having multipath, NLOS, and inaccuracy in algorithms were used in our system. Time-synchronous property would also play a major role in determining the location of the mobile terminal accurately.

Besides using metrics directly, there is another method to determine the location of the mobile terminal, fingerprinting. Fingerprinting basically means building database with respect to the received signal's metric related to the present location of the mobile terminal. Operation of such systems has 2 phases, named off-line phase and on-line phase.

Off-line phase is the data collection phase or learning phase. The major portion of this phase involves roaming around in the site and collecting data. Recording the set of information as a function of the user's location covering the entire zone of interest, we can form a database consisting of fingerprints. Each fingerprint corresponds to fingerprint information associated with a known user's location.

Real-time phase or on-line phase is the phase of matching the obtained fingerprint for the specific location and compared with the database. A pattern matching algorithm is then used to identify the recorded information closest to the new specific fingerprint.

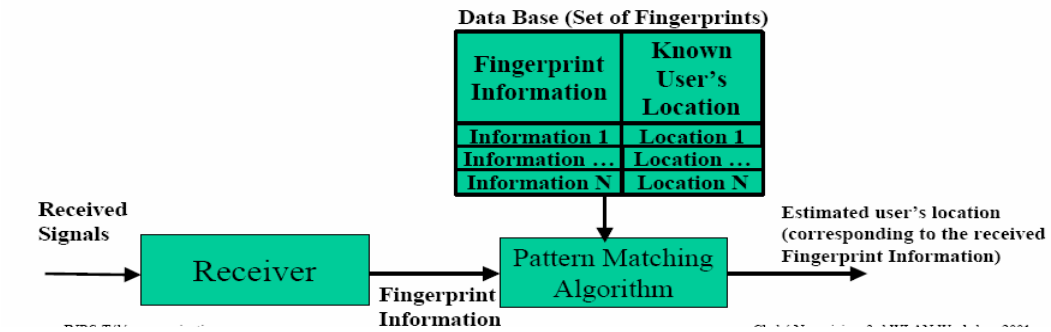

Figure 9: On-site phase of fingerprinting

3 A testbed for performance evaluation of indoor positioning systems

In this chapter we discuss a testbed for Real-Time Performance Evaluation of an Indoor Geolocation Systems using PROPSim channel simulator. Nowadays most of the positioning systems use RSS-based fingerprinting method which is dependent on building a database. The reason for developing such a testbed is that today there is no way to evaluate the performance of a positioning system except running massive measurements.

Running massive measurement to build database for positioning systems is both time consuming and costly to do. Roaming around the whole site, we should collect enough metrics and information for each point. Accuracy of the positioning system is closely related to the number of nodes in database and distribution of them. For example for a typical indoor area, at least we should gather information for nodes as close as 2-3 meters to each other if we want to have an accurate positioning system. Collecting this much information is very time consuming. This way also is not repeatable and every time we do the measurement we may get different error.

With this testbed we can run the whole experiment in the lab and evaluate the performance evaluation of a positioning system without gathering real data from different locations. Also the scenario is repeatable with this testbed, so we can compare the results of one positioning system with another one and compare the results. This method also reduces the cost of developing such a database.

The intuition here is to use a real-time channel simulator for performance evaluation of a positioning system. With the aid of this real-time channel simulator we can simulate the channel between transmitter and receiver antenna. We can divide the

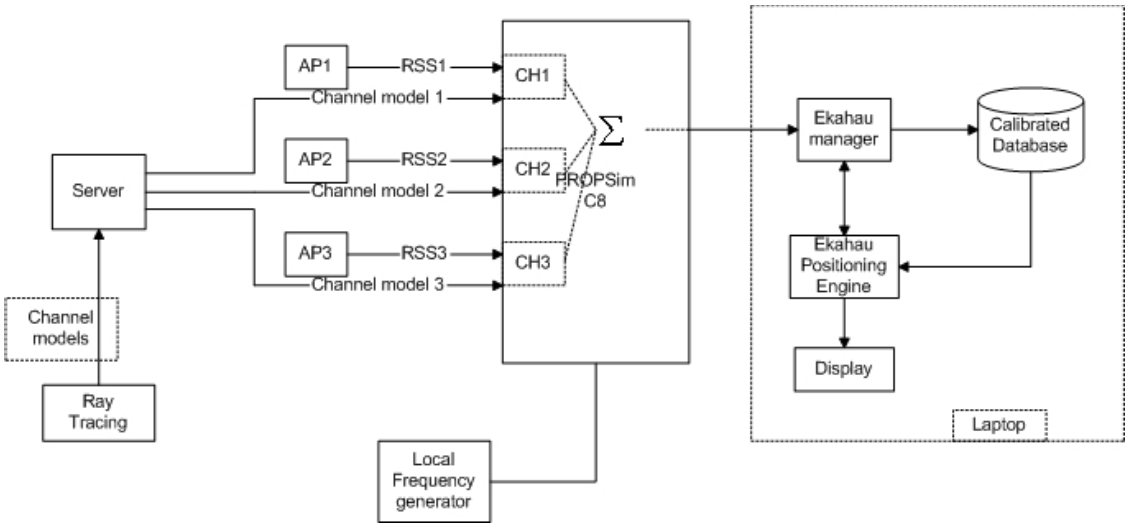

Figure 10: Block Diagram of the testbed

testbed into two main parts, software and hardware. Hardware of the testbed mainly consists of three parts, named channel simulator, wireless antennas, and server. Channel is responsible to simulate the channel between transmitter and receiver. Wireless antennas are mainly wireless access points transmitting and receiving.

Figure 10 shows us the block diagram of the testbed. As you can see the main block in this block diagram is the PROPSim real-time channel simulator.

We can categorize these blocks into two groups, software-oriented and hardware-oriented. Ekahau and Ray Tracing blocks are software oriented while PROPSim is hardware. We will get into the details of them in this section.

3.1 EKAHAU positioning engine

3.1.1 General description

The Ekahau positioning engine (EPE™) is a powerful software tool which is able to locate the target and provide the coordinates (x, y, and floor) corresponding to each client. The main concept of this research is using this real-time testbed to evaluate the performance of EPE™ software in the laboratory. EPE™ uses RSS-based system with fingerprinting method which enables us to locate the mobile terminal. Figure 11 shows us block diagram of the infrastructure of EPE™ system. This software communicates with any Wi-Fi-based (802.11 a/b/g) access points in the building to locate an object. Installing EPE™ is able to work with existing infrastructure that has been built for telecommunication purposes. The software consists of four components.

Ekahau client™: which is a small software module running on the client device.

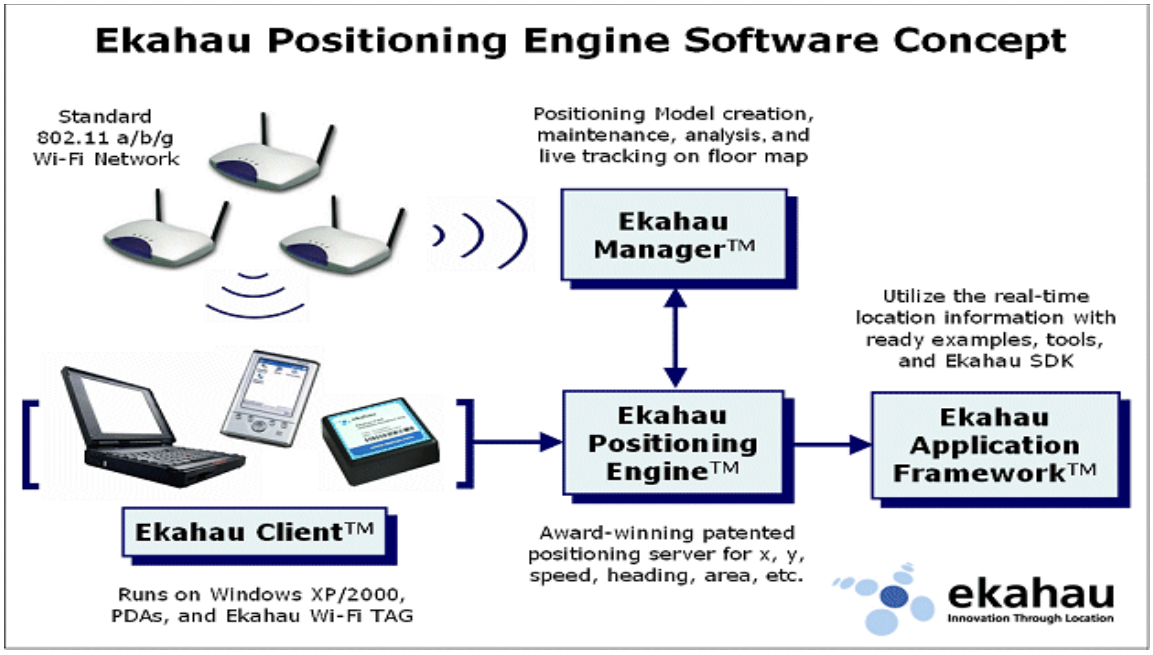

Figure 11: Ekahau Block diagram

The client device can be a PDA, laptop, or Wi-Fi tag.

Ekahau positioning engine™: This service would run on a PC or laptop as server to calculate the coordinates and/or other properties of the client.

Ekahau manager™: is an application for recording field data for a positioning model and calibration, analyzing data and tracking the object.

Ekahau application framework and SDK™: is a set of helpful tools and easy programming interface for authorized applications to quickly utilize EPE location information.

In using Ekahau software, we must first define a new positioning model by inserting the floorplan of our chosen scenario. Then we define paths and trails for our

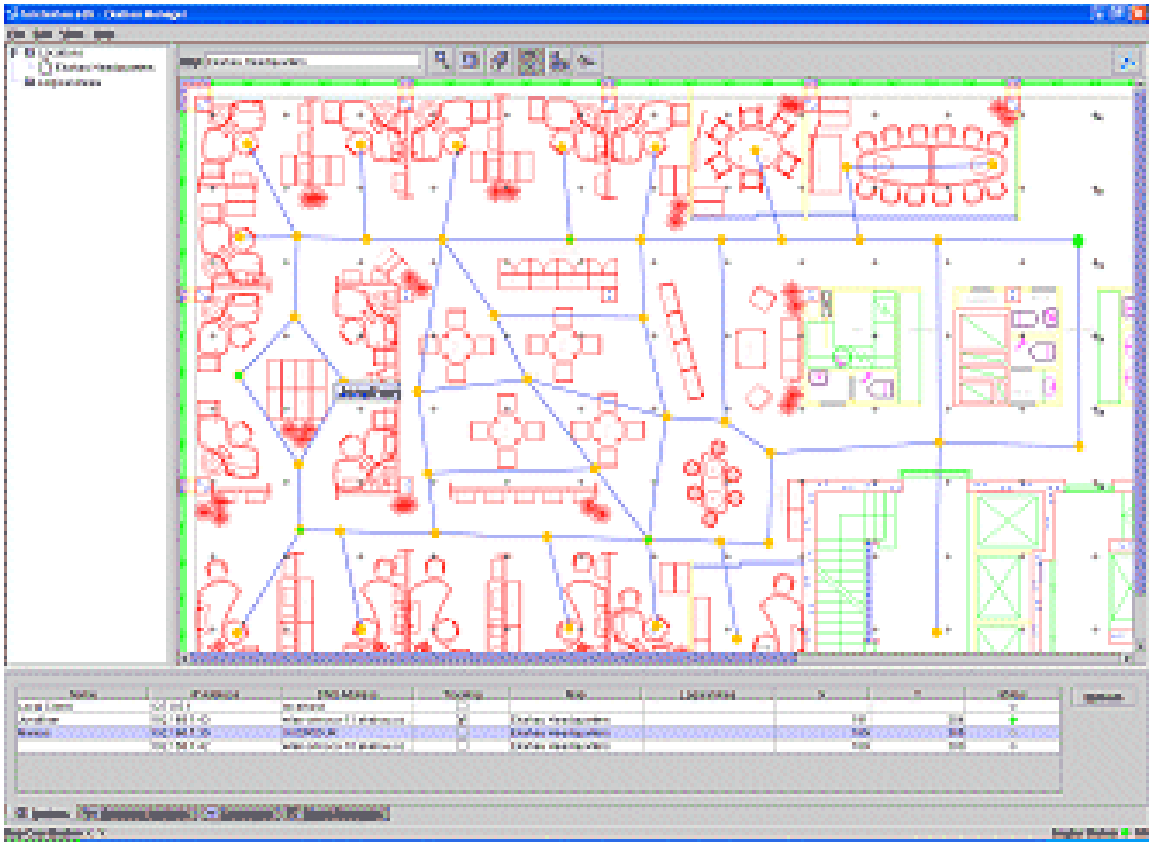

Figure 12: Ekahau Representation

model. These paths are the routine paths within we may wish to track specified objects. Subsequently, we must calibrate our model by walking around the building along those paths that we have defined for software and start collecting data by measuring the received power from different accessible access points. Every few meters we can gather information by reading the received power from access points. This will build our database as we walk through all the routine paths in the building and collect data. The more data we collect, the more accurate our location estimation can be. After collecting data, we can locate the desired target which can be either a laptop or PDA. We can visually analyze the positioning error vectors and statistics to find areas where additional access points or calibration sample points are needed.

3.1.2 Ekahau Algorithm

The conceptual development of location estimation has been solved geometrically by engineers since ancient times [15]. This approach needed exact measurement of the preferred metric since triangulation had been used. The new and unique approach to this problem is the statistical model approach which has been used in Ekahau software to locate the desired object.

In this approach signal metrics, such as RSS or AOA are treated as random variables which are statistically dependent on the location of transmitter, receiver, and the propagation environment. Because of this dependency an observation of a sample of signals metric leads us to a conclusion about the metric which will lead us to the location afterward.

The difference between these two approaches is quite evident. In geometric approach people are interested in mapping the metric into location while in statistical

modeling approach it is vice versa. The reasoning goes from location to signal properties in statistical modeling approach by means of dependency of signal properties and location. The problem is an inverse problem which is going to be solved then. In statistical terms, the propagation model is sampling distribution whose parameters are directly related to the location.

The problem of incompatible measurement is not present in the statistical modeling approach in contrast to geometric approach. It should be mentioned that although every unlike obtained measurement is possible but if the propagation model does not fit the actual propagation phenomena and environment the result of location estimation will not be accurate. To enhance the accuracy of statistical modeling approach one can use other propagation models rather than using more complicated transmitters or receivers to enhance the metric observation. Even more sophisticated models like neural networks or ray tracing can be used to enhance the accuracy of this approach.

3.1.2.1 Description of the problem

Given a set of n distinct locations $L = \{l_1, l_2, ,..., l_n\}$; *Where* $l_i = (x_i, y_i)$ $i \in \{1,2,...,n\}$ in area A and their associated observations $O = \{O_1, O_2, ,..., O_n\}$ *where* $O_i = \{o_{i1}, o_{i2}, ..., o_{ik}\}$ $i \in \{1,2,...,n\}$ for k observations in location l_i, $o_{i,j} = (r_{i,j,1}, r_{i,j,2}, ..., r_{i,j,m})$ where $r_{i,j,t}$ $t \in \{1,2,...,m\}$ vector representing RSS values from m different access points *(AP_1-AP_m)*, find the most probable location of a receiver which observes $o_{measured} = (\hat{r}_1, \hat{r}_2, ..., \hat{r}_m)$.

Figure 13 shows a similar scenario with $m = 5$ (AP_1-AP_5). We want to estimate the location of MH, given $o = (\hat{r}_1, \hat{r}_2, \hat{r}_3, \hat{r}_4, \hat{r}_5)$ is the RSS values it receives from AP_1-AP_5 respectively, and each point on the grid represents a reference point in which there is an associated RSS profile in a reference database.

We denote probability of observing $o = (r_1, r_2 ..., r_m)$; where $r_1, r_2 \ldots r_m$ represent

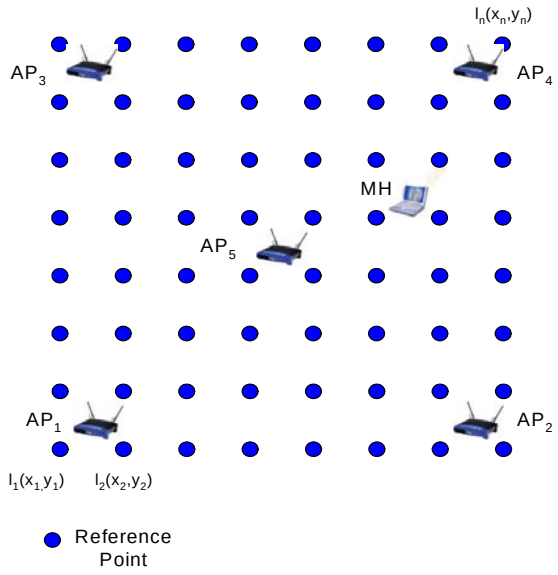

Figure 13: Example of geolocation problem

RSS values from $AP_1, AP_2 \ldots AP_m$ respectively; as $p(o)$ and *p(l)* are the prior probability of being at location *l*. we obtain the posterior distribution of the location by applying Bayes rule:

$$P_l(l \mid o) = \frac{P_o(o \mid l)\, p(l)}{p(o)} = \frac{P_o(o \mid l)\, p(l)}{\sum_{l' \in L} p(o \mid l')\, p(l')} \tag{1}$$

where $P_l(l \mid o)$ and $P_o(o \mid l)$ are conditional probability distribution functions of being in location l given observation $o = (p_1, p_2 ..., p_m)$, and probability distribution function of observing o given location *l*, respectively. Generally, the prior probability density function of *p(l)* is a mechanism to incorporate previous tracking information to this system. Here for simplicity we assume that *l* has a uniform distribution. Total probability $p(o)$ does not depend on the location *l*, thus it can be considered as a normalization factor. Application of these assumptions reduces (1) to (2) where η is a constant normalization factor.

$$P_l(l \mid_o) = \frac{P_o(o \mid l).p(l)}{p(o)} = \eta . P_o(o \mid l) \tag{2}$$

The probability density function $P_o(o \mid l)$ is called the likelihood function which in a discrete domain gives the probability of an observation, given the location. Equation (2) shows that estimation of the likelihood function leads to finding $P_l(l \mid o)$. The posterior distribution function $P_l(l \mid o)$ can be used to find the optimum location estimator based on any desired loss function. In particular, using expected value of the location, minimizes the mean squared error, and (3) shows this estimation.

$$E[l \mid o] = \sum_{l' \in L} l' . P_l(l' \mid o) = \sum_{l' \in L} l' \, . \, \eta . P_o(o \mid l') = \eta . \sum_{l' \in L} l' \, . \, P_o(o \mid l') \qquad (3)$$

Note that we can extend the above discussion to a two dimensional case where $l = (x,y)$ and hence:

$$\begin{aligned} E[x \mid o] &= \eta . \sum_{x' \in X} x' \, . \, P_o(o \mid x') \\ E[y \mid o] &= \eta . \sum_{y' \in Y} y' \, . \, P_o(o \mid y') \end{aligned} \qquad (4)$$

Thus, in order to implement this system we need to generate a likelihood function from a set of training data. There are several examples of such likelihood functions. Here we present the kernel method.

3.1.2.2 Kernel Method

We assign a probability mass to a kernel around each observation for a given location. Suppose we have *k* observations at point *l*. Equation (5) defines the prospective likelihood function.

$$P_o(o \mid l) = \frac{1}{k}\sum_{i=1}^{k} K(o, o_i)$$
$$o = (r_1, r_2, ..., r_m) \qquad (5)$$
$$o_i = (r_{i,1}, r_{i,2}, ..., r_{i,m})$$

$K(o,o_i)$ denotes the kernel function. If we assume that RSS (r_t) from access point t is a Gaussian ($\mu = p_t, \sigma^2$) random variable; where σ^2 is an adjustable parameter; moreover consecutive values of p_t are independent, we can use a Gaussian kernel defined by (6).

$$K_n(o, o_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(o-o_i)^2}{2\sigma^2}}$$
$$K_n(o, o_i) = K_n(r_1, r_2, ..., r_m, r_{i,1}, r_{i,2}, ..., r_{i,m}) \qquad (6)$$
$$K_n(r_1, r_2, ..., r_m, r_{i,1}, r_{i,2}, ..., r_{i,m}) = \frac{1}{(\sqrt{2\pi}\sigma)^m} e^{-\frac{1}{2\sigma^2}\sum_{j=1}^{m}(r_j - r_{i,j})^2}$$

Applying (6) in (5) results in:

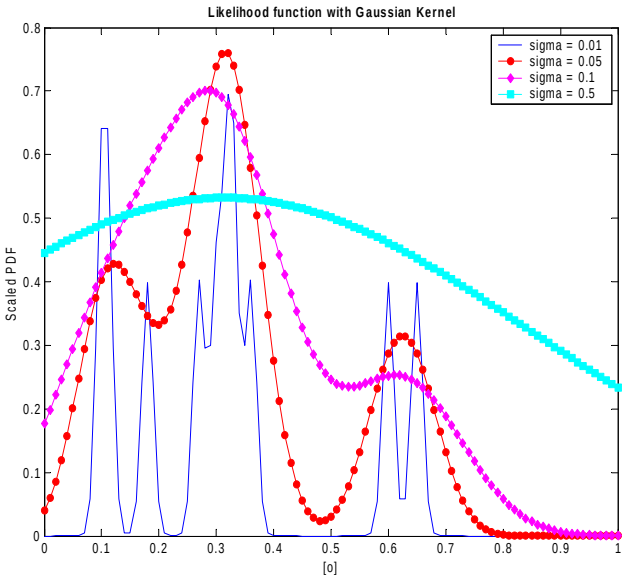

Figure 14: Kernel generated plots

$$P_o(o \mid l) = \frac{1}{k}\sum_{i=1}^{k}\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(o-o_i)^2}{2\sigma^2}}$$

$$P_{r_1 r_2 \ldots r_m}(r_1, r_2, \ldots, r_m \mid l) = \frac{1}{k}\frac{1}{(\sqrt{2\pi}\sigma)^m}\sum_{i=1}^{k}e^{-\frac{1}{2\sigma^2}\sum_{j=1}^{m}(r_j - r_{i,j})^2} \quad (7)$$

Example: Lets consider the following observations (0.1 , 0.11 , 0.18 , 0.27 , 0.3 , 0.32 , 0.33 , 0.36 , 0.6 , 0.65) in a location *L*. This corresponds to RSS values from a single access point in a number of trials at a fixed location *L*. Figure 14 shows the resulting likelihood functions for different values of σ. [1]

3.1.3 Implementation

3.1.3.1 Generating the Kernels

In order to apply (7) in its most generic case, we need to collect *k* different training vectors in each reference point for a reasonably large value of *k*. One way to reduce the excessive amount of measurements and also reduce the complexity of the system is to group the reference points in different clusters $\{c_1, c_2, \ldots, c_N\}$. Two different strategies for clustering are discussed in the next section. Our assumption is that training vectors in a cluster are collected from a single location (center of the cluster). Using this assumption we apply (7) in each cluster and create *N* likelihood functions defined by (8).

[1] This example has been worked out by Mr. Ahmad Hatami for the purpose of algorithm development based on [15]

$$P^{(j)}_{r_1,r_2,\ldots,r_m}(r_1,r_2,\ldots,r_m \mid c_j) = \frac{1}{k_j}\frac{1}{(\sqrt{2\pi}\sigma)^m}\sum_{i=1}^{k_j} e^{-\frac{1}{2\sigma^2}\sum_{l=1}^{m}(r_l - r_{i,l}^{(j)})^2} \quad (8)$$

Where

C_j : Center of the j'th cluster

K_j : Number of reference points in j'th cluster

$R_{i,j}^{(l)}$: RSS from l'th access point in i'th point of cluster j

3.1.4 Positioning algorithm

Step 1: Generate *N* kernel functions as mentioned in the previous section.

Step 2: Compute the normalization factor η for the measurement $o_{meas} = (\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_m)$

$$\eta = \frac{1+\sum_{j=1}^{N} k_j}{(n+1).\sum_{j=1}^{N}\frac{1}{k_j}\frac{1}{(\sqrt{2\pi}\sigma)^m}\sum_{i=1}^{k_j} e^{-\frac{1}{2\sigma^2}\sum_{l=1}^{m}(\hat{r}_l - r_{i,l}^{(j)})^2}} \quad (9)$$

Step 3: Find the expected value for x and y coordinates.

$$\begin{aligned} E\left[x \mid \hat{r}_1, \hat{r}_2, \ldots, \hat{r}_m\right] &= \eta.\sum_{i=1}^{N} x_i.\, P^{(i)}_{r_1,r_2,\ldots,r_m}(\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_m \mid_{x_i}) \\ E\left[y \mid \hat{r}_1, \hat{r}_2, \ldots, \hat{r}_m\right] &= \eta.\sum_{i=1}^{N} y_i.\, P^{(i)}_{r_1,r_2,\ldots,r_m}(\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_m \mid_{y_i}) \end{aligned} \quad (10)$$

3.1.5 Real-time performance evaluation of Ekahau

Like any other geolocation system Ekahau has some limitations in terms of number of training points and number of access points when it comes to real world. For the purpose of studying the accuracy of this engine we did some measurement in the first floor of Atwater Kent Laboratories in WPI. We started with placing seven access points in the floor which would be able to communicate with Ekahau.

These access points were initiated to work in different channels, the configuration which gives us the least interference between channels. Two of them were working on channel eleven and the next two were working on channel six and the others were

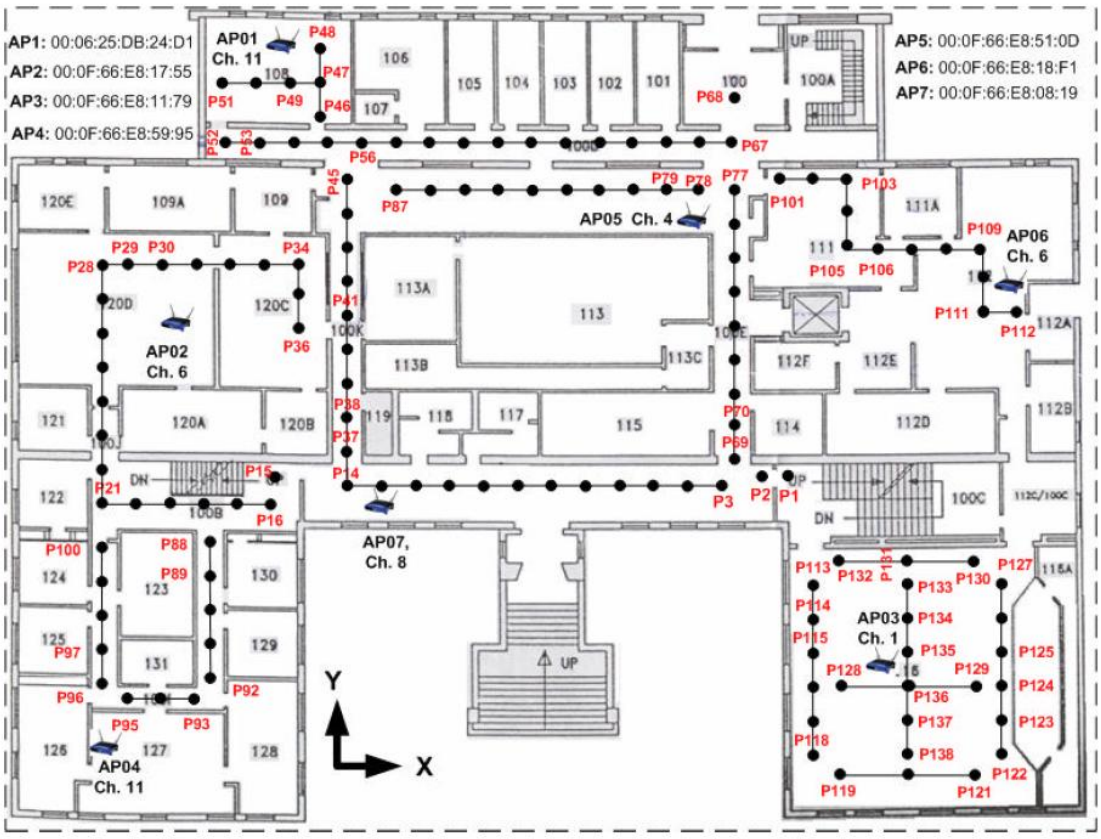

Figure 15: Configuration of access points and training points

working on channels one, four, and eight. The access points with the same working channel were placed as far as possible from each other. This configuration with the details of channels and their location is being shown in Figure 15.

Next step was to decide on the number of training points for our scenarios. We came up with two scenarios with the same configuration of access points but different number of training points. The first scenario included just four training points while the second one had thirteen training points. Figure 15 also illustrates the location of four training points corresponding to the first performance evaluation scenario.

For this case we tried to pick the four training points in a way that all of the estimating points could be categorized in one of these points neighborhood. Since each training point was at least close to one of the access points in the corresponding power profile the received power of the respective channel

Ekahau's estimation of the location was compared to the actual location of the mobile terminal which in our case was a laptop. The procedure was repeated for 138 points which would give us the best estimation of the performance evaluation of the positioning engine.

The error was measured in terms of horizontal error and vertical error which gives a degree of freedom in analyzing the error which comes next. The orientation of error has been considered in the procedure as well. We named the origin the lower left corner of the map, so if the estimated location is to the right of the actual location the error would be considered as a positive value and if it was to the left of actual points, error was considered as a negative value. The same procedure has been applied for Y-direction

which positive error meant that the estimated point is upper in map comparing to the actual location and negative if the estimated location is lower.

Table 1 gives us the statistics of error.

As it can be read from the table, X-error is more distributed, ranging from -31 meters to 30 meters. This can be justified by considering the location of training points. As you can see training points are close to each other in Y-axis and far from each other in Y-axis. In other words, since the width of the floorplan is less than the length of it, range of possible values for Y-error is less than X-error.

Since this experiment was done with just inefficient number of training points of four, the results are not impressive. The large amount of variance is telling us the error is very dispersed in X, Y directions. Although perfect positioning engines are those which are able to get trained with as few training points as possible, but apparently for this special engine four training points were not enough. Most of the applications can not take the risk of having 10 meters of error in estimating the location of object. For getting an estimate of how close the engine can estimate the location of object Table 2 will help us.

Table 1: Error statistics for sample campaign I

4TP	*MEAN*	*VARIANCE*	*MIN*	*MAX*
X-error	-1.64	88.08	-31.63	30.44
Y-error	4.04	40.1544	-17.56	24.88

Table 2: Error statistics for sample campaign I

4 TP	*MEAN*	*VARIANCE*
X-error	10.7306	88.08
Y-error	6.2516	40.1544

Although the table shows us that the average error in X direction is around 10 meters and in Y direction is about 6 meters, but if we consider the error as the distance of the estimated location to the actual location, i.e. $\sqrt{X_{err}^2 + Y_{err}^2}$ we can find that the error in distance is 16 meters which is definitely not suitable for many of the applications.

Figure 16 shows us the CDF of this campaign which is the distribution of the error. It can be seen that the graphs looks like normal distributed cumulative density functions centered on zero. In chapter 4 we will go through the details of the analysis of the graph errors.

The second measurement campaign consisted of 13 training points and the same configuration for the location of access points and estimating points. It could be predicted that the result the result of this campaign are more accurate are reflect less dispersion in spatial error. Table 3 gives us some statistics of this campaign.

Table 3: Error statistics for sample campaign II

13TP	*MEAN*	*VARIANCE*	*MIN*	*MAX*
X-error	1.6155	89.4620	-22	42
Y-error	1.7079	69.8067	-31.4	28.6

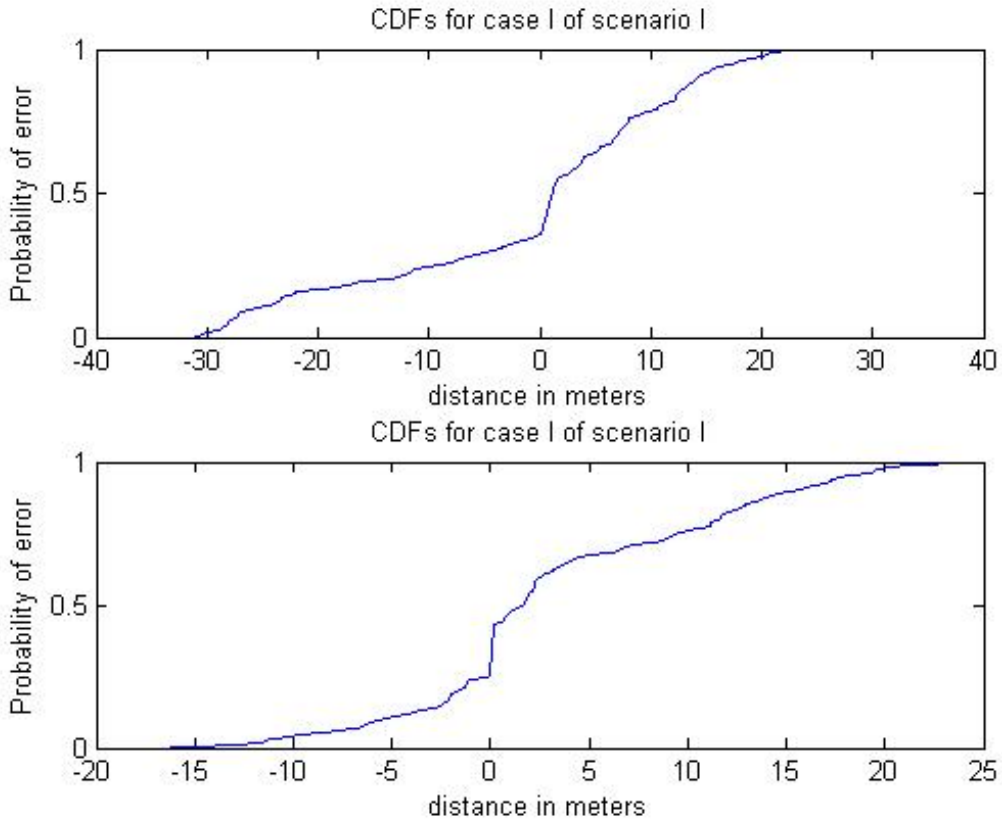

Figure 16: Ekahau performance in sample campaigns

Amazingly enough by increasing the number of training points by a factor of approximately 3, error in X and Y direction did not seem to change a lot. There were points in the floorplan which had large errors. The positioning engine's algorithm was unable to locate these points accurately. This should be noted that the purpose of this thesis is not to investigate the positioning engines and improve their capabilities but to develop a testbed which behaves the same way as if the positioning system was working in real environment. Table 4 below lets us take a closer look at the result of this campaign in terms of error.

Table 4: Error statistics of campaign II

13 TP	*MEAN*	*VARIANCE*
X-error	4.9888	89.4620
Y-error	4.6727	69.8067

The information in this table shows us that there was an error reduction in our system. One can easily notice that the X-error has been reduced to 5 meters from 10 meters previously, and Y-error has been reduced to 4.5 meters instead of 6.25 meters. The fact that both campaigns show the same behavior in terms of dynamic range of error and variance would let us conclude that there were few points in this campaign which had large errors. This result could be a glitch in their software as well as a glitch in their algorithm. Otherwise in the other points the positioning engine was working fine comparing with first campaign.

Figure 17 illustrates the cumulative distribution function of the error for the second campaign. It can be observed from this figure that error has been reduced in its

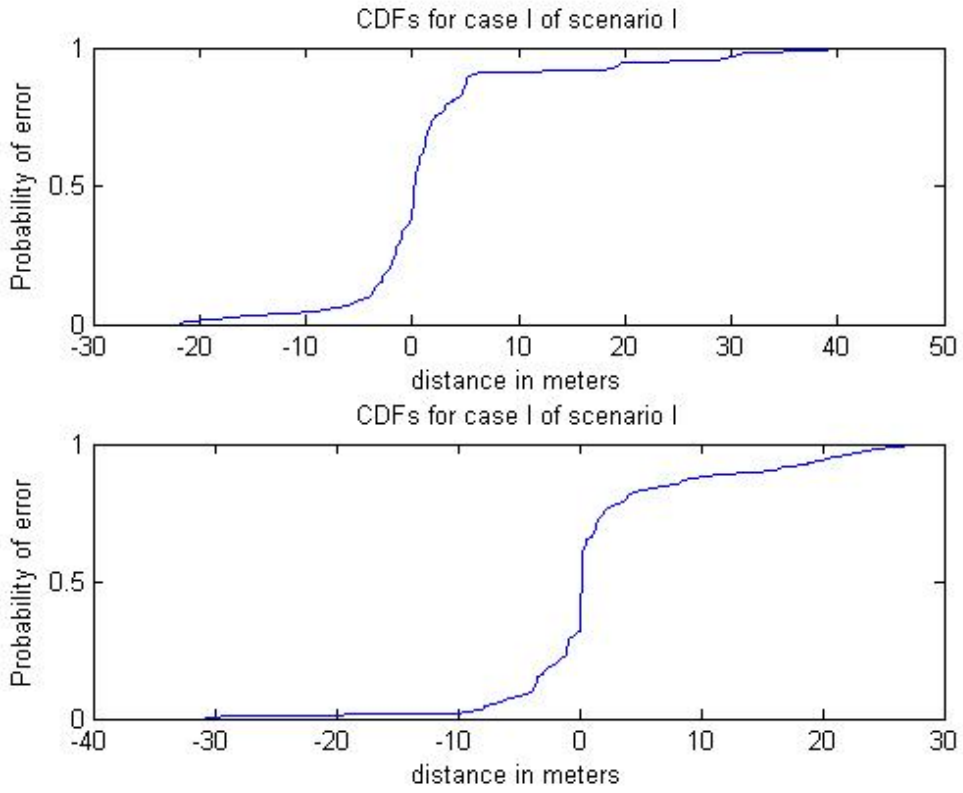

Figure 17: Ekahau Performance, second campaign

average for both X and Y directions but the dispersion has been remained the same for both X and Y directions. Since the graphs are steeper around zero one can conclude that the average of error has been decreased. The compression of the graph around zero actually yield the fact that the positioning engine was working better for the second campaign making smaller errors in both X and Y directions. More detailed discussion is left for chapter 4 in which we discuss the result of the testbed and the error analysis.

3.2 PROPSim channel simulator

The core for Real-Time channel simulation is PROPSim. PROPSim C8 wideband channel simulator simulates the real-time radio channel between a transmitter and

receiver. It is a very versatile and powerful tool for the development of wireless systems. Basically the channel simulator works based on the tapped-delay line method where it can generate different models and channels between the input and output of the each PROPSim's channel. The user's RF input signal is downconverted into analog complex baseband signals. These signals are filtered then converted to digital signals using A/Ds. The multipath fading and simulation and summing of taps is carried out in digital parts utilizing DSP technology. After fading, the original signal is D/A converted and upconverted back to the original RF frequency.

Each radio channel is considered as a combination of different paths arriving from transmitter into receiver with different amplitudes, different delays, and different phases. PROPSim also uses this approach to simulate the radio channel between its input and output which are necessarily the transmitter and receiver of the user's system. The properties of each tap are definable. Delay, mean amplitude in dB, and fading are the parameters that user can define for each tap which represent a path between transmitter and receiver. Delay of each tap can be set as static (fixed), random hopping, sinusoidal, or linear. Amplitude of each tap is fixed. Fading has the most diverse set of options. One can set the fading to classical (Jakes model), constant, flat, Nakagami, pure Doppler, Rice, lognormal, Suzuki, Gaussian, or even custom fading model. Defining no fading for taps of a channel will make that channel to have static delay which means delay is the same all the time. Figure 18 shows us an example of definition of such a channel.

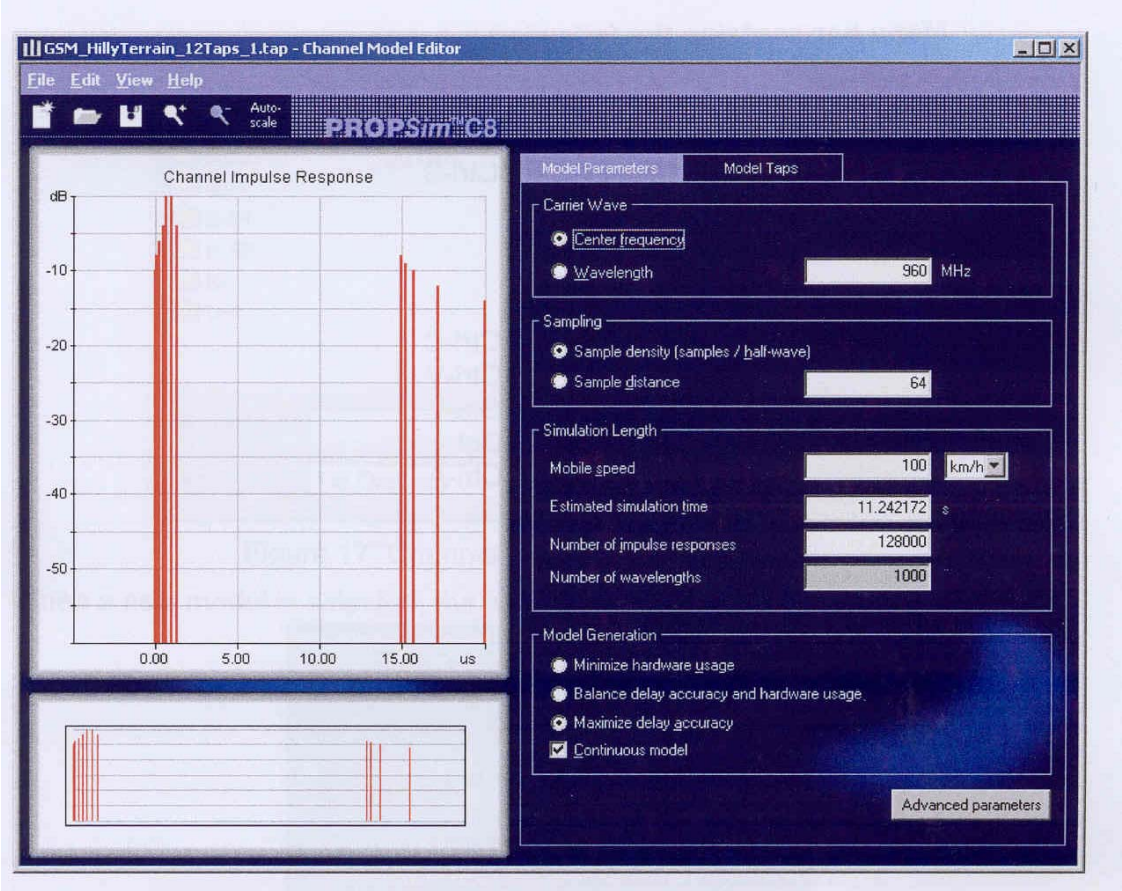

Figure 18: A sample channel model generated with PROPSim

Besides properties of each tap the whole model has some properties as well. Amongst them one can highlight center frequency which is the frequency of the input signal and wavelength, sampling density, sample distance, mobile speed, estimated simulation time, number of impulse responses, number of wavelengths, design accuracy option, number of channels, distribution seed (when the same model is regenerated with the same seed, the result is exactly the same as the original. If the seed is changed, new channels are not correlating with the original ones). By ignoring the effect of fading in PROPSim taps actually we have ignored the effect of movement in positioning. In dynamic methods of positioning this causes problems for tracking the desired object since the effect of movement of the object and the effect of moving objects around both have

been neglected. But in static methods like fingerprinting one can use this feature to just simulate the channel between transmitter and receiver. The channel would then become an element of the database which the desired metric can be extracted from. Other parameters of a tap are also definable which make PROPSim suitable for DOA or AOA methods.

There are eight independent channels in PROPSim and each channel has its own input and output. Each channel's bandwidth is 70 MHz so it is quite enough for RSS or AOA measurement methods but for TOA method there will be shortage in our setup system.

From user's point of view, there are predefined channels available in PROPSim's software such as JTC and GSM models but one can define his own models based on Ray Tracing method or even new methods which are going to be developed. Since in this thesis the main problem was around indoor positioning systems, WLAN frequency was then used for center frequency of each channel. These frequencies are in 2.4-2.48 GHz band. Each channel has to have its own specific center frequency, but due to limitations I used the same center frequency for all of the channels in my experiment which degrades the outer channels a little bit. So in order to being more accurate about the measurement one can set the local frequency of each channel which is an input to system to the desired center frequency and set the parameter in simulation to that specific frequency. Whereas Ekahau has this property which tells the user about the channel being used for calibration, one can easily find the center frequency of the channel which should be set for PROPSim's respective channel. It is very straightforward to connect the signal which comes an access point to an input of PROPSim. Most of the new access points have this

ability to shut down the wireless antenna and send the signal out through a connector. But what has been done in this project was that we chose access points as fixed stations which had the property that instead of antenna PCMCIA cards function as transmitters and receivers. Taking advantage of this idea the problem of connection then solved by connecting PCMCIA cards to PROPSim's BNC input connectors.

Each PROPSim channel has two ports, one input and one output. The RF local frequency block is necessary for every channel to work properly. The input to each channel is being transferred into baseband with the help of this local frequency generator. Then it is been transformed into digital and then it goes into the channel which is an FIR filter. So the core of PROPSim is digital baseband filtering and simulating.

3.3 Place Tool Ray Tracing Software

PlaceTool™ is ray tracing software which is used to find the entire set of paths between transmitter and receiver. According to the floorplan of the building and reflection and transmission coefficient of each wall PlaceTool decides on the number of the arrived paths, their respective amplitude, and their delay. One can also include details of the floorplan of the test environment, like tables, metal cabinets, file cabinets and etc. We used PlaceTool software developed at CWINS to simulate the channels as the inputs of PROPSim.

Defining the floorplan as a combination of lines with different coefficients is the first step to find the multipath between transmitter and receiver. Placing the transmitter and receiver in the map afterward will be the next step to complete the procedure. Finally the software can calculate the multipath, their respective amplitude and delay. Importing these files to MATLAB will let us change them in a readable format by PROPSim

software. Since ray tracing is a static environment, *i.e.* does not give you fading properties of the path, I set the fading properties of channel in MATLAB process. The output files should contain the amplitude, delay and type of delay, and fading which all has been done in MATLAB.

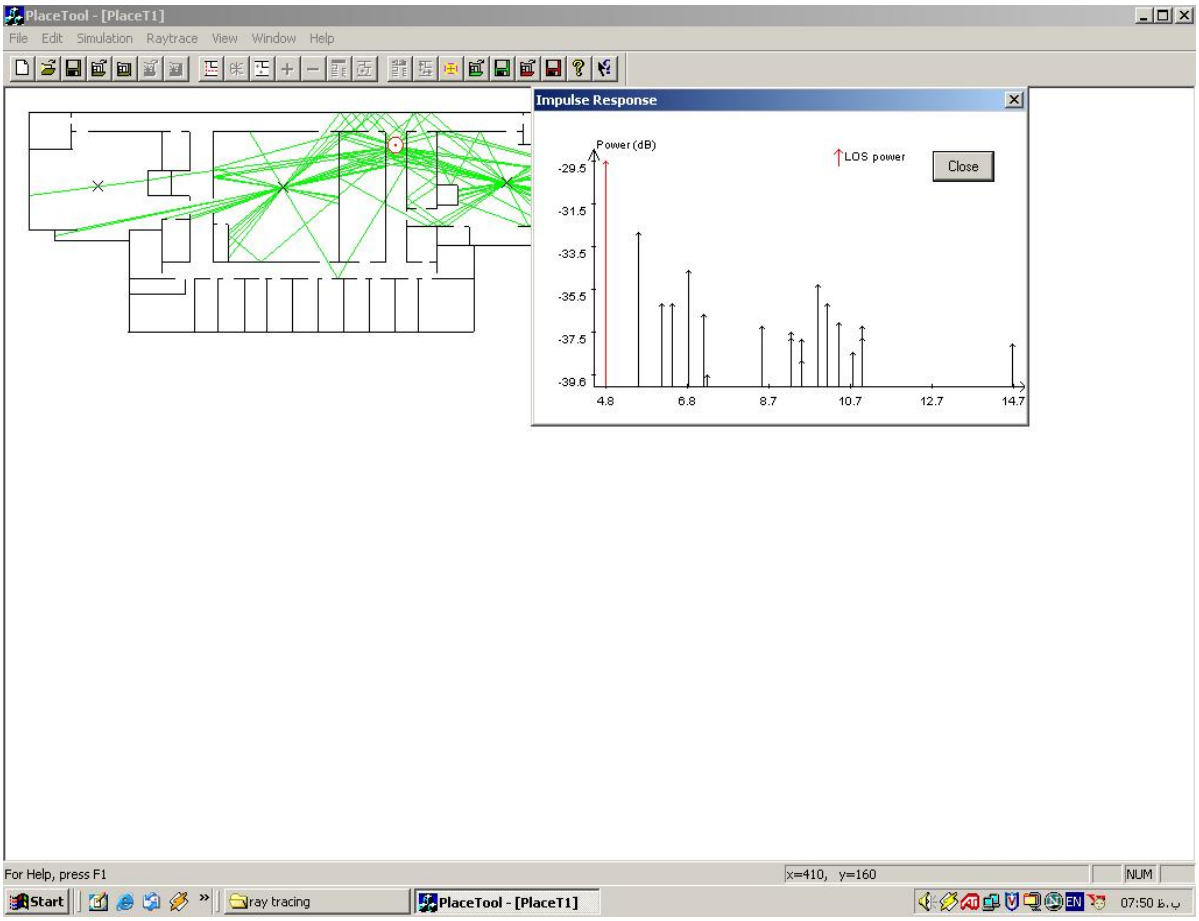

Figure 19: PlaceTool ray tracing software

4 Performance analysis

In this chapter we go into the details of the scenarios that we defined for testbed and the results. We try to compare different scenarios in terms of the difference between number of access points and number of training point. Another experiment that is conducted is related to the configuration of access point and effect of that parameter on the accuracy of the system. Statistics of the error is represented for each scenario separately which let us compare them later on. All the scenarios were done in the third floor of Atwater Kent Laboratories in WPI.

4.1 Introduction

The purpose of this section is to discuss the results of the simulation for performance evaluation and compare the results with real world measurement. The first experiment consisted of three different scenarios regarding the number of access points used in each scenario. Each scenario was divided into three cases in which the number of training points used Ekahau is changed. In all of the cases the first step is to define the location of access point(s) in PlaceTool ray tracing software. Defining the location of training points in PlaceTool and running the software gives us the entire channel response between each training point and access point(s). By importing these channel responses into Matlab® we can process them and turn them into PROPSim readable files. Next step is to use each of the processed files to simulate the channel between receiver and transmitter in PROPSim. After channel simulation Ekahau can be trained on the desired point. Repeating these steps for all of the training points will give us an option to train Ekahau with as many training points as we want. The interface between access point(s)

and PROPSim is MC-card-BNC cable and same cable is used for the interface between PROPSim and the PCMCIA card in Ekahau.

4.2 Empirical results versus simulation

The first case we try to investigate is case I from scenario I in which we had only one access point and just four points as Ekahau's training points. The locations of these training points have been shown again in Figure 20. After calibration four numbers of independent points were selected to see how accurate Ekahau determines the location of the mobile terminal. The error was then calculated by the means of displacement of the estimated location from actual location. Obviously in this case the result was poor due to the fact that just 1 access points was considered and the number of training points was also few. The result for this case has been given in Table 5.

Table 5: Error statistics of case I from scenario I

SCENARIO	*MEAN(M)*	*VARIANCE(M^2)*	*RMSE(M)*	*P(90%)(M)*
Case I	13.1903	31.8125	14.3288	26

Figure 20: Location of access points and training points for the case I from scenario I

Where RMSE stands for root mean squared error and P(90%) is the distance the with probability of 90 % error is less than that. So as it is obvious from table 1 that in 90% of places, location estimation errors were less than 26 meters.

The concept of this graph and its parameters and what they discuss will be discussed later on in this chapter. Briefly from this curve an approximation can be made to our distance error and one can model the error with respect to the parameters that he

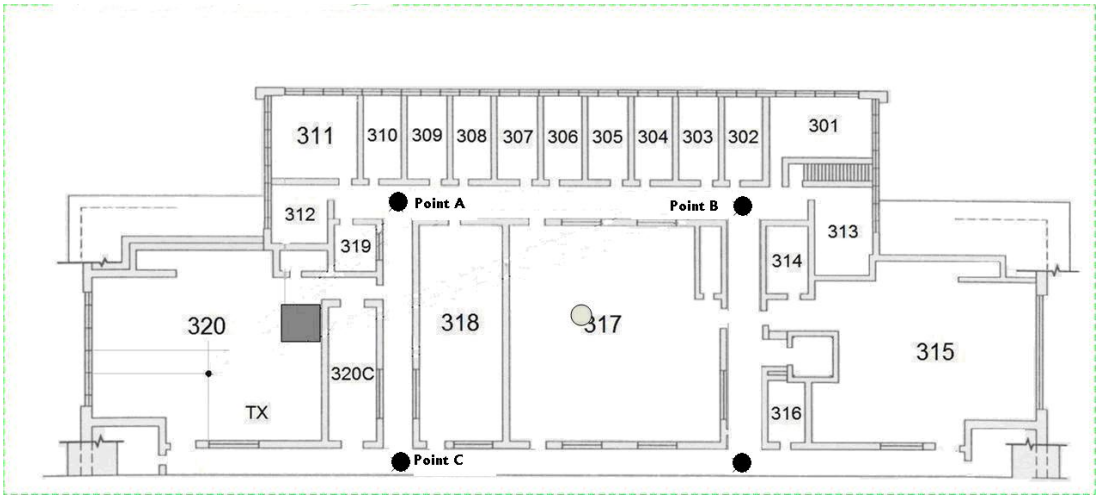

Figure 21: About points A, B, and C which have the same power profile but large displacement

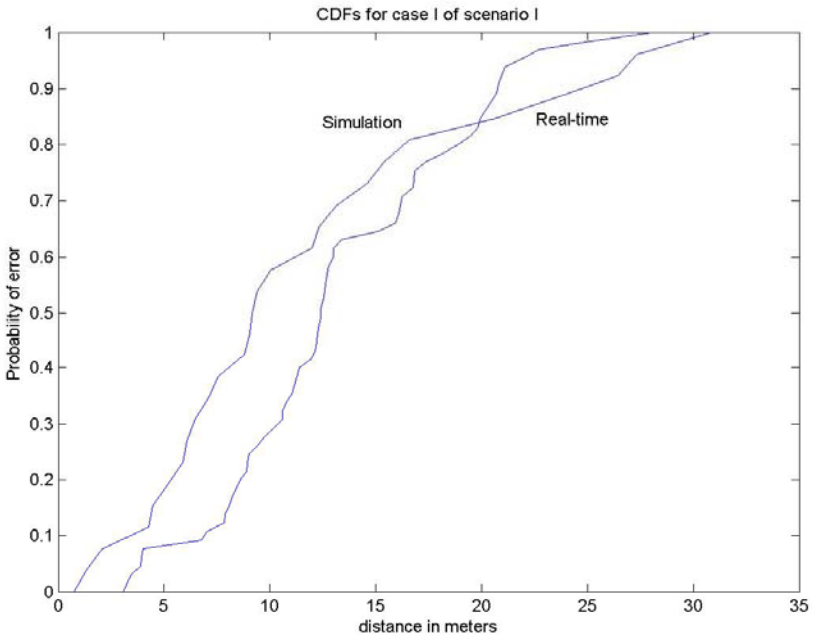

Figure 22: Comparison between simulation and real-time result for case I from scenario I

gets from the curve.

As it is obvious from table in this case due to lack of RSS information for Ekahau, it could not estimate the exact location of the mobile terminal. For illustrating that how these large errors could happen, Figure 21 will help us. As you can see in this case the received power from the same access point in points A and B were equal. So in estimation procedure one of the points can easily be picked as the desired point falsely.

After simulation I created the scenario in real world by putting an access point exactly on the same spot as we simulated the case with. Then I calibrated the Ekahau

Table 6: Simulated Error statistics for case I from scenario I

SCENARIO I	*MEAN(M)*	*VARIANCE(M^2)*	*RMSE(M)*	*P(90%)(M)*
Case I	11.4938	66.4658	14.0039	24.5

software with the same four points that I calibrated the software with in simulation. The statistics of this case is shown in Table 6

As you can see the results of the real world calibration and simulation calibration match to each other in mean, RMSE and P(90%). The only difference is in variance that since the case had just one access point and only four training points, it could be predicted that the location estimation errors would be large enough to make such a high inaccuracy in location estimation.

Figure 22 above illustrates the comparison between CDFs of the two cases and one can observe the fact that simulation for location estimation follows the exact pattern that real world measurement performs.

The second case for comparison between real-time measurement and simulation is

Table 7: Error statistics for case III from scenario III

SCENARIO III	*MEAN(M)*	*VARIANCE(M^2)*	*RMSE(M)*	*P(90%)(M)*
Case III	1.2301	4.196	2.3761	5.6

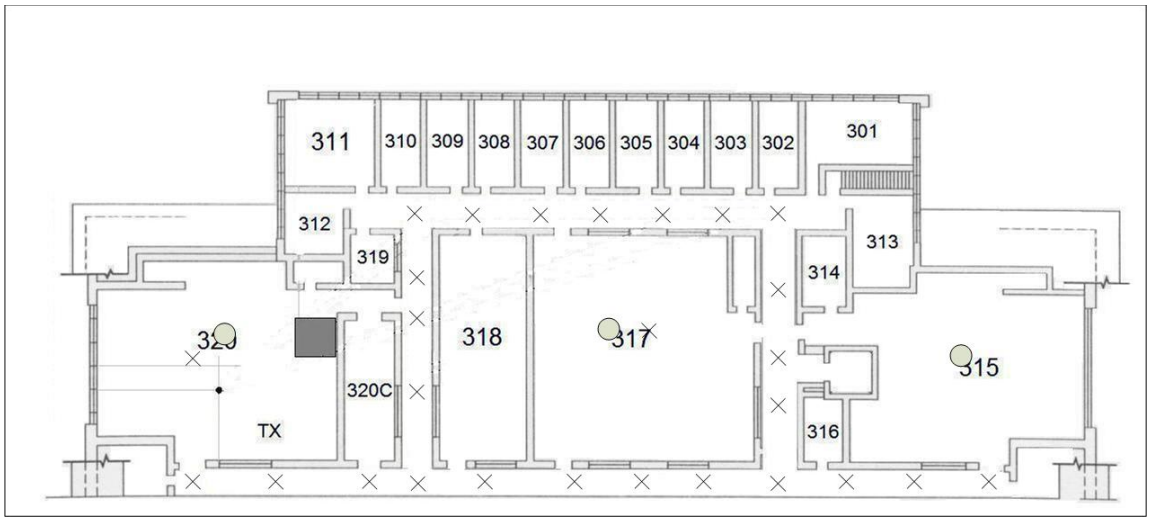

Figure 23: Location of access points and training points for case III from scenario III

the case where we had three access points, scenario III, and we had twenty seven points for training of Ekahau. Figure 23 illustrates the location of access points and training points for this case. Since this case had the most number of access points and training points, we could expect that the error would be the least between all of the other cases. Table 8 provides the statistical information about this case. As we can see great enhancement in terms of location estimation has been achieved by both increasing the number of access points and number of training points. Details of improvements are going to be discussed in the next two sections of this chapter.

Further the real world measurement with the same set of access points and training points will give us the same result with minor differences in statistics. Table 8 provides the information for real world measurement.

In Figure 24 we can compare the CDFs of the real world measurement and simulation of location estimation.

The justification given above about the simplest case and the most comprehensive case in our simulation profile gives insight into how we can relate these two sets of location estimation. Although there were minor differences in average and variance of error, one can claim that simulation with a perfect approximation will result in the same location estimation that real world calibration does.

Table 8: Simulated Error statistics for case III from scenario III

SCENARIO III	*MEAN(M)*	*VARIANCE(M^2)*	*RMSE(M)*	*P(90%)(M)*
Case III	1.6729	2.313	2.2453	3.5

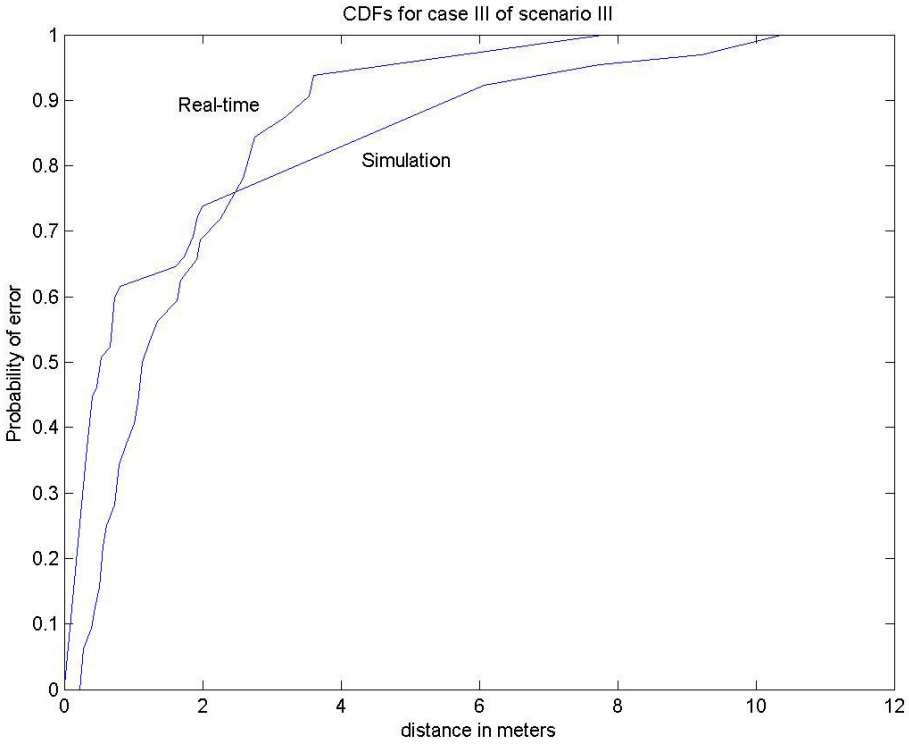

Figure 24: CDF of error for case III from scenario III

The purpose of designing the testbed was to find a way to estimate the location without having to carry out massive measurement campaign and this justification will allow us to use this testbed instead of real calibration in a specified site.

Analyzing the effects of various environmental conditions and effects of set up requirements for location estimation is also made easier by using this testbed which we discuss in the next few subsections.

4.3 Effect of number of training points

In this section we discuss the effect of the number of training points used for Ekahau. In every scenario we started with four number of training points and then we

Table 9: Error statistics for different cases of scenario

SCENARIO I	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Case I	13.1903	31.8125	14.3288	26
Case II	13.7574	24.789	16.1235	25
Case III	9.5309	21.9344	13.3944	21

examined the case with ten number of training points and finally we used twenty seven number of training points to train Ekahau.

First we discuss scenario I in which one access point was used for calibration. Figure 20 illustrates the location of access points and training points for case I, i.e. the number of training points were four.

Figure 26 shows the locations of the only access point involved and the set of training points for case II with 10 as a number of training points. And finally one can find the locations of access point and training points by noting the fact that location of the access point was the same for all the cases of this scenario and locations of training points were similar to the location of training points in Figure 26.

Table 9 provides the statistics of these three cases while Figure 25 illustrates the CDFs of error for scenario I.

So as it can be seen from Figure 25 that although by increasing the number of training points from 4 to 27 we enhance the location estimation procedure but the fact that database was built with data from just one access point has affected the improvement and it can be seen that the error in location estimation has been just reduced to 9 meters instead of 13 meters. This illustrates the importance of the number of access points used in the training sequence. The greater the number of access points we use, the more accurately the software can predict the location.

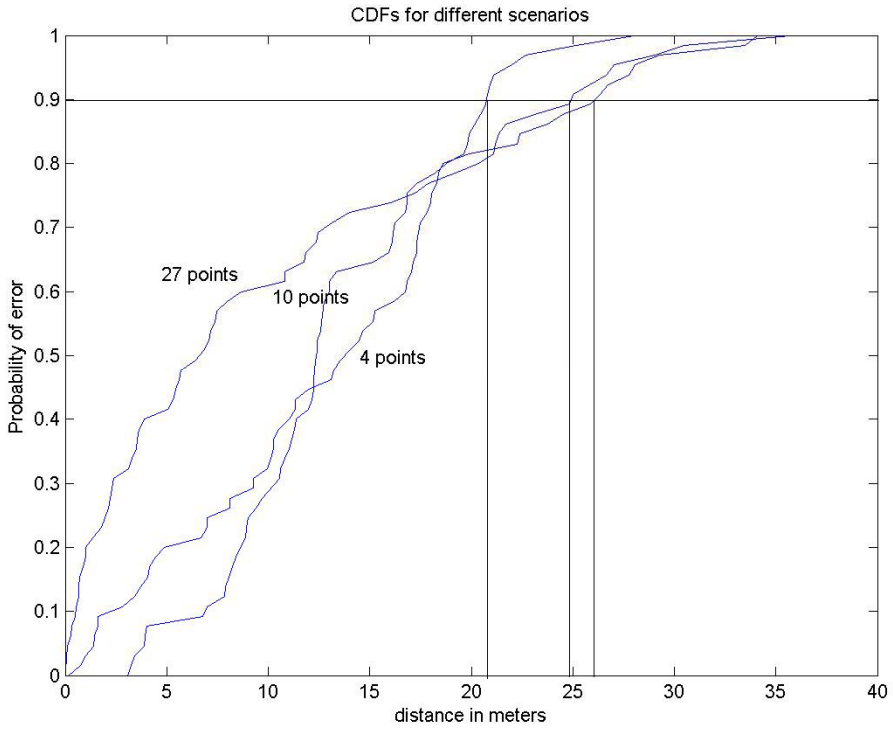

Figure 25: Comparison of three cases from scenario I

Figure 26: Location of access points and training points for case II of scenario II

Another good diagram which illustrates the comparison of the three different cases is Figure 27 in which all of the averages and standard deviations have been shown together.

In Figure 27 it can be seen that though the average of error decreases from the first case till last case but standard deviation increases. An explanation for this

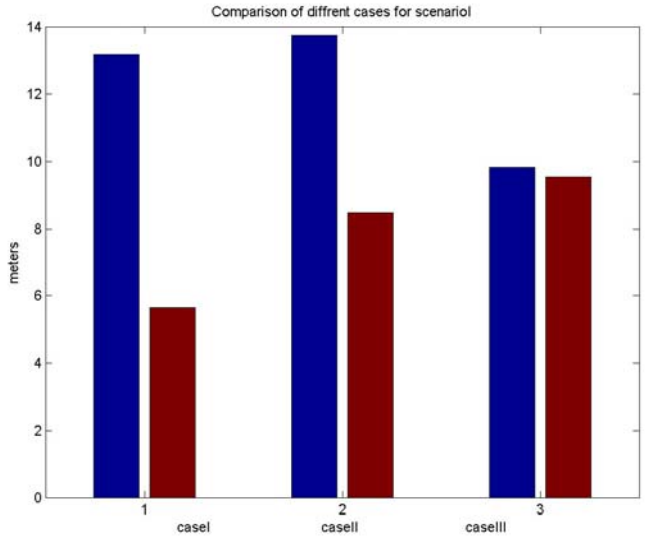

Figure 27: Comparison of mean and variance of error

phenomenon is that as we increase the number of training points, number of points with similar power profile increases and so does error. In addition standard deviation of error increases as we increase the number of training points. This is due to the fact that with 27 training points either we are close to the real location of mobile terminal or we have a large error.

As we saw in earlier chapters one access point is not enough not only for TDOA and AOA metric based positioning systems but also for fingerprinting-based methods. This is the main reason for having such large distance errors in this scenario. The Ekahau software calibrates itself with just one received power and there are lots of points in the same floor that might have the exact power profile, i.e. in outdoor areas with one access point in the origin points which are located on a circle with a constant radio will have the same power profile. In indoor areas there are several points with the same power profile

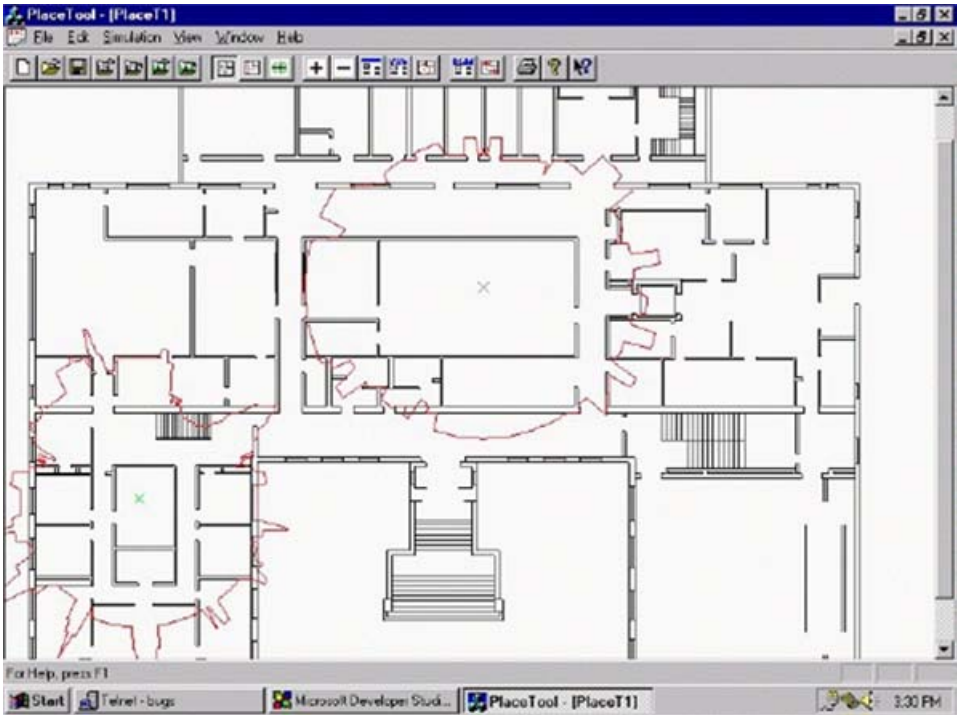

Figure 28: Two set of different perimeters that have the same power profile

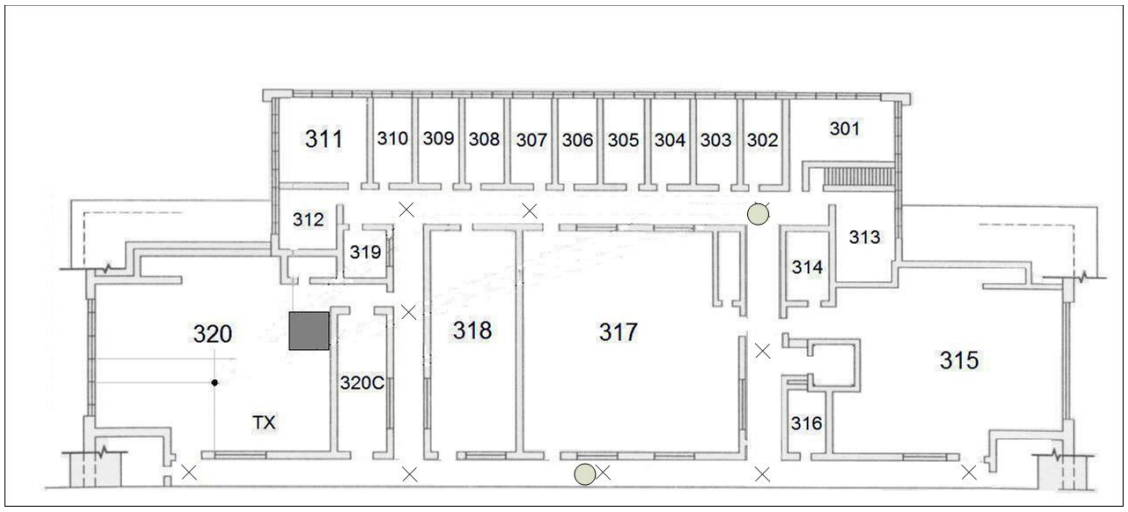

Figure 29: location of access points and training points for case II of scenario II

as it can be seen from Figure 28.

For the second scenario we placed two access points both in hallways. Similar to scenario 1 I separated the cases based on number of access points. In case I we put 4 as a number of training points and in case II 10 and in case III 27, respectively. Since we had two access points this time, Ekahau built its database more accurately. Associated with each training points Ekahau was able to save two values for received power at a time.

Figure 29 illustrates the location of access points and training points for case II of scenario II. It is worth mentioning that the set of locations of training points for each case was exactly similar to scenario I, so we were able to compare scenarios with each other.

Table 10 gives us the statistics of scenario II for different set of training points. As one can observe from table by increasing the number of training points, error decreases in terms of mean.

One can observe that by increasing the number of training points from 4 to 27 mean of error has almost got halved, but still the results are not impressive. Indoor positioning system needs to be more accurate due to the customer requirements. Even two access points are not enough to build a reliable database.

Figure 30 shows the CDFs for different cases in scenario II. As you can see from this figure at the best 10 % of the time location estimation can be done with errors less than1 meter, but in the other hand again in the best case 90 % of the time error is just less than 8 meters which means system is still not reliable for location estimation. What people are looking for is a more stable positioning system.

Table 10: Error Statistics for different cases of scenario II

SCENARIO II	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Case I	6.7953	15.2099	7.8202	11.9
Case II	4.8541	18.3895	6.4555	10
Case II	3.1392	12.9932	4.7593	8.4

By examining Figure 31 in which all the averages and standard deviation are being shown one can easily observe that adding more access points will enhance the error significantly.

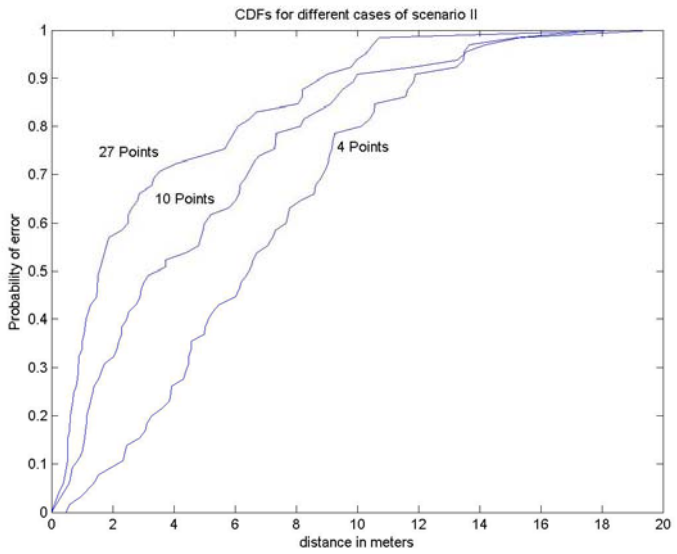

Figure 30: CDF comparison for different number of training points from scenario II

It can be seen from the figure that the average of error decreases as we have expected but standard deviation remains almost the same for all of the cases.

For scenario III we placed three access points in third floor of Atwater Kent

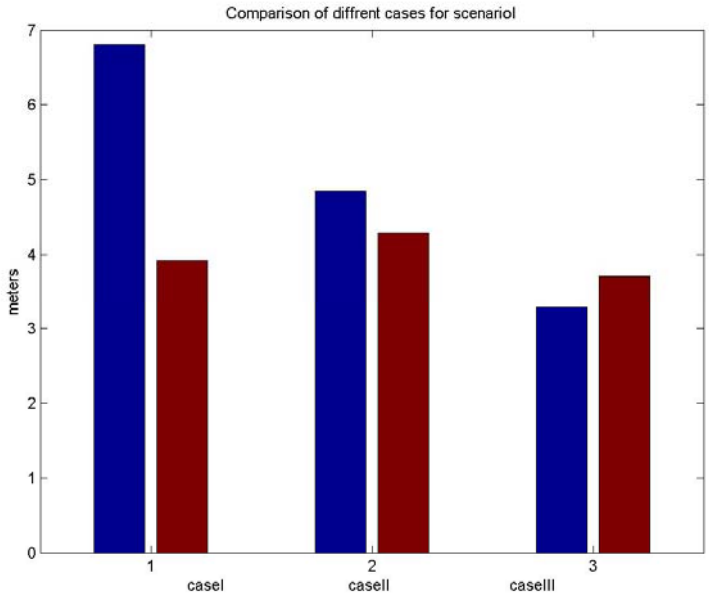

Figure 31: Comparison of mean and variance of error for scenario II

Laboratories and we simulated the scenario with PlaceTool. Again we divided the scenario into three different cases according to the number of training points. The locations of training points were chosen exactly as what we had for previous scenarios so we could compare them in terms of number of access points as well.

Figure 32 shows case III of this scenario. The locations of access points in this scenario were chosen in a way that user randomly might choose them to cover the entire floorplan. Statistics of error can be found in Table 11. Figure 33 also will help us understanding the difference between different cases in this scenario by comparing their CDFs.

Now with three access points we can observe that with enough training points we can almost reach the level of acceptable accuracy that we wanted. Specifically for case III in which we had three access points and 27 training points, the Ekahau positioning engine was able to locate the mobile terminal with the accuracy of 1.5 meters. Although it is said in their documentation that using more than three access points would able software to work more accurately, but one can observe that even with three access points 1.5 meter accuracy can be achieved.

Table 11: Error Statistics for different cases of scenario III

SCENARIO III	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Case I	6.7283	16.189	7.824	11.8
Case II	3.3729	10.053	4.6127	8.6
Case II	1.2301	4.196	2.3761	5.6

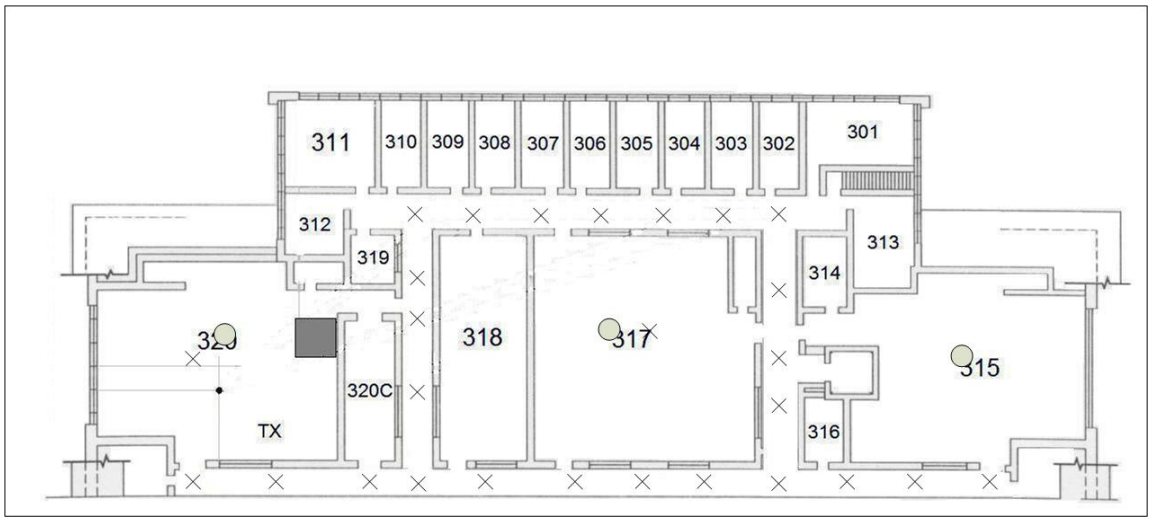

Figure 32: Location of the access points and training points for case III from scenario III

Figure 34 shows us the comparison of different cases of this scenario. It can be observed that both average and its standard deviation of the error decrease by increasing the number of training points as was expected. Another observation is that as the number of access points and number of training points get closer to the requirements of the

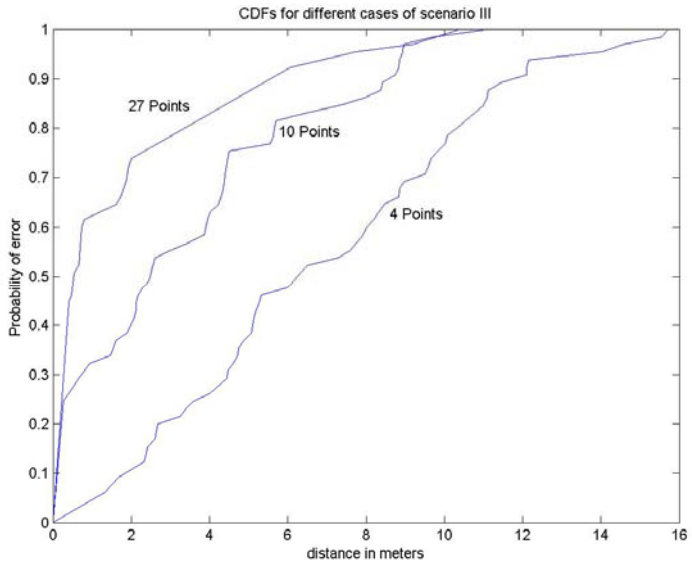

Figure 33: Comparison of CDFs of different cases of scenario III

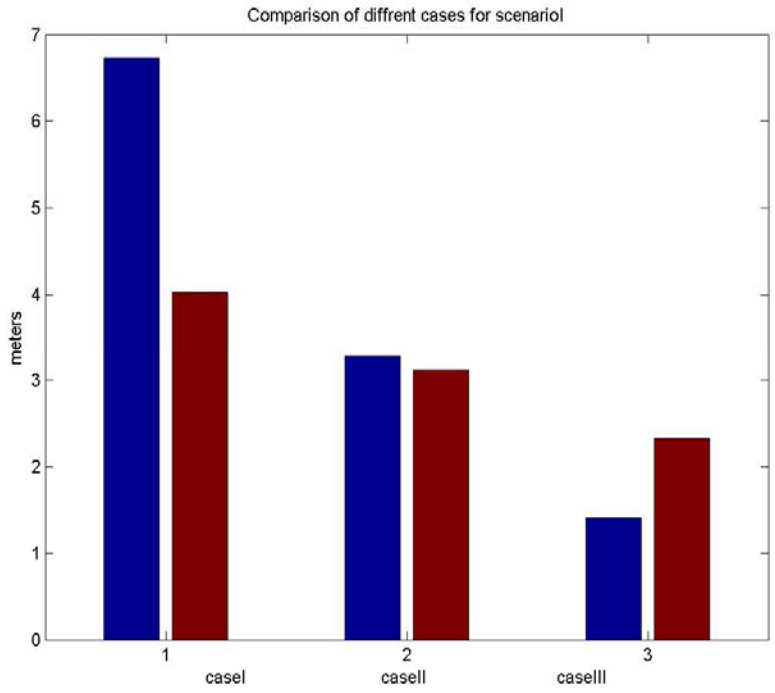

Figure 34: Comparison of statistics for different cases of scenario III

positioning system, the system tends to show off its normal behavior.

4.4 Effect of number of access points

It is also interesting to review the results from the aspect of number of access points. In this section we present the effect of number of access points for a fixed number of training points.

First we start with the case of having four training points. By increasing the number of access points we can see that results are getting better. Table 12 summarizes the case.

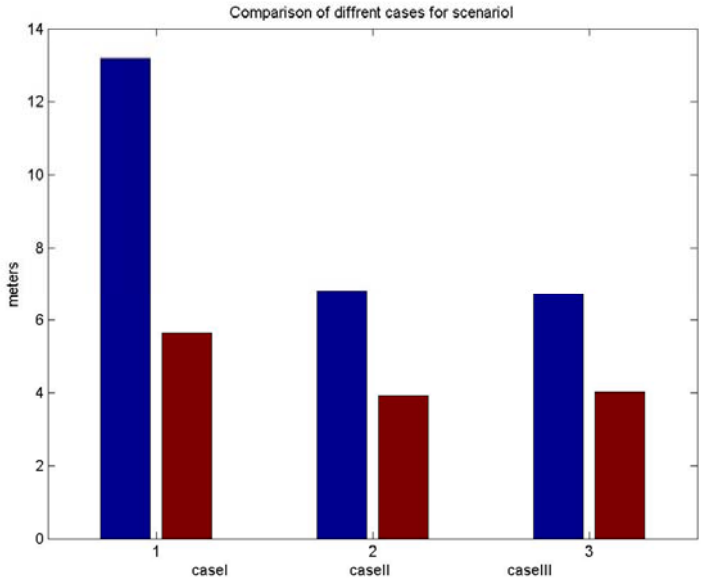

Figure 35: Comparison of statistics for four training points

Figure 35 illustrates the average and standard deviation of error for different scenarios which helps us visualizing the difference between three scenarios in term of average and mean when the number of access points is being held unvarying. With the same set of training points which in this case is four. It is interesting to note that by increasing the number of access point from one to three both average and standard deviation decreases which was expected. But it is also worth observing that between

Table 12: Error Statistics of the same case from different scenarios

CASE I	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Scenario I	13.1903	31.8125	14.3288	26
Scenario II	6.7953	15.2099	7.8202	11.9
Scenario III	6.7283	16.189	7.824	11.8

scenario II which had two access points and scenario III which had three access points there is almost no difference in terms of average and standard deviation of error. An explanation of this could be due to the lack of information for building database for positioning software. We will see later on that by having more number of training points this phenomenon vanishes.

Here it can be seen even with four training points it is still possible to locate the mobile terminal within a distance of 7 meters. Although it is totally unimpressive but still the effect of adding more access points is evident. Results are getting more interesting while we move to the cases with more number of training points.

For instance for a case with 10 as the number of access points we have moved from 13 meters to 3 meters in terms of mean of error. Table 13 gives us the statistics of this case.

And the statistical graph is shown in Figure 36 below. It can be seen that by increasing the number of access points in this case which we had just 10 as the number of training points both average and standard deviation of error decrease.

Table 13: Statistics of 10 training points

CASE II	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Scenario I	13.7574	24.789	16.1235	25
Scenario II	4.8541	18.3895	6.4555	10
Scenario III	3.3729	10.053	4.6127	8.6

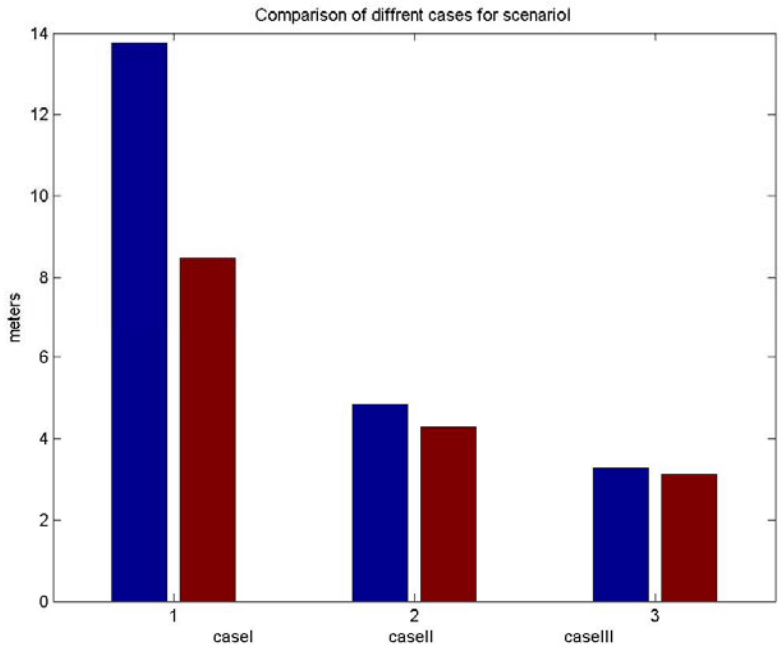

Figure 36: Statistics of 10 training points

The last case in this category consisted of 27 training points and number of access points varying from one to three. Results are shown in Table 14. Significant enhancement can be observed from the table. Figure 37 helps us to compare the cases better. It can be seen that by increasing the number of access points from one to three, error almost gets tenth. Standard deviation also decreases significantly in this case.

Table 14: Statistics of 27 Training Points

CASE III	*MEAN(M)*	*VARIANCE(M)*	*RMSE(M)*	*P(90%)(M)*
Scenario I	9.5309	21.9344	13.3944	21
Scenario II	3.1392	12.9932	4.7593	8.4
Scenario III	1.2301	4.196	2.3761	5.6

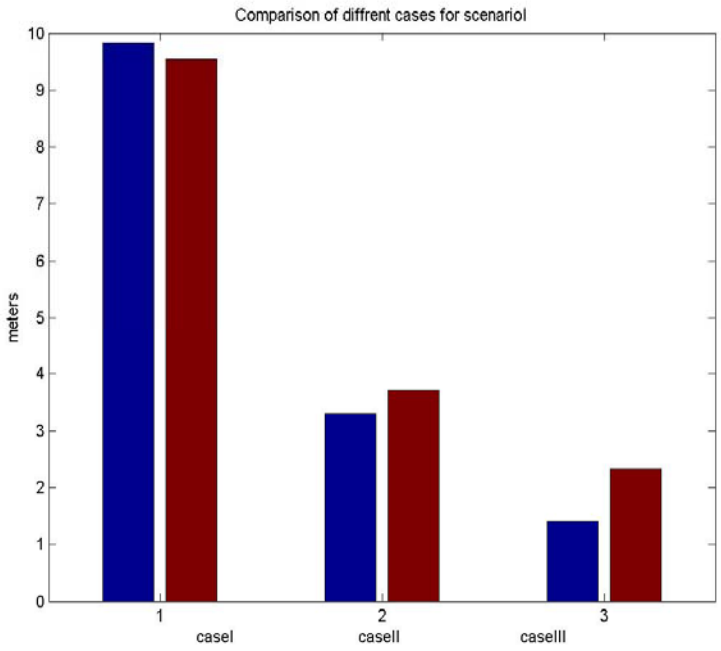

Figure 37: Comparison of statistics of 27 training points

4.5 Effect of the location of the access points

In this section we represent the result of some experiments which were done to find the effect of the location of the access points on the accuracy of the geolocation process. We created three scenarios for the purpose of studying this parameter. One of the patterns was exactly to scenario III from previous section, while the other two were different. Figure 38 represents the configuration of the access points for different patterns.

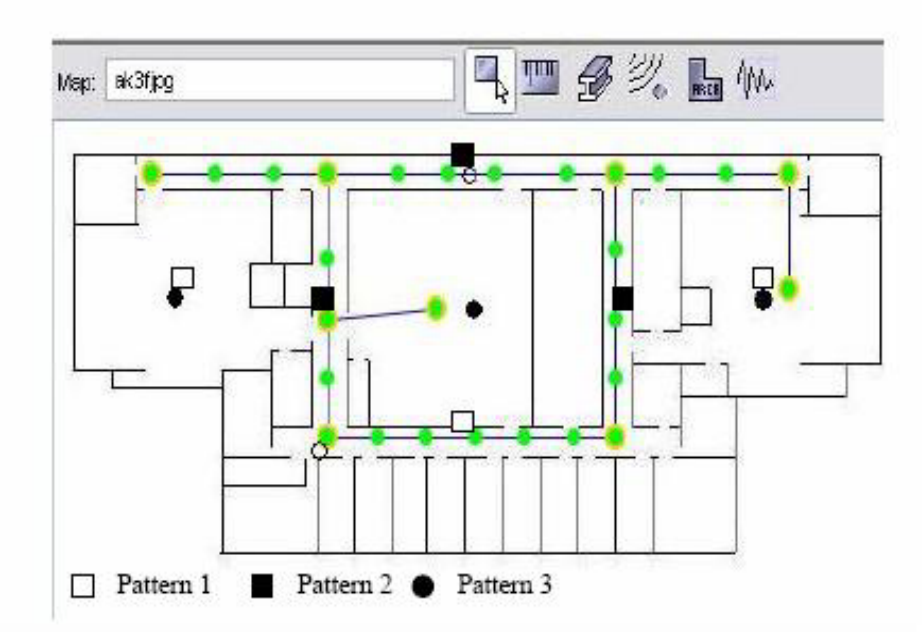

Figure 38: Various patterns for infrastructure of the WLAN network

It can be seen that in pattern two access points are close to each other comparing with the other configurations. It is expected that for this configuration error rate increases comparing to the other configurations. First pattern has an advantage over third pattern in a sense of having one access point in a hallway. So in this case all of the training points in the hallway have LOS to the access points which enable them to receive a large power from access point. This it self will lead to a more accurate database theoretically.

Table 15: Statistics for different patterns of infrastructure

SCENARIO\STATISTICS	*MEAN*	*STANDARD DEVIATION*
Pattern 1	4.46	4.76
Pattern 2	8.25	8.7
Pattern 3	3.41	4.08

Table 15 gives us the statistics of these scenarios. The disadvantage of this pattern is that the training points that are located in the other hallway are far from this access points. So in the process of building database, these points lack from getting power from all of the access points resulting in a poor database. We will investigate the statistics of these scenarios in details. It should be mentioned that in all of the configurations same set of training points has been used to make the scenarios comparable to each other. The grid network for training points consists of 25 points while error has been measured over 100 points distributed all over the place in floorplan.

As it can be seen from the table second pattern which had the worst configuration between all of the configurations resulted in 8.25 meters of error in mean. Here it can be

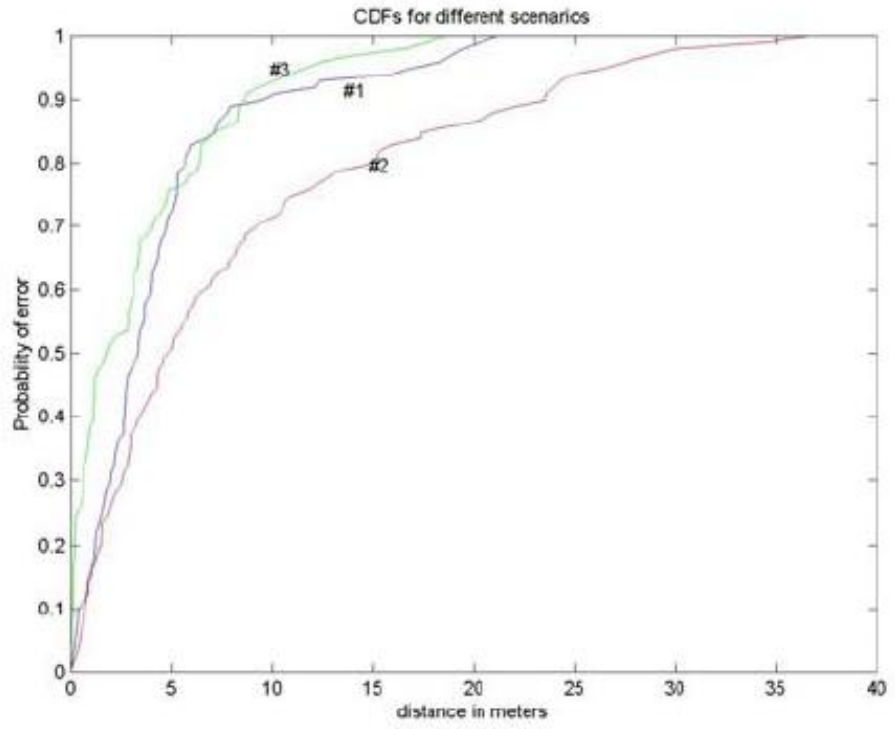

Figure 39: CDF graphs of error for different scenarios on the effect of configuration

seen that although the scenario happened in one story of the Atwater Kent Laboratories and three access points were used, result is very poor. The result of this experiment mainly suggests us that the configuration of the access points for better accuracy in geolocation may not be necessarily the same as the configuration of the access points that delivers the best data-rate for telecommunication purposes.

Figure 39 illustrates the CDF of the error for different scenarios of this experiment. It can be observed that since second and third patterns are close to each other in configuration, errors follow the same shape for both of them. The only difference is that in small distances the probability of error is slightly less for pattern 3.

5 Conclusion and Future Work

In this thesis development of a unique testbed for real-time performance evaluation of positioning systems was described. A sample positioning engine was analyzed using this testbed and location-estimation error was analyzed, has been described in chapter 4. After evaluating the performance of the sample positioning engine the following conclusions can be drawn.

Due to the harsh indoor environment, doing repeatable comparative performance evaluation measurements is almost impossible. Another fact which affects the performance evaluation of indoor positioning systems is that defining same scenarios for different systems is a difficult task to do. Therefore there is a need for unified testbed which would be able to define a same scenario for every positioning system, so one would be able to compare different systems.

The testbed is composed of three main blocks, Ekahau positioning software, PROPSim real-time channel simulator, and PlaceTool Ray Tracing software. Procedure starts with simulating the channel between the access point and training point in PlaceTool. Then the results are manipulated with MATLAB® to export to PROPSim. PROPSim simulates the channel with tapped-delay-line hardware for the input signal from access point. The output of PROPSim directly goes to the PCMCIA card as an input of Ekahau. Ekahau reads the power from PCMCIA card and calibrates its database. Finally arrival of new power profile would let Ekahau search for the best estimated location.

According to the experiments which are described in chapter 4 the result of the real-time performance evaluation and simulated performance evaluation match to a great

extent. The results are highly correlated to each other and this correlation turns out to be even more if the configuration of the testbed approaches its ideal case which consisted of more access points rather than three and more training points rather than thirteen. Next remarkable conclusion is that error tends to become a normal in its distribution in each of X- or Y-directions which leads us to the fact that distance error is more likely to have a Rayleigh distribution.

The effect of number of access points has been discussed. By increasing the number of access points system approaches greater accuracy, as it was expected. When the number of access points is less than three, it is hard for the positing system to accurately locate the mobile since it uses a version of triangulation or fingerprinting methods.

The effect of number of training points has been discussed in details. As we expected by increasing the number of training points positioning takes place more precisely. The effect of training point is more important than the number of access points, though a combination of them will yield the best result while using the most efficient resources.

Configuration of access points is also important. By changing the location of access points in a building one may achieve greater accuracy in terms of positioning with the same data-rate that the infrastructure was built for telecommunication purposes.

Future work can be done on channel model development, in which instead of Ray Tracing channel models one can simply use one's own model and see how accurately it works for Indoor Geolocation. Another feature which this testbed is capable of handling is that one can define the same scenario for another positioning system and by comparing

the results one is able to say which positioning system is more accurate. Use of this Testbed is in a way that one could be able to use another methods of positioning except fingerprinting. One can define AOA in PROPSim and consequently one is able to use AOA methods of positioning. With wider bandwidth one might be able to evaluate the TOA method as well, since TOA approaches need a lot of bandwidth to work accurately.

6 References

[1] K. Pahlavan and A. Levesque, *Wireless Information Networks*, John Wiley & Sons, 1995

[2] K. Pahlavan, P. Krishnamurthy, J. Beneat , "Wideband Radio Propagation Modeling for Indoor Geolocation Applications" IEEE Communication Mag. Vol 36, no 4, Apr. 1998, pp 60-65.

[3] M. Hassan-Ali and K. Pahlavan, "A New Statistical model for Site-Specific Indoor Radio Propagation Prediction Based on Geometric Optics and Geometric Probability" *IEEE JSAC Wireless,* Jan 2002

[4] R. Tingley and K. Pahlavan, "Time-Space Measurement of indoor Radio Propagation", *IEEE Trans. Inst. Measurements,* Vol. 50, No. 1, Feb 2001, pp. 22-31.

[5] K. Pahlavan, X, Li, J. P. Makela, "Indoor Geolocation Science and Technology" IEEE communication Mag. Vol 40, no 2, Feb. 2002, pp 112-118

[6] www.ekahau.com

[7] B. Alavi, K. Pahlavan, "Modeling of Distance Error for Indoor Geolocation", Wireless Communications and Networking, 2003. WCNC

[8] G. M. Giaglis, "On the Potential Use of Mobile Positioning technologies in Indoor Environments", 15th Bled Electronic Commerce Conference, 2002.

[9] G. Held, "Data over Wireless Networks: Bluetooth, WAP, and Wireless LANs," McGraw-Hill Publishing Company, 2000

[10] Y. Tseng, S. Wu, W. Liao, and C. Chao, "Location awareness in AdHoc wireless Mobile, Networks", IEEE Computer Mag. Vol 36, 2001.

[11] J. Hightower, G. Borriello, “Location Systems for Ubiquitous Computing”, IEEE Computer Mag. Vol 34, 2001.

[12] R. Want, B. Schilit, “Expanding the Horizons of Location-Aware Computing” IEEE Computer Mag. Vol. 34, 2001

[13] A. Brewer, N. Sloan, and T.L. Landers, “Intelligent Tracking in Manufacturing”, Journal of Intelligent Manufacturing, Vol. 10, 1999.

[14] C. Nerguizian, C. Despins, S. Affes, “A Framework for Indoor Geolocation using an Intelligent System” 3rd WLAN Workshop, 2001, INRS Telecommunications.

[15] T. Roos, P. Myllymaki, and H. Tirri,“A Statistical Modeling Approach to Location Estimation”, IEEE Transaction on Mobile Computing, Vol. 1, 2002.

[16] J. Kolu, T. Jamsa, ”A Verification Platform for Cellular Geolocation Systems” Elektrobit Oy, Finland, May 2002.

[17] T. Jamsa, J. Kolu, S. Tsukamoto, and T. Matsumoto, “Real-Time Simulation of Adaptive Array Antenna using Broadband Vector Channel Simulator” Wireless Persianl Multimedia Communication, 2002.

[18] T. Jamsa, T. Poutanen, and J. Meinila, “Implementation Techniques of Broadband Radio Channel Simulators” Vehicular Technology Conference, 2003.

7 Appendix A[1]

In Ekahau the received signal power is the metric with which the software would work. The log-loss model is used then to build the propagation model. The log-loss model has three parameters, two regression parameters and the variance of error since a zero-mean error term has been used in this software. The mean value of received power is then given by:

$$\mu(d, p, \theta) = p + \beta_0 + \beta_1 \ln d \quad (1)$$

In (1) d is the distance between transmitter and receiver and p is the transmitted power and θ is the set parameters used.

Another term which can be used in formula (1), as an indicator to the direction of antenna, is parameter δ. Hence the distribution of r as received power is:

$$f(r|d, \delta, p, \theta) = \frac{1}{\sqrt{2\pi}\sigma} \xi\left(\frac{r - \mu(d, \delta, p, \theta)}{\sigma}\right) \quad (2)$$

Where $\xi(x) = \exp(-\frac{1}{2}x^2)$

We already know that for indoor areas due to severe multipath problem the parameters for log-loss model can not be derived universally so they should be extracted from set of received power profile.

Suppose that we have a received signal power vector $\mathbf{r} = (r^{(1)}, r^{(2)}, ..., r^{(n)})$, respective distance vector is $\mathbf{d} = (d^{(1)}, d^{(2)}, ..., d^{(n)})$, the deviation vector

[1] This appendix is a summary of reference [15]

is $\delta=(\delta^{(1)},\delta^{(2)},...,\delta^{(n)})$, and transmitted power is $\mathbf{p}=(p^{(1)},p^{(2)},...,p^{(n)})$ the maximum likelihood method is then used to extract data and find the desired metric and subsequently location.

Maximum likelihood estimation (MLE) of these parameters from empirical data would give us the missing parameter.

Since what we have is linear regression model, standard methods for solving MLEs for linear regression models can be applied. Hence we have:

$$\ell(\theta)=\prod_{i=1}^{n} f(r^{(i)}\left|d^{(i)},\delta^{(i)},p^{(i)},\theta\right)=\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi}\sigma}\xi(\frac{r^{(i)}-\mu^{(i)}}{\sigma})$$

Where $\mu^{(i)}\equiv\mu(d^{(i)},\delta^{(i)},p^{(i)},\theta)$

The likelihood function can be rewritten as follows:

$$\ell(\theta)=(\frac{1}{\sqrt{2\pi}\sigma})^{n}\exp(-\frac{SSE}{2\sigma^{2}})$$

where SSE is Sum of Squared Error and by definition it is given by

$$SSE=\sum_{i=1}^{n}(r^{(i)}-\mu^{(i)})^{2}$$

One can prove that maximum likelihood estimation of $\beta_0,\beta_1,$ and β_2 which are $\hat{\beta}_0,\hat{\beta}_1,$ and $\hat{\beta}_2$ is independent of the estimation of σ which is $\hat{\sigma}$.

By using matrix notation and defining $\hat{\beta}=\begin{bmatrix}\hat{\beta}_0\\ \hat{\beta}_1\\ \hat{\beta}_2\end{bmatrix}$, $\mathbf{Y}=\begin{bmatrix}r^{(1)}-p^{(1)}\\ r^{(2)}-p^{(2)}\\ \vdots\\ r^{(n)}-p^{(n)}\end{bmatrix}$, and $\mathbf{Z}=\begin{bmatrix}1 & \ln d^{(1)} & \delta^{(1)}\ln d^{(1)}\\ 1 & \ln d^{(2)} & \delta^{(2)}\ln d^{(2)}\\ \vdots & \vdots & \vdots\\ 1 & \ln d^{(n)} & \delta^{(n)}\ln d^{(n)}\end{bmatrix}$,we can find MLE of $\hat{\beta}$ as follows:

$$\hat{\beta}=(\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T\mathbf{Y}$$

where $\mathbf{Z}^T$ means transpose of matrix $\mathbf{Z}$ and $\mathbf{Z}^{-1}$ is the inverse of matrix $\mathbf{Z}$. Finally we have $\hat{\sigma}=\sqrt{\frac{SSE}{n}}$.

The formulas above are true in the case that all of the values for received power present and no data is missing. Generally this is not true and some part of the received power is missed due to the restrictions that measurement system and devices have, so called physical restrictions. Some of these restrictions are

Rounded received power

The received power due to the inaccuracy of measurement system is not always exactly the value that has been received and most of the time it is rounded to the nearest integer value.

Insufficient number of values for received power

Due to limitations in physical devices we are not able to receive all of the values of power, i.e. most of the time just received power from limited number of channel is available to measure.

Here is the idea how we can overcome this problem of incomplete data. Assume we have the observed vector for received power $\mathbf{o}=(o^{(1)},o^{(2)},...,o^{(n)})$ which represents the received power from n different potentially accessible channel. For simplicity and without loss of generality we can assume that the first m values are for the case that we have measured the received power but it has been rounded to the nearest integer value and the next n-m values are corresponding to the case that we have not received any power from that channel so we know that the value is less than what we have measured by measurement system. Hence we have the following relationships

$$o^{(i)}-\frac{\varepsilon}{2}\le r^{(i)}\le o^{(i)}+\frac{\varepsilon}{2} \text{ for } i\in\{1,...,m\}$$

$$r^{(i)}\le o^{(i)}+\frac{\varepsilon}{2} \text{ for } i\in\{m+1,...,n\}$$

where the accuracy of system is determined by determining value ε.

The incomplete MLE for this vector is defined as follows:

$$\ell_I(\theta)=\prod_{i=1}^{m}\int_{o^{(i)}-\frac{\varepsilon}{2}}^{o^{(i)}+\frac{\varepsilon}{2}} f(r|d^{(i)},\delta^{(i)},p^{(i)},\theta)dr.\prod_{i=m+1}^{n}\int_{-\infty}^{o^{(i)}+\frac{\varepsilon}{2}} f(r|d^{(i)},\delta^{(i)},p^{(i)},\theta)dr \quad (3)$$

Since there is no analytical solution for maximizing this function the Expectation-Maximization method, so called EM algorithm, is used to find a local maximum for MLE. The EM algorithm can be applied whenever it is possible to evaluate the expected value of the logarithm of the complete data likelihood (log-likelihood). In order to do that we need to specify the distribution of the missing received power values. In the EM algorithm the distribution is obtained by fixing the parameters to some hypothetical values, denoted by θ_t. The expectation of log-likelihood function is then obtained by solving the following equation:

$Q(\theta,\theta_t)=E_{\theta_t}\ln\ell(\theta)$ (4) in which $Q(\theta,\theta_t)$ is the log-likelihood function, E_{θ_t} denotes the conditional expectation given θ_t, and $\ell(\theta)$ represents the likelihood function in (3). Next step which is called maximization step is to find the new value of parameters, θ_{t+1}, by the following equation:

$$\theta_{t+1}=\arg\max_{\theta} Q(\theta,\theta_t)$$

It has been shown that since the likelihood of the parameters never decreases these iterations will converge to a local maximum, if the algorithm converges. Repeating these steps consequently will yield a local maximum.

By taking logarithm of (4) and simplifying that equation one can show that

$$Q(\theta,\theta_t)=n(-\frac{1}{2}\ln 2\pi-\ln\sigma)-\frac{1}{2\sigma^2}SESE \quad (5)$$

where $SESE\overset{df}{=}\sum_{i=1}^{n}E_{\theta_t}(r^{(i)}-\mu^{(i)})^2$

It is proven that the value of β maximizing (5) is $\hat{\beta}=(Z^TZ)^{-1}Z^TY$

Where $\hat{\beta}$ and Z are given by as above

$$Y\overset{df}{=}\begin{bmatrix}E_{\theta_t}r^{(1)}-p^{(1)}\\E_{\theta_t}r^{(2)}-p^{(2)}\\\vdots\\E_{\theta_t}r^{(n)}-p^{(n)}\end{bmatrix}$$

For binned observations it is proven that.

$$E_{\theta_t}r^{(i)}=\frac{(\xi(a^{(i)})-\xi(b^{(i)}))\sigma_t}{\sqrt{2\pi}(\Phi(a^{(i)})-\Phi(b^{(i)}))}+\mu_t^{(i)}$$

where Φ is the cumulative distribution function of a zero-mean, unity variance Gaussian distribution. $\mu_t^{(i)}$ is the mean received power value according to the log-loss model with parameters θ_t

$$\mu_t^{(i)}=\mu(d^{(i)},\delta^{(i)},p^{(i)},\theta_t)$$

and $a(i)$ and $b(i)$ are given by

$$a(i)=\frac{o^{(i)}-\frac{\varepsilon}{2}-\mu_t^{(i)}}{\sigma_t}$$

$$b(i)=\frac{o^{(i)}+\frac{\varepsilon}{2}-\mu_t^{(i)}}{\sigma_t}$$

Since the value of $r^{(i)}$ is known to be within the range $o^{(i)}\pm\frac{\varepsilon}{2}$, its expected value is also within the same range. Thus it can be assumed that the expected value can be approximated by $o^{(i)}$.

For truncated observations the expectation is given by

$$E_{\theta_t} r^{(i)}=\frac{\xi(b^{(i)})\sigma_t}{\sqrt{2\pi}\Phi(b^{(i)})+\mu_t^{(i)}}$$

where notations are the same with previous equations.

By substituting these values in SESE and simplifying the result, for binned observations we can approximate the expected squared error with $(o^{(i)}-\mu^{(i)})^2$ since the range of $r^{(i)}$ is known to be within $o^{(i)}\pm\frac{\varepsilon}{2}$ and for truncated observations a reasonable approximation is $-\frac{\sigma_t^2 b^{(i)}\xi(b^{(i)})}{\sqrt{2\pi}\Phi(b^{(i)})}+\sigma_t^2$.

Knowing the propagation parameter $\hat{\theta}$, now we can find the distribution function of the location variable by bayes rule.

$$p(l \mid \mathbf{r}, \hat{\theta}) = \frac{g(\mathbf{r} \mid l, \hat{\theta})\pi(l)}{\int g(\mathbf{r} \mid l', \hat{\theta})\pi(l')dl'}$$

where $g(\mathbf{r} \mid l, \hat{\theta})$ is the likelihood function of vector r given by

$g(\mathbf{r} \mid l, \hat{\theta}) = \prod_j g_j(r_j \mid l, \hat{\theta})$ where r_j is the received power values.

Function π is also the prior distribution function of location variable.

Since some of the elements of r are truncated we can not apply the formulas above to practical cases. Instead we should use observed vector o. The relation between these two vectors is as follows:

$$\begin{cases} o_j - \frac{\varepsilon}{2} < r_j < o_j + \frac{\varepsilon}{2} & \text{if binned} \\ r_j < o_j + \frac{\varepsilon}{2} & \text{if truncated} \end{cases}$$

thus

$$g(\mathbf{o} \mid l, \hat{\theta}) = \prod_{j\,binned} \int_{o_j - \frac{\varepsilon}{2}}^{o_j + \frac{\varepsilon}{2}} g_j(r \mid l, \hat{\theta})dr \cdot \prod_{j\,truncated} \int_{-\infty}^{o_j + \frac{\varepsilon}{2}} g_j(r \mid l, \hat{\theta})dr$$

and

$$p(l \mid \mathbf{o}, \hat{\theta}) = \frac{g(\mathbf{o} \mid l, \hat{\theta})\pi(l)}{\int g(\mathbf{o} \mid l', \hat{\theta})\pi(l')dl'}$$

since the denominator is not a function of location l, location variable is directly proportional to the numerator. In theory since location variable might be continuous in

$\Re^2$, no prior uniform function could be found for π. But in practice the location variable is always restricted to some areas hence uniform prior function for π can be used.

8 Appendix B

This appendix describes the operation steps that should be taken to run a simulation on the testbed.

The first step in starting a simulation is to run PlaceTool, Ray Tracing software. By clicking on the PlaceTool icon a new placement wizard will open. Next step is to load you new floorplan of the building that you wish to simulate. This is done by going to **file** and then **load new floorplan….**

This will let you to browse for your new floorplan and load it in your simulation environment. After loading the floorplan next step is to define the type of ray tracing you wish to perform. By going to **edit** menu and the **indoor ray tracing** you can choose amongst different indoor ray tracing procedures. For the purpose of this thesis the first option, **Edit-I Point Ray Trace**, is the best option. After choosing this option every click with your mouse results in placing a base station as transmitter or receiver. In our discussion base stations are representatives of access points located in the building. Right-clicking will result in placing a receiver and left-clicking will result in placing a transmitter and will start the simulation. Before placing the transmitter and starting the simulation one may want to save the configuration of the receivers for next simulation. Placing the transmitter has an option that you can set the location of the transmitter according to the grids that user can see in the lower right corner of the screen. For the case of having three access points we should repeat the process three times for the same configuration of receivers. The simulation will give us the multipath profile of the indoor wireless channel between transmitter and receiver. PlaceTool automatically saves these files in its root directory with the prefix that you specify for each simulation.

Since these PlaceTool generated files are not compatible with the power profile that PROPSim imports we should reformat the file in such a way that it is going to be readable by PROPSim channel modeling software. This is done with the help of Matlab®. A piece of code written in Matlab® lets us do the conversion. The code is attached to the end of this appendix. Output of this Matlab® code is three .tap files which are readable by PROPSim. It specifies all of the properties of each tap so we can even change these properties in Matlab® code. At last it should be mentioned that all of the parameters of each tap is also definable in the output of the Matlab® code. Now our files are ready to be used as PROPSim's channel profiles.

Next step is to start with the PROPSim to simulate the multipath between transmitter and receiver. First we start with channel modeling software. By opening a new channel model editor we can import the multipath profile from Matlab® to PROPSim environment. This is necessary if you have not changed the attributes of the simulation like center frequency, Channel update response, sample density, mobile speed (not necessary in this testbed), and estimated simulation time. If these steps have not been taken and these parameters have not been set you can open the .tap file and set the parameters and then save it.

A graph of the channel editor is shown in Figure 40. On the left part of graph you can see the multipath phenomenon and on the right hand side model parameters are shown.

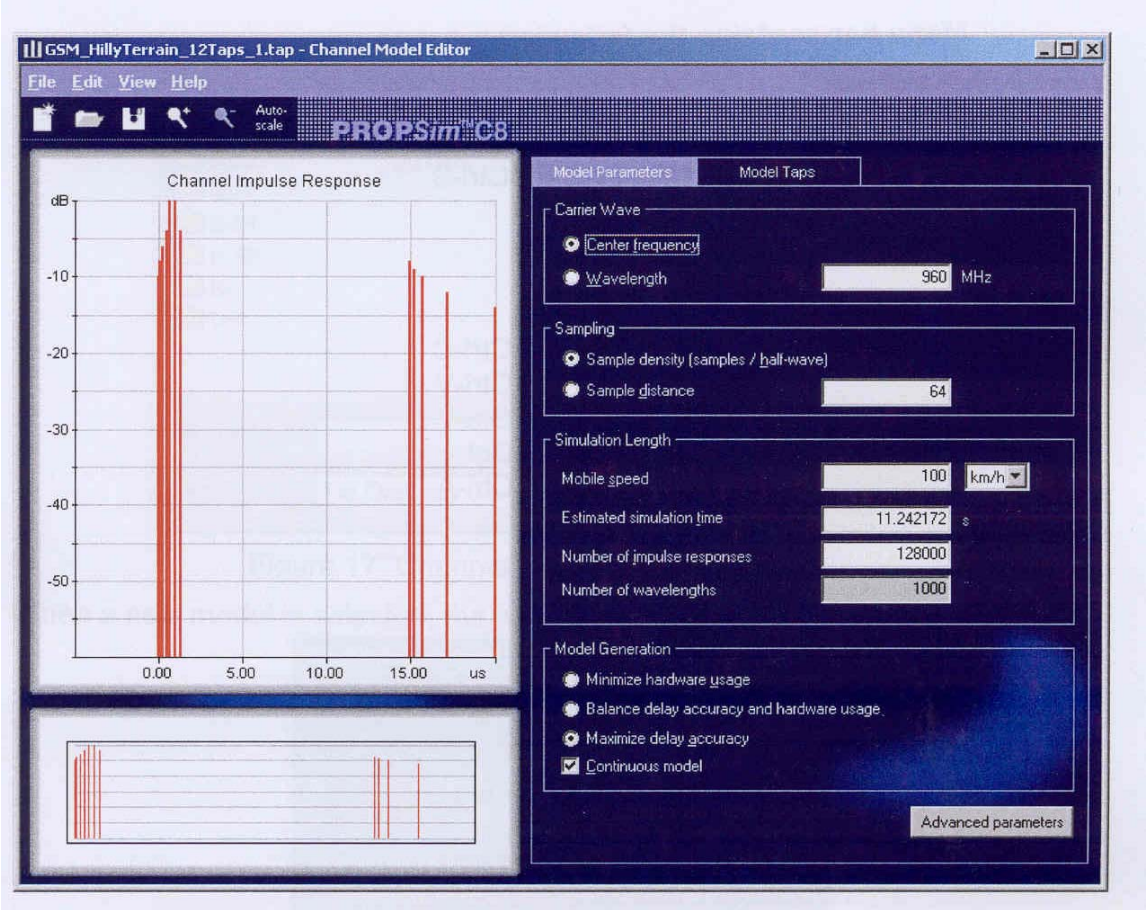

Figure 40: Snapshot of Channel model editor

Next step is to run the simulation editor. This part of PROPSim software is responsible for setting the parameters of the environment between transmitter and receiver. According to the number of access points that we have in our scenario the block diagram of this editor will change. We discuss about the most complicated scenario in which we had 3 access points. For inserting blocks first user should click on the edit button in mode row and then from add row user should select desired number of inputs, channels, or outputs. In this case we had three inputs and three different channels and one output. Each access point is going to be connected to an input and channel model is

imported from PlaceTool. Since Ekahau is using just one receiver then we just need one output in out block diagram. Connecting the respective inputs and channels and outputs are done by clicking on the connect button in the mode row and then clicking on the first box and then second box and software automatically connects them together. Inputs and outputs can be connected to channel models but not together. Figure 41 shows these steps graphically.

For Multiple Inputs Multiple Output (MIMO) purposes one may add different inputs to different channels and different outputs to different channels also. In that case correlating matrix of the channels should also be provided.

On the right hand side of simulation editor some properties of the selected block

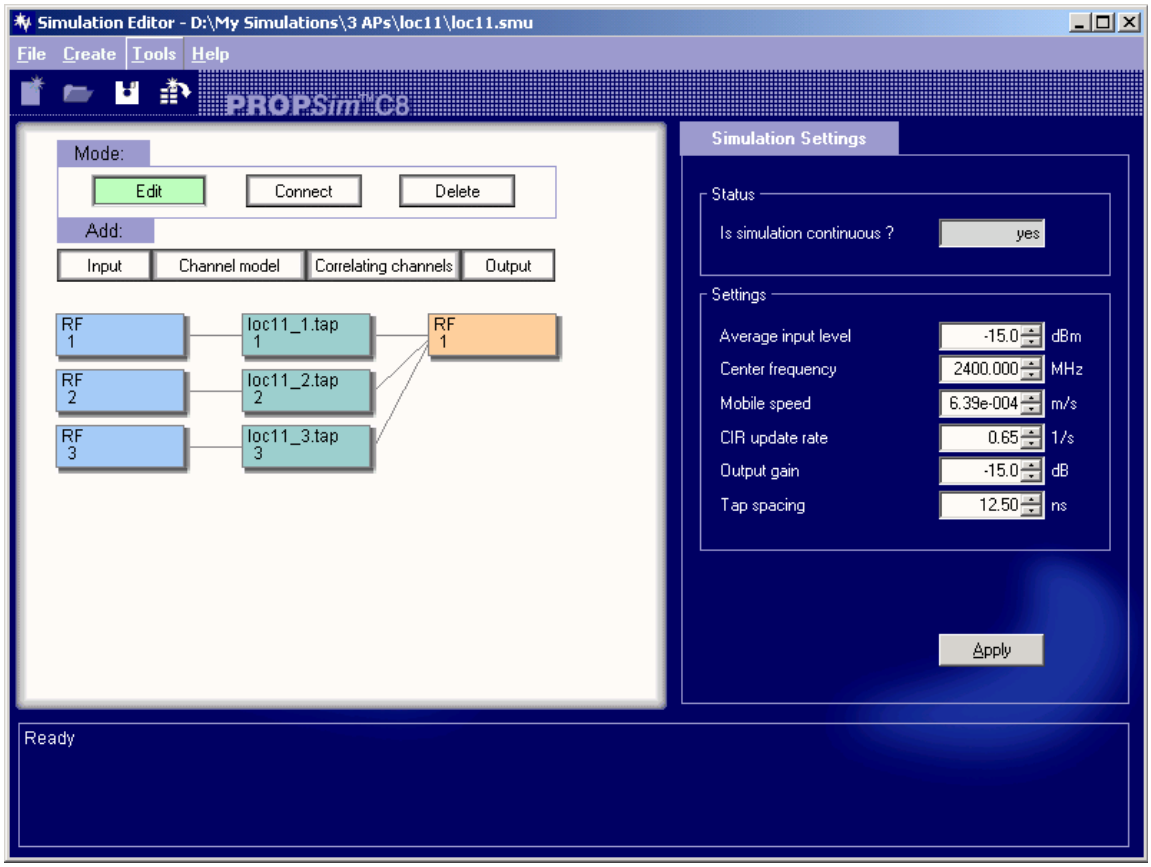

Figure 41: Snapshot of simulation editor

are shown which user can set them for his own experiments. For channel properties, since each channel should have a model inside, we should click on the desired channel and then form property tab select browse button to import the desired channel between transmitter and receiver. Canter frequency, average input level, mobile speed, Channel Impulse Response (CIR) update rate, and output gain can also be set for all of the channels in property tab (Group Settings). Besides this group setting tab user can individually set all of the abovementioned factors for each channel. Each and output has its own property tab which let user to set their properties individually as well.

Last step is to run the simulation. For this purpose we need to run the simulator control user interface which lets us start the simulation, stop it, pause it, and even follow the simulation step by step. By opening a simulator control box we need to import the stored file from simulation editor. After saving this file user can run the simulation. Note that once the simulation starts even if user stops the simulation the channel hardware inside the PROPSim will keep the same tapped-delay-line configuration. So basically from our point of view the channel between transmitter and receiver is what we were interested in. this property also lets us using the PROPSim continuously since the specification says the input powers to PROPSim should not exceed 15 dbm. The output power of common access points in market is 20 dbm which according to the specifications of PROPSim can not be tolerated by the hardware of PROPSim for a long time.

When simulation is done and being run for a while, channels between transmitter and receiver are set properly and user can use the positioning system. Ekahau, in our experiments, was the positioning software which is working with RSS property of the

signal along with fingerprinting method. The output of the PROPSim is connected to the input of the PCMCIA card of the laptop which Ekahau is being run on. Ekahau reads the received power from the PCMCIA card and builds its database. It automatically measures the power from different channels and stores them as different elements in its database.

Starting with Ekahau, user should first make sure that both the client and the manager positioning engine software are installed on the machine and the version of the driver that machine uses for communicating with the PCMCIA card matches the Ekahau requirement. The PCMCIA cards which we used in these experiments were Proxim-Gold card. The reason that there cards been used is that they have a small MC-type connector on them. If user connects another MC-type connector to the PCMCIA connector the internal antenna of the card is shut down automatically which makes these cards suitable for our purposes. It is worth mentioning that although this internal antenna is shut off but we still have leakage from the connectors and the cables and PCMCIA card itself. To reduce all of the effect of these leakages user may prefer to shield the PCMCIA card.

After making sure about the software part of the Ekahau we start the manager positioning engine software. If driver matches Ekahau's requirements then a green line appears in the lower right hand side of the manager positioning engine software along with a open text besides that which represents the status of the engine. Next step is then to import the floorplan of the building under test. This is done by going to **file** menu. Under **file** menu the **new map** option will get the user to browse for the floorplan of the building. New floorplan can be in .jpeg, .jpg, or .png format which makes user unable to use Autocad generated floorplans. The best approach to this problem is to save Autocad file as an .jpg file with all of the details that Autocad would give its user. In the same box

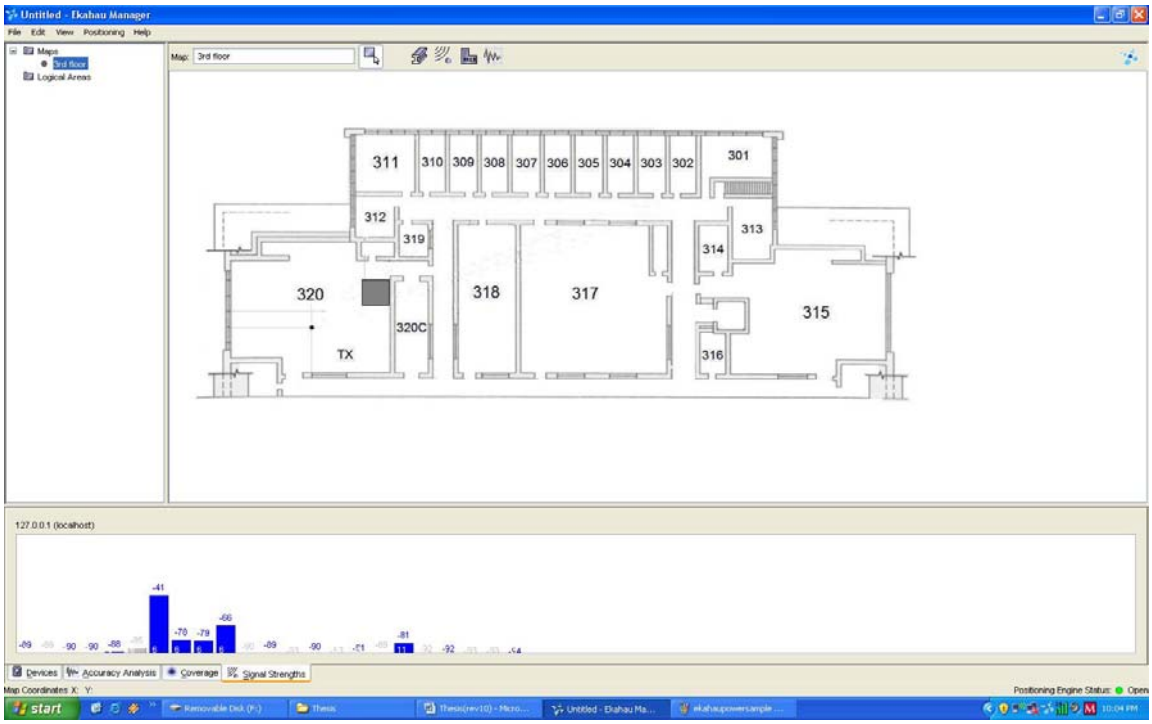

Figure 42: Snapshot of Ekahau

which has been opened to browse for the map user can define the map scale ratio of the floorplan. Map scale ratio gives the number of pixels in Ekahau in terms of meters in real world, so one can measure the error easily by just using pixels in Ekahau. We will get back to the error measurement later in this appendix.

After importing the floorplan right-clicking on the plan in the left toolbar of the manager positioning software will let to define the properties of the floorplan including the map scale ratio. One of the most important use of this tab is to select or deselect the access points that user wants to include in his experiment. For example if user is using his own access points in a building which already has a wireless network, then by going to the access point tab of the properties tab user may be able to select or deselect any visible access point. Since Ekahau categorizes the access points by their network name and MAC address, it is fairly easy to select or deselect user's access points. This is not

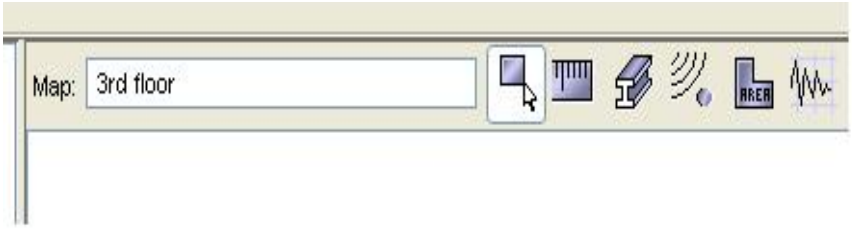

Figure 43: Button for different configuration of Ekahau software

applicable till the first time Ekahau measures the received power from wireless PCMCIA card. So basically the first point in training phase is the point that Ekahau measures and stores the received power from all of the visible access points, but user can do the training part again on the same point after deselecting the undesired access points. Since roaming around the building will make some other access points visible for Ekahau, it is better to check this option from time to time to make sure that built database is just based on the received power from desired access points.

There are six buttons besides the name of the map in Ekahau manager positioning software. We describe the functionality of each in this paragraph in details. Figure 43 shows these buttons and their respective location. The first button is the **edit** button which makes the user enables to edit the properties of the floorplan and network. This has already been done. The second button is the **measure** button. We have discussed this one as well but it is good to remember that if user changes the size of the map he can easily change the scale with this button. Another functionality of this mode is that when manager is in measure mode, by clicking on the floorplan and move the mouse and clicking on another spot in floorplan the window will give us the distance between the two locations. One usage of this mode is to measure the distance error between the actual place and the estimated place. For example if user is standing on place A on the floorplan but Ekahau shows location B on the floorplan as the estimated location, the distance error

is simply the length of the line connecting locations A, and B. The third button is the **rails** button. This mode is responsible to define the paths on which user will build Ekahau's database. By clicking on the map a yellow dot will appear and moving the mouse and clicking on another spot on the floorplan will let the user to draw a line. Basically Ekahau presumes that user will stand on this line while he is updating and building his database. Figure 44 illustrates the procedure of defining rails for Ekahau along with calibration and error measurement. The fourth button is the **calibration** button. By this time everything should be ready in terms of doing ray tracing, exporting the files to PROPSim channel simulator, editing the channel between the access point and the PCMCIA receiver card, running the simulation to make the channel ready, setting the floorplan's settings in Ekahau, and defining the paths in floorplan. This mode will let user to stand on one of the locations on the defined rails and calibrate Ekahau's database. By clicking on this button a window will open which asks you about the access points you want to use. If you choose all, Ekahau will use all of the visible access points to calibrate his database, while clicking on active profile will let the user to use his own access points. After that if you move the mouse on the floorplan, on the defined rails a green dot will appear which is the sign of Ekahau being ready for calibration. By left-clicking Ekahau starts to measure the received power from different access points. It is recommended that during calibration user rotates slowly. This helps Ekahau building its database more accurate since different paths come from different directions. By rotating 360 degrees, Ekahau will receive almost any profile of each access point so it can build its database more precisely. Repeating this procedure increases the number of training points and in fact more training points will lead us to more accurate location estimation

as we have discussed about it in previous chapters. Figure 44 also shows us an example of calibration for a sample floorplan.

The next buttons are **logical areas** and **accuracy analysis** buttons which for this experiment we did not have to use them. **Logical areas** button will define some areas for Ekahau which tells the software that the estimated location should be within those areas and **accuracy analysis** will give user some information about the statistics of the accuracy of the positioning procedure.

Next step is to run the positioning engine. This is done by clicking on the track active device button on the upper right hand side of the software. This button activates the positioning algorithm described above to locate the user according to the new received power. In reality of user moves around the building a blue dot will appear on the map graphically screening the location of the user. If user moves from one point to another this blue dot changes its shape to an arrow to show the direction of the user and speed of user. If user runs from a location to another, the length of the arrow becomes bigger.

The last step is to measure the error. One way to measure the error is to measure the distance of the estimated location and the actual location graphically as mentioned above. Second way is to use a YAX protocol to connect to the manager positioning engine to read the X, and Y elements of the estimated location and compare them to the actual values.

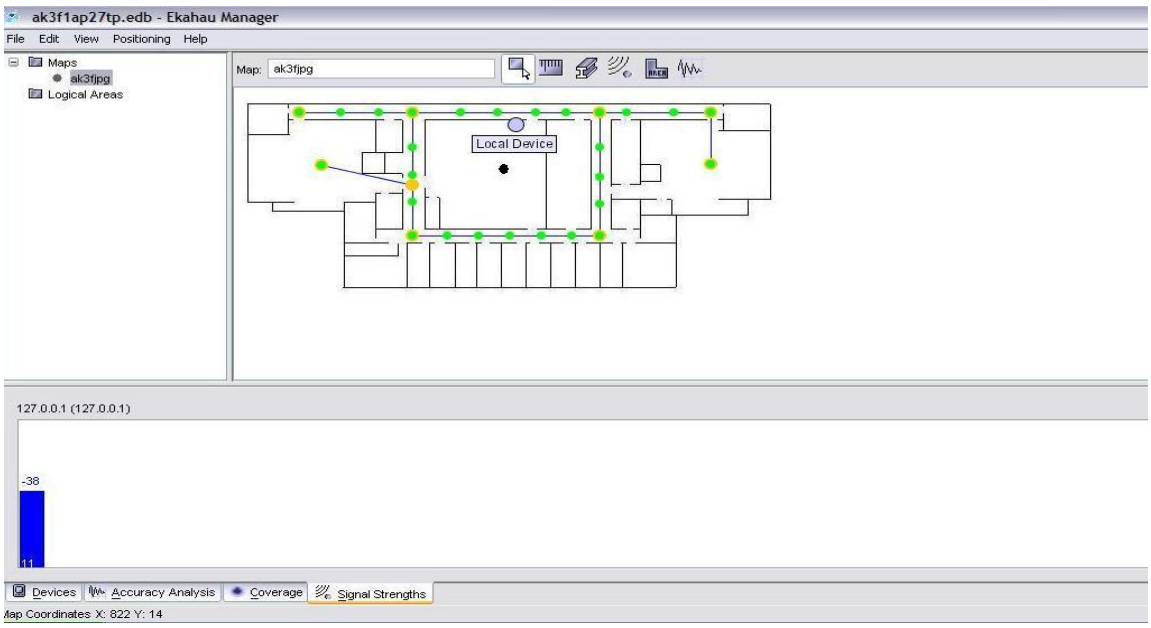

Figure 44: Ekahau's snapshot with rails and calibration points and estimated location

9 Appendix C

A sample MATLAB® code is provided for transforming the Ray Tracing files to PROPSim readable files.

```
clear;
close all;
clc;
jay = sqrt(-1);

%#############################################################
% I/O to enter the values from the .raw file
% filename = 'prefix_1.raw';
% Alternative to a fixed file name, uncomment the text below:
filename1 = input('Enter the filename => ','s');
filename2 = input('Enter the filename => ','s');
filename3 = input('Enter the filename => ','s');
%------ enter the whole file in one string ----------------
fid1 = fopen(filename1, 'r');
  s11 = fscanf(fid1, '%c');
fclose(fid1);
fid2 = fopen(filename2, 'r');
  s21 = fscanf(fid2, '%c');
fclose(fid2);
fid3 = fopen(filename3, 'r');
  s31 = fscanf(fid3, '%c');
fclose(fid3);
%------ remove the header characters ----------------------
k = 1;
while s11(k) ~= '_'
   k = k + 1;
end;
while s11(k) == '_'
   k = k + 1;
end;

j = 0;
for i=k:length(s11)
   j = j + 1;
   s12(j) = s11(i);
end;

k = 1;
while s21(k) ~= '_'
   k = k + 1;
end;
while s21(k) == '_'
   k = k + 1;
end;
```

```
j = 0;
for i=k:length(s21)
   j = j + 1;
   s22(j) = s21(i);
end;

k = 1;
while s31(k) ~= '_'
   k = k + 1;
end;
while s31(k) == '_'
   k = k + 1;
end;

j = 0;
for i=k:length(s31)
   j = j + 1;
   s32(j) = s31(i);
end;
%------ convert to numbers and sort ----------------------
s13 = str2num(s12);
s23 = str2num(s22);
s33 = str2num(s32);
rt1 = sortrows(s13,1);
rt2 = sortrows(s23,1);
rt3 = sortrows(s33,1);

tk1  = rt1(:,1);                % time values in seconds
ak1  = rt1(:,2);                % amplitude values (linear/no units)
phk1 = rt1(:,3);                % phase values (in radians)
dep_angle1 = rt1(:,4);      % departure angle values (in radians)
arr_angle1 = rt1(:,5);      % arrival angle values (in radians)
n_cross1   = rt1(:,6);      % number of crossings
n_ref1     = rt1(:,7);      % number of reflections

tk2  = rt2(:,1);                % time values in seconds
ak2  = rt2(:,2);                % amplitude values (linear/no units)
phk2 = rt2(:,3);                % phase values (in radians)
dep_angle2 = rt2(:,4);      % departure angle values (in radians)
arr_angle2 = rt2(:,5);      % arrival angle values (in radians)
n_cross2   = rt2(:,6);      % number of crossings
n_ref2     = rt2(:,7);      % number of reflections

tk3  = rt3(:,1);                % time values in seconds
ak3  = rt3(:,2);                % amplitude values (linear/no units)
phk3 = rt3(:,3);                % phase values (in radians)
dep_angle3 = rt3(:,4);      % departure angle values (in radians)
arr_angle3 = rt3(:,5);      % arrival angle values (in radians)
n_cross3   = rt3(:,6);      % number of crossings
n_ref3     = rt3(:,7);      % number of reflections
%#############################################################

npaths1 = length(tk1);              % total number of taps
ck1     = ak1 .* exp(-jay * phk1);      % construct the complex taps
ck_db1  = 20 * log10(abs(ck1));     % tap amplitude values in dB
```

```
maxck_db1 = max(ck_db1) ;              % the maximum value of the
amplitudes to add to all taps

npaths2 = length(tk2);               % total number of taps
ck2     = ak2 .* exp(-jay * phk2);      % construct the complex taps
ck_db2  = 20 * log10(abs(ck2));     % tap amplitude values in dB
maxck_db2 = max(ck_db2) ;              % the maximum value of the
amplitudes to add to all taps


npaths3 = length(tk3);               % total number of taps
ck3     = ak3 .* exp(-jay * phk3);      % construct the complex taps
ck_db3  = 20 * log10(abs(ck3));     % tap amplitude values in dB
maxck_db3 = max(ck_db3) ;              % the maximum value of the
amplitudes to add to all taps

maxck_dbi = max(maxck_db3,maxck_db2) ;              % the maximum value
of the amplitudes to add to all taps
maxck_db = max(maxck_dbi,maxck_db1) ;             % the maximum value
of the amplitudes to add to all taps


ck_db_new1 = ck_db1 - maxck_db ;       % normalizing the taps so the
max amplitude is always zero
ck_db_new2 = ck_db2 - maxck_db ;       % normalizing the taps so the
max amplitude is always zero
ck_db_new3 = ck_db3 - maxck_db ;       % normalizing the taps so the
max amplitude is always zero

% ---------------------- Purely for making nice plots ------------------
---
tmin1 = 10*(floor(1e9*tk1(1)/10)-1)*1e-9;
tmax1 = 10*ceil(1e9*tk1(npaths1)/10)*1e-9;

min_ckdb1 = min(ck_db1);
min_db11  = 10*floor(min_ckdb1/10);
min_db1   = max(min_db11, -100);

max_ckdb1 = max(ck_db1);
max_db1   = 10*ceil(max_ckdb1/10);

tmin2 = 10*(floor(1e9*tk2(1)/10)-1)*1e-9;
tmax2 = 10*ceil(1e9*tk2(npaths2)/10)*1e-9;

min_ckdb2 = min(ck_db2);
min_db12  = 10*floor(min_ckdb2/10);
min_db2   = max(min_db12, -100);

max_ckdb2 = max(ck_db2);
max_db2   = 10*ceil(max_ckdb2/10);

tmin3 = 10*(floor(1e9*tk3(1)/10)-1)*1e-9;
tmax3 = 10*ceil(1e9*tk3(npaths3)/10)*1e-9;
```

```
min_ckdb3 = min(ck_db3);
min_db13  = 10*floor(min_ckdb3/10);
min_db3   = max(min_db13, -100);

max_ckdb3 = max(ck_db3);
max_db3   = 10*ceil(max_ckdb3/10);
% ---- this part is just for test how one can write data into a file --
--

len=length(ak1);
for i= 1 : len
    tapnum1(i)=i-1;
    tapdel1(i)=tk1(i);
    tapre1(i)=real(ck1(i));
    tapim1(i)=imag(ck1(i));
end

fid4 = fopen('answer1.tap','w');
fprintf(fid4,'; PROPSim Channel Model File, version 1.1 \r\n ');
fprintf(fid4,' \r\n' );
fprintf(fid4,'[Model] \r\n');
fprintf(fid4,'CenterFrequency = 2400000000 Hz \r\n');
fprintf(fid4,'CirUpdateRate = 26092.1247799443 Hz \r\n');
fprintf(fid4,'SampleDensity = 64 \r\n');
fprintf(fid4,'CirCount = 100000 \r\n');
fprintf(fid4,'DistributionSeed = 18467 \r\n');
fprintf(fid4,'Optimise = maximise delay accuracy \r\n');
fprintf(fid4,'ChannelCount = 1 \r\n');
fprintf(fid4,'Continuous = true \r\n');
for l=1:len
fprintf(fid4,' \r\n' );
fprintf(fid4,'[Tap %g] \r\n',tapnum1(l));
fprintf(fid4,'Description =  \r\n');
fprintf(fid4,'Delay = constant, %e \r\n',tapdel1(l));
fprintf(fid4,'MeanAmplitude = constant, %e \r\n',ck_db_new1(l));
fprintf(fid4,'StandardModel = classical \r\n');
fprintf(fid4,'AmplitudeDistribution = rayleigh \r\n');
fprintf(fid4,'DopplerSpectrum = jakes \r\n');
end
fclose(fid4);

len=length(ak2);
for i= 1 : len
    tapnum2(i)=i-1;
    tapdel2(i)=tk2(i);
    tapre2(i)=real(ck2(i));
    tapim2(i)=imag(ck2(i));
end

fid5 = fopen('answer2.tap','w');
fprintf(fid5,'; PROPSim Channel Model File, version 1.1 \r\n ');
fprintf(fid5,' \r\n' );
fprintf(fid5,'[Model] \r\n');
fprintf(fid5,'CenterFrequency = 2400000000 Hz \r\n');
fprintf(fid5,'CirUpdateRate = 26092.1247799443 Hz \r\n');
fprintf(fid5,'SampleDensity = 64 \r\n');
```

```
fprintf(fid5,'CirCount = 100000 \r\n');
fprintf(fid5,'DistributionSeed = 18467 \r\n');
fprintf(fid5,'Optimise = maximise delay accuracy \r\n');
fprintf(fid5,'ChannelCount = 1 \r\n');
fprintf(fid5,'Continuous = true \r\n');
for l=1:len
fprintf(fid5,' \r\n' );
fprintf(fid5,'[Tap %g] \r\n',tapnum2(l));
fprintf(fid5,'Description =  \r\n');
fprintf(fid5,'Delay = constant, %e \r\n',tapdel2(l));
fprintf(fid5,'MeanAmplitude = constant, %e \r\n',ck_db_new2(l));
fprintf(fid5,'StandardModel = classical \r\n');
fprintf(fid5,'AmplitudeDistribution = rayleigh \r\n');
fprintf(fid5,'DopplerSpectrum = jakes \r\n');
end
fclose(fid5);

len=length(ak3);
for i= 1 : len
    tapnum3(i)=i-1;
    tapdel3(i)=tk3(i);
    tapre3(i)=real(ck3(i));
    tapim3(i)=imag(ck3(i));
end

fid6 = fopen('answer3.tap','w');
fprintf(fid6,'; PROPSim Channel Model File, version 1.1 \r\n ');
fprintf(fid6,' \r\n' );
fprintf(fid6,'[Model] \r\n');
fprintf(fid6 ,'CenterFrequency = 2400000000 Hz \r\n');
fprintf(fid6,'CirUpdateRate = 26092.1247799443 Hz \r\n');
fprintf(fid6,'SampleDensity = 64 \r\n');
fprintf(fid6,'CirCount = 100000 \r\n');
fprintf(fid6,'DistributionSeed = 18467 \r\n');
fprintf(fid6,'Optimise = maximise delay accuracy \r\n');
fprintf(fid6,'ChannelCount = 1 \r\n');
fprintf(fid6,'Continuous = true \r\n');
for l=1:len
fprintf(fid6,' \r\n' );
fprintf(fid6,'[Tap %g] \r\n',tapnum3(l));
fprintf(fid6,'Description =  \r\n');
fprintf(fid6,'Delay = constant, %e \r\n',tapdel3(l));
fprintf(fid6,'MeanAmplitude = constant, %e \r\n',ck_db_new3(l));
fprintf(fid6,'StandardModel = classical \r\n');
fprintf(fid6,'AmplitudeDistribution = rayleigh \r\n');
fprintf(fid6,'DopplerSpectrum = jakes \r\n');
end
fclose(fid6);
```