UNIVERSITY OF OSLO
Department of Informatics

Location Estimation Methods for Open, Privacy Preserving Mobile Positioning

Master's Thesis

Brendan Johan Lee

May 2011

Location Estimation Methods for Open, Privacy Preserving Mobile Positioning

Brendan Johan Lee

May 2011

They who can give up essential liberty to
obtain a little temporary safety,
deserve neither liberty nor safety.

Benjamin Franklin

ABSTRACT

The future is mobile and location aware. More and more of our gadgets are portable and have an online presence. For our location-aware mobile future to be safe, we need to demand that our privacy and anonymity be protected. Currently, each and every location aware-system or feature requires us to give new people, corporations and entities access to one of our most intimate attributes, our location.

The main solutions to ameliorate this have been by cloaking or hiding users from service providers or by moving trust to other "more trustable" parties. We want to minimize the need for trust. Your location is your own, and you should not have to pay with your privacy to determine it.

Our focus lies on location estimation services—services that calculate your location based on measurements done on your network equipment—as they are the main drive behind the location-aware future. You can freely choose, discriminate against, and cloak yourself from services asking for your location, whereas removing the ability to determine your own location effectively impedes location awareness.

We are interested in producing a freely available, open source, privacy preserving, community sourced, and safe location estimation service that minimizes the need for trust. In this thesis we focus on three things: Designing such a system, testing different ways of estimating locations, and determining the best way of estimating locations for the designed system.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisors Alexander Eichhorn and Carsten Griwodz who have provided an excellent environment (or perhaps I should call it a habitat) to learn the art of research. Their enthusiasm and help was invaluable.

To my friends from the lab: Thank you for great conversations, plenty of laughs and many a burning of the midnight oil together. Thanks to Marius N Olsen and Gisle Enåsen for helping me gather field-data, and Haakon Gjersvik Eriksen for his excellent proofreading.

I would like to thank my mother, my father, and my sister, not only for their support and help during the work on this thesis, but for teaching me to question the world and find pleasure and gratification in knowledge.

Finally, to my fiance Heidi: without your support, encouragement and help this thesis would never have come to be.

Oslo, May 16th 2011
Brendan Johan Lee

CONTENTS

GLOSSARY

Assisted GPS (A-GPS) system that can speed up the start up performance of a GPS in a mobile phone. Uses mobile network to determine coarse position, which in turn can provide GPS-satellite almanac enabling quicker fix. 14, 15, 52

Angle Of Arrival (AOA) method used to position MS in mobile networks. 13, 14

Absolute Radio-Frequency Channel Number (ARFCN) a code that specifies a pair of physical radio carriers and channels used for transmission and reception, one for the uplink signal and one for the downlink signal. Defined in GSM spec 05.05 sec 2. 56, 57

Base-station Color Code (BCC) 3 bit value identifying a base station within a GSM-network. Part of the BSIC value. Defined in GSM spec 03.03 sec 4.3.2. xiii, xvi

Broadcast Control Channel (BCCH) in GSM networks a channel that sends a repeating pattern of system information messages that describe the identity, configuration and available features of the BTS. Used by MS to identify the cell and gain access to it. xiv, 10

Bit Error Rate (BER) in GSM networks a measurement of error: The percentage of received bits in error compared to the total number of bits received. 56

Paging repeat period (BS PA MFRMS) the paging repeat period in GSM networks. 56

Base Station Identity Code (BSIC) truncated form of cell identity. 6 bits long. 3 bits NCC, 3 bits BCC. Defined in GSM spec 03.03 sec 4.3.2. xiii, xvi, 56, 57

Base Station Subsystem (BSS) the section of a GSM network responsible for handling traffic and signaling between MS and NSS. 7, 12

Basic Service Set Identifier (BSSID) identifies a IEEE 802.11 wireless LAN access point (AP). Normally the MAC-address of the AP (see also SSID). xvii, 55, 79

Base Transceiver Station (bts) in GSM networks also known as base station or cell site. BTS is a radio mast or site that serves one or more cells. A BTS consists of a network and switching subsystem, base station subsystem and operational support system. i, v, xiii, xiv, xvi, xvii, 6–8, 12–15, 30, 34, 37, 43, 50, 56, 79, 90

Path Loss Criterion (C1) in GSM networks serves to identify candidate cells during cell re-selection. A value greater than 0 identifies a candidate. If serving cell C1 falls below 0, cell re-selection must be performed. xiv, 56, 57

Cell Reselection Criterion (C2) in GSM networks serves to rank candidate cells during cell re-selection. xiv, xvii, 56, 57

Cell Bar Access (CBA) in GSM networks indicates if a given cell is blocked. If CBA is set MS is not allowed to access the cell. However, if CBQ is set the cell can be accessed as a last resort. 56, 57

Cell Bar Qualifier (CBQ) in GSM networks indicates the priority of a given cell. If CBQ is set MS can access the cell, but only as a last resort if no other cells provide sufficient quality. xiii, 56, 57

Code Divided Multiple Access (CDMA) — channel access method for wireless radio communication. CDMA employs spread-spectrum technology with coding system to allow several users to share a band of frequencies and transmit at the same time. xvi, xvii, 15

Cell re-selection — in GSM networks MS camp i.e. listen to the BCCH of the serving cell. When the serving cells signal becomes to weak to camp on the MS must switch to one of its neighbors. This process is called cell re-selection. xiii, xiv, xvi, xvii, 15, 53, 90

Cell type indicator — indicates what type of cell a given cell is (GSM, GPRS, etc.). 56, 57

Cell Global Identity (CGI) — method used to position MS in mobile networks. 7, 8, 34–36, 40, 43, 44, 68, 69, 94–96, 100, 101

Cell Global Identity - Timing Advance (CGI-TA) — a synonym for the TOA method of positioning MS in mobile networks. 12

Cell ID (CID) — in GSM-networks identifies a unique cell within a LA. Uniquely identifies a cell universally when combined with LAC, MNC and MCC. xv, 6–8, 56–58

Common Pilot Channel (CPICH) — a downlink channel each cell broadcasts a known bit sequence at a constant power in UMTS. Used by handsets to determine the cells PSC and RSCP when maintaining lists of neighboring cells. 16

Cell Reselection Offset (CRO) — in GSM networks when calculating C2 for cell re-selection CRO is added to C1. A higher CRO value makes a BTS more attractive to an MS. 57

dBm — abbreviation for the power ratio in decibels of the measured power referenced to one milliwatt. xvii, 16

Database Correlation Method (DCM) — method used to compare fingerprints and weight them in such a way that one can determine which fingerprint is the most comparable and then used that fingerprints true location as a location estimation. 10, 11, 34–36, 42–45, 68, 72, 76, 78, 80, 81, 84, 90, 91, 94–97, 100, 101

Differential GPS (DGPS) — enhancement to GPS that uses a network of fixed, ground-based reference-stations to broadcast the difference between the positions indicated by the satellite systems and the known fixed positions. 9

Downlink Signaling failure Counter (DSC) — in GSM networks a value that is decreased by 4 when a message is unsuccessfully decoded and increased by 1 when successfully decoded. If the value reaches 0 a cell re-selection is performed. xvi, 56

Enhanced Cell Global Identity (E-CGI) — method used to position MS in mobile networks. 8, 12, 35, 40, 43, 44, 68, 69, 74–76, 85, 90–92, 94–96, 100, 101

Estimated Precision Climb (EPC) — in GPS specifies the estimated climb error. 59

Estimated Precision Direction (EPD) — in GPS specifies the estimated directional error. 59

Estimated Precision Horizontal (EPH) — in GPS specifies the estimated horizontal error. See also HDOP. xv, 59

Estimated Precision Speed (EPS) — in GPS specifies the estimated speed error. 59

Estimated Precision Time (EPT) — in GPS specifies the estimated timestamp error. 59

Estimated Precision Vertical (EPV) — in GPS specifies the estimated vertical error. See also VDOP. xviii, 59

Fingerprint — a collection of observed or estimated network measurements at a known or unknown location at a given time. xiv, 10, 11, 16, 30, 31, 35, 37, 60, 63, 67, 69, 72, 74–76, 78, 80, 81, 85, 94, 95

Frame offset — in TDMA networks used to prevent overlapping bursts of transmission. 57

Free space propagation
: free space propagation model assumes a transmit antenna and a receive antenna to be located in an otherwise empty environment. Neither absorbing obstacles nor reflecting surfaces are considered. 8

Geocaching
: a modern GPS-based game based on orienteering. See http://www.geocaching.com. 29

Global Positioning System (GPS)
: global navigation satellite system that provides location and time information. v, xiv, 1–3, 9, 14, 15, 23, 28, 30, 35, 36, 50–54, 65, 67, 76, 78, 79, 95

Global System for Mobile communications (GSM)
: the most commonly used standard for mobile telephone systems. i, xiii–xviii, 3, 4, 6, 7, 12, 15, 16, 18, 25, 28, 31, 32, 35, 43, 44, 50–54, 65, 69, 76, 79, 94

Handover
: handover or handoff refers to the process of transferring an ongoing call or data session from one channel connected to the network to another. 9

Handset
: in GSM networks often used as a term for a mobile phone to differentiate between an actual mobile phone and a MS which includes computers, etc.. xv, xvi, 3

Horizontal Dilution of Precision (HDOP)
: in GPS specifies the horizontal additional multiplicative effect of GPS satellite geometry on GPS precision. See also EPH. xiv, 59

International Mobile Equipment Identity (IMEI)
: unique number identifying a GSM, W-CDMA or iDen MS. Sent to operator when SIM registers on a network. Used to blacklist stolen or interfering devices. 28, 52

International Mobile Subscriber Identity (IMSI)
: in GSM networks a unique number identifying a user. Stored on SIM. xv, xvii

Location Area (LA)
: in GSM-networks a Location Area identifies a collection of cells within a physical area. The size and borders of the area are determined by the network operator. A CID must be unique within an LA. xiv, xv, 6

Location Area Code (LAC)
: in GSM-networks a unique identifier of an LA. A CID must be unique within a LAC. Uniquely identifies a Location Area universally when combined with MCC and MNC. xiv, xvi, 6, 56–58

Location-Based Service (LBS)
: service accessible with MS through a mobile network utilizing the ability to make use of the geographical position of the MS. Often used for map services, games, location search, etc. 1–3, 17, 18, 20, 21, 23, 25, 29, 34, 95, 101

Map-matching
: a process where one assumes a user is following a known path and then uses map-data to increase the accuracy of the location estimation. 11, 68

Mobile Country Code (MCC)
: in GSM-networks Mobile Country Code identifies what country a mobile operator operates in. Oft used in conjunction with MCC. Part of the IMSI. xiv, xv, 6, 56, 58

Mobile Equipment (ME)
: in GSM networks a GSM-device that is fully mobile (such as a mobile phone, a GSM-enabled computer, etc). Identifies the unit itself without a SIM. (see also handset and MS). xvi, 3

Mobile Network Code (MNC)
: in GSM-networks identifies a unique network operator in a country. Uniquely identifies an operator universally when combined with MCC. Part of the IMSI. Virtual operators don't have MNC, only operators who own a physical network. xiv, xv, 6, 56, 58

Mobile Station (MS)
: in GSM networks a GSM-device that is fully mobile (such as a mobile phone, a GSM-enabled computer, etc). Consists of ME and SIM. (see also handset). i, xiii–xviii, 3, 4, 6–10, 12–16, 31, 43, 50–52, 68, 69, 84, 90, 96, 100, 102, 104

Multipath interference
: phenomenon where a wave travels to a detector via two or more paths and two or more components of the wave interfere. xvi, 39

Network Color Code (NCC)
: 3 bit value identifying a GSM-network. Part of the BSIC value. Defined in GSM spec 03.03 sec 4.3.2. xiii, xvi

Neighboring cell
: in GSM networks a MS keeps track of up to 8 cells that neighbor with the serving cell to be able to perform a cell reselection to one of them when needed. xvi, 8, 10, 51

Network equipment
: throughout this thesis this term is used to denote equipment operating in any wireless network including, but not limited to, GSM, UMTS, and WLAN. When discussing equipment or methods unique to GSM or UMTS networks we use the term mobile station. 4, 11, 18–20, 35, 44, 45, 50, 61, 63

Network Measurement Report (NMR)
: collective term for all of the many different values recorded and used for traffic handling and signaling in GSM networks. Non-exhaustive examples are: neighboring cell list, BCC, NCC, BSIC, DSC, and RLT. 8, 10, 35, 43, 44, 68, 103

Network Switching Subsystem (NSS)
: the core component of a GSM network. Responsible for among others call switching and mobility management. xiii, 7, 12

Pico cell
: small cellular BTS covering a small area such as inside buildings (offices, malls, trainstations) or even inside airplanes. Typically used to extend coverage or add network capacity in areas with dense cellphone usage. 7

Primary Scrambling Code (PSC)
: UMTS networks are CDMA networks. Scrambling codes are used so allow multiple clients to share a single frequency. Scrambling codes are divided into Primary Scrambling Codes, and Secondary Scrambling Codes. Each cell is provided a Primary Scrambling code and a set of Secondary Scrambling Codes. The Primary Scrambling Code is broadcasted and used for signaling. The secondary Scrambling Codes are temporarily assigned to handsets engaging in active communications with the cell. xiv, xvii, 15, 16, 58

Routing Area Code (RAC)
: in GSM-network equivalent to LAC but for packet switching rather than line switching (not all cells support packet switching). 57

Rayleigh fading
: a stochastic model for radio propagation anomalies due to multipath interference. At least one of the paths must be changing and none of the paths should be dominant. Also see Rician fading. xvi, 39

Rician fading
: a stochastic model for radio propagation anomalies due to multipath interference. At least one of the paths must be changing and one of the paths must be dominant e.g. line of site. Also see Rayleigh fading. xvi, 39

Radio Link Timeout Counter (RLT)
: in GSM networks during active calls a value that is decreased by 1 when a message is unsuccessfully decoded and increased by 2 when successfully decoded. If the value reaches 0 the call is dropped. xvi, 56

Roaming
: in wireless telecommunications, roaming is a general term referring to the extension of connectivity service in a location that is different from the home location where the service was registered. xviii

Received Signal Code Power (RSCP)
: denotes the power measured by a handset or receiver on a particular physical communication channel, where physical channel in CDMA networks corresponds to a particular spreading code. RSCP is mainly used in UMTS systems, but can be used in any CDMA system. xiv, 16, 58

Radio Receive Level (rxlevel)
: radio receive strength in wireless networks. Measured in dBm. 8, 14, 53, 55–58

Rxlevel access minimum
: in GSM networks when performing cell re-selection rxlevel access minimum is a value indicating the minimum received signal strength the MS must observe from the BTS before it may re-select to it. 57

Received Field Strength Full (rxlevel_f)
: in GSM networks radio receive strength full. Measurement based on all frames. 56

Received Field Strength Sub (rxlevel_s)
: in GSM networks radio receive strength sub. Measurement based on four frames. 56

Received Quality Full (rxqual_f)
: in GSM networks radio receive quality full. Measurement based on all frames. 56

Received Quality Sub (rxqual_s)
: in GSM networks radio receive quality sub. Measurement based on four frames. 56

Serving cell
: in GSM networks specifies the cell an MS is actively listening to at any given time. xvi, 7, 51

Subscriber Identity Module (SIM)
: in GSM networks a smart card that contains the entire customer related information (identification (IMSI), secret key for authentication, etc.). xv, xvi, 3, 50, 53

Secondary Scrambling Code (SSC)
: see PSC. 15

Service Set Identifier (SSID)
: name that identifies a particular 802.11 wireless LAN. Can span multiple BSSID (see also BSSID). xiii, 55, 78

Periodic location update interval (t3212)
: the periodic location update interval in GSM networks. 56

Timing Advance (TA)
: in GSM networks corresponds to the length of time a signal takes to reach BTS from MS. i, 8, 11–13, 35, 43, 56, 68, 103

Time Division Multiple Access (TDMA)
: channel access method for shared medium networks. Users on the same channel are assigned time slots. xiv, xvii, 12

Time Difference Of Arrival (TDOA)
: method used to position MS in mobile networks. 13

Temporary offset
: in GSM networks when calculating C2 for cell re-selection a penalty can be given by subtracting the temporary offset for a given penalty time. This makes the BTS less attractive to the MS for the duration of the penalty time. 57

Time alignment
: since GSM is a TDMA the MS must meet its timeslot. Due to propagation time the MS must postpone or advance its transmission based on distance from BTS. BTS informs MS how much by sending a time alignment value. 57

Timing measurement
: since TDMA based mobile networks share frequencies in time slots, accurate measurements of how long it takes for a signal to travel from sender to receiver is important. Such measurements are called timing measurements. i, 12, 13

Temporary Mobile Subscriber Identity (TMSI)
: temporary subscriber ID used in GSM and UMTS networks randomly assigned by the network. Used to avoid tracking by sending IMSI only when necessary. 56

Timeslot Number (TN)
: identifies the time slot in a TDMA network. 56

Time Of Arrival (TOA)
: method used to position MS in mobile networks. xiv, 12, 13

Transmit Power Level (txlevel)
: radio send strength in wireless networks. Measured in dBm. 56

Uplink Time Difference Of Arrival (U-TDOA) — method used to position MS in mobile networks. 13

Universal Mobile Telecommunications System (UMTS) — third generation (3G) mobile telecommunications system based on GSM. i, xiv, xvi–xviii, 3, 7, 12, 15, 16, 25, 28, 31, 35, 43, 50–52, 54, 65, 69, 76, 94, 103

Vertical Dilution of Precision (VDOP) — in GPS specifies the vertical additional multiplicative effect of GPS satellite geometry on GPS precision. See also EPV. xiv, 59

Virtual operator — in GSM networks a virtual operator is an operator that does not own their own network infrastructure but rather leases a network from a different operator. Not to be confused with roaming. xv

Vocoder — in GSM networks indicates what type of speech codec to be used. 56

Wideband Code Division Multiple Access — the most common code division technique used in UMTS. xv

War driving — the act of searching for and registering wireless networks by a person in a moving vehicle or on foot using a portable computer, PDA or cellphone. 8, 10, 53, 95

Wireless Local Area Network (WLAN) — links two or more devices using some wireless distribution method usually providing an access point to the wider internet. xvi, 3, 10, 11, 18, 31, 32, 35, 41, 43, 44, 50–52, 54, 65, 69, 72, 74–76, 78, 79, 84–86, 90, 92, 93, 95, 96, 100

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

During the last few decades we fantasized about and worked towards ubiquitous computing; a reality where we are surrounded by computers; a reality where computers not only know where they are, but also know where we are; the computers are so integrated in our daily lives that we do not notice their presence. This dream has become reality .

When our devices need to know where we are they have two options: use a built-in positioning device like the Global Positioning System (GPS), or use a mobile location estimation service that translates their virtual location in the networks they are connected to into a real world location.

But simply knowing our real world location provides little value to our lives. To create value we must associate our location to something: a map, other people or a point of interest nearby. The result is a need for Location-Based Services (LBSs): Services that when provided with our location offer something of value in return: A map, a list of people we might want to meet, the store we are looking for or the time table of the closest bus.

Not everybody wants others to know where they are all hours of the day. Hence there is a need for location cloaking: Services designed to hide us from the LBS. We can continue to use their services, but trick them so they really do not know where we are.

There exists, both in the literature and in the reality of our daily lives, a conflict of interest between these services. On one hand we have the LBS providers who provide services based on a user's location. On the other hand we have cloaking services that help preserve the user's anonymity. Most cloaking services rely on degrading data, often by providing fake data. This works fine if the user knows their location and wants services based on it; in other words the user's device contains a GPS.

If not, the use of a mobile location estimation service is required. Providers catering to this audience must use a database of network data. Mobile network operators generally regard their own network topography databases as trade secrets and even if we potentially could convince them to share we would expect to pay a high price. In addition few if any networks are static. They continuously evolve. Therefore mobile location estimation providers must collect their own data. They might do it themselves, but that is expensive and requires continuous maintenance since networks continuously change and evolve. Instead they can convince users that

have GPS chips in their devices to do it for them. This is known as crowd sourcing.

Here we immediately see a problem. Since cloaking services generally rely on degrading data, they in turn degrade the data mobile location estimation services crowd source and use in their databases. The result is that users wishing to protect their own privacy unintentionally contribute to degrading the quality of the services they themselves use.

The division between mobile location estimation services and LBSs is not clear cut. Traditionally mobile location estimation providers are also LBS providers, and most independent LBS providers rely on integrating an API from a mobile location estimation provider.

1.1 Motivation

By separating out the mechanism that provides location estimates as a separate entity, one can ensure that LBSs and existing cloaking methods can work together without involuntarily degrading each other's services. Furthermore one can eliminate the need to customize cloaking services specifically for different LBSs, allowing a freer choice of not only cloaking methods, but also LBSs. Finally a user should be free to own their own location, and not have to pay for it with their own privacy.

Besides privacy one of our major concerns regarding existing location estimation systems regards availability and ownership. We can divide current services into two categories: Services provided by network operators and services provided by third party providers. Services provided by network operators are normally limited to the operators own network, and do not work when roaming on a foreign network. These services are often quite expensive to use, and users have no control over their own data. Services provided by third party providers are commonly based on data contributed by the users themselves. Even though the users contribute the data, they have no control over it. The systems are closed source. A user has no knowledge of what data is transferred, or how the data is used. She has no insight in how securely the data is handled or what, if any, third parties are privy to the information. She has no guarantee that the system will be available at any given time in the future, or that the provider will not change the licensing scheme overnight. Finally she might even be paying for data she herself contributed.

A community driven, open access mobile location estimation service can eliminate these problems. The system would be available anywhere people choose to collaborate on the project and contribute data, can quickly adapt since the users themselves both collect and use the data and the license can only be changed if all of the users contributing data agree to the change. That leaves the privacy issue. The system must ensure the users privacy. It might even inform users about privacy issues and demonstrate to them the importance of their privacy. The system would also allow testing new and existing location estimation methods on larger sets of data. Most published tests were run on either simulated data, or on a very small subset of data. The fact that most large collections of data are owned by large commercial companies limits the amount of research done on the data. An open access system like the one proposed here would invite anybody to do research and tests on the data. The system would also invite the community to create their own applications that use and contribute to the system.

Even though more and more devices are sold with built in GPS locators, far from all users have access to or choose to have access to data from a GPS locator. GPS locators are still generally only included on high end and midrange devices. In addition users are free to turn off their GPS locators, something many users do. GPS locators are often turned off to save power since devices have a limited power supply, and users generally want their devices to function as long

as possible on the go. In addition many users simply do not trust a device that potentially has the capability to track their location covertly, and therefore choose to turn off their GPS locators.

GPS locators have over the years become more and more trustworthy and fast thanks not only to new chipsets, but also the introduction of new technology. However GPS locators still require a more or less an unobstructed view of the sky, and do not always function immediately after powering up. Finally GPS locators cannot, and will never function fully indoors, a fact that does not prevent users wanting to use LBSs indoors

The conclusion is that nowhere in the foreseeable future can LBSs rely solely on users providing their location using GPS locators. This in turn means a continued demand for mobile location estimation.

1.2 Problem Statement

More and more applications and services are based on users' geographical location. However the data such location estimation is based on is mainly not available to the end users themselves and belongs to the operators or service providers. This is a major issue. We feel that a user should own their own location and not have to rely on a corporation to obtain it.

Secondly there are major privacy concerns involved in the fact that the operators own the data, and in theory can do anything they like with it, change their license or even block a user's access without consequences. Our goal is to eliminate such issues by creating a system where the users own their own data and do not have to rely on, or trust, a corporation to be able to obtain their own geographical location.

We want to investigate building a *privacy-preserving*, *community-driven* and *open-source* system for locating mobile devices in networks without the need for a GPS receiver, special hardware, or any support from the network operator. We want to determine the best location estimation method for such a system. And we want to gather a large set of field-data and use it to test existing location estimation systems.

1.3 Contributions

Working to give people back the ownership of their own location we have proposed and analyzed the effects of creating a free, open access, and privacy preserving mobile location estimation service. We have defined a set of principles such a service must be based on, investigated the privacy issues of the service, and investigated the internal and external workings of such a service. We have also investigated how this type of service can ameliorate the issues of cloaking services and LBSs interfering with each other.

We have further proposed a novel location estimation method tuned towards openness and privacy preservation. Our method reduces the frequency and amount of data transfer, and the amount of data stored. It embodies the simplicity of the simplest location estimation methods combined with the power of the more advanced methods. The method is extremely flexible and adaptive to different types of handsets[1] and data.

[1] In Global System for Mobile communications (GSM) and Universal Telecommunications System (UMTS) networks the terms Mobile Station (MS) and Mobile Equipment (ME) are used, where ME denotes a network capable device itself and MS denotes a network capable device together with a Subscriber Identity Module (SIM). To avoid confusion we have used the term *network equipment* throughout this thesis to denote equipment operating in any wireless network including, but not limited to, GSM, UMTS, and Wireless Local Area Networks (WLAN), and the term *mobile station* when discussing equipment or methods unique to GSM or UMTS networks.

We have implemented a powerful and flexible test system designed to test both our own proposed estimation method and many types of existing estimation methods. The system is designed to be flexible and handle the widest possible range of data types and network equipment and compare a widest possible range of location estimation methods on equal terms. Furthermore, the system is designed to be easily adapted to new types of data and estimation methods in the future. Finally the system is designed to gather and work on large sets of data, allowing tests on more complete sets of data than we often find described in the literature.

We have created a logging device specifically made for measuring GSM networks and gathering data in the field. We created software for use with the logging device in addition to software to log networks directly on mobile stations running on three different mobile platforms: Android, Symbian Series 60 and OpenMoko.

We gathered two large sets of data and used them to test our own estimation method, in addition to two well known and simple estimation methods and the most common advanced estimation method. The tests were performed on a large collection of data from a wide range of mobile stations. We have compared all of the tested estimation methods in terms of precision, success rate and processing time.

Finally, we also created visualization software used for analyzing and illustrating collected data, comparing different algorithms and estimation methods and comparing collected data to the known position of antennas by plotting the data on top of maps or satellite imagery.

1.4 Outline

For the remainder of this thesis our main focus lies on location estimation methods. They must be in place before making the system secure, but cannot be selected arbitrary without considering security and privacy.

In chapter 2 we briefly explore the fundamentals of GSM networks and show how these fundamentals can be exploited to locate devices within the network. We take a closer look at how such location estimation is done, before looking at some of the work being done to mitigate the conflict of interest between location estimation services and privacy.

In chapter 3 we suggest creating a *privacy-preserving*, *community-driven* and *open-source* system for locating mobile devices in networks, take a closer look at the requirements for such a system, and suggest a design.

In chapter 4 we introduce our own location estimation method, *Intersecting Areas*, tuned towards open and privacy preserving mobile location estimation systems. We thoroughly discuss the method, and suggest several alternative implementations.

In chapter 5 we present the implemented test system used to gather data and test mobile location estimation methods. Furthermore we present the tools implemented for use with the test system including data gathering tools, a database, visualization software and a collection of maintenance scripts.

In chapter 6 we choose a selection of location estimation methods to test and compare with our suggested location estimation method. We present a series of tests of precision, success rate and processing time, and thoroughly discuss the results.

Finally, in chapter 7, we summarize the results and contributions of this thesis, discuss the results of our efforts, and present topics for future work.

CHAPTER 2

BACKGROUND

In this chapter we briefly explore the fundamentals of GSM networks and show how these fundamentals can be exploited to locate devices within the network. We take a closer look at the most common methods used to locate devices within the network, in addition to methods that allow the devices themselves to determine their physical location using the network. We investigate how privacy and anonymity relates to such mobile location estimation. Finally we look at some of the work being done to mitigate the conflict between mobile location estimation services and privacy.

2.1 GSM Networks - Overview

GSM networks are radio networks consisting of a large number of cells, allowing frequency reuse. Each cell is served by a fixed transmitter known as a base transceiver station (BTS) or cell site. One BTS normally serves three cells, but may also serve a lower or higher number. This is called sectoring.

Cells are in the literature described as hexagons. This, however, is a simplification created to ease the description of networks. In reality cells have no specific shape, and their actual shape is dictated by the surrounding physical environment.

Radio networks divided into cells are referred to as cellular networks. The major advantage of cellular networks is improved network capacity. Since frequencies are reused, one can fit more traffic on a given bandwidth compared to traditional radio networks.

GSM networks are divided into areas, known as Location Areas (LAs) and assigned a Location Area Code (LAC). Each LA contains a number of cells. The actual physical size of, and the number of cells residing within, an LA is decided by the network operators, and varies greatly from network to network. Figure 2.8 page 22 shows the LAs we observed in one network in a given area. The purpose of LAs is to reduce signaling traffic. If every mobile station was to update the network with information about which cell it was listening to every time it switched cells, the network would be flooded with signaling traffic leaving no capacity for actual voice and data traffic.

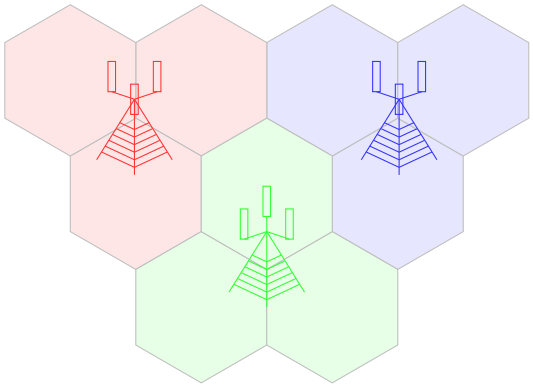


Figure 2.1: Cellular network

Instead the network is updated only when the mobile station travels between LAs. The mobile station also informs the network when it is powered down. In addition, the mobile station must inform the network that it is still in any given LA at certain intervals, thus mobile stations that power down incorrectly or lose contact with the network completely are detected. How often a mobile station must provide such updates is decided by the network provider who broadcasts the given limit on signaling channels. The value can vary not only from operator to operator but also from cell to cell.

When the network needs to contact the mobile station due to an incoming call or SMS, the network informs the LA the mobile station is currently registered with. Every cell in that LA then sends a message over the signaling channel asking the mobile station to identify which cell it currently is listening to.

Each cell in a given LA has it's own unique identifier, known as Cell ID (CID). Each location area in a network has it's own unique identifier known as Location Area Code (LAC). Each country has it's own unique identifier, Mobile Country Code (MCC), and each network within a given country has it's own unique identifier, Mobile Network Code (MNC). Together these four identifiers uniquely identify any cell in any GSM network anywhere in the world.

2.2 Mobile Location Estimation in GSM Networks

Methods for locating mobile stations in a GSM network are roughly divided into three categories: Mobile-based, network-based and mobile-assisted. Mobile-based techniques only utilize data available on the mobile station itself and do not require any extra functionality in the network. Network-based techniques only utilize data available in the network without involving the mobile station in any way, hence the mobile station itself has no knowledge of the fact that it is being located. The third and final category of methods, called mobile-assisted[1], involves a cooperation between the mobile station and the network.

First we take a look at the most common mobile-based location estimation methods, then move on to the most common network-based and mobile-assisted methods. We focus on location estimation in GSM networks, but the same methods, sometimes with minor changes, are used in UMTS networks.

2.3 Mobile-based Location Estimation in GSM Networks

Mobile-based location estimation is a collective category of location estimation methods that use data gathered on the mobile station. Several of the methods in this category also use data from external databases, or even send information to external servers for processing without being classified as network-based methods. The network in network-based refers to the GSM-network itself, more specifically the Base Station Subsystem (BSS) and the Network Switching Subsystem (NSS).

The boundaries between mobile-based, network-based and mobile-assisted location estimation methods are not strict. Some methods can be classified in several different categories depending on what data they are based on or other details of the method.

2.3.1 Cell Global Identity (CGI)

The simplest form of location estimation, known as Cell Global Identity (CGI) is based solely on the serving CID. Using the unique identifier of the serving cell one can estimate the location of the mobile station if the location of the BTS is known. This method is relatively inaccurate as the size of a cell may vary from 35 kilometers in rural areas, through a couple of hundred meters in urban areas, down to a few meters for pico cells indoors.

All mobile stations have access to the globally unique identity of the cell they are listening to, and most mobile stations that allow external software applications expose this value through their API. If one has access to the exact or calculated coverage area of the serving cell, one can determine that the mo-

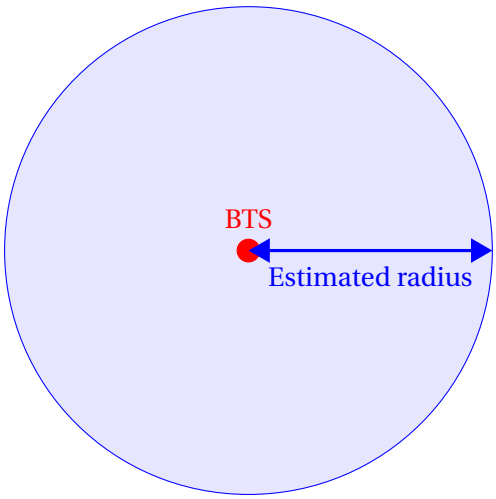


Figure 2.2: Cell Global Identity positioning method

[1] This category is sometimes referred to as hybrid methods.

bile station resides somewhere it. In figure 2.2 page 7 that demonstrates the function of CGI one can determine that the mobile station is located somewhere within the blue area.

The knowledge of the coverage area can be gathered in a wide variety of ways for example: war driving, crowd sourced war driving, network planning tools, known propagation models or data provided by the operator.

2.3.2 Enhanced Cell Global Identity (E-CGI)

There are several methods of improving the accuracy of CGI. Enhanced Cell Global Identity (E-CGI) is a collective term for combining CGI with other measurements to improve accuracy. Such measurements can be the received signal strength (often denoted Radio Receive Level (rxlevel)), the Timing Advance (TA), the sectorization of the BTS or any of the Network Measurement Report (NMR) measurements. E-CGI is normally classified as a mobile-based location estimation method, but since it is a collective term, some E-CGI methods can be mobile-assisted or even network based depending upon their implementation.

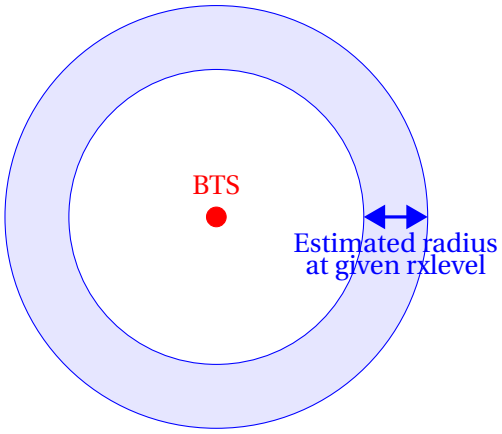


Figure 2.3: Enhanced Cell Global Identity positioning method using received signal strength

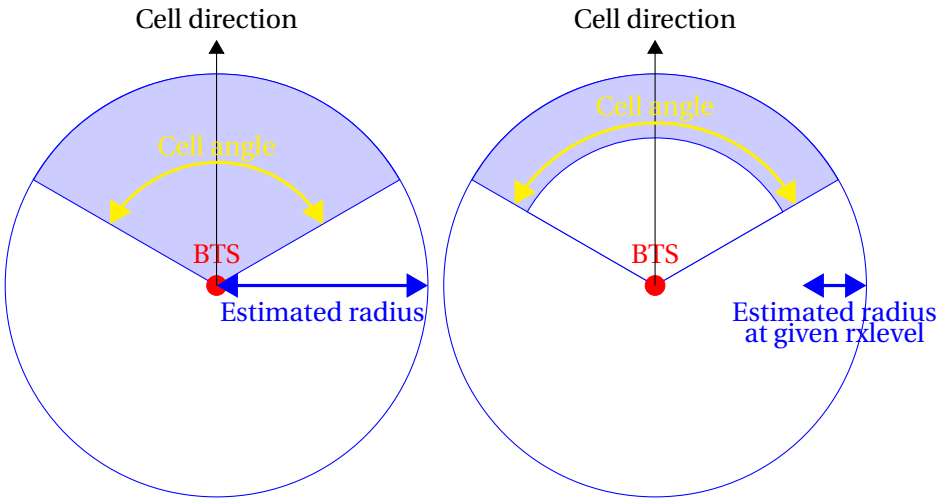
We now look at some of the common types of E-CGI. Considering the large amount of measurements available in NMR, in addition to the possibility of using other forms of measurements, we cannot cover all E-CGI methods in this text. Therefore we only concentrate on the most widely used methods.

One way is utilizing information about the received signal strength. In near free space propagation the received signal strength can be assumed to be proportional with the distance to the BTS since the effect of long term fading is the only significant fading effect. Free space propagation, however, cannot be assumed. In fact in urban areas near free space propagation isn't very common. Even in free sight situations in rural areas, true free space propagation does not occur since the ground reflects the signal. There are three ways of solving the problem of non-free space propagation: One can base the location estimation on actual measured field data. If one has access to information about the surrounding topography one can use network planning tools to create an estimated propagation model. Finally we can choose to use a known model such as the Okumura-Hata model[1, 2], which can be adapted to specific environmental conditions[3]. Figure 2.3 page 8 demonstrates enhancing CGI using received signal strength.

If the direction and angle of reception of the receiving antenna at the BTS is known this can be used to gain further accuracy. Figure 2.4a page 9 shows enhancing CGI with sectoring, while figure 2.4b demonstrates combining sectoring with received signal strength for higher accuracy.

Sectoring information is generally only available to the network operators themselves. Some operators use an algorithm for their CIDs that encodes information about the general direction the antenna is listening, namely north, south-east or south-west. Unfortunately this is not a standard. Not all operators do this, and there are several different ways of encoding this into the CIDs. Hence one cannot take the availability of such information for granted.

A mobile station must always maintain a list of the signal strength of the six strongest neigh-



(a) Sectoring only

(b) Combined sectoring and received signal strength

Figure 2.4: Enhanced Cell Global Identity positioning method with sectoring

boring cells. This list is used to make decisions about when the mobile station should start listening to a different cell or, if a speech or data channel is active, to know when to initiate a handover. This means that one, at any given time, has access to information about the identity and received signal strength of up to seven cells. This data can be utilized to preform a cross bearing to estimate the location. Unfortunately few mobile stations grant access to neighboring cell information through their standard API.

2.3.3 Global Positioning System (GPS)

Many modern mobile stations have built in Global Positioning System (GPS) chips. GPS receivers calculate their location by precisely timing signals from medium earth orbit satellites that orbit earth twice a day. After the selective availability system, created to limit the accuracy of GPS for non-military use, was turned off in 2000 GPS can be quite accurate. Precision can be a couple of meters, but can also vary up to several hundred meters. However, GPS-receives report an estimated inaccuracy of location. More precise measurements can be acquired using Differential GPS (DGPS), a system that involves ground stations sending correctional signals via the FM radio band. These measurements are actually precise enough to land a commercial jet using GPS[4].

GPS has some big advantages compared to other location estimation systems. Besides the fact that GPS works 24 hours a day anywhere in the world, the precision is normally high compared to other systems. There are, however, several disadvantages, mainly the fact that one needs a clear view of the sky. This limits GPS positioning to outdoor situations and limits the accuracy in cities and wooded or mountainous areas. GPS positioning requires power consuming hardware, and even though the power consumption of GPS-chips has been reduced the past few years, the battery drain is quite noticeable. GPS-systems also can take a long time to start functioning after a cold-start[2].

[2]Cold starting a GPS means starting a GPS from an initial powerless state where the GPS has lost its memory.

2.3.4 Database Correlation Methods (DCMs)

Database Correlation Methods (DCMs) comprises a collection of methods that use a database of measurements to improve the accuracy of mobile location methods. The database used for DCM normally consists of some form of what we here define as fingerprints: *a collection of observed or estimated network measurements at a known or unknown location at a given time.*

For instance, a fingerprint can simply be the serving cell at a certain unknown location, all observed WLANs and the observed received signal strength at a known coordinate at a certain time, or the whole NMR at a certain location. As long as the measurement contains at least one measurement of some form of some wireless network, it is considered a fingerprint.

The terminology is not novel[5, 6], but the definition and use of the word fingerprint varies in the literature. We therefore feel the need for a more concrete definition, and settled on the above.

When a DCM method is provided a fingerprint with an unknown location it uses a correlation algorithm to compare said fingerprint to fingerprints with known locations residing in a database and determine the closest match. The known location of the closest match is then asserted as the location of the provided fingerprint.

The fingerprints in the database can be gathered in multiple ways. One can build a database based on estimated information. If one has access to information about the surrounding topography network planning tools can be utilized to generate a database [7, 8], or lacking such information or tools a known propagation models like the Okumura-Hata model[1, 2]. Field measurements can be gathered through war driving or crowd sourcing. Finally the network operators already have large collections of network data, both estimated and measured in the field, used when planning, maintaining, troubleshooting and expanding their networks. They do not, however, normally grant access to third parties as such data is considered a trade secret.

DCM is not limited to cellular phone networks, but may be used on any wireless network where access points can be uniquely identified such as WLAN networks and bluetooth networks. The more static the network access points are, the more efficient this method is. Thus networks where access points tend to be mobile, such as Bluetooth networks, generally do not increase accuracy and can in fact reduce the accuracy of the system.

DCM enables location estimation with fewer measurements than the average network-based location estimation method and also avoids multipath problems[9]. The simplest form of DCM, presented elsewhere[5], uses least mean squares comparison. This method performs best in dense urban areas and indoors, however it can be used to improve the accuracy of any of the location estimation techniques described in this chapter[5].

2.3.5 Database Correlation Method from Laitinen et al

Laitinen et al 2001[5] present a simple DCM algorithm:

$$d(k) = \sum_i (f_i - g_i(k))^2 + p(k)"$$

where f_i is the signal strength of the request fingerprint on the i^{th} Broadcast Control Channel (BCCH), $g_i(k)$ is a signal strength of the k^{th} database fingerprint on the same channel, and the summation is taken over channels that are found in both fingerprints. Each channel found in only one of the fingerprints contributes to the penalty term $p(k)$"[5]. From here on we use the term *score* when referring to $d(k)$. The method is intended to be used on a database that contains, and with mobile stations that have access to, neighboring cell information. Furthermore

the algorithm only utilizes received signal strength. The algorithm can be used directly on fingerprints containing other types of measurements such as TA, but must be adapted if to be used on fingerprints containing multiple attributes[3] such as both received signal strength and TA. The method can also be used without adaption on fingerprints containing measurements of other types of networks, such as WLAN-networks, as long as they only contain a single attribute.

Laitinen et al do not describe their use or implementation of the penalty term $p(k)$" beyond the above mentioned. It is natural to assume that the optimal value of the penalty term varies depending on what data the algorithm is used on. Since the algorithm utilizes the received signal strength, and signal strength is measured in different ways on different networks, the general scores before penalty would vary depending on network type. When the score before penalty varies from network type to network type, the optimal penalty variable would also vary from network type to network type. Since radio implementation and antenna configuration affects received signal strength, one also expects the optimal penalty term to vary between different network equipment.

2.3.6 Statistical Location Estimation

Promptly, after using DCM methods for mobile location estimation was introduced, using statistical methods to improve on DCM was suggested. First out was Kalman filtering[10, 11]. Kalman filters rely on observations over time, containing random variations and inaccuracies, to produce values closer to the true values without noise.

Later Kalaf-Allah et Kyamakya suggested using Bayesian estimation as a method of performing DCM for location estimation in mobile networks[12, 13, 14, 15, 16]. They compared several different Bayesian algorithms with some rather interesting results. Bayesian algorithms rely on estimations over time. Thus one needs to collect a range of fingerprints from a user. Each location estimation is then based on not only the current fingerprint, but the estimated location of the previous fingerprints.

The latest research is on creating DCM techniques using neural networks to improve WLAN location estimation[17].

2.3.7 Map-matching

Another method that can be used to improve the accuracy of mobile location estimation is map-matching[18, 19]. Map-matching is a process where one assumes a user is following a known path. Simply put one moves the estimated location so that it ends up on top of whatever path it is closest to. Naturally map-matching becomes more accurate when one stores and take into account previous estimated or known locations. By knowing the users previous locations one can determine what path the user is following, and use location estimation to determine how far along that path the user has moved. Exceptions are made when the user strays too far off the path or comes close to an intersection.

Although map-matching can significantly improve accuracy it is limited to situations where digital map-data is available and the user actually is following a known path. This basically limits map matching methods to situations where the network equipment is following a road, path, or similar.

[3] When talking about attributes of fingerprints in this paragraph, we are only talking about attributes that can be utilized as indicators of distance such as received signal strength and TA

estimation. It location, and we path. As an on estimation. More information from a measuring device to further situations when location periods such as when a car

2.4 Network-based and Mobile-assisted Location Estimation in GSM Networks

We have looked at some of the most common mobile-based location estimation methods. Now we delve into the most common network-based and mobile-assisted methods.

When using data available only to the BSS or NSS or extra hardware located at the BSS or BTS the method is classified as a network-based, or mobile-assisted method.

What category the methods fall under can vary depending upon how the methods are used. Some of the methods can even be used in ways that would classify them as mobile-based, for example the Time Of Arrival (TOA) method can be a mobile-based, mobile-assisted or network-based method depending upon how it is implemented.

Most of the common network-based and mobile-assisted methods rely on using timing measurements available in GSM networks. Therefore we start with a short introduction to timing measurements.

2.4.1 Timing Measurements

GSM is a Time Division Multiple Access (TDMA) network. Timing measurements are used to synchronize when data should be sent in a TDMA network. The parameter used in GSM networks is called Timing Advance (TA).

TA values can be utilized to gain a more accurate location estimation as one can expect the time offset to be proportional to the distance from the BTS. TA values have a granularity that translates to approximately 550 meters in GSM networks and 80 meters in UMTS networks[4]. Simply using timing measurements would place the mobile station inside a circular locus between 0 and 550 meters, 550 and 1100 meters, etc. from the BTS in a GSM network. Several methods to improve on these figures exist. TA values are only available during active conversation or data transfer.

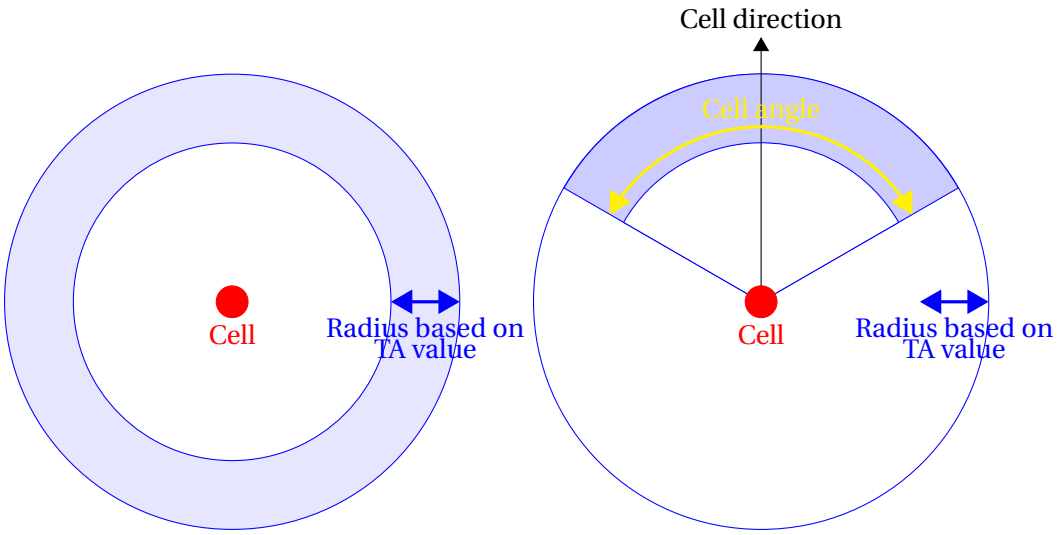
We now look closer at some of the most common location estimation methods that utilize timing measurements.

2.4.2 Time of Arrival (TOA)

Time Of Arrival (TOA) is, simply put, a measurement of propagation time. One measures the time it takes data to travel from the BTS to th mobile station or vice versa. Based on this measurement one positions the mobile station within a circular locus around the BTS. Figure 2.5a page 13 illustrates TOA used on an omni-directional cell, while figure 2.5b shows the same technique used on a sectored cell. The method determines that the mobile station is located somewhere inside the blue area. Notice how sectored cells improve the accuracy.

This technique is based on the same principle as E-CGI using received signal strength values, but instead utilizes TA values for improved accuracy. Hence this technique is sometimes referred to as Cell Global Identity - Timing Advance (CGI-TA).

[4] See appendix A for more information



(a) Omnidirectional cell (b) Sectored cell

Figure 2.5: Timing Of Arrival positioning method

TOA requires accurate time synchronization between the mobile station and the BTS. A more common approach is to do round trip time measurements eliminating the need for time synchronization[20].

2.4.3 Time Difference of Arrival (TDOA)

The Time Difference Of Arrival (TDOA) method is based on the same principle as TOA, but uses measurements from two or more cells. A hyperbolic loci is derived estimating the mobile stations' location. This method requires the BTSs involved to have synchronized clocks[20].

If the calculations are done on the mobile station one must ensure that the signal is sent from all of the BTSs simultaneously, and if the calculations are done at the BTSs one must ensure that the measured time of arrival at the BTSs are based on synchronized clocks. The latter is often referred to as Uplink Time Difference Of Arrival (U-TDOA) since the timing measurements are all done on the uplink from the mobile station to the BTS.

Figure 2.6 page 14 illustrates using TDOA to locate a mobile station based on the TA values from three different BTS. The timing measurements (T1-3) are utilized to calculate the distance the mobile station is from each BTS. The intersection of these three distances, which represents the location of the mobile station is then calculated.

2.4.4 Angle of Arrival (AOA)

Angle Of Arrival (AOA) involves measuring the angle the signal from a mobile station reaches the BTS. This measurement requires a special array of antennas at the BTS. The time difference between the arrival of signal at each antenna is used to calculate the angle of the incoming signal.

When using only a measurement from one BTS additional information, like received signal strength, is required as the measurement results in positioning the mobile station somewhere in a straight line from the BTS. If one has measurements from several BTSs, the intersection of

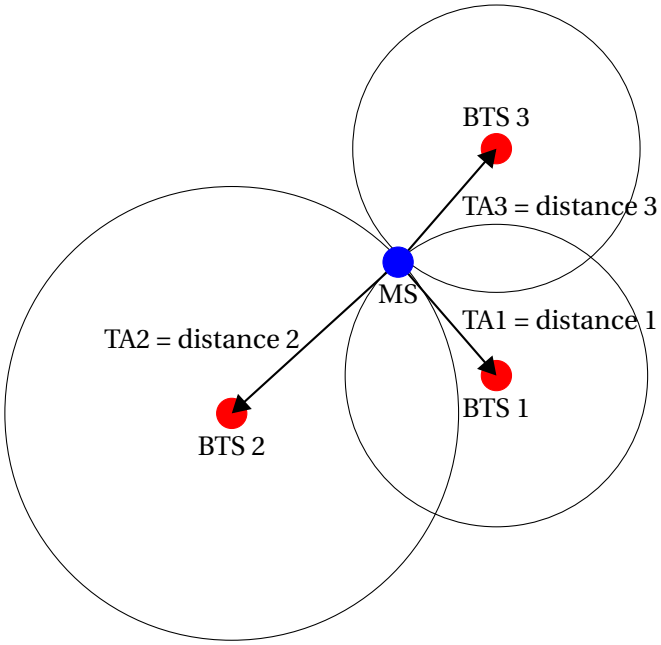


Figure 2.6: Time Difference Of Arrival positioning method

these straight lines are used to estimate the location. Again this requires the BTSs involved to have synchronized clocks.

Figure 2.7a page 15 shows using AOA from one BTS combined with received signal strength to locate a mobile station. Using this method one can conclude that the mobile station is located somewhere along the stretch where the black line, based on AOA, intersects with the blue area, based on rxlevel. Figure 2.7b illustrates using AOA from multiple BTSs to locate a mobile station.

For this method to be accurate the distance between the BTS and the mobile station should not be too small. The distance between the antennas in the array also affects the accuracy. The greater the distance between the antennas, the higher accuracy one can expect.

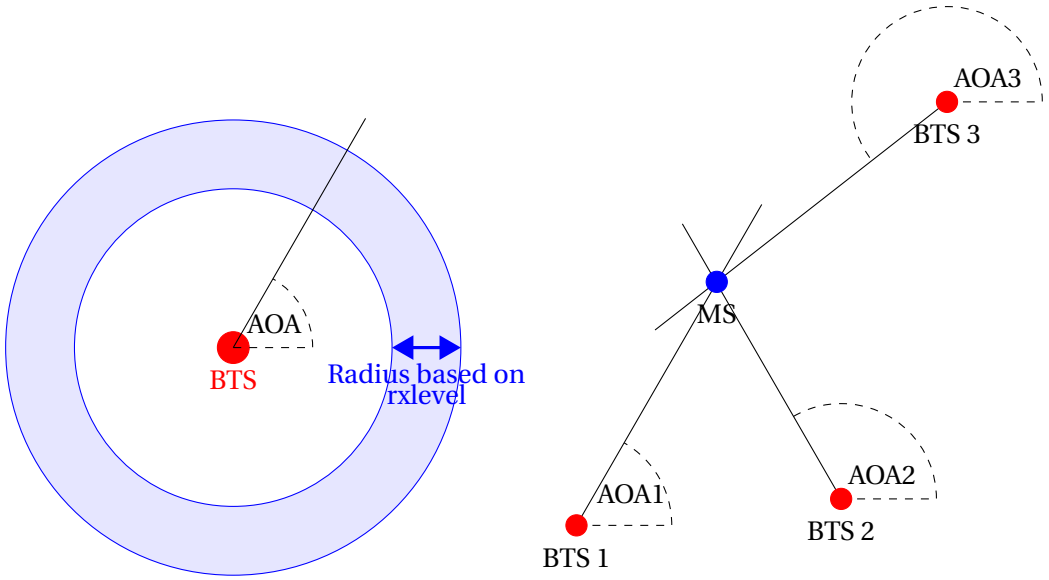
The use of AOA in the field is limited[21] mainly due to the costs and impracticality of the extra antenna arrays[5].

2.4.5 Assisted Global Positioning System (A-GPS)

Assisted GPS (A-GPS) is a system created to improve positioning on cellphones with integrated GPS receivers. A-GPS utilizes the mobile network to help the GPS receiver determine it's initial position when first turned on and assists positioning when the reception from the GPS-satellites is degraded. A-GPS may also work indoors[22] in some situations.

The mobile station contacts the A-GPS system any time it is in need of assistance. Some form of network-based or mobile-assisted location estimation is performed by the A-GPS server which has it own GPS. The server estimates the location of the GPS-satellites in the sky based on the estimated location of the mobile station and its own GPS. Error correction values are normally calculated. All of the values calculated by the server are then transmitted back to the mobile station.

Exactly what information and values are calculated and transferred varies from system to



(a) Single BTS and rxlevel (b) Multiple BTSs

Figure 2.7: Angle Of Arrival positioning method

system. A-GPS is a collective term for GPS positioning aided by a remote server. On one end of the scale we have systems that assist the GPS receiver retrieve a fix, on the other end we have systems where the mobile station transmits all the data from the GPS receiver to the A-GPS server which in turn does all of the calculations thus offloading CPU-load from the mobile station to the server.

2.5 A Note on Neighboring Cells

It is important to note that neighboring cell information works completely different in GSM and UMTS networks. In the GSM system mobile stations maintain a list of the up to 8 neighboring cells. Lists of neighboring cells are broadcasted by the BTS and the lists are compiled by the network operators. The received signal strength of neighboring cells is monitored by mobile stations and used to determine when to perform a cell re-selection. In addition neighboring cell information is often used to ensure that fast moving mobile stations listen to larger cell, while slow moving mobile stations listen to smaller cells in the same area. In most APIs that grant access to neighboring cell information the retrievable values are the full ID of the cell, and the received signal strength.

The neighboring cell information in UMTS networks uses a completely different approach. UMTS networks are Code Divided Multiple Access (CDMA) networks, and therefore use scrambling codes to divide frequencies between multiple simultaneous users. Scrambling codes are assigned as Primary Scrambling Code (PSC) and Secondary Scrambling Code (SSC). Each cell is assigned one PSC and multiple SSCs. The PSC is announced by the cell and used for broadcasting, while the SSCs are used when mobile stations are actively engaged in communicating with the cell.

Mobile stations in the UMTS network maintain a list of neighboring PSCs instead of iden-

tifying neighbors by their ID as is done in GSM networks. In addition the signal strength measurements are not measured in dBm (dBm) as in GSM networks. Instead the mobile station measures the Received Signal Code Power (RSCP) on the Common Pilot CHannel (CPICH).

The result is that neighboring cell information in GSM networks is completely incompatible with neighboring cell information in UMTS networks. RSCP is not easily converted to dBm. There is also no way to convert PSC to cell ID information or vice versa without access to the network providers internal information.

This means that fingerprints containing neighboring cell information from a UMTS network cannot be used when analyzing fingerprints from a GSM network and vice versa. Note that this only applies to the neighboring cell information, not the serving cell information.

2.6 Privacy

We have investigated some of the technical details of location estimation in mobile networks. Now we move our focus to the social implications and problems surrounding mobile location estimation, starting with privacy, before moving on to investigate the work being done to ameliorate the problems.

The Oxford dictionary of philosophy defines privacy as:

> In moral and political theory, private conduct is that which it is no business of the public, and particularly the public institution of law, to notice. Similarly, private information about a person would be that to which there can be no right of public access.[23]

Within law a similar definition is often used: "The right to be left alone and to keep certain matters secluded from public view."[24] Article 8 of the European convention on human rights[5], the British human rights act 1998[6] and the Norwegian "Lov om styrking av menneskerettighetenes stilling i norsk rett"[7] all recognize privacy as a human right. This includes privacy of communications.

Privacy is normally a term used when describing an individual's, or a group of individuals', relations to the public. The term is also commonly used to describe relations between individuals and relations between an individual and a group or company, even though the pure definition of the term does not include this use. What is considered private conduct varies from society to society, and even from individual to individual. Certain private information that one would never consider sharing with the public, one has no problem sharing with a company. How much, and what, private information one feels at ease sharing is highly individual. Therefore we find the term privacy too vague in this context.

In our opinion a person's right to privacy should be taken for granted, thus we consider it to be the moral responsibility of any company to protect their customers' privacy. Furthermore it should be considered a company's moral responsibility to inform their customers of their right to privacy, and help their customers achieve such privacy. But there is no doubt that not all companies share our vision. This leaves us with the need of a new term. We need a term that covers not only an individual's right to privacy in their dealings with the public, but an individual's right to protection when dealing with a company. A term not only covering the company's

[5] http://conventions.coe.int/treaty/en/Treaties/Html/005.htm (2011-04-03)
[6] http://www.legislation.gov.uk/ukpga/1998/42/contents (2011-04-03)
[7] http://www.lovdata.no/all/hl-19990521-030.html (2011-04-03)

moral obligation to protect their customers private conducts from society, but also covering the customers protection from the company itself.

2.7 Anonymity

Within the field of information technology anonymity is defined as "the state is being not identifiable within a set of subjects, the anonymity set"[25], where the anonymity set is the set of all possible subjects with potentially the same attributes. Although the term does not describe our above-mentioned requirements, all of these requirements can be seen as a subset of anonymity. A subject being anonymous implies that all of the above requirements are fulfilled.

Throughout this text the term privacy is used as defined, regarding an individual's relation to the public. The term anonymity is used as its strict definition provided above. When a method or service conceals the subject, but does not supply anonymity, it provides partial anonymity.

2.8 Privacy Policies

One suggested solution to the many privacy issues regarding Location-Based Services (LBSs) is the so-called privacy policies[26, 27]. Privacy policies involve service providers creating policies where they state exactly how a user's data is used, and can be used in the future. In addition, privacy policies often include privacy profiles allowing a user a higher level of control of their data. In their simplest form privacy policies simply inform the user on how their data is used[28]. More advanced privacy policies allow users hands on control. The simplest form is an on–off mechanisms allowing users to regulate when their location should be available to the software. More advanced versions offer fine-grained controls allowing users to regulate when and who (where who normally is analogous to which application) has access to location data, or even near complete control of their data[26].

Today most service providers offer privacy policies, at least in their simplest form. In certain areas such privacy policies are even required by law[8,9,10]. We are even seeing the large service providers such as Google [11] and Apple [12], more or less voluntarily, implement more advanced privacy policies.

Such privacy policies are to an end-user's advantage for several reasons. Policies requiring users consent may make users aware that they are disclosing their location, and might even force them to research why disclosing their location can be problematic thus making informed choices. In addition policies provide the users with a powerful tool if their privacy is breached. In a legal case following such a breach they have a legal document stating what the service provider has guaranteed them, and they need only prove that the company has not followed their own policies.

However, one might determine such privacy policies to be of no consequence regarding one's own security. Privacy policies require complete trust in the service provider i.e. trust us or don't

[8] The California Online Privacy Protection Act of 2003 http://leginfo.ca.gov/cgi-bin/displaycode?section=bpc&group=22001-23000&file=22575-22579 (2011-03-29)

[9] Children's Online Privacy Protection Act of 1998 http://www.ftc.gov/ogc/coppa1.htm (2011-03-29)

[10] Financial Services Modernization Act of 1999 http://business.ftc.gov/privacy-and-security/gramm-leach-bliley-act (2011-03-29)

[11] http://www.google.com/intl/en/privacy/ (2011-04-10)

[12] http://www.apple.com/privacy/ (2011-04-10)

use our service. And though having an aid in a lawsuit when one's privacy has been breached is a good tool, it is not a tool that improves security.

2.9 Location Cloaking

A different suggested solution to the many privacy issues regarding Location-Based Services (LBSs) is location cloaking. Location cloaking involves network equipment hiding its true location from the LBS. Normally the network equipment hides itself by making itself indistinguishable from other network equipment, hiding its true location among several other bogus locations, or simply providing the location not as a point in space but rather as an area in space (city, region, country etc.). Hiding in this manner can be seen as putting on a cloak of uncertainty, thus making oneself partially invisible to the LBS, hence the term location cloaking.

Location cloaking can in general be split into two main categories: centralized services and decentralized services. Centralized services involve the network equipment sending a request to a trusted third party who in turn cloaks the requests before passing it on to an LBS. The LBS handles the request as normal, passes the result back to the cloaking service which in turn passes it back to the network equipment. A decentralized service on the other hand does not involve a trusted third party. Instead the network equipment handles the cloaking itself, either alone or in collaboration with other network equipment, and communicates directly with LBS.

A sub-category of decentralized services, and perhaps the most common type of decentralized service, is peer-to-peer services. Normally this involves multiple network equipment creating a peer-to-peer network, often communicating through WLAN, Bluetooth or other non-GSM networks. The peer-to-peer network then uses a cloaking method to hide the individual network equipment from the LBS.

A third category, combining the best of both worlds, is the hybrid approach. The hybrid approach normally involves a peer-to-peer network coordinated by, or communicating through, a trusted or non-trusted third party. Zhang et Huang suggest such a hybrid approach[29]. Their approach uses algorithms for load balancing between the peer-to-peer network and a trusted third party, and they claim that this not only protects the system from overload and bottlenecks but protects it from malicious attacks.

A wide variety of different cloaking methods have been suggested, tested and implemented in the literature. Here we give a short overview of some of the important terminology used withing the field, followed by an overview of some of the main methods used.

2.9.1 Location-based Range Query (LBQ)

The term location-based range query describes one of the most common uses of LBS today. The term describes a client flagging its interests in certain types of objects within a given range of its own location[28]. For example "where is the closest hospital?", "Show me restaurants that serve Italian food rated with at least five stars open on Friday evenings close to my hotel!". LBQ services were one of the first types of services that promoted the use of LBS to mobile phone users. Thus many of the cloaking methods described in the literature concentrate on cloaking this type of queries. Although LBQ is still popular, the new and rising type of LBS is social networking. We foresee a shift in the field to more general cloaking methods that can hide a user's location independently of what type of service they use.

2.9.2 K-anonymity

The term k-anonymity is a central term in location cloaking on mobile networks. K-anonymity, first introduced in the field of relational data privacy research[30], addresses the issue

> How can a data holder release a version of its private data with scientific guarantees that the individuals who are the subjects of the data cannot be re-identified while the data remain practically useful?[31]

In location cloaking on mobile networks a subject is considered k-anonymous

> if and only if the location information presented is indistinguishable from the location information of the least $k-1$ other subjects[32].

A large range of articles use k-anonymity when evaluating and implementing location cloaking methods. K-anonymity is normally used as a measurement of how dependable a cloaking service is[29, 33, 34]. In addition many suggested cloaking services allow their users to set their own minimum k-anonymity value[35, 32], a feature often referred to as a privacy profile. Yet others combine these two ways of using k-anonymity[36].

2.9.3 L-diversity

L-diversity is closely related to k-anonymity. Formally l-diversity prevents sensitive attributes being disclosed by providing diversity among the sensitive attributes of the anonymized group[37, 33].

In mobile location estimation k-anonymity deals with making a subject indistinguishable from $k-1$ other subjects, while l-diversity deals with making a user's location indistinguishable from $l-1$ similar locations. The idea is that the subject defines their l-diversity value, which is used to ensure that their location is indistinguishable from $l-1$ sensitive locations or sensitive objects. For example a subject present at a hospital presenting a location obfuscated in such a way that they may either be present at the hospital, one of two gas stations, a grocery store, a train station or one of two police stations, has an l-diversity of $7-1$. As with k-anonymity l-diversity can be used as a user specified setting, or as a measurement of the performance of a system.

2.9.4 Onion Routing

Onion routing is a well-known and trusted method of cloaking one's identity. It was used on computer networks[38] long before the need for location cloaking within mobile networks arose. Establishing a peer-to-peer network of network equipment that use onion routing to cloak each other's locations[32] is a simple and easy to implement method of providing location cloaking on mobile networks. However, this method must normally rely on a trusted third party to establish the network. In addition establishing and maintaining the network, and the relatively high amount of data traffic onion routing generates, consumes large amounts of power. Power on network equipment is a high value commodity.

2.9.5 Path Perturbation

Path perturbation, also known as path confusion (and more generally obfuscation), has been suggested as a cloaking method on mobile networks[39]. Path perturbation involves using algo-

rithms to cross different users' paths in locations where two or more users meet, thus confusing attackers. This method does not completely cloak a users location, but rather prevents an attacker from tracking a user's movements over time. Therefore this method must be combined with other cloaking methods if it is to ensure full anonymity.

2.9.6 Obfuscation, Geographic Space Obfuscation and Graph Obfuscation

Duckham et Kulis define obfuscation as "the means of deliberately degrading the quality of information about an individual's location in order to protect that individual's location privacy."[40]. This term can be seen as covering most, if not all, of the location cloaking field, and can be seen as a synonym to location cloaking. The term geographic space obfuscation specifies that the obfuscation is limited to one's location in 3-D space, not in time.

The term graph obfuscation describes methods of obfuscation that use an algorithm on a graph to cloak a user's location. Graph obfuscation as a method has been suggested by among others [40], [35], [39], and [34].

2.9.7 Hilbert Cloak

Kalnis et al suggest using the Hilbert space-filling curve[41] to map the 2-D locations of mobile users to 1-D space in their cloaking services MOBIHIDE and PRIVE[34, 42, 33]. Both of their cloaking services use obfuscation, by hiding an individual's location within the location of several users, and guarantee k-anonymity through their algorithm based on the Hilbert space-filling curve. The services both use a trusted third party to organize peer-to-peer networks of network equipment who use the algorithm to protect their location.

2.9.8 Dummy Messages

One suggested method for cloaking users location is using dummy messages[43, 44]. This seemingly simple technique involves generating bogus locations and paths, thus hiding the user within a set of fake users. The method can be used through a trusted third party, but the main benefit is that it can be used without the need for a trusted third party or a peer-to-peer network by generating dummies directly on the network equipment. Although this method sounds simple, creating realistic dummies that act normal over a long period of time is a complex task. For this method to ensure security, it must be impossible to determine which users are dummies and which are true users, even over a long period of time.

2.9.9 Whole Network Approach

Mokbel et al suggests involving the whole network in the cloaking process[36]. Their cloaking approach involves not only a trusted third party cloaking server, but an LBS that is cloaking aware i.e. created to handle cloaked requests. Their LBS is tuned specifically to deal with anonymous queries in cloaked areas. In addition they use different methods to approach different types of queries. Queries are divided into three types: Private queries over public data (where is the nearest gas station?), Public queries over private data (how many people are in a given area?), And private queries over private data (where is my closest buddy?). This approach also allows users a privacy profile. Although this is by far one of the most complete approaches, it does not address the issue of needing a trusted third party. In addition a close working relationship between the third party and the LBS itself is assumed and the LBS must be cloaking aware.

2.9.10 Location Uncertainty

We have named the final cloaking method we present here location uncertainty. Most of the methods we have looked at so far hide the user from the LBS. Instead it is possible to degrade the location information in space and time, providing a lower resolution[28]. This can be as simple as presenting one's location to the LBS at a much lower granularity, such as providing it with the city, county or even country we are in. Or we could choose to record one's location over time and inform the LBS that we reside somewhere within the area surrounding all of one's previous locations.

The downside of this method is that in many instances degrading one's location also degrades the quality of service the LBS can provide. This is particularly true when dealing with LBQ services. Asking for a restaurant that serves fish and vegetarian options somewhere in the Czech Republic is quite different from asking for the same type of restaurant close to the old town square in Prague.

This type of cloaking is, however, far from useless. Social networking services often need a far lower granularity. For example finding people with the same interests as you in a city or even a whole country is still quite useful.

2.10 Summary

This chapter covered how cellular networks function and how the characteristics of them can be used to locate connected devices spatially. We explained the more common ways of locating devices and discussed how some of these methods can be used from the devices themselves, how others only work when operated from the network and how some methods require a cooperation between the network and the device.

Further we have seen how mobile location estimation methods have social implications and introduce problems concerning the individuals privacy and anonymity and discovered a thriving field of research trying to ameliorate the problems mobile location estimation introduces.

We saw how most of the work done involves hiding an individual from the provider of LBSs, known as obfuscation. And how many of them rely on one's ability to trust a third party, and the rest normally rely on some form of peer-to-peer network.

The chapter also covered the common terminology and concrete practices used within the fields of mobile location estimation and location cloaking.

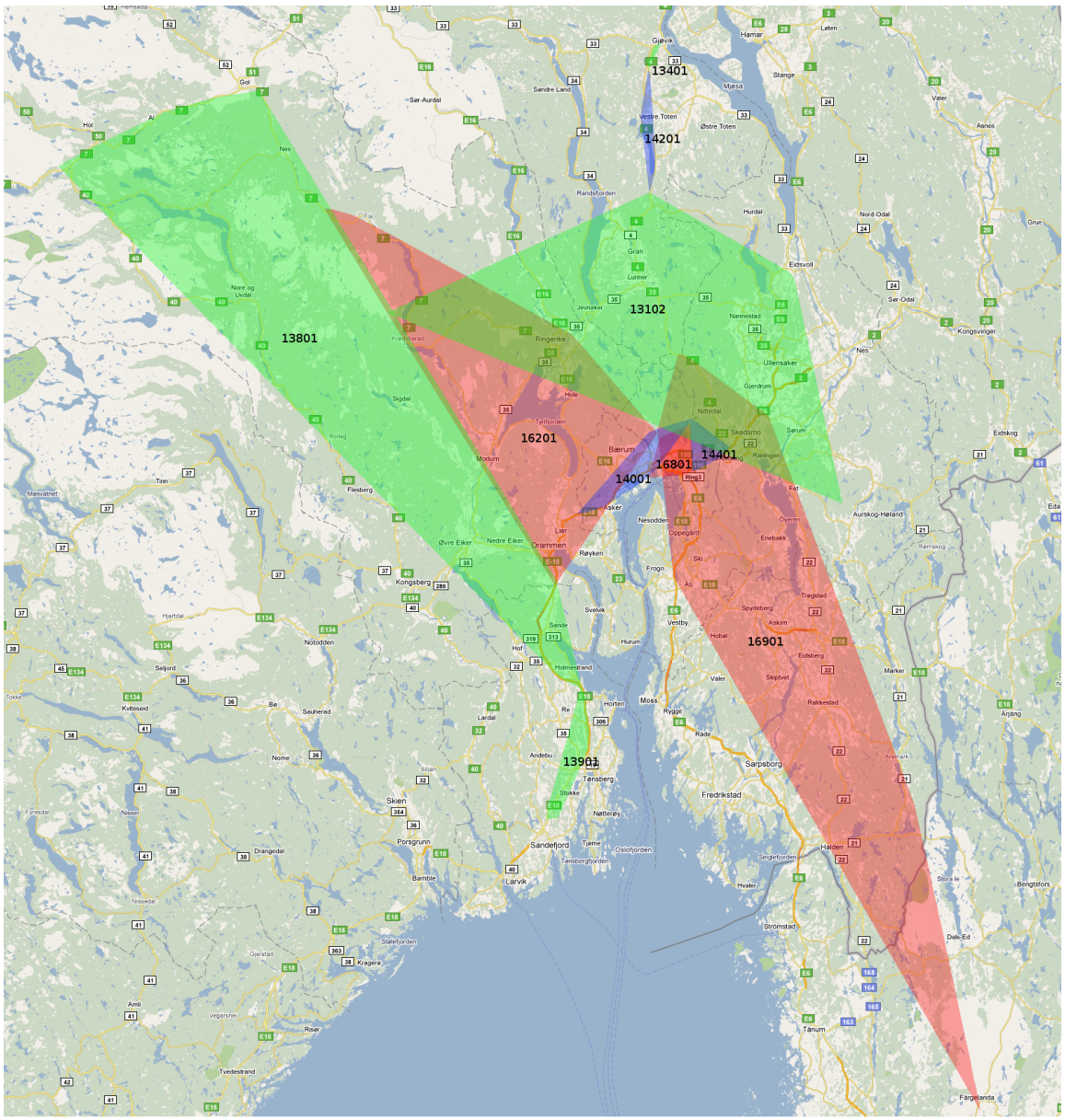


Figure 2.8: Observed Location Areas in the Telenor GSM network around Oslo

CHAPTER 3

PROPOSING A PRIVACY PRESERVING MOBILE LOCATION ESTIMATION SYSTEM

In the introduction we suggested creating a *privacy-preserving*, *community-driven* and *open-source* system for locating mobile devices in networks without the need for a GPS receiver, special hardware, or any support from the network operator as a means of resolving the conflict between LBSs and cloaking services.

People's main motivation for using a cloaking services is to protect their own privacy. If users are to accept a mobile location estimation service independent of LBSs and cloaking services, it must unquestionably protect their privacy.

In this chapter we take a closer look at the requirements for this type of system, and suggest a design.

In this thesis we focus solely on a system based on the traditional server—client model where data is transfered between clients and a server, data is stored on the server, and calculations are performed client-side or server-side. In regards to privacy preservations, other models, where data-storage and calculations reside solely on the client or within the network are of interest, but lie beyond the scope of this thesis. Furthermore, we have not discussed if the calculations are to be done server-side or client-side, but left this as an open issue to be determined after a location estimation method has been chosen.

3.1 Objectives

Looking back at our motivation in chapter 1.1 on page 2 we construct a set of categories of objectives. None of the objectives take precedence over the others; each objective is considered to be of equal importance.

- *Privacy*. The system must ensure that any data stored does not compromise any users' privacy. The system must ensure that any data transfered between the system and any user is transfered in such a manner that the user's privacy is not compromised.
- *Open access*. The system shall provide open and free access for non-commercial use. Data contributed by the users belong to the users.
- *Precision*. The system should use algorithms ensuring the highest possible accuracy on a widest possible range of devices.
- *Accessibility*. The system should be available to the widest possible range of users on a widest possible range of devices.
- *End user perspective*. The system should only use data collected by users. No network operator data is assumed.
- *Data transfer and Storage*. The system must use a data transfer scheme that preserves the quality of the data, ensures high response times and minimizes the amount of data transferred. The system should minimize the amount of storage needed.
- *Quality control*. The system must protect its own integrity without compromising the users' privacy. The quality control system must be able to exclude invalid data maliciously inserted into the system and data from defect devices. Furthermore, the system must handle unintentional random noise and fluctuations.

The objectives are entangled. The solutions to one objective will often restrict or determine the options for objectives in other areas.

Now that we have an overview of the objectives we continue by taking a closer look at each of them separately.

3.1.1 Threats to the System

Before we can discuss preserving privacy within the system we need to define threats to the system. The treats are divided into two groups. First threats to the users' privacy, then threats to the systems reliability in terms of accuracy.

1. an attacker observes a single user over time to derive the user's identity (profiling)
2. user's location is derived by an attacker by observing user over time
3. user's location is derived by an attacker by directly intercepting a message containing enough data to determine their current location
4. observations over time are used to derive an individual's current, past or future location
5. user trending with or without deriving identity (who meets whom for how long and where)
6. area trending (determine popular areas with or without user identities)

1. one user turned attacker hijacks other users updates by intercepting them and providing the system with the update presented as his own
2. attacker forges replies from service
3. attacker supplies service with bad or bogus data

We have identified the following attacks that can threaten the system:

1. eavesdropping on traffic
2. collecting/logging data at the point of destination
3. forging updates
4. forging replies
5. stolen database
6. database released by design

Finally we have the following overview of the potential attackers that represent a threat to the system:

- intelligence
- law enforcement
- cell phone network provider

The attackers mentioned above all have means of tracking users completely independent of the system. By owning and operating a device operating on GSM or UMTS networks one must accept the fact that these three attackers have access to one's location.

- Internet service provider
- LBS providers
- cloaking service providers
- the provider of the location estimation system itself
- unfaithful employees at any of the above
- individuals or groups sniffing network traffic

3.1.2 Privacy

Three conclusions can be drawn when analyzing the lists above: One must protect the traffic between a system and its users, one must protect any stored data and must ensure that no stored data can be traced back to any user. We now formalize and discuss these statements.

The system must ensure full sender anonymity through unlinkability: The system must ensure that a particular message is not linkable to any sender and that to a particular sender, no message is linkable[25]. In addition to providing anonymity unlinkability ensures that "a user may make multiple uses of resources or services without others being able to link these uses together"[45]. Providing unlinkability satisfies our requirement that a user be anonymous within the system. Looking back at the list of threats above we soon discover that this is not sufficient. The most obvious point here is that many of the threats are completely unrelated to the identity of the user. Also note that unlinkability within a system does not guarantee unlinkability outside the system. Even if a message is unlinkable with a user of the system, said message might be linkable to a users true identity.

The system must provide sender anonymity throughout the whole chain. From the instant the message leaves the sender, until it reaches the location estimation provider. The user must be cloaked from any attacker, any third-party the message passes through and also from the location estimation provider.

The system might not provide relationship anonymity as relationship anonymity implies recipient anonymity. This can be seen as an issue of trust. If full sender anonymity is provided through unlinkability, there is no need for a sender to trust the recipient in regards to his anonymity. Since sending data over mobile networks is costly, both in the terms of price and power usage, the issue of trust might arise for some senders who wish to ensure that their efforts are not for nought. From the point of view of the service provider their anonymity might not be important. However there is a possibility that some senders would prefer full anonymity, thus ensuring that they are not identified as users of the service. Normally this issue arises in instances where the use of the service is banned. We therefore leave the question of full anonymity open.

Providing unobservability, where a message is indistinguishable from any message at all, would be sufficient to provide protection from all of the mentioned threats. Unobservability implies anonymity[25]. In addition unobservability ensures that users are unlinkable outside as well as inside the system. If a message is not discernible from "random noise" one cannot retrieve a location from it.

Whether or not the method needs to provide unobservability, that a method is indistinguishable from random noise[25], depends on the method itself. Unobservability implies anonymity, but the price of unobservability is high. Any method providing unobservability is a complex method. Thus any method that can provide anonymity without having to provide unobservability is preferred.

3.1.3 Open Access

Many of the issues we are concerned about, besides privacy, can be overcome by an open system using an open community owned licensing policy. Most open licenses are created such that all of the contributors have a say, and the license cannot be changed without each and every contributor agreeing to the change. However, most open licenses are created for creative works, not collections of data. Extra care must be taken when choosing a license to ensure it is appropriate for collections of data and truly protects not only the data, but also the users.

An open access data system tends to imply extra privacy issues. This has been factored in to our objectives. Looking back at the list of attackers in section 3.1.2 page 25 you see that we have included the provider of the location estimation system as an attacker, to ensure that users' anonymity is protected, not only from external attackers, but from internal attackers as well as the system itself. Furthermore the list of attacks include the database being stolen or released by design.

3.1.4 Precision

We previously determined the privacy requirements of the system. Before it is possible to choose security measures we must know what type of data needs to be transferred and what type of data needs to be stored. This in turn depends on what methods are selected for location estimation and how they are adapted to provide precision.

The system should provide the highest achievable precision without compromising users' privacy. Privacy preservation is to be favored above precision. Furthermore, precision and availability can be opposed. Many high-performance location estimation methods are limited to equipment with access to certain data while most low-performance location estimation methods are widely available. A balance between the two must be found: The highest precision available on the largest amount of equipment.

3.1.5 Accessibility

As stated the system should work with the widest possible range of devices, in the widest possible range of geographical areas and be accessible by anybody. This requirement impacts the choice of location estimation methods. The algorithms used for estimating locations must not only ensure the highest possible precision, but work on the largest possible range of devices.

The solution lies in the use adaptive algorithms or multiple algorithms. The accuracy depends on how much data is available. Clients with a very limited amount of network measurement data receive an estimate, though an imprecise one, instead of being rejected. Clients with access to a large amount of network measurement data receive an estimate based on all of the data, rather than some of the data being ignored.

Therefore, flexible methods that perform well over a wide range of data are preferred; as are different methods that use the same type of data formatting.

3.1.6 End User Perspective

The chosen end user perspective has a major impact on the final design of the system. It effectively limits what type of location estimation methods are suitable; the system is limited to mobile-based estimation methods and other non-network-based methods.

Furthermore, this limits the data that can be used to data that can be collected by users of the system. This generally means that the system must be limited to use data that is obtainable on common mobile devices. However, the system is not limited to common devices; it can be perfectly legitimate to use specialized devices to collect data, as long as they are commonly available.

3.1.7 Data Transfer and Storage

Again this objective affects the choice of location estimation methods; algorithms that generate small amounts of traffic and algorithms that require less amounts of stored data are preferred.

Securing the storage and transfer of data to ensure users' privacy and anonymity is important. It is, however, not possible to determine how this impacts the system until storage and data transfer methods have been chosen, which in turn cannot be determined until location estimation methods have been chosen.

What we already do know is that, in general, the amount of data transferred can impact measures in place to enforce a user's anonymity, like encryption; the more data transferred, the higher the load on the encryption system is. Furthermore the frequency data is transferred can also impact the system; the more often data needs to be transferred in either direction, the higher the load is. We also know that the more data needs to be stored on the server, the more care needs to be taken to ensure that the data cannot be linked to an individual.

One final point needs to be made about the storage of data with regards to choosing location estimation methods; less granularity can lighten the load on the security measures. The more

details stored, the easier it is to infer the data back to an individual. We illustrate this with an example: An individual owns a blue car—is much less likely to be inferred back to an individual than—an individual owns a blue car, lives somewhere on main street, likes steaks for lunch and works at an airport.

3.1.8 Quality Control, Trust and Incentive

A community sourced location estimation system cannot be without quality control. When users provide data fundamental to the location estimation the need arises to ensure that said data is valid.

Invalid data can potentially corrupt the whole location estimation system and render it useless. Such corrupt data can stem from several different sources, some intentional, others unintentional; a malfunctioning device can record incorrect network measurements or GPS readings; a malicious user can intentionally contribute fake, malformed or altered data, so can competitors or anybody with a beef against the system.

We will not dwell too long on this issue as an ultimate decision cannot be reached before the location estimation method and data storage method are in place. How one can successfully achieve quality control without breaching users' anonymity cannot be fully determined until the type of data to be controlled is determined. We would, however, like to suggest a quality control method we presume works well in an open, community sourced, privacy preserving location estimation system.

A simple way to achieve a medium level of quality control is to keep a history of users' contributions, and trust users based on their previous achievements. In its simplest form this would involve quarantining data from new users until they have contributed a certain amount of data deemed to be valid. This means keeping track of users.

Keeping track of users can potentially be extremely dangerous in a privacy preserving system. In the very least one must ensure that no critical data can be linked to a user, perhaps by simply maintaining a counter of how many successful contributions a user has made.

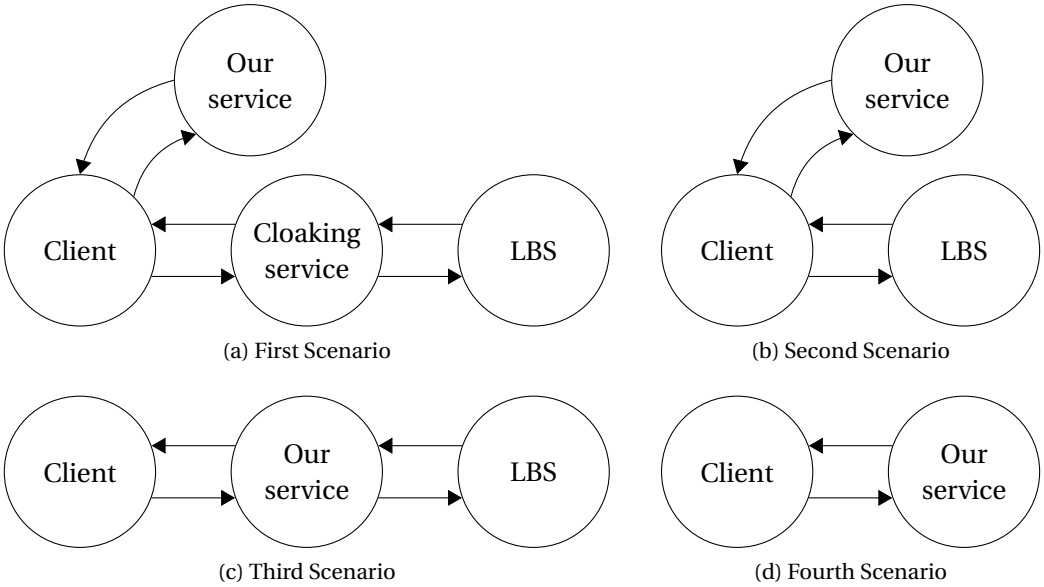
In GSM and UMTS networks, users can be uniquely identified by their International Mobile Equipment Identity (IMEI). Unfortunately an IMEI can easily be traced back to an individual. To avoid this one would use a cryptographic hash function on the IMEI before transmitting it off the users device. This would allow the location estimation provider to uniquely identify users, without being able to link them to identities.

In addition we suggest, rather than a simple counter, using a digital wallet to rate users. Each time a user submits a valid, usable contribution, the user is payed for their contribution into their digital wallet. When determining trust in a user, for example to decide if their contribution should be quarantined or not, the cumulative value of their wallet is checked. The more "money", the higher the trust.

Using a digital wallet in this manner solves one of the larger issues of this type of community sourced system; how does one provide users with incentive to contribute to the system. Why would users with GPS, a great means of mobile positioning, bother to contribute measurements?

As mentioned, GPS has its weaknesses; GPS cannot be used indoors or in areas with a very obstructed view of the sky, and can be shut down by government at any time. In addition many users choose to only turn on their GPS when they absolutely need it. Hence many, if not all, users of GPS need other location estimation services from time to time.

Therefore, a digital wallet can be used to encourage contributions. Users could be given an initial sum of "money" and charged for using the system. In addition they would be payed for



(a) First Scenario (b) Second Scenario

(c) Third Scenario (d) Fourth Scenario

Figure 3.1: User scenarios of a open, privacy preserving estimation system

contributing to the system. Hence using the system more than a pre-determined amount would require contributions.

To avoid the system of trust and the system of incentive working against each other, two wallets are needed; one wallet to establish trust and a second wallet to pay users for contributions. If a single wallet were to be used, using the system would demote the systems trust in a user. Not trusting users because they use the system is in no way logical.

If care is take to avoid any link between actual location data in the database and any user data, in addition to not including user data in the data openly released into the wild, this scheme does not pose a threat to users anonymity. The scheme may, however, create issues when selecting a method to protect users from eavesdroppers on the communication link.

Before moving on we have two more issues we must address: How do we intend the system to be used and how should the system gather data?

3.2 How the System Should Be Used

The system should be quite flexible as the main goal is to separate what normally has been perceived as nested functions into separate independent entities. Users can use the service in several different ways. A user might retrieve a location estimation from the service, and request service from an LBS through a location cloak of choice (see figure 3.1a page 29). A user could choose to retrieve a location estimation from us, and send this estimation directly on to an LBS without cloaking it (see figure 3.1b). It would also be possible for the service to act as both the estimated location provider and a cloaking service between the user and an LBS (see figure 3.1c). Finally a user could use the service to retrieve their estimated location for other reasons that using a LBS (see figure 3.1d). This would normally imply that the user wants to process the estimated location locally, for example when mapping personal movements, geocaching, or using locally stored map software.

3.3 Gathering Data

We have identified and investigated several different schemas to gather data. These schemas can be generalized into four different mechanisms.

3.3.1 Direct Data Upload

Direct Data Upload is the "traditional" way of building a location database. Here one simply sees the update and query mechanisms as two separate features. Clients wishing a location estimation submit a query, and receive either a location estimation or data providing them the means to estimate their own location. Clients who wish to contribute simply send the system their current physical location together with a fingerprint. When, how often and in what contexts updates are provided is not specified. Clients could provide an endless stream of data, one could require users to provide a certain amount of updates before they are allowed to use the query function, or one could even covertly collect updates without the users knowledge or permission.

The main disadvantage of this method is that there is no way of predetermining if an update is required or not. Thus duplicates are continuously transmitted to the server generating a much higher level of network traffic than needed.

If one were to introduce the suggested digital wallet—rewarding users for contributing to the system—this data gathering method introduces a catch-22: Either one has to reward users that contribute data one cannot use, or users cannot pre-determine if their contributions will grant a reward. There is no way of knowing if a contribution enhances the system before it has been contributed.

3.3.2 Pre-Populated Database

It is possible to pre-populate a database using either network planning tools or known radio propagation models. The first requires a prior knowledge of the three dimensional topography of the area, and both requires a prior knowledge of the exact location of BTSs.

Access to this type of data in all areas where the system is used cannot be assumed. Such data is also generally commercial, and not compatible with a platform of openness.

3.3.3 Amending Queries — Submitting Fingerprint

Amending queries — submitting fingerprint is one of two methods where there is no distinct separation between a client querying the system and a client updating the system. Instead any client querying the system can also update the system by providing any data the system is missing, provided they have their GPS-radio turned on and working. The client sends a query containing a fingerprint, the system uses said fingerprint to calculate their estimated location, and returns this location to them. If the client has a GPS-fix he can inform the system of his current true location based on the GPS-radio, and the system can use this true location to amend the database. This method relies on the system either retaining a copy of the original fingerprint, and somehow linking it to whatever update it later might receive, or the client re-transferring the original fingerprint with their update.

3.3.4 Amending Queries — Submitting Measurement

Amending queries — submitting measurement is the second method where there is no distinct separation between the client querying the system and the client updating the system. This method relies on the actual calculation of the location estimation to be performed on the mobile station. The result is less network traffic since the need to re-transport the whole fingerprint can in many cases be eliminated. Only the parts of the fingerprint missing data needs to be re-transported. Determining if an update is needed is much easier. When using the first method the system can only determine that the mobile station's physical location differs from the estimated location. Using this method the system can simply investigate each and every measurement in the fingerprint to determine if an update is needed. This results in much less network traffic, and this method also allows for a completely fair digital wallet system where the user can predetermine if their update is beneficial and will grant them an award.

3.4 Bootstrapping

In a system that only relies on data collected by users, and users do not supply raw data the problem of bootstrapping a new area arises. When a contributor enters a completely new area, an area the system has no data on, there is no way for the system to provide a user with an initial estimate.

We looked into several different methods of bootstrapping an area. One method could be to allow trusted users to supply data on new areas directly. Unfortunately this method introduces several problems. This complicates not only the interface to the database, but also the cloaking and methods of communication between clients and us. In addition new users, i.e. users who have not earned the systems trust yet, cannot supply data for new areas. Any new user who resides in an area the system has yet to collect data on, cannot use the system until a user who has earned the systems trust happens to enter it and supply initial measurements. The systems coverage only expands as trusted new users "pioneer" new areas. Finally this means that the system must have an interface for supplying data directly in addition to a query/update-system.

A second method of bootstrapping an area could be a system that verifies data similar to the system BOINC[1] uses where each processed unit must be done independently by two clients before it is accepted. Other than the obvious disadvantage of needing at least two active clients to bootstrap an area this method has a major drawback. This method requires that the system ensures that both users are unique, so that a single malicious user cannot game the system by sending multiple updates. However a system that can provide verification of such uniqueness, violates users anonymity. Furthermore, since BOINC is offloading workloads, they choose which users receive what work; a single workload can be offered to one trusted and one untrusted user. The location estimation system, on the other hand, is not offloading workloads, but receiving updates observed by users, and can therefore not pick and choose; this type of bootstrapping implies one would have to accept an update from two untrusted users as a safe update.

A third method would be to request the information needed for bootstrapping from existing service providers. Sending a single request for information on a single WLAN-network or GSM/UMTS-cell in a new area, and retrieving their results would be sufficient to bootstrap a new area. Such queries could either be done preemptively in bulk or on demand, i.e. whenever a user queries from within a new area. This method could even be extended to supply mean-

[1] http://boinc.berkeley.edu (2010-04-07)

ingful location estimations even in areas where the system does not have sufficient data to do this, by simply querying an existing service provider any time it does not have sufficient data to respond accurately to a query. However, this would be a complete violation of our fundamental platform of free and open data. The downside of this method is that the system would be relying on one or more third parties to supply bootstrapping information. Thus any changes said third parties might implement may impact the system. In addition any changes to their licensing policy could also impact the system. However, such changes only affect the systems bootstrapping mechanism, not the system as a whole. Finally one must be very careful when selecting such third party to make sure that one is not in violation of their terms of use.

Fortunately there are many[2] community driven services that collect wireless network measurements, both from GSM networks, WLAN networks, and other wireless networks, releasing them under the creative commons share alike licence[3]. Using these services for bootstrapping would not introduce licencing problems; the main disadvantage is that their coverage is far from complete.

3.5 Summary

In this chapter we proposed a privacy preserving mobile location estimation system as a means of solving the conflict between cloaking services and location estimations services, and providing individuals the privacy they deserve.

We laid out a set of objectives for the proposed system and determined threats to, and attackers of, the system, before closely examining the individual groups of objectives.

Finally we investigated other aspects of the proposed system including how we expect the system to be used, how the data should be transfered and stored and how the system should bootstrap and collect data.

Through suggesting a privacy preserving mobile location estimation system we discovered that before implementing such a system, and determining the details about storage, transfer, calculations and how the system should protect users' anonymity and privacy a location estimation method must be in place. The rest of this thesis focuses on finding the location estimation method best suited for a open and privacy preserving mobile location estimation system.

[2] E.g. opencellid http://www.opencellid.org (2010-04-19) and openBmap http://www.openbmap.org (2010-04-19)

[3] http://www.creativecommons.org (2010-04-19)

CHAPTER 4

THE *Intersecting Areas* LOCATION ESTIMATION METHOD

In this chapter we introduce our own location estimation method developed for, and tuned towards, open and privacy preserving mobile location estimation services. The method was named *Intersecting Areas*.

The *Intersecting Areas* method is designed to eliminate the need to store sensitive user-data. Rather than storing crowd sourced network samples directly, the *Intersecting Areas* method uses samples to update and maintain a training set stored in such a way that no data is linkable to any user. Furthermore the *Intersecting Areas* method limits the amount of crowd sourced updates needed.

First we describe the motivation behind developing a new location estimation method, why a new method is needed to create privacy preserving and open estimation services. Next we walk through the process of creating the method. We look at all of the considered options; investigate which options were selected, and why. We describe the intersection areas method in detail. Finally we look at the advantages of the method.

4.1 Motivation

The last decade much effort has been put into improving, and discovering, new mobile location estimation methods. This work has been driven by a desire for better precision, but also the need for universal systems that perform well, not only in the widest possible geographical area, but also on as many devices as possible.

A lot of work has also been done to protect users' privacy and anonymity when using LBS. But little work has been done regarding protecting users' privacy and anonymity when using location estimation services, and optimizing location estimation for privacy preservation.

When designing an open, privacy preserving mobile location estimation method we soon discovered how tightly connected privacy preservation is to location estimation methods; the frequency and type of data stored and transferred impacts how and which privacy measures can be implemented; the choice of estimation method affects the choice of privacy preservation method; not all privacy measures are interchangeable with all estimation methods.

We concluded that there is a need for these issues to be addressed, and that existing and new location estimation methods should be analyzed in terms of privacy preservation. In addition, location estimation methods focusing on privacy preservation should be created.

4.2 Inspiration

To create a location estimation method focusing on privacy we started by addressing the amount and frequency of data transferred and amount and granularity of data stored. A privacy preserving estimation method should minimize the amount and frequency of data transferal. The amount of data stored should also be kept to a minimum, but more important the granularity of the data stored should be as low as possible. High granularity of data makes it easier to link stored data back to a user[1].

In terms of granularity one of the best existing algorithms is the CGI algorithm. Only the identification of a BTS and the observed coverage of said BTS needs to be stored. CGI also is good when considering data transfer amount and frequency. The only uplink data needed is the id of the serving cell, and the only downlink data needed is the area the given cell covers.

A well implemented user sourced CGI system needs a much lower frequency of updates than other location estimation systems. Querying the system can be used to determine if an update is needed. If the updater's true location is outside of the area it received as a location estimate, an update is required. The granularity of an update in a CGI system is also low; the only data needed is the contributers true location and the ID of the serving cell.

The achievable precision using CGI is at the very low end of the scale. In terms of accuracy and precision, CGI is the worst performing of the location estimation methods.

In terms of granularity the worst existing algorithms are the different DCM techniques. Using DCM involves collecting as many fingerprints as possible with as much detail as possible in each fingerprint. Thus the only limit to the granularity is the sample frequency available on the device providing an update.

Furthermore, the frequency and amount of data transferred when using DCM is extreme; the more updates provided, the better the algorithm becomes. When investigating the amount of data stored, things get even worse; there is no limit to how much data is stored whatsoever.

[1] For example: *An individual owns a blue 1997 Ford Escort* is more linkable than *an individual owns a blue car*

Concerning precision DCM is the best of the best, only surpassed by DCM extended using statistical methods. We already know that using statistical methods on stored data to improve location estimation methods is huge problem for an open, privacy preserving location estimation system and would render the task of guaranteeing the users anonymity near impossible. Any data free to be released into the wild must be able to withstand statistical attacks.

So DCM and CGI are on either ends of the scale in. This made us ask ourselves: Can we somehow combine the two to gain the pros of both while losing the cons?

4.3 Combine and Conquer

A community sourced CGI location estimation system could work in the following way: Each unique cell is given one entry in the database. Connected to each entry is an area; the area surrounding all of the observations of the cell.

The area is returned to any device requesting an estimation based on that cell. If the device has a GPS unit and determines that its physical location is outside of the area it received, it should inform the system of its physical location so that said location could be added to the stored area.

The method can be extended to utilize the received radio signal strength quite easily: Instead of a single entry for each cell, an entry for each cell–signal strength pair would be used, each entry with its own area. This is now a community sourced E-CGI location estimation system.

The *Intersecting Areas* location estimation method starts with this concept of areas.

We want to extend this concept to encompass the strength of DCM. The strength lies mainly in two factors: the amount and richness of data.

4.4 Introducing the Area

We eliminate the use of fingerprints on the server. In the *Intersecting Areas* method fingerprints are only used to transfer network samples from network equipment to the server. On the server fingerprints are replaced by areas.

We define an area as *a geographical two-dimensional region corresponding to a single unique datum of a network sample that can be linked to physical location.*

An area can consist of the ID of a single serving cell in a GSM or UMTS network, the combination of the ID of a single serving cell and received radio signal strength from the cell or the ID of a *single* neighboring cell with or without received signal strength information. Further, an area can consist of a combination of the ID of a single serving cell combined with a TA, or any other NMR, sample of the cell.

An area is not limited to samples of GSM and UMTS networks: The unique ID of a WLAN can be an area. The same unique ID combined with received signal strength can be an area. Samples of any wireless network can be used, as long as the access points in the network can be uniquely identified.

A sample of received signal strength on its own cannot be an area. Nor can a TA or any other NMR sample on its own be considered an area. These samples violate the requirement of uniqueness, since the same samples can be observed in multiple access points. Furthermore, they cannot be linked to a physical location on their own.

4.5 Characteristics of an Area

An area starts out as a single point corresponding to the first location the sample linked to the area was observed. Any area with less than three points is an *orphan*. Orphans are not true areas, and do not contribute to a location estimation. Only when an area matures to at least 3 points is it included in location estimations.

Areas are mutable. They evolve over time as more samples are contributed to the system. Users contributing to the system, ask for a location estimation as normal. In addition they have GPS activated. If their GPS-based location is outside of the estimated location area returned from the server, they send an update to the server informing it of the discrepancy. The server then updates the area including the new location.

Updating areas in this way implies that areas grow rapidly in their infancy. As more samples are contributed the growth declines, the frequency of updates drops, and successive updates contribute to defining the borders of the area more accurately rather than defining the area itself. This is in direct contrast to standard DCM methods where the need for updates never decreases.

Areas differ from standard DCM methods in two other beneficial ways. When using DCM there is no upper limit to the amount of fingerprints stored on the server; the more fingerprints stored the better. In the *Intersecting Areas* method areas are only stored once. This means that any unique sample datum is only stored in the database one time independently of how many times it has been observed and how many fingerprints have been transferred containing the datum. Furthermore, the concept of areas removes the linkability between users and entries in the database. DCM relies on storing individual fingerprints, each connected to a single point of location. This means that in a community sourced DCM system each individual entry in the database has been contributed by a single user. When using our concept of areas, no single entry in the database is connected to a single point. Furthermore, no single entry in the database is connected to a single user; any entry in the database, other than initial areas consisting of a single point, are generated by combining data from an unknown number of users. While DCM relies on storing data contributed by users directly, *Intersecting Areas* relies on adding the users' contributions to an existing set of data, and only when needed.

4.6 From Database to Location Estimation

We now have a data storage and retrieval model that combines the simplicity of CGI with the richness of DCM. The next step is to determine how the data should be used. How do we go from database entries to a concrete location estimation?

Determining what data to use to calculate the estimated location in the intersected areas method is trivial. Since each datum is stored as a value–area pair in the database the following method is used to determine what data to base the calculation on:

Listing 4.1: Retrieving Data from Database

```
foreach datum in fingerprint:
    retrieve corresponding area from database

    if datum not in database:
        request update from client
        add datum with empty area
```

The result is an array of areas. These areas should now be converted to a single location estimation.

The simplest solution would be to return the array to the client, provide a standard API for converting the areas, and allow the client to convert them to a location itself. Doing this would naturally increase the frequency and amount of data transfer needed, and would still require us to determine the standard conversion method.

The advantage of doing the calculation client-side, seen from an open, community driven perspective is that anybody would be free to implement their own superior algorithms. In addition, client-side calculations simplifies the process of updating the system: Since a client receives a list of areas, they can determine exactly what areas are missing a location and send updates only for said areas.

From a technical point of view the location where the calculation is performed is not important. For these reasons we do not specify where the calculations are to take place, but rather suggest how the calculations can be done.

4.7 From Multiple Areas to a Single Area and Point

As we saw in the previous section, the raw data retrieved from the database when analyzing a fingerprint to calculate a location estimation is an array of geographical areas. This collection of areas must be converted to a single location estimation.

The first step is to find the intersection of the collection of areas. Any datum containing less than 3 points is ignored as it is not an area (yet). The intersection of the remaining areas is then calculated.

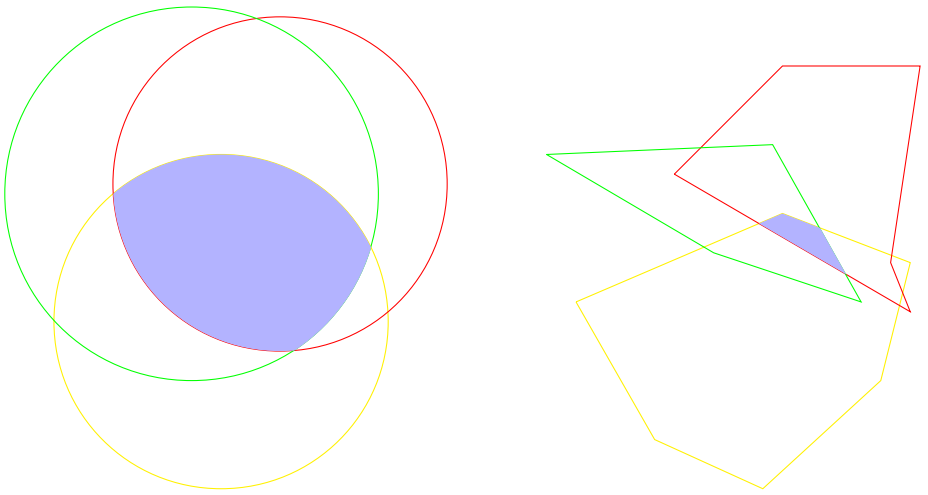
A problem arises if there are non-overlapping areas in the collection. This exception can be handled in multiple ways: One can simply disregard any non-overlapping areas on a first-come-first-serve basis. One can disregard any non-overlapping areas based on a weight—like the size of the areas or the number of points an area consists of. Or one can calculate multiple areas and choose the final area based on the number of areas intersected to create it, the size of the area, or the number of points the area is created from.

When all of the relevant areas have been compared, the resulting intersection is the estimated location area. The “center” of this area is then derived by averaging each extreme point of the area. This "center" point is the estimated location. The client receives both the estimated location area and the estimated location point, and can determine how they should be interpreted.

4.8 Calculating and Storing Areas

Most services we have investigated returned a circular area with a given radius as an estimated area. There are two ways of interpreting such a circle: The first way is to interpret the center of the circle as the estimated location, and the radius of the circle as the estimated uncertainty. The second way is to interpret the center of the circle and the estimated location of the BTS as the circle as the estimated coverage.

One could store each area in the database as a circle with a given radius, and use the intersection of the circles as an estimated location area, and the center of the intersection as an estimated location. Figure 4.1a page 38 illustrates this use of circles.



(a) Circles (b) Convex Hulls

Figure 4.1: Circular Intersection vs. Convex Intersection

Using circles, however, has one major disadvantage; surrounding points with a circle can result in fairly large areas of false positives. Think of a scenario with only two samples 500 meters apart. The system would then estimate a coverage area consisting of a circle with a radius of 250 meters and a center located between the two points; two single points results in a coverage area of 0.2km².

To improve the accuracy we suggest using convex hulls surrounding all of the samples for a given area. Figure 4.1b page 38 illustrates this use of convex.

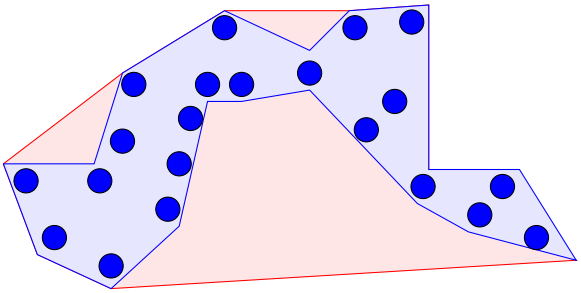


Figure 4.2: Area of false positives when using convex hulls

Convex hulls improve the accuracy of the method and, besides increasing the amount of storage space needed on the server for an area, have all of the advantages of using circles: Calculating the intersection of convex hulls is not complicated. No extra data transfer on the uplink is needed, and the frequency of data transfer stays the same. The amount of data on the downlink stays the same if the calculations are done server-side, but increases slightly if the calculations are done client-side. Finally, the granularity of the data on the server stays the same.

4.9 Further Improvements

In some instances an area calculated using convex hulls can result in areas of false positives in a similar way as circles—see figure 4.2 page 38—though the areas of false positive would be much smaller.

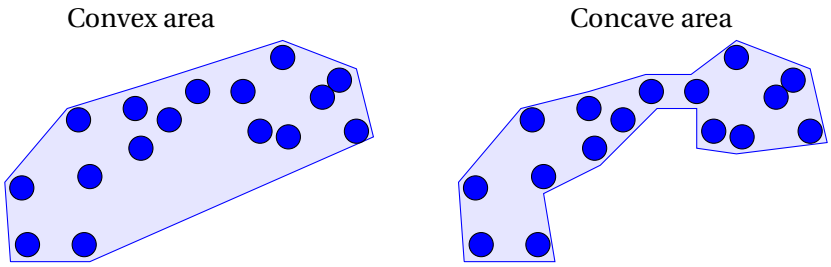


Figure 4.3: Convex vs. Concave area

4.9.1 Concave Hulls

One can improve on the accuracy of convex hulls by using concave hulls (most likely generated using alpha shapes[46]). Figure 4.3 page 39 illustrates the increased precision when using concave hulls instead of convex hulls, while figure 4.4 page 39 compares convex hulls, concave hulls and assumed free space propagation.

The problem with concave hulls is that they are extremely complicated compared to convex hulls. In addition the algorithms used to calculate concave hulls are not static. The simplest way of calculating concave hulls is to draw straight lines between each pair of closest neighboring extreme points. A more accurate, but more complex, method draws curves through each pair of closest neighboring extreme points. Both of these algorithms have tunable parameters, like the maximum amount of curves or lines, how bent a curve can be, etc. Naturally how precise the area is calculated relies on how well the parameters are tuned. Tuning the parameters must be done manually and the best values varies from instance to instance.

While a convex hull of a given set of points can be considered an absolute, a wide range of different concave hulls exists for the same set of points. Using concave instead of convex hulls would improve the accuracy of the *Intersecting Areas* method—to a large extend in some situations—but the price is high; using convex hulls would result in much more complex and fragile system.

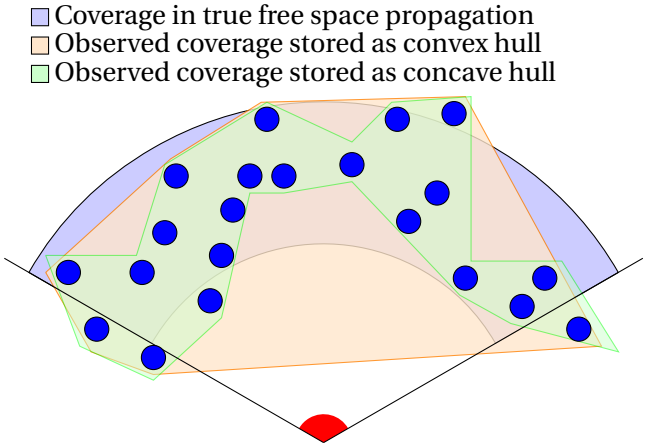
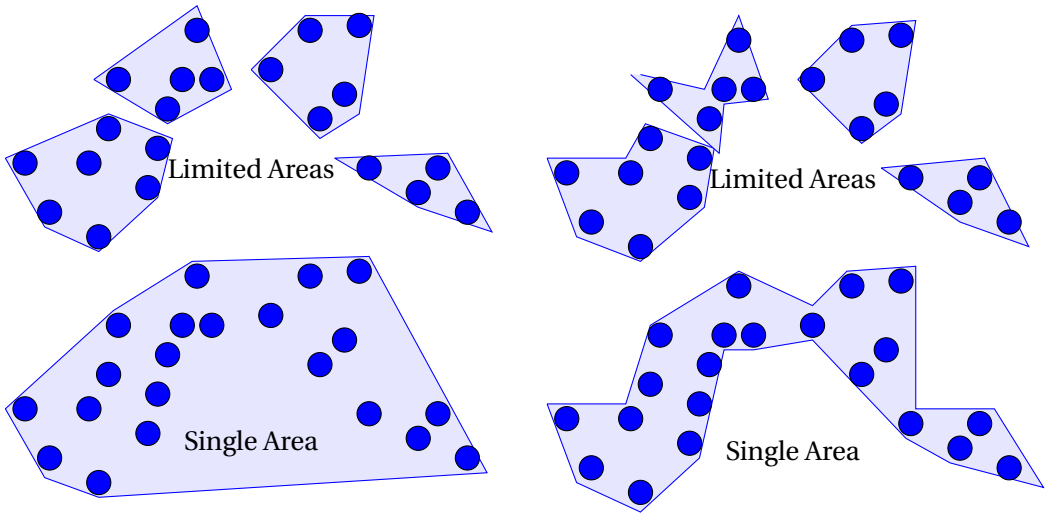


Figure 4.4: Assumed layout of a normal cell at a certain rxlevel

When dealing with larger areas, we assume concave hulls generally outperform convex hulls, at least in areas where one comes close to free space propagation. This is especially true since one stores separate areas for each received signal strength value. We cannot reach the same conclusion areas with a lot of Rician fading or Rayleigh fading—fading caused by multipath interference. In such areas the best performing method varies from cell to cell.

The only difference between using concave hulls and using convex hulls is the calculation algorithms. In terms of amount and granularity of storage and amount and frequency of updates the two are equal.



(a) Limiting convex hulls

(b) Limiting concave hulls

Figure 4.5: Limiting areas to improve accuracy

4.9.2 Limiting Areas

A different way of improving the method is to limit how large an area can be, by splitting the area into smaller areas as it grows. Areas could be limited to a maximum surface area or a maximum distance between any two points. Technically this would involve allowing a single datum to be connected to multiple areas, with no limit to how many areas can be connected to a single datum.

Figure 4.5a page 40 illustrates limiting the area size when using convex hulls. The method is not limited to convex hulls and can also be applied when using concave hulls, as figure 4.5b illustrates.

We expect that limiting areas to a smaller size, when used on larger cells, could improve precision compared to using convex hulls alone.

When limiting areas, the performance difference between convex and concave hulls should be small, as long as the area size limit is small enough. Combined with the fact that calculating concave hulls is much more complex and fragile than calculating convex hulls, limited convex areas is a much sounder choice.

Limiting areas increases the amount of data transferred if calculations are done client-side; calculating on the server side does not. The frequency of data transfer stays the same as when only using concave or convex hulls. The amount of data stored on the server increases, but the granularity of the data does not.

4.9.3 Fall-back to E-CGI and CGI

Since the *Intersecting Areas* method is based on areas like CGI and E-CGI it can be extended to fall back on these simpler location estimation methods when lacking enough data to successfully estimate a location. Fall-back to E-CGI is trivial to implement and requires no additional processing or data-storage, as the same areas used for E-CGI are also used for *Intersecting Areas*. Fall-back to CGI, however, requires additional data-storage and processing since *Intersecting Areas* does not use areas calculated from serving cell observations without any additional datum.

4.10 Error Estimation

Since mobile location estimation systems are exactly that— *estimation* systems—adding an indicator of error enhances how they can be used; providing a measure of how accurate the estimation is allows the client application to make sounder decisions on how to use the data. A simple example is a map application determining if it should show the users location as a point or an area based on the error estimation. A more complex example is a location based game that grants users points for finding certain locations. It could adapt the gameplay based on error estimation; a high error estimation would require the user to stay within a certain area for a certain time, while a low error estimation would simply require the users to reach given point in space.

We have identified three simple error estimations within the *Intersecting Areas* method. More complex error estimation could be achieved by statistically analyzing how a live system performs over time.

The first simple error estimation involves reporting the difference in how many of the samples in the submitted fingerprint do not correspond to entries in the database. The *error score* is increased for every sample present in a submitted fingerprint that the server has no record of.

As we saw in the previous section some areas can be non-intersecting. The number of non-intersecting areas can be used as an error estimation.

The third error estimation involves counting how many contributions have been made for each area used to calculate the location estimation. The more times an area has been updated, the surer we can be that the boundaries of the area are accurate.

If the date of the last update is stored, the contribution-count value can also be used to determine how fresh an area is. The inverse of the value—number of queries that did not result in an update—together with the last date of access provides an even more accurate estimation of freshness. An indicator of freshness would be used to avoid stale information, caused by moved or discontinued access points, from contaminating the database.

When implementing this third error estimation method, the word *counting* is critical. This feature must be implemented as a simple counter of the number of updates of each area. Storing fingerprints or updates instead of a counter would undermine the very reason why we created this location estimation method; *to ensure anonymity and privacy*.

The update counter can also be used to enhance the location estimation process itself. When dealing with sample-rich fingerprints an update counter can be used to determine one or more samples that should be ignored due to their accuracy. For example in a situation where the system is presented a fingerprint containing samples of 17 different WLAN—two of which have only a few updates in the database, the other hundreds—the system should ignore the two areas with only a few updates. In most situations these two areas would add nothing to the precision of the estimation, and in some situations they could actually harm the precision; they should be ignored.

None of these error estimation methods affect the relation between data and privacy much. All three of them slightly increase the amount of data transferred on the downlink, but not the frequency. The uplink is unaffected. None of them affect the granularity of the database, while the third method increases the amount of stored data slightly.

4.11 Privacy Preservation

Part of the motivation behind creating this location estimation method was to create a precise and flexible estimation method that focuses on simplifying the task of ensuring users' privacy and anonymity. So what are the benefits of this method with regards to privacy and anonymity, and what is the benefit of using the method when creating an open system?

Any system that relies on processing or storing location data on a central server is vulnerable to eavesdropping and statistical attacks on transferred data. At some point location data, or data used to calculate a location must be transferred. The easiest well known way of dealing with this is encryption.

Encryption is generally expensive in terms of CPU-usage and transfer speed. Reducing the amount and frequency of data transfer does not in itself provide added privacy, but the reduction does help reduce the negative effects of encryption. Smaller amounts and less frequent data transfer means less CPU-usage for encryption, and, when encryption time is correlated to the amount of data encrypted, less data in each bundle of transferred information speeds up the transfer. The *Intersecting Areas* method reduces the amount and frequency of data transfer on the uplink when users update the system massively compared to DCM. When a user is querying the system as normal the amount and frequency of data is compared to DCM the same.

As mentioned a central location estimation method alone cannot negate the need to protect the users' transmission of data from eavesdropping and statistical attacks. What the location estimation method can provide is a reduction of granularity of the transferred data. In the event a single message is compromised, the less the granularity of the message is, the harder it is to link the message to a given entity. When comparing the *Intersecting Areas* method to DCM the granularity of updates is drastically reduced, while the granularity of regular queries and replies stays the same.

We see that while the *Intersecting Areas* method comes with some benefits when ensuring the privacy of data transfer in the system, it is as a whole not a big improvement over DCM.

Where the *Intersecting Areas* method really does come to it's rights is in the data storage department. Encrypting and protecting data transfer is a widely researched area. Instead of mainly focusing on data transfer, the *Intersecting Areas* method focuses on the combination of anonymity and openness.

Protecting anonymity in a community based service demands that great care is taken to secure any stored data from statistical attacks, and any information entered cannot be traced back to individuals.

The *Intersecting Areas* method addresses this in multiple ways. Both the amount, frequency and granularity of stored data is greatly reduced when compared to DCM. No individual user contribution is ever stored in the database as such. A user contributing data is only contributing a single point to a collection of points connected to a single unique network sample, and the point is not connected to any user in any way. In DCM, on the other hand, a user is contributing a location and a multitude of network samples each and every time they contribute to the database, and the updates are individually stored as is in the database.

In addition, the *Intersecting Areas* method immensely reduces the frequency at which data from any user is stored in the database. While DCM thrives on updates and by nature should store any data it receives in the database, the *Intersecting Areas* method only stores data when it is absolutely needed. And data is only absolutely needed when a device observes a unique network sample somewhere outside of the area said network sample has previously been observed.

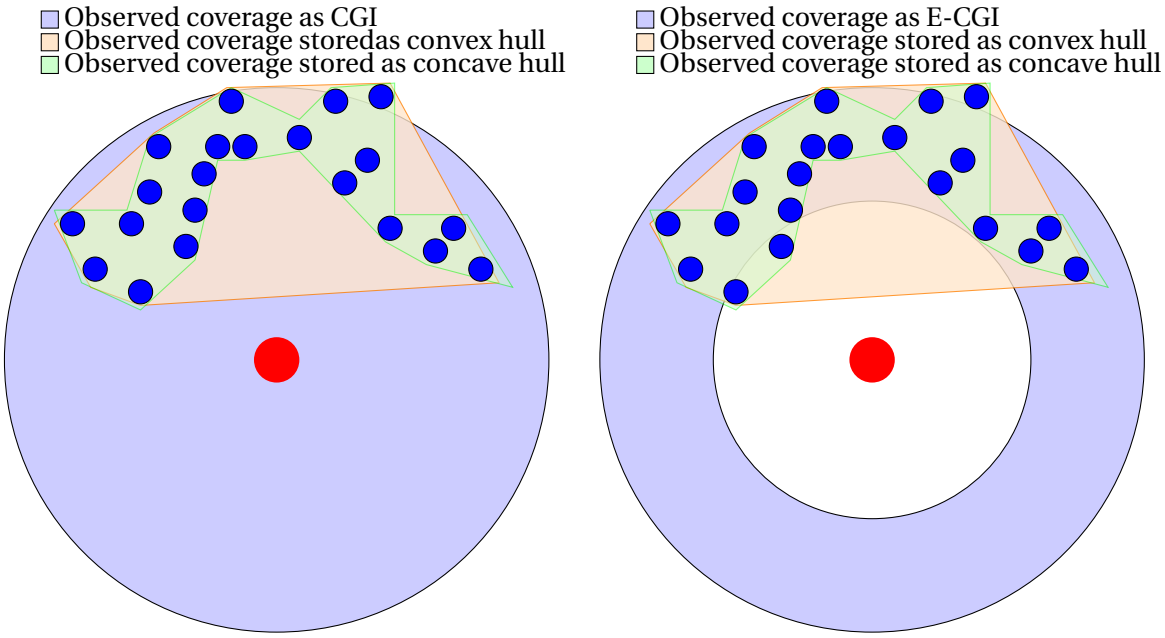
Through this, the *Intersecting Areas* method can be used to create a community sourced loca-

tion estimation database free of data linked to individuals; a database that can freely be released in the wild without compromising the anonymity of the contributors.

4.12 Summary of Benefits

We gave ourselves the task of creating a location estimation method that captures the simplicity of CGI and E-CGI with the strength of DCM while ensuring that the estimation method works well with an open, privacy preserving location estimation service.

The *Intersecting Areas* method is based on the principles of CGI and E-CGI, inheriting much of their simplicity. The whole system is based on areas, in the same manner as CGI and E-CGI, but allows the use of other types of areas and uses the intersection of multiple areas to improve accuracy. Figure 4.6 page 43 illustrates how the method compares to traditional CGI and E-CGI where the location of the BTS is known.



(a) Compared to CGI

(b) Compared to E-CGI

Figure 4.6: Comparing convex and concave limiting to CGI and E-CGI using received signal strength

This location estimation method is extremely flexible, and can handle any type of network samples, as long as the sample is related to a network or network node. We initially assume the use of GSM or UMTS serving cell identity, signal strength of serving cell, neighboring cells, signal strength of neighboring cells, and the identity and signal strength of WLAN networks. However, it is trivial to extend the system to handle any other type of samples such as TA, any type of NMR, or bluetooth networks.

In addition, *Intersecting Areas* can fall back on E-CGI or CGI when not enough network measurements are available. Thus it can offer the success rate and mobile station availability of E-CGI and CGI.

Furthermore, the *Intersecting Areas* method is flexible in terms of what network equipment can use the system; the system handles a request containing only samples of the serving GSM cell in exactly the same manner as a request containing a full range of neighboring cell information, WLAN samples and full NMR samples. All data is stored separately, but in exactly the same format. Neither the storage, nor the algorithms themselves need to be changed or tuned to special cases.

Comparing the *Intersecting Areas* method to CGI and E-CGI, *Intersecting Areas* should prove increased precision in a manner close to DCM.

Compared to DCM the *Intersecting Areas* method has several advantages: The amount and frequency of data transfer is lower. The amount of data stored on the server is exceptionally lower. And due to less CPU-usage and lower storage and memory usage we expect better response-times. Furthermore, the granularity of the data stored on the server is much lower than when using DCM, not to mention when combining DCM with other statistical methods. The result is a system that stores and transfers much less data critical to privacy and anonymity, lightening the burden of implementing security. When comparing *Intersecting Areas* to DCM, in terms of ease of securing privacy and anonymity, we observe the following:

- the amount, and frequency, of data transferred on the downlink is much the same
- the amount, and frequency, of data transferred on the uplink is much lower. DCM requires saving as many fingerprints as possible in the database, while the *Intersecting Areas* method only requires an update when the network equipment's physical location is outside of the previously observed area.
- the amount of data on the uplink, when using *Intersecting Areas*, decreases over time. The more updates received, the less updates needed in the future. In a mature system the need for updates is low.
- the amount of data stored on the server is much lower. DCM databases grow linearly while the *Intersecting Areas* method's database grows at a much slower rate. In a mature system the database ceases to grow almost completely.
- the granularity of data stored on the system is much lower. In the event that someone manages to trace data back to a user, the data will tell her very little about the user.
- the sensitivity of the stored data is low. Any datum stored is a compilation of points retrieved from many users, and there is no link between said data and any user. An area is a collection of points and there is no means to determine who contributed to an area, much less who contributed a single point.

4.13 Summary

In this chapter we suggested a location estimation method for open, community driven, systems, tuned towards privacy preservation. The main features of the *Intersecting Areas* method are:

- requires relatively low data transfer size and frequency
- embodies the simplicity of the CGI and E-CGI methods

- embodies the power of DCM methods
- small memory and processing power footprint compared to DCM
- extremely flexible and adaptive to different network equipment and different data
- easier to secure in terms of anonymity and privacy compared to DCM

The *Intersecting Areas* method relies on gathering areas of observations of different samples. Each individual sample is assigned to an area which is stored as a convex or concave hull. Clients send a collection of all network samples they observe. The system retrieves the areas corresponding to each sample and calculates the intersection of all of the retrieved areas. This intersection is returned as an estimated location area, and the center of the intersection is used as the estimated location.

Three different types of error estimations can be calculated, allowing the client to determine how to interpret and use the returned location estimation. One of these error estimations can be used to enhance the location estimation calculation as well. On a mature implementation one can use statistical measures for more precise error estimations.

Figure 4.7: Live example of the *Intersecting Areas* location estimation method. The intersection of the red areas are found to produce the green estimated location area. The yellow point is the estimated location and the blue point is the true location.

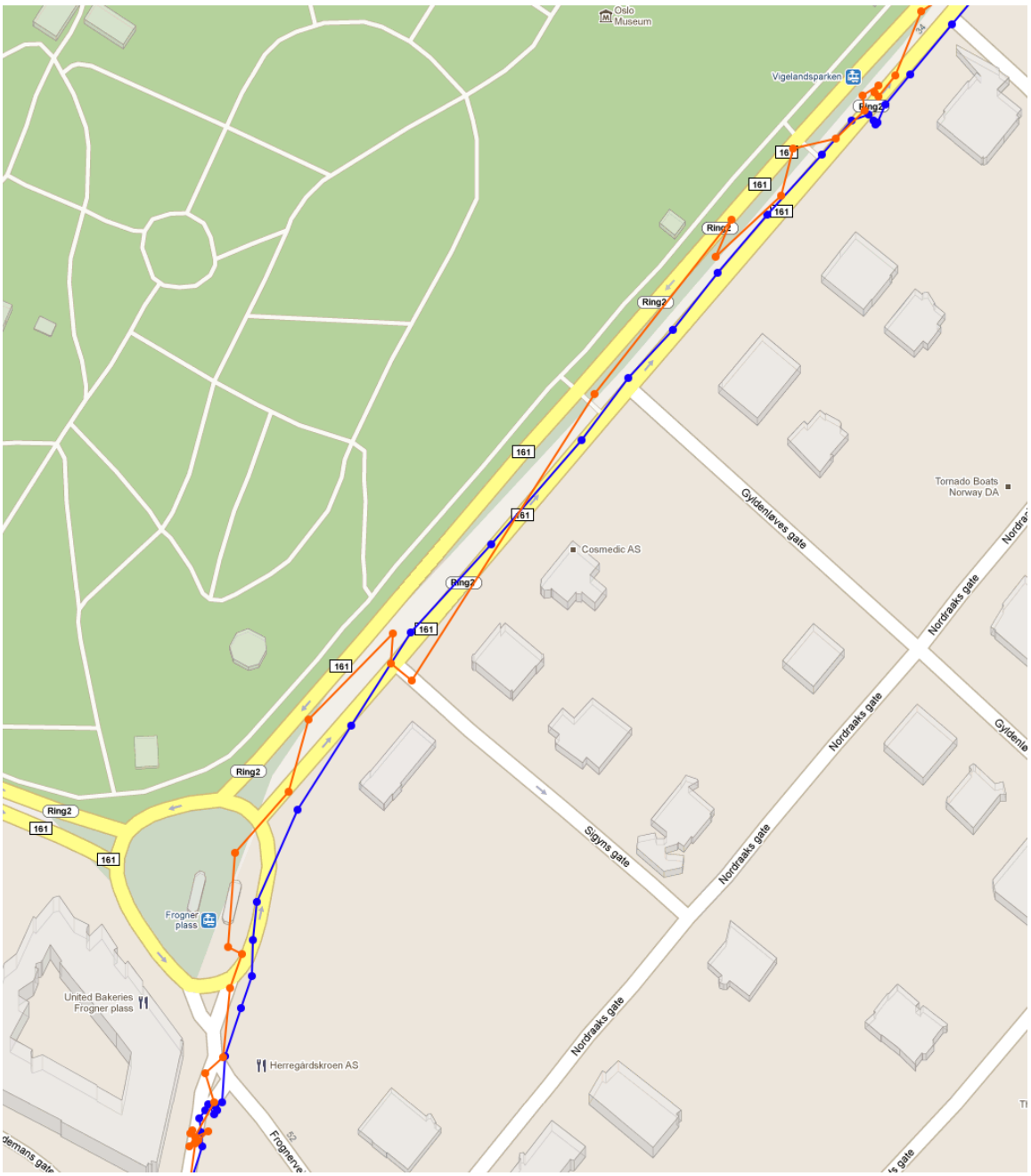


Figure 4.8: Live example of the *Intersecting Areas* location estimation method over a route. The blue line represents the true path and the red line represents the estimated path.

CHAPTER 5

IMPLEMENTING A LOCATION ESTIMATION TEST SYSTEM

The first step, after designing the system, was to create an advanced test system that allowed us to gather data and test location estimation methods. The system was designed to be as flexible as possible, to allow us to run a wide range of tests, gather a widest possible range of data, but also so the system easily can be adapted to new situations and test scenarios in the future.

Our test system consists of four main parts: a large collection of data-gathering tools, a core system, a database, and a data visualization tool. In this chapter we briefly document each part of the system.

5.1 Data-Gathering Tools

To be able to test an open and community driven location estimation system we needed data; we needed a large collection of network measurements from the field. Therefore, the first thing we created was a set of data-collection tools, designed to collect a wide range of data from a wide range of devices.

The data-gathering tools are the main source of data. The first version was implemented as a Python[1]-application that runs on a mobile station and collects data in the field. The application uses an internal or external GPS to record the network equipment's true geographical location. In addition all measurements from any GSM, UMTS and WLAN within range are recorded.

The data-gathering tools all log data to regular text-files that later are parsed and entered into the database. Each log-file has a descriptive human-readable header describing the content.

Five different versions of the application were created: one for the Symbian Series 60 (pys60), two different OpenMoko versions, an Android version, and a PC-version that uses an external modem. What data is stored varies from platform to platform; we store all available data on each platform. All of the applications were written using the Python language, except for the Android version written in Java.

We created such a wide range of data-gathering tools for several reasons: The amount of data available through standard APIs varies from mobile station to mobile station and platform to platform; collecting data from multiple mobile stations and platforms allows us to compare not only different mobile stations, but also a wider range of location estimation methods that rely on different types of data.

5.1.1 The Laptop Application

The laptop application, written in Python, is a command line application intended to be used on small laptops or notebooks. It was mainly intended to be used when gathering data using a vehicle. The application relies on an external GSM-modem and an external GPS.

The application has been successfully run on a 5V 8W Aleutia E2 Minipc[2] using a laptop battery backup pack allowing data gathering for days on end in areas only accessible by foot.

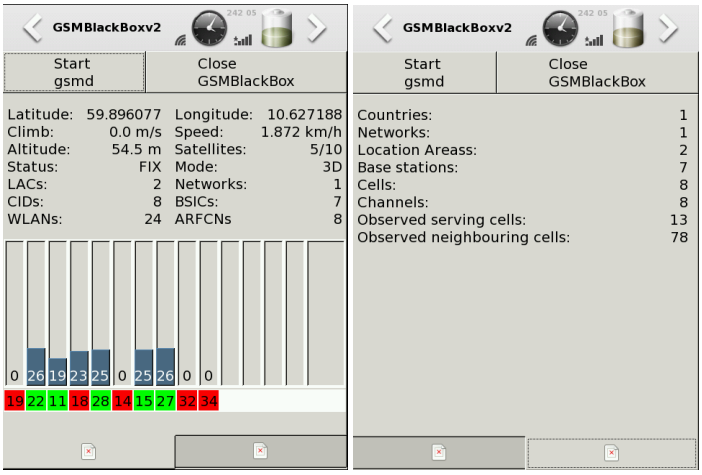
The laptop application is intended to be used with the EZ10-QUAD-PY[3] GSM modem produced by Telit[4]. This modem allows the application some unique possibilities. When run without a SIM-card the modem can scan all GSM networks within range and measure all broadcast channels from all BTSs. In addition it can list all broadcast and non-broadcast channels belonging to each BTS. Furthermore, it measures and reports on all non-broadcast channels, a feature not available with other hardware. The laptop software takes advantage of the first two possibilities. However, it does not log non-broadcast channels since none of the mobile stations used in this project support and can take advantage of this.

[1] http://python.org (2010-08-07)
[2] http://www.aleutia.com/wp-content/Photos/2009/02/E2-Datasheet.pdf (2010-07-03)
[3] http://www.gatetel.com/PDF/EZ10/80269st10024a_EZ10-QUAD-PY_Prod_Descr_r0.pdf (2010-07-03)
[4] http://telit.com (2010-07-03)

5.1.2 The OpenMoko Application

The OpenMoko[5] running on the Neo FreeRunner[6] was selected due to the large amount of network measurements available on the platform. The OpenMoko-application includes extended information about the serving cell, in addition to information about the (maximum) eight neighboring cells on GSM networks. No UMTS data is available. In addition the use of an external or internal GPS provides a wide range of GPS measurements.

A second OpenMoko-application was implemented as an alternative to the laptop application. It is identical to the laptop application in all ways except for the fact that it runs on the OpenMoko and has a graphical user interface. The intention of this application is to save weight and battery consumption when logging networks in inaccessible areas.



(a) Main Overview (b) Mobile Network Overview

Figure 5.1: OpenMoko data gathering tool

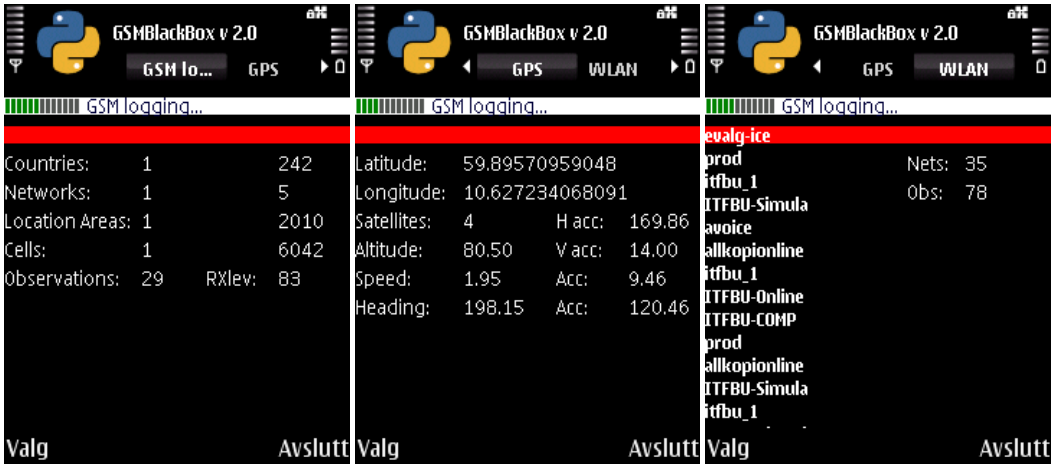
5.1.3 The Symbian Series 60 Application

Though the amount of data available on the Symbian Series 60 platform is limited, it was chosen for its wide availability.

The only GSM or UMTS information gathered is the serving cell identification and the received signal strength of the serving cell. No other mobile network data is available through the standard API. A fair amount of WLAN-data is gathered and using an external or internal GPS a wide range of GPS-data is gathered.

UMTS and WLAN data is not available on all Symbian Series 60 mobile stations. When run on a mobile station missing one, or both, of these features, the application simply skips this data and logs whatever data is available.

[5] http://openmoko.com (2010-03-01)
[6] http://www.openmoko.com/freerunner.html (2010-03-01)



(a) Mobile network screen (b) GPS screen (c) WLAN screen

Figure 5.2: The Symbian Series 60 data gathering tool

5.1.4 The Android Application

The Android[7]-platform was choose for its stability, availability and the possibility of measuring neighboring cell information. Written in Java[8], the Android application is the only data-gathering application not written in Python.

The application has access to fairly limited GSM or UMTS data. In addition it has, as the only one of our data-gathering applications, access to neighboring cell information on *both* GSM and UMTS networks.

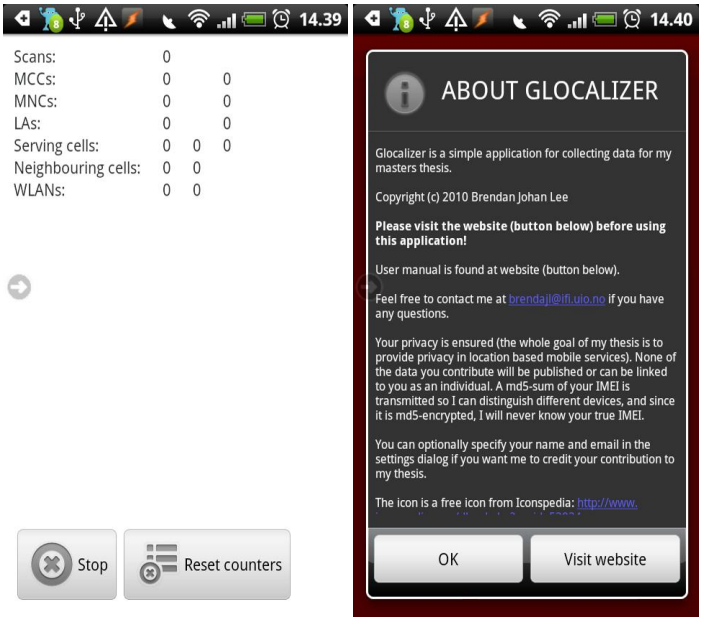
The application gathers a limited amount of data using the internal A-GPS on the mobile station. No option to use an external GPS exists.

Finally, the application gathers a limited amount of WLAN data.

The application is published on Android marked, and the only data-gathering application we released into the wild, allowing people outside of the project to contribute measurements.

Since people outside the project were given the ability to help collecting data this application the application preserves privacy in the spirit of the project. People can contribute their names and email addresses if they wish, but can also choose to be anonymous. The device's IMEI address is encoded in the filename so that one can determine which log-files came from an individual device. The IMEI address is MD5 encrypted. This allows one to determine which files come from a single device, without gathering the users private information. Finally the stored log-files are freely accessible on the device's memory card. Users can review what data the files contain before sending them, or even remove data they determine is sensitive before sending it. This means that data from the application is not trustworthy when coming from unknown sources. The analysis of the accuracy of the algorithms therefore excludes data from sources we do not personally know.

[7] http://android.com (2011-03-19)
[8] http://java.com (2011-04-23)



(a) Logging (b) Policy

Figure 5.3: Android data gathering tool

5.1.5 Hardware Logger

The hardware logger is a physical device built to be used with the laptop application. The device is built inside an external harddrive cabinet and contains:

- Sequoia EZ10-QUAD-PY GSM modem with Tellit GM862-Quad-Py chip[9]
- Globalsat ET-333 GPS-chip[10]
- a 9–24 to 5V converter
- a USB-hub
- 2 RS232—USB adapters
- a TTL—RS232 power level adapter
- external GPS antenna
- external GSM antenna

The unit can be plugged into any device that can act as a USB host and run the laptop application or the OpenMoko-modem application for instant network logging.

Besides the benefits listed in section 5.1.1 page 5.1.1, the EZ10 modem can, when supplied with a SIM-card be locked to a single serving cell and be refused to perform cell re-selection. This can be used to log the precise coverage area of any given cell, at different rxlevels by war driving the area surrounding the cell.

[9] http://www.sequoia.co.uk/components/product.php?d=3&c=99&f=4&p=51&fmt=grid (2010-05-10)

[10] http://www.globalsat.com.tw/products-page.php?menu=2&gs_en_product_id=4&gs_en_product_cnt_id=40 (2011-05-10)

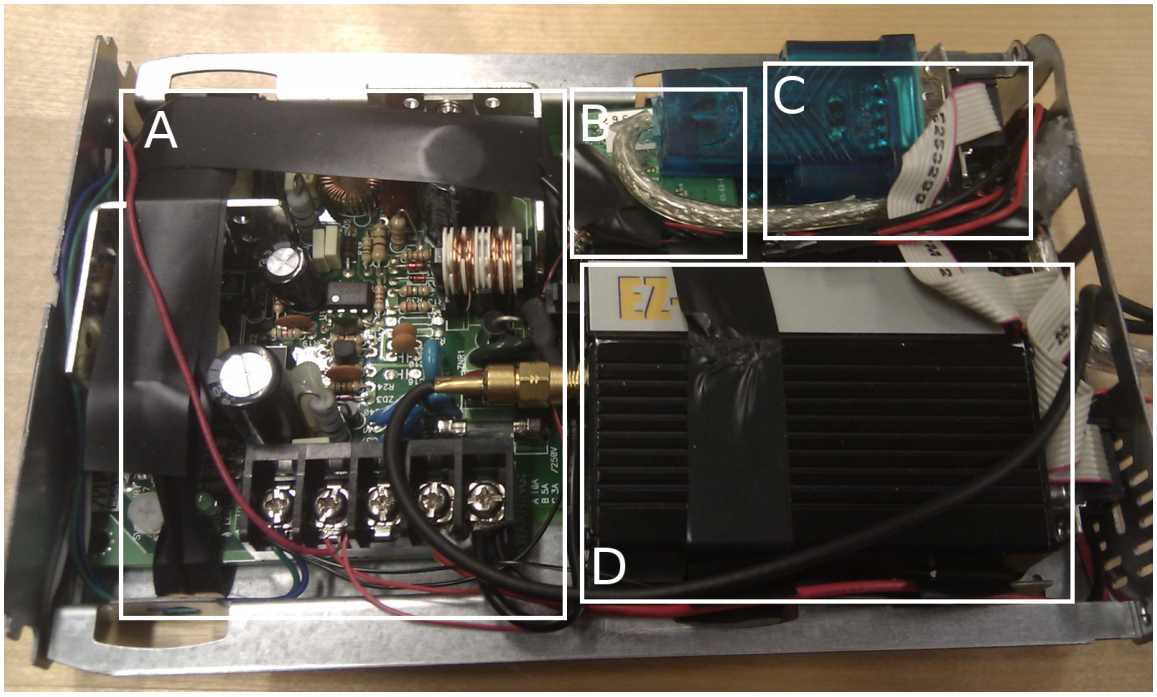


Figure 5.4: The hardware logger. a) Power converter b) GPS-chip c) RS232—USB and TTL—RS232 adapters d) GSM modem

5.1.6 Gathered Data

All of the data-gathering software was created to gather the widest possible range of data on observed WLAN networks, observed GSM and UMTS networks, and GPS measurements, available through the standard API on the devices the software was written for. GPS measurements include calculated speed and altitude in addition to all error estimation values when available. In addition the GPS-timestamp and local device-timestamp is always stored. On all platforms and all versions the GPS coordinates are stored in the WGS84 Decimal Degrees format. What data is available varies widely from platform to platform. The following tables list the exact data collectable on the different platforms:

Value	Modem	OpenMoko	Symbian S60	Android
Basic Service Set Identifier (BSSID)	✔	✔	✔	✔
Service Set Identifier (SSID)	✔	✔	✔	✔
mode	✔	✔	✗	✗
channel	✔	✔	✔	✗
quality	✔	✔	✗	✗
Radio Receive Level (rxlevel)	✔	✔	✔	✔
noiselevel	✔	✔	✗	✗
encryption	✔	✔	✔	✔
wpa_ie	✔	✔	✗	✗
bnc_int	✔	✔	✗	✗
beaconinterval	✗	✗	✔	✗
connectionmode	✗	✗	✔	✗
supported rates	✗	✗	✔	✗
capabilities	✗	✗	✗	✔

Table 5.1: WLAN data gathered

Value	Modem	OpenMoko	Symbian S60	Android
Cell ID (CID)	✔	✔	✔	✔
Mobile Country Code (MCC)	✔	✔	✔	✔
Mobile Network Code (MNC)	✔	✔	✔	✔
Location Area Code (LAC)	✔	✔	✔	✔
Radio Receive Level (rxlevel)	✔	✔	✔	✔
Paging repeat period (BS PA MFRMS)	✗	✔	✗	✗
Periodic location update interval (t3212)	✗	✔	✗	✗
Temporary Mobile Subscriber Identity (TMSI)	✗	✔	✗	✗
Absolute Radio-Frequency Channel Number (ARFCN)	✔	✔	✗	✗
Path Loss Criterion (C1)	✗	✔	✗	✗
Cell Reselection Criterion (C2)	✗	✔	✗	✗
Base Station Identity Code (BSIC)	✔	✔	✗	✗
Downlink Signaling failure Counter (DSC)	✗	✔	✗	✗
Transmit Power Level (txlevel)	✗	✔	✗	✗
Timeslot Number (TN)	✗	✔	✗	✗
Radio Link Timeout Counter (RLT)	✗	✔	✗	✗
Timing Advance (TA)	✗	✔	✗	✗
Received Field Strength Full (rxlevel_f)	✗	✔	✗	✗
Received Field Strength Sub (rxlevel_s)	✗	✔	✗	✗
Received Quality Full (rxqual_f)	✗	✔	✗	✗
Received Quality Sub (rxqual_s)	✗	✔	✗	✗
Cell Bar Access (CBA)	✔	✔	✗	✗
Cell Bar Qualifier (CBQ)	✔	✔	✗	✗
cell type indicator	✔	✔	✗	✗
vocoder	✗	✔	✗	✗
Bit Error Rate (BER)	✔	✗	✗	✗
number of ARFCN on cells BTS	✔	✗	✗	✗
list of ARFCN on cells BTS	✔	✗	✗	✗
number of channels on cells BTS	✔	✗	✗	✗
list of channels on cells BTS	✔	✗	✗	✗

Table 5.2: GSM serving cell data gathered

Value	Modem	OpenMoko	Symbian S60	Android
Cell ID (CID)	n/a	✔	✗	✔
Location Area Code (LAC)	n/a	✔	✗	✔
Radio Receive Level (rxlevel)	n/a	✔	✗	✔
Absolute Radio-Frequency Channel Number (ARFCN)	n/a	✔	✗	✗
Path Loss Criterion (C1)	n/a	✔	✗	✗
Cell Reselection Criterion (C2)	n/a	✔	✗	✗
Base Station Identity Code (BSIC)	n/a	✔	✗	✗
frame offset	n/a	✔	✗	✗
time alignment	n/a	✔	✗	✗
Cell Bar Access (CBA)	n/a	✔	✗	✗
Cell Bar Qualifier (CBQ)	n/a	✔	✗	✗
Routing Area Code (RAC)	n/a	✔	✗	✗
cell type indicator	n/a	✔	✗	✗
Cell Reselection Offset (CRO)	n/a	✔	✗	✗
temporary offset	n/a	✔	✗	✗
rxlevel access minimum	n/a	✔	✗	✗

Table 5.3: GSM neighboring cell data gathered

Value	Modem	OpenMoko	Symbian S60	Android
ARFCN	✔	✗	✗	✗
rxlevel	✔	✗	✗	✗

Table 5.4: GSM non-broadcast channel data collected

Value	Modem	OpenMoko	Symbian S60	Android
Mobile Country Code (MCC)	✗	✗	✗	✓
Mobile Network Code (MNC)	✗	✗	✗	✓
Location Area Code (LAC)	✗	✗	✗	✓
Cell ID (CID)	✗	✗	✗	✓
Radio Receive Level (rxlevel)	✗	✗	✗	✓

Table 5.5: UMTS serving cell data gathered

Value	Modem	OpenMoko	Symbian S60	Android
Primary Scrambling Code (PSC)	✗	✗	✗	✓
Received Signal Code Power (RSCP)	✗	✗	✗	✓

Table 5.6: UMTS neighboring cell data gathered

Value	Modem	OpenMoko	Symbian Series 60	Android
time	✔	✔	✔	✔
time accuracy (EPT)	✔	✔	✗	✗
horizontal accuracy (EPH)	✔	✔	✔	✗
horizontal dilution of Precision (HDOP)	✗	✗	✔	✗
vertical accuracy (EPV)	✔	✔	✔	✗
vertical dilution of Precision (VDOP)	✗	✗	✔	✗
bearing over ground	✔	✔	✔	✔
bearing accuracy (EPD)	✔	✔	✔	✗
speed over ground	✔	✔	✔	✔
speed accuracy (EPS)	✔	✔	✔	✗
climb	✔	✔	✗	✗
climb accuracy (EPC)	✔	✔	✗	✗
latitude	✔	✔	✔	✔
longitude	✔	✔	✔	✔
altitude	✔	✔	✔	✔
visible satellites	✔	✔	✔	✗
used satellites	✔	✔	✔	✗
position accuracy	✗	✗	✗	✔

Table 5.7: GPS data gathered

5.2 Backend

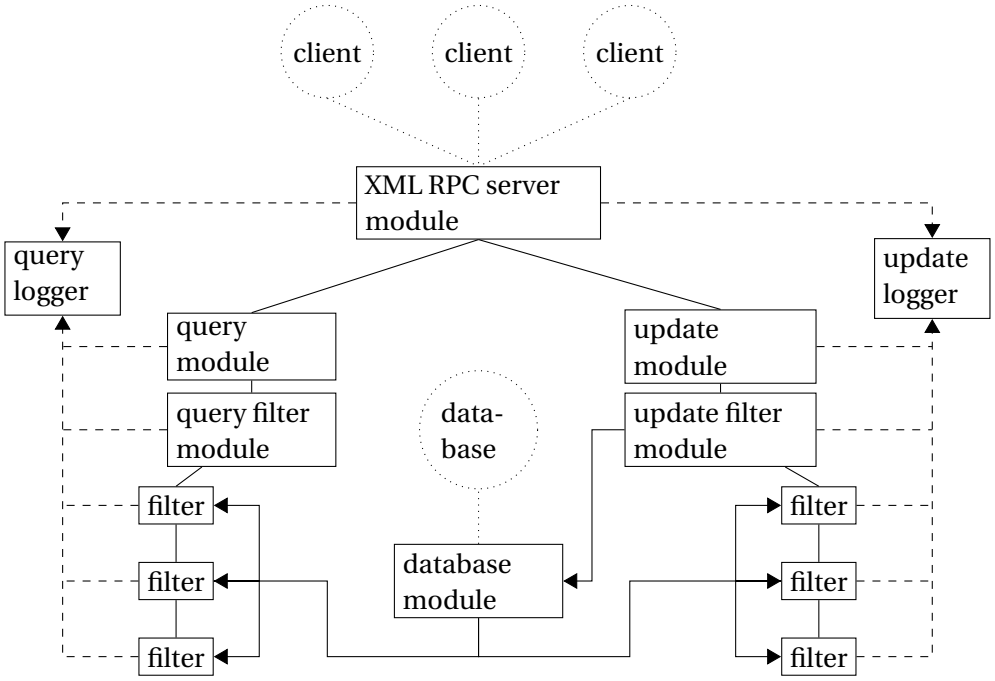


Figure 5.5: The backend module hierarchy

The core of the test system is the backend which provides a simple interface used to access, update, and maintain location data in a database. Internally the backend is a hierarchy of modules. Data flowing through the backend travels down through a module hierarchy reaching a filter, which in turn analyzes and handles the data. Any returned information travels back up through the hierarchy.

The filter system provides a simple plug-in based method of processing data, which includes ensuring quality of data, analyzing data, and generating responses. It provides a flexible test platform for developing and testing algorithms.

The flow of data through the backend can be divided in three categories: server settings and server information requests, queries and updates.

Server settings involve changing default server parameters runtime. Server information requests involves retrieving information from the server regarding server parameters, status reports, etc.

A query, a request for an estimated location based on one or more fingerprints, received by the server module is passed down through the *left side* of the module hierarchy (see figure 5.5 page 60). The result of the query is passed back up and returned to the client.

An update, a request to add new data to the database is passed down through the *right side* of the module hierarchy (see figure 5.5 page 60). The result of the update is generally not passed on to the client; the client only receives an acknowledgement.

5.2.1 The Log System

The log system provides step-by-step logging of all data received by the backend, all output the backend generates, in addition to all actions performed by the backend. The logs provide enough information to re-create any live interaction with the server at a later time, and provides the means of storing live user sessions and replaying them at a later time, possibly with different settings. This allows simulations of live sessions while experimenting with different filters, different algorithms, etc.

- **Debug**: Debug information from the server. This information should be ignored when re-creating user sessions.
- **Info**: General information needed to re-create user sessions, includes received data, filters used, returned data, etc.
- **Warning**: Warning information includes any warnings about received data, returned data, used filters, etc. Warnings can in general be ignored when re-creating user sessions.
- **Error**: Any errors regarding received data, returned data, filters, etc. An error log entry is triggered when a single query/update or parts of a query/update fails. This includes illegal access, invalid data, missing filter, etc. Errors can be of importance when re-creating user sessions, and normally an error should result in skipping one or more queries/updates.
- **Critical**: The critical keyword is reserved for programming errors resulting in a state where the server cannot continue to handle requests. Any server errors or warnings not resulting in such a state should be marked as a debug-level log entry.

5.2.2 Modularization

All of the Python-software written for this project has been modularized as much as possible, allowing module re-use throughout the project.

In addition the backend is completely modularized. The backend consists only of modules and filters—also implemented as modules. The advantage of this approach is flexibility; the backend is extremely flexible. Any part of it can be replaced or re-written by simply replacing a module. This approach makes it simple to switch to other database software or other types of databases, test new algorithms, add new network equipment platforms, or change storage formats.

In the following sections an overview is given of the most important modules used in the backend.

The Server Module

The server module is the main engine. It is implemented using the event-driven network engine Twisted.[11] This module receives all incoming requests, handles any requests for information regarding the server and requests to change server settings, and passes query requests on to the query module and update request on to the update module. Naturally this module also handles all responses to connected clients, authenticating clients, black-listing, etc. All of the software has built-in help-functions describing options and configuration settings.

A range of configuration options determine how the server performs, contains information on the database in use, etc.

[11] http://twistedmatrix.com/trac (2011-04-20)

Query

The *query module* receives any incoming queries from the *server module*, logs the event via the *query logger module*, and passes the query on to the *query filter module* for processing. Any data to be returned to the client is passed back through the *query module*, which again logs the event, before passing the reply to the *server module*.

Update

In almost all aspects the *update module* handles incoming updates in the same manner as the *query module* handles incoming queries. The one main difference is how database access is handled. Since queries do not need write access to the database, the filters themselves handle all of the database access. Updates, on the other hand, need write access. Filters do not have write access to the database, so any data a filter needs to add to the database must be passed back down to the *update filter module*. If the filter supplies valid data, the *update filter module* adds the data to the database, and logs the event through the *update logger module*.

Database

All of the server's interaction with the database must pass through this module. No other module may at any time access the database directly. New database modules can be produced to change how the server interacts with the database. Not only can database interaction be changed, but the database itself can easily be replaced. This supplies support for load balancing, separate databases for pending updates and filtered requests, and even interfacing with other databases. For detailed information on the database, see 5.3 page 62.

5.2.3 Filters

Filters are created as Python modules. This allows new filters to be created and loaded without restarting the server. In addition this means that filters, once loaded, reside in memory resulting in fast responses. But the main benefit of filters is the flexibility. Two types of filters exist: Query filters and update filters. Query filters have one single task; convert an incoming query to a location estimation. Update filters have two tasks, analyze an incoming update to determine if it is a valid and usable update, and convert the incoming data to a format that is recognized by the *update filter module* for adding to the database. Thus adding new location estimation methods and algorithms to the test system is simple; simply implement a new query filter, a new update filter and, in some instances, a new database module or an extension to the existing database module.

5.3 Database

The database was created with three rules in mind: It should follow the general rules for relational databases and confirm to the 3rd normal form. The database should be created in a manner that easily allows adding new data not in the original design. Any data in the database, or combination of such data should be easy to retrieve.

The database was not created to be a production database, or optimized for speed or performance. Instead the database was created to be as flexible as possible for collecting data and testing different algorithms and ways of processing the data.

The database is maintained using a series of scripts. Several different scripts are used to add new data to the database while maintaining the integrity of the existing data. Each script is dedicated to adding data from a specific network equipment platform.

The *algotestground* part of the database is a staging area for converting data for algorithms with other requirements than the standard fingerprint.

Snapshots of the database are manually taken any time a major change is made to the database structure, or the data itself. In addition a snapshot is taken before each bulk of data tested using different algorithms. Since an algorithm's performance varies based on the content of the data, such snapshots are important. The snapshots allow the algorithms to be re-run at a later time with the same set of data they originally were run on so that results can be confirmed.

The complete database schema and an overview of the technology used can be found in appendix C page v.

5.4 Visualization

We implemented the glocalizer map application used for analyzing and illustrating collected data, comparing different algorithms and methods of estimating locations, and comparing collected data to the known position of antennas. Simply put glocalizer is an application that renders maps complete with one or more overlays consisting of collections of physical locations.

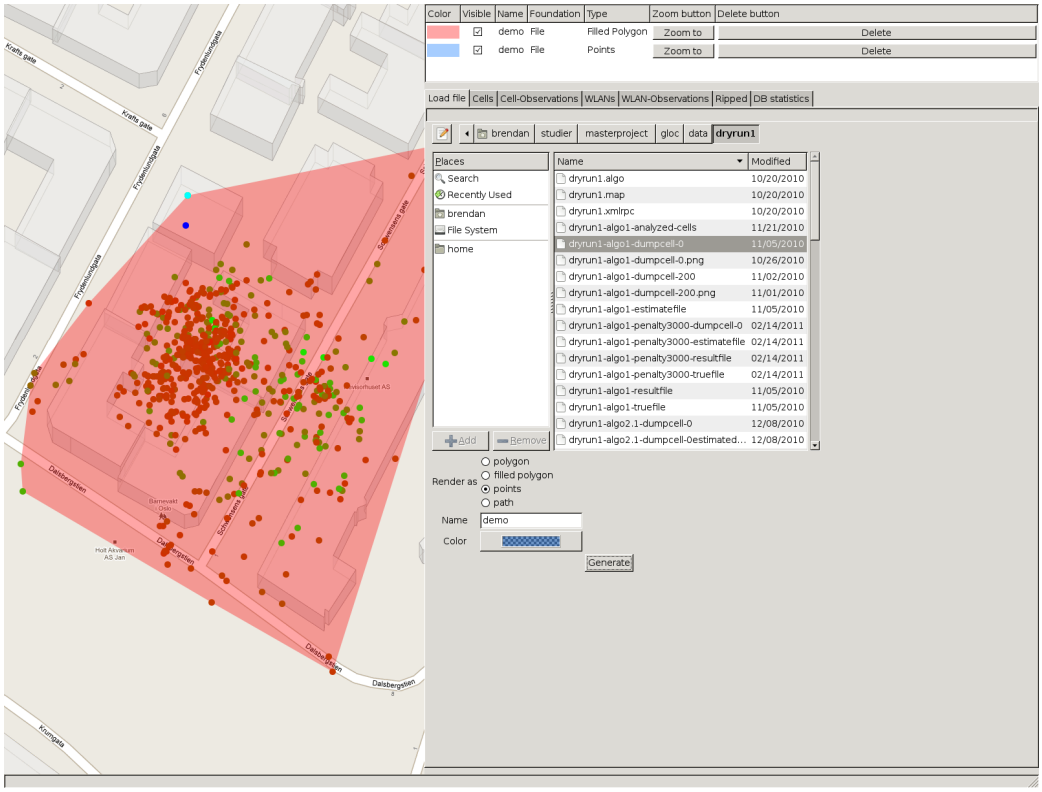
The application is based on the open source application GMapCatcher[12]. The core of the GMapCatcher project, the map tile download, storage and retrieval utility, is untouched, while the rest of the application has been rewritten and customized for use within our project. The most important change is that the map rendering is now done using Cairo[13] to enable fast rendering of maps with a large number of overlays.

The application supports rendering maps from several different sources (Google Maps[14], Yahoo Maps[15], Microsoft Maps[16], Open Street Maps[17], and a wide range of services based on Open Street Maps data), in addition to several different type of maps (street map, satellite map, etc.). Once a map is rendered, the application supports geo-coding using Googles geo-coding service, allowing the map to be centered and zoomed to a named location. The map can also be zoomed and navigated manually.

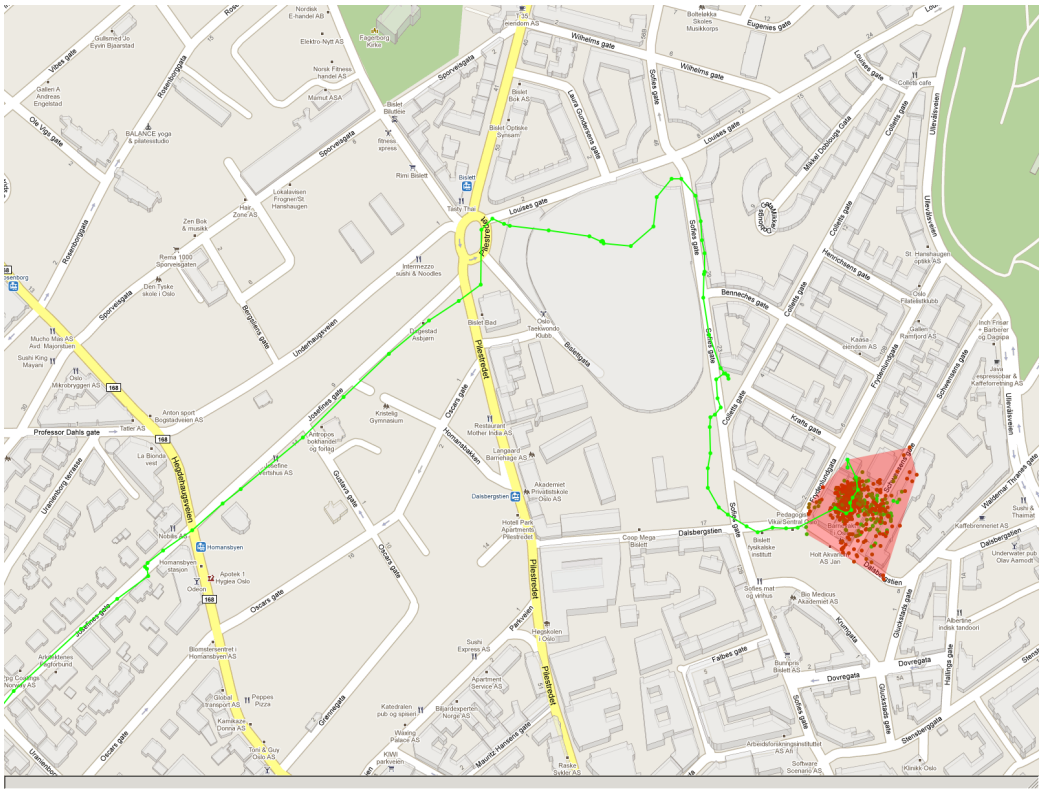
An overlay consists of one or more physical points represented by latitude and longitude in the WGS84 Decimal Degrees format. Each overlay is manually assigned a name and a color, and is in addition to being rendered on the map also places in a list of current overlays. Each individual overlay may be deleted or have its visibility toggled. In addition one can automatically zoom and center the map display to an existing overlay.

Several different types of overlays exist. The simplest being a small dot on the map for each point supplied. An overlay can also be rendered as a polygon or filled polygon, either directly based on the supplied points, or as a convex hull surrounding the supplied points. Finally an overlay can also be a *path*. A path is represented by a small dot on the map for each point supplied, each connected by a line. The line is determined by the order the points are supplied.

[12] http://code.google.com/p/gmapcatcher/ (2010-04-04)
[13] http://cairographics.org/ (2011-05-14)
[14] http://maps.google.com (2011-02-02)
[15] http://maps.yahoo.com (2011-02-02)
[16] http://microsoft.com/maps (2011-02-02)
[17] http://openstreetmap.org (2011-02-02)



(a) Using tools



(b) Map with overlays

Figure 5.6: Screenshots of the glocalizer visualization application

The glocalizer application reads overlay data from file in addition to interfacing and retrieving data directly from the database described in section 5.3 page 62.

5.5 Scripts

Besides the software mentioned throughout this chapter, a wide range of scripts have been designed and implemented. The scripts are used to perform menial and smaller tasks within the system. This includes, but is not limited to, database maintenance, log-file conversions, importing log-files to the database, analyzing location estimation methods and their performance, generating data-sets, and plotting charts and graphs.

The scripts are implemented in Python or Bash. Small scripts performing single tasks have self-explanatory names. Longer more complex scripts performing larger tasks, or with many configuration options all come with a built in help-system. Many of the larger, more complex, scripts come with smaller single-action wrapper-scripts.

5.6 Summary

In this chapter we discussed the implementation of our location estimation test-system. We saw how the system consists of four main parts: a wide range of data collection tools running on a wide range of platforms, a flexible database, an extremely flexible core-system interfacing with the database and providing a simple means of implementing and testing different location estimation methods, and finally a powerful data-visualization tool used to render data on maps.

We saw that the five different data collection tools can gather a wide range of GSM, UMTS and WLAN measurements tied to GPS measurements in the same physical location at the same time.

In addition we looked at a hardware-unit created specifically for gathering GSM data, and the software written to take advantage of the unit.

Finally we showed how the whole location estimation test-system was modularized to make it flexible end easy to extend and adapt in the future.

CHAPTER 6

TESTS AND RESULTS

In this chapter we first choose a set of algorithms and then test and compare them using the test system presented in chapter 5. Two sets of data were gathered to test the algorithms: One gathered with Symbian Series 60 devices and one gathered with Android devices. The OpenMoko platform proved highly unstable with common operating system crashes and long reboot times and was therefore not used. The hardware logger was used to map areas in a grid-like fashion reserved for future testing. In our initial tests we wished to investigate common off-the-shelf devices.

When testing the algorithms we focus on precision, calculation time and success rate. The precision is determined by calculating the difference between an algorithms estimation location of fingerprints and their true location measured by GPS. Calculation time is the average CPU-time spent by an algorithm to estimate a single location. Success rate, or performance, is a measure of how many fingerprints each algorithm successfully calculates a location for, and how many fingerprints each algorithm cannot produce a result for.

The chapter is concluded with a discussion of the results of the *Intersecting Areas* method compared to the other tested location estimation methods.

6.1 Requirements

An open and community sourced mobile location estimation system must be built using only mobile-based location estimation methods. Any method that relies on measurements or equipment available only to the network operators cannot be used, and are therefore not relevant for testing.

We compare our own estimation method with basic well used estimation methods, in addition to more advanced estimation methods.

Since we not only test the overall performance of existing location estimation methods, but also compare them to our own suggested method, to choose methods that rely on the same types of measurements as our own method.

When testing and discussing the algorithms we are interested in three characteristics: The precision or accuracy, the processing time, and the performance of success rate.

6.2 Selections

Our own *Intersecting Areas* method was implemented using convex hulls without limiting areas. This is the simplest implementation, and a good starting point. We handled the problem of non-intersecting areas by ignoring them on a first-come-first-serve basis as initial testing indicates this to be a rare occurrence. Keeping in mind that the frequency could increase with growing amounts of data, the need for handling this problem in a different way might become relevant in a full scale implementation.

Of existing methods CGI, E-CGI, and DCM are natural choices since our method attempts to embrace the beneficial features of each of these methods. While only one variant of CGI exits, the same is not true with E-CGI and DCM; several different variants of E-CGI exist—such as using TA values, received signal strength, etc.—while a whole range of DCM and DCM-enhancements exits.

Choosing a E-CGI method is quite simple. E-CGI using received signal strength is widespread, mainly because received signal strength is available on a wide range of mobile stations. Far fewer mobile stations provide API-access to TA values or other NMR values. The most widespread method, available on the largest range of mobile stations is a natural choice.

Among DCM methods, we choose the early published, and fairly simple, method suggested by Laitinen et al presented in section 2.3.5 page 10. The method is simple and easily adaptable. It is thoroughly described in the literature and does not require any form of heuristics, statistics or other features that would require a location history. Many proposed DCM methods rely on access to a users previously estimated locations. One of the requirements of the proposed system is that it may never retain any memory of a users location true or estimated.

As discussed, any methods that require knowledge of an individuals previous location are not compatible with an open and privacy preserving location estimation system. Hence statistical estimation methods such as Bayesian estimations and Neural Networks are not candidates for the tests. Map-matching is not a general method, but rather a method of improving results after the estimation, and is therefore excluded.

6.3 Selected Algorithms

Here we present a detailed description of each of the location estimation methods we tested using the system described in chapter 5, and the data they were tested on. Note that from here on they are referred to as algorithms as several of them are based on the same estimation methods, but rely on different types of data.

6.3.1 Algorithm 1 - Cell Global Identity

The first algorithm is the most basic: CGI. The observed serving cell identification is retrieved from the incoming fingerprint. The area representing all previous observations of the cell is retrieved from the database. The "center" of the area is found by averaging the extreme points of the area, and used as the estimated location.

We chose to store the areas used for CGI as convex hulls rather than circles to give CGI a fair competition with our own method.

The algorithm can be used with the very simplest phones with the simplest APIs and should be able to be used on most modern mobile stations that allow any form of third party software.

The algorithm works on any fingerprint as long as it contains the ID of the serving cell.

We expect the precision of this algorithm to be, by far, the lowest of all the algorithms tested since the method relies on only the serving cell which has a maximum range of 35km in GSM networks and 8km in UMTS networks (see appendix A page i). In terms of performance the situation is opposite: Any fingerprint would normally contain serving cell information, so the only point of failure is the systems complete lack of a previous observation of the given cell.

Figure 6.1 page 70 shows a live example of the CGI method from our test system.

6.3.2 Algorithm 2 - Enhanced Cell Global Identity

E-CGI utilizing received signal strength, is the second basic algorithm included in our tests. The method is identical to our implementation of CGI described in section 6.3.1 page 69, apart from using the area representing all previous observations of the serving cell–received signal strength pair rather than the area representing all observations of the serving cell.

This algorithm works on fingerprints containing the ID and the received signal strength of the serving cell.

The performance of this algorithm should, in our tests, match the performance of algorithm 1 as all of the network equipment used provided access to received signal strength in addition to serving cell. Enhancing with received signal strength should greatly improve the precision when compared to CGI since the granularity is much higher (we observed received signal strengths between -40 to -113 dBm). We expect this algorithm, in terms of precision, to have the second lowest results of our tests.

Figure 6.2 page 71 shows a live example of the E-CGI method from our test system.

6.3.3 Algorithm 3 - Database Correlation Method on Serving Cell and WLAN

This algorithm is based on the DCM method presented in[5]. In this version the algorithm utilizes the serving cell and WLAN data only. A query is run to find all fingerprints in the database containing an observation of the serving cell or at least one WLAN observation present in the query fingerprint. The algorithm presented in section 2.3.5 page 10 is used to assign a score

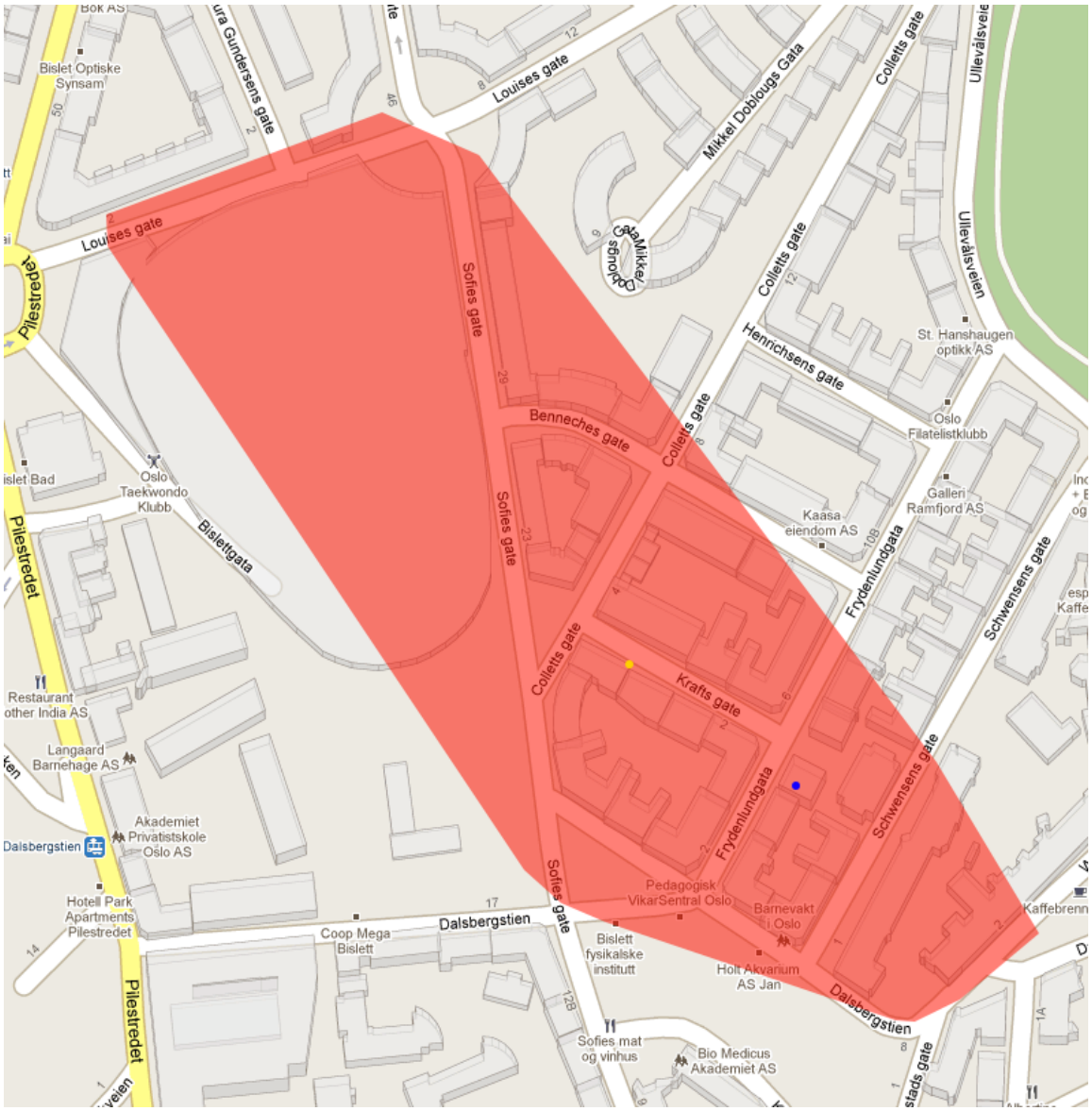


Figure 6.1: Live example of CGI location estimation method. The red area is the observed coverage of the serving cell. The yellow point represents the estimated location and the blue point represents the true location.

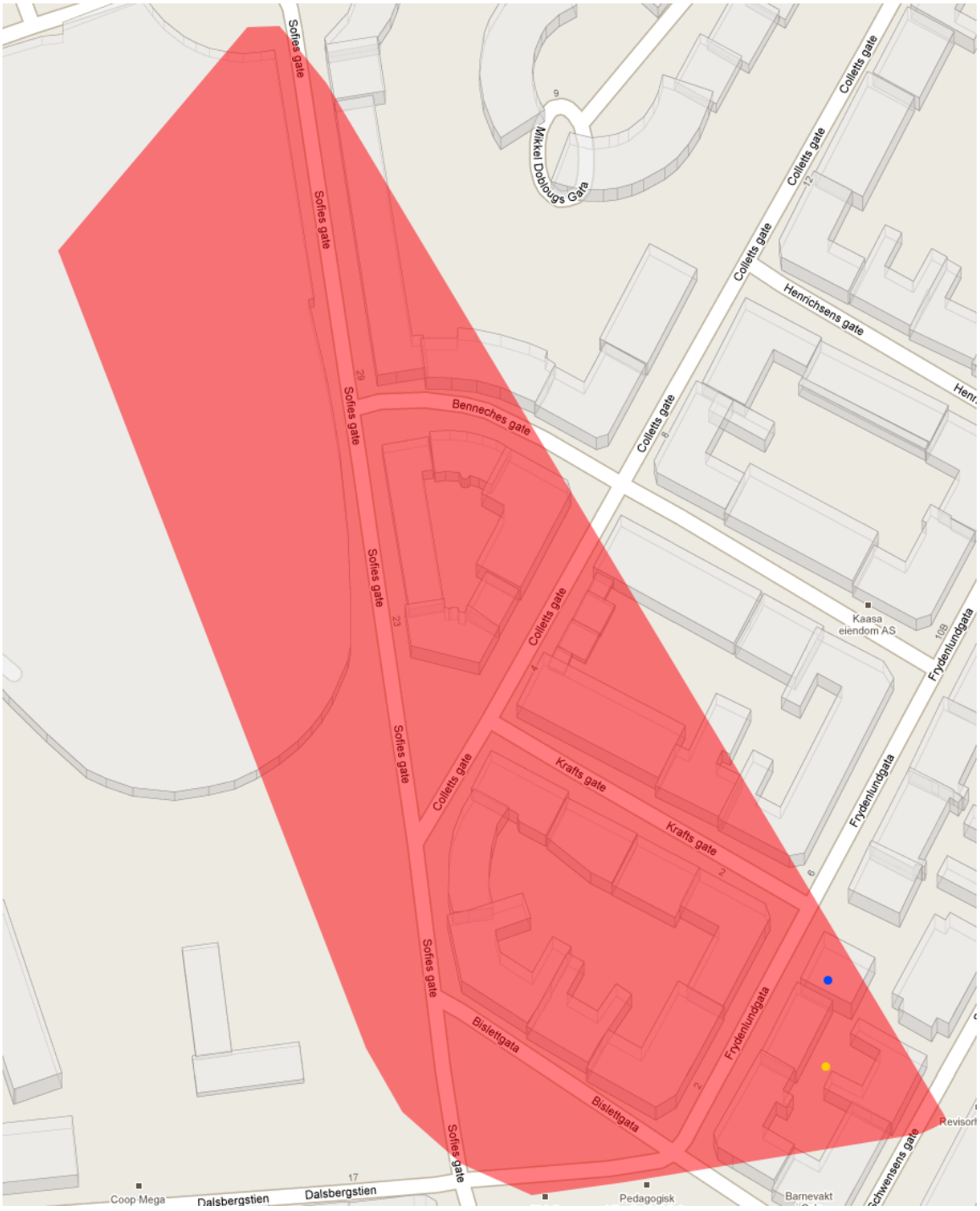


Figure 6.2: Live example of the E-CGI location estimation method. The red area is the observed coverage of the serving cell at the given received signal strength. The yellow point represents the estimated location and the blue point represents the true location.

based on the differences in received signal strength to each of the candidate fingerprints retrieved from the database. A penalty is removed from the score for each WLAN observation present in only the candidate fingerprint or only the query fingerprint. The actual location of the highest scoring candidate fingerprint is used as the estimated location.

This algorithm only produces a result when fingerprints contain serving cell, received signal strength of serving cell, and at least one WLAN measurement. The algorithm can easily be adapted to work with fingerprints only containing WLAN measurements and fingerprints missing the received signal strength of the serving cell. Since no such data exists in our data-sets, these adaptions were not done.

The DCM methods are expected to have the highest precision in our tests. We expect algorithm 3 only to be surpassed by algorithm 5 which utilizes both WLAN and neighboring cell information and therefor has a higher granularity. The DCM methods are expected to have the lowest performance. This algorithm will only yield a result when used in an area containing at least one WLAN network.

Figure 6.3 page 73 shows a live example of the DCM method from our test system.

6.3.4 Algorithm 4 - Database Correlation Method on Serving and Neighboring Cells

Algorithm 4 is the first of the algorithms using neighboring cell information. The same DCM technique used in algorithm 3, described in section 6.3.3 page 69, is used. The database fingerprint retrieval method is the same and the penalty score system is the same. The data used is not: Instead of utilizing WLAN observations, this algorithms utilizes neighboring cell observations.

The algorithm only works on fingerprints that contain serving cell, received signal strength of serving cell, and at least one neighboring cell observation.

We expect this to have the lowest results, both in terms of precision and in terms of performance, of all the DCM methods in our tests. This algorithm will only work on fingerprints containing neighboring cell information. Most of our data is gathered in urban and sub-urban areas. In areas highly saturated with WLAN networks algorithm 3 will outperform algorithm 4. In areas with little or no WLAN density algorithm 4 will outperform algorithm 3. The same goes for algorithm 4 and 5.

6.3.5 Algorithm 5 - Database Correlation Method on Serving Cell, Neighboring Cells and WLAN

Algorithm 5 is the final DCM method we have tested. Instead of utilizing only WLAN measurements or only neighboring cell information, this algorithm utilizes both.

The algorithm only works with fingerprints containing a serving cell—signal strength measurement. In addition the fingerprint must contain at least one WLAN or neighboring cell measurement.

We expect this algorithm to show the highest results of the DCM methods both in terms of precision and performance. It has the highest data granularity and will work on both fingerprints only containing neighboring cell information and fingerprints only containing WLAN information.

6.3.6 Algorithm 6a - Intersecting Areas - Serving Cell and WLAN

This is the first algorithm based on our proposed location estimation method. In this version one only utilizes serving cell and WLAN observations.

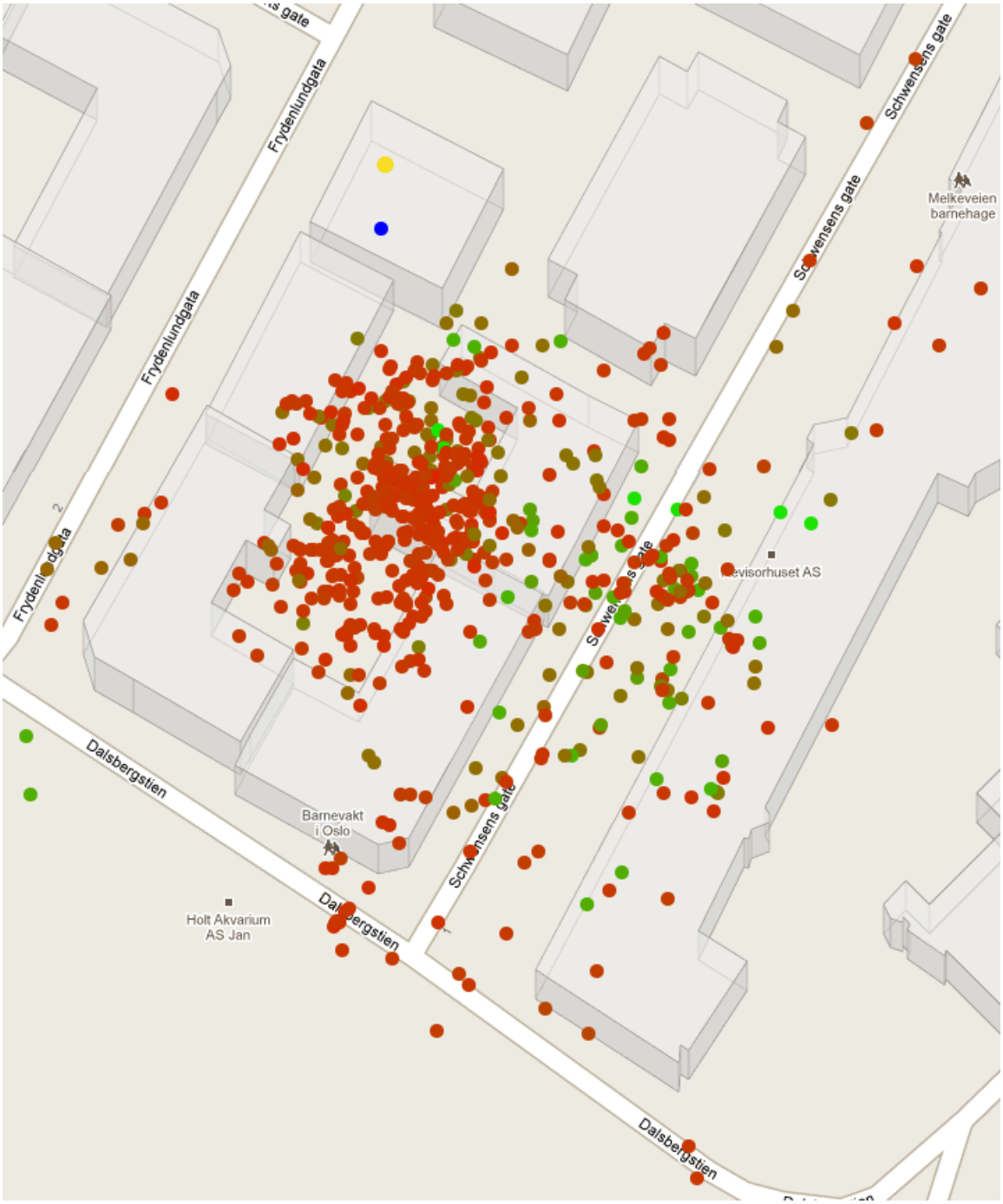


Figure 6.3: Live example of the DCM location estimation method. Each point represents a candidate fingerprint ranging from the lowest score in red, to the highest score in green. The yellow point denotes the estimated location i.e. the highest scoring fingerprint. The blue point denotes the true location.

The database contains a collection of entries corresponding to unique serving-cell signal-strengths. Each entry contains an area represented by the points of a convex hull surrounding all points the given serving cell has been observed at a given signal strength. The database also contains a corresponding table for WLAN signal strengths. In a live implementation these entries would be continuously updated. When testing algorithms a static database is used.

The system retrieves all of the areas from the database matching the serving cell–received signal strength and WLAN–received signal strength pairs in the query fingerprint. The intersection of these areas is used as the estimated location area, and the center of the area is used as the estimated location.

This algorithm only works when fingerprints contain serving cell, signal strength of serving cell, and at least one WLAN measurement. The algorithm can easily be adapted to work with fingerprints only containing WLAN measurements and fingerprints missing the signal strength of the serving cell. Since no such data exists in our test data, we have not implemented these adaptions.

The precision of this algorithm is expected to lie close to the precision of algorithm 3, as does the expected performance.

Figure 4.7 page 46 and figure 6.4 page 75 show live examples of the *Intersecting Areas* method from our test system.

6.3.7 Algorithm 6b - Intersecting Areas - Serving Cell and WLAN - with E-CGI Fallback

Algorithm 6b is an extension to algorithm 6a. Since we implemented both E-CGI and *Intersecting Areas* using convex hulls, and we use the same method to calculate the center of an area in both algorithms, the two are interchangeable. The *Intersecting Areas* method allows fallback to E-CGI. Running our method on a single area skipping the intersection part is essentially E-CGI. We have implemented a simple switch that allows our method to fall back to E-CGI when processing a query fingerprint that does not contain the data needed by the *Intersecting Areas* method.

We expect the precision of this algorithm to lie somewhere between the precision of algorithm 2 and algorithm 6a. The precision will depend on the proportion of fingerprints with the data needed by the *Intersecting Areas* method to the fingerprints without the data needed by the *Intersecting Areas* method. The performance is expected to be identical to the performance of algorithm 2.

6.3.8 Algorithm 7a - Intersecting Areas - Serving and Neighboring Cells

Algorithm 7a is the second algorithm based on the *Intersecting Areas* location estimation method. In this variant of the method the serving cell–signal strength area is intersected with neighboring cell areas.

The algorithm only works with fingerprints containing a serving cell–signal strength measurement and a measurement of at least one neighboring area.

The precision of this algorithm is expected to lie close to the precision of algorithm 4. The performance should be close to, but a bit lower than, algorithm 4. The reason is that while algorithm 7a needs at least three observations of a neighboring area to successfully train, algorithm 4 only needs a single observation.

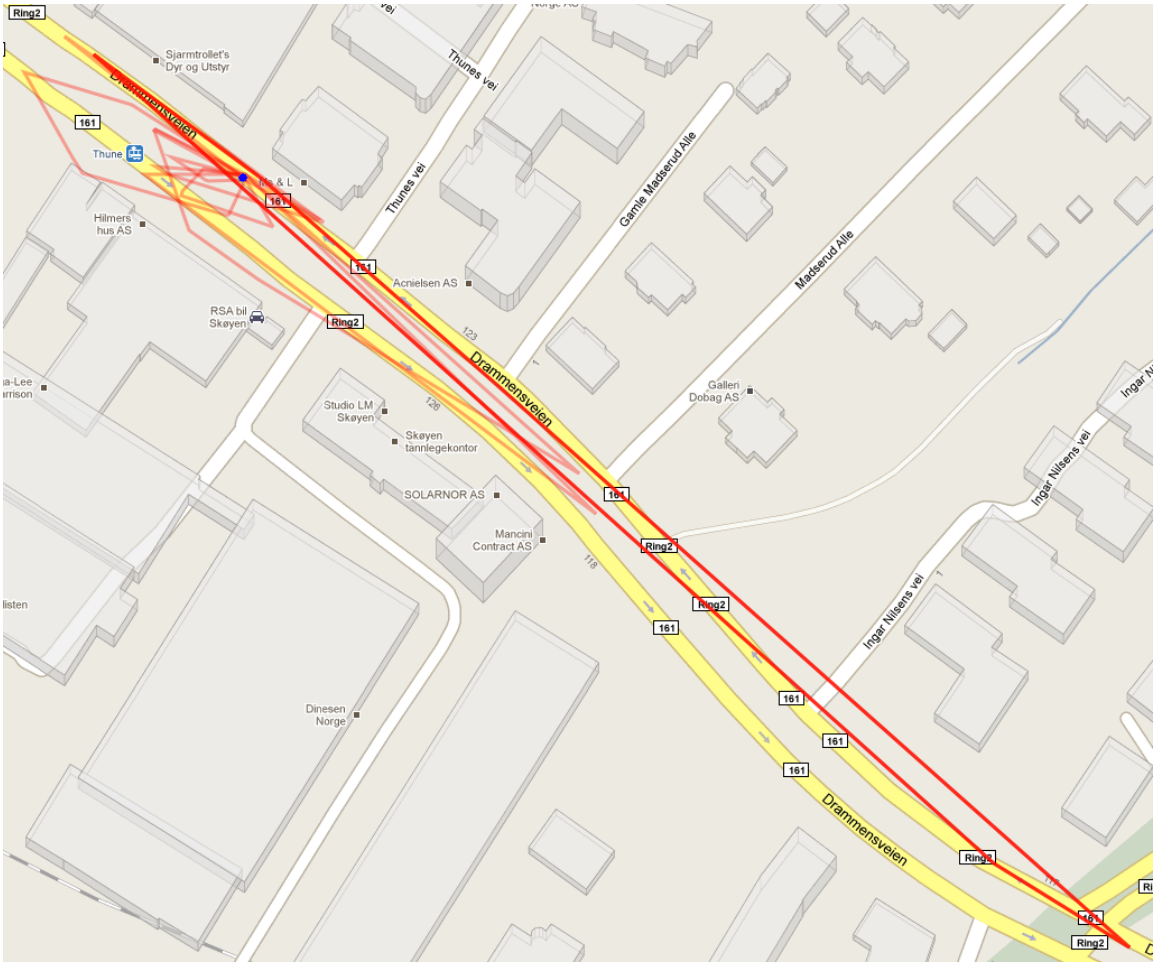


Figure 6.4: Second live example of the *Intersecting Areas* location estimation method. The intersection of the red areas are found to produce the estimated location area. The estimated location area, true location and estimated location all overlap in this example and are denoted by the blue dot.

6.3.9 Algorithm 7b - Intersecting Areas - Serving and Neighboring Cells - with E-CGI Fallback

In the same manner as algorithm 6a the *Intersecting Areas* method can fall back to E-CGI when encountering query fingerprints lacking the measurements needed to perform the *Intersecting Areas* method.

We expect the precision of this algorithm to lie somewhere between the precision of algorithm 2 and algorithm 7a. The performance should be identical to algorithm 2.

6.3.10 Algorithm 8a - Intersecting Areas - Serving Cell, Neighboring Cells and WLAN

This algorithm is also based on the *Intersecting Areas* location estimation method. The same technique is used. Instead of relying on only WLAN data or only neighboring cell data in addition

to serving cell information, both are used.

This algorithm only works on fingerprints containing a serving cell–signal strength pair and at least one WLAN–signal strength-pair or a neighboring cell measurement. The algorithm works on fingerprints lacking neighboring cell information if they contain at least one WLAN–signal strength pair, and vice versa.

Again we expect the precision and performance of this method to lie close to the DCM algorithm run on the same data: algorithm 5.

6.3.11 Algorithm 8b - Intersecting Areas - Serving Cell, Neighboring Cells and WLAN - with E-CGI Fallback

Again the *Intersecting Areas* method can fall back to E-CGI when encountering fingerprints not containing the data needed to perform the *Intersecting Areas* method.

Again we expect the precision of this algorithm to lie somewhere between algorithm 2 and algorithm 8a, the precise location determined by the ratio of fingerprints successfully analyzed by the *Intersecting Areas* method and fingerprints resulting in a fallback to E-CGI.

6.4 The Data-Sets

Two large collections of field-measurements were gathered. The resulting two datasets were kept separate throughout all of the testing. Both of the datasets were collected in a variety of different situations, both while walking, driving, riding public transportation and bicycling

The first dataset was gathered using Nokia devices running Symbian Series 60: N80, N95 8GB, E51 and E65. On the N95 8GB the internal A-GPS was used. On the other devices three external GPS receivers were used: Two Holux M1000C and one Holux GR-240. The data was mainly gathered in urban and sub-urban areas. Small amounts of data gathered in rural areas were included. Due to the limitation of available data through the Sybmian Series 60 API, this dataset only includes information on the serving GSM or UMTS cell with received signal strength, and identification, features and received signal strength of any WLAN networks observed in addition to GPS data. See table 5.1 page 55, 5.2 page 56, 5.5 page 58 and 5.7 page 59 for detailed information on the measurements this dataset includes.

The *Symbian dataset*, as this dataset will be referred to from this point on, totals 627.597 fingerprints. 129.662 of the fingerprints include WLAN measurements.

The second dataset was gathered using HTC devices running Android 2.2 and 2.1: Desire and Legend. The data was gathered in urban and sub-urban areas, with less sub-urban data than the Symbian dataset. This dataset, from this point on referred to as the *Android dataset*, includes information on the serving GSM or UMTS serving cell, neighboring cell information and visible WLAN networks, in addition to GPS data. See tables 5.1, 5.2, 5.3, 5.5, 5.6 and 5.7 pages 55 through 59 for detailed information on the measurements included in the Android dataset.

The Android dataset totals 29.666 fingerprints. 24.371 of the fingerprints include WLAN measurements.

The different algorithms tested require different types of measurements to be present in a query fingerprint to succeed. Table 6.1 page 77 provides an overview of algorithms eligible for each dataset.

	Required data			Datasets	
Algorithm	Serving	WLAN	Neighbor	Android	Symbian
1 - Cell Global Identity	✔			✔	✔
2 - Enhanced Cell Global Identity	✔			✔	✔
3 - DCM	✔	✔		✔	✔
4 - DCM	✔		✔	✔	✗
5 - DCM	✔	✔	✔	✔	✗
6 - Intersecting Areas	✔	✔		✔	✔
6.1 - Intersecting Areas E-CGI Fallback	✔	✔		✔	✔
7 - Intersecting Areas	✔		✔	✔	✗
7.1 - Intersecting Areas E-CGI Fallback	✔		✔	✔	✗
8 - Intersecting Areas Se	✔	✔	✔	✔	✗
8.1 - Intersecting Areas E-CGI Fallback	✔	✔	✔	✔	✗

Table 6.1: Overview algorithms: Data required and dataset eligibility

6.5 The Tests

We performed two sets of tests on the collected data. The first set of tests provided an initial indication of the precision and performance of each individual algorithm. In addition the first test was used as a quality control to determine what data should be used for the second set of tests.

The second set of tests was run on subsets of each of the datasets after removing data that did not pass the quality control. In addition the second set of tests were more robust, and timing data was gathered to allow the speed of each algorithm to be compared.

The first set of tests were highly time consuming when compared to the second set of tests, mainly due to a lot of time spend analyzing and comparing garbage data. In retrospect future test implementations should do as much garbage detection as possible during data import. Garbage detection should preferably be performed using a mathematical optimization algorithm.

Each set of tests was performed twice. Once on the Symbian dataset and once on the Android dataset. Each of the two sets of tests are described in detail below.

6.5.1 First Set of Tests

This set of tests was divided into three stages:

1. data import and training
2. calculations/testing of algorithms
3. statistical and manual analysis

All three stages were first performed individually on the Symbian dataset. The system was then reset, and the same three stages performed on the Android dataset.

During the *data import and training* stage each log-file gathered using the data gathering tools was parsed and imported as fingerprints in the database. Malformed data was removed or manually fixed. The small amount of malformed data removed was mainly due to faulty GPS drivers, unexpected application shutdowns and similar occurrences. The malformed data manually fixed was mainly WLAN SSIDs containing illegal characters of characters not recognized by the system.

After all of the data had been imported in the form of fingerprints, *data transformations and training* was performed. All of the imported fingerprints were converted to the entry types, mainly areas represented by convex hulls, needed for non-DCM location estimation methods.

Once import and training was completed, the main stage of the tests started: *testing the algorithms*. Each algorithm was tested over each and every log-file belonging to the dataset.

The log-files were parsed one by one. Each log-file entry was converted to a fingerprint. That fingerprint was then temporarily removed from the database and the trained system. The algorithm was then run using the fingerprint as input and the database as background. The estimated location, and any other data relevant to the algorithm such as penalty value, was logged together with the true location of the fingerprint tested and the difference between the two locations. The results were catalogued based on algorithm and log-file of origin. Finally the fingerprint was re-entered into the database and the trained system before moving on to the next entry in the log-file. This was repeated until all of the log-files used to produce the dataset had been tested.

The whole process was repeated for each individual algorithm. Once all of the algorithms suitable for the Symbian dataset had been tested over all of the log-files used to create the Symbian dataset, the whole process was repeated on the Android dataset for all of the algorithms.

Once all of the tests had been performed using both datasets, the results were analyzed for each of the datasets individually. First the initial precision and performance of the different algorithms were calculated for an early indication the algorithms' performances. Second the results on each log-file were examined and recorded to be used later to determine what data to base the second set of tests on.

The performance of each individual algorithm was determined by counting the number for fingerprints used to train the system and the number of fingerprints successfully analyzed by that algorithm and calculating the difference.

The precision was calculated using the difference in the true location, determined by GPS and the estimated location, generated by each algorithm, of each individual fingerprint. The mean, standard deviation, and five-number summary[1] were calculated for the whole set of observed differences of each individual algorithm. These measurements were used as an early indicator of the precision of each algorithm, and later compared to the second set of tests to ensure the results matched.

Finally the mean and standard deviation was calculated for each individual algorithm on each individual log-file. Outlier-files were determined and recorded statistically based on the mean performance of each file. This data was later used when determining the basis for the second set of tests.

[1] The five most important percentiles: sample minimum, lower quartile, median, upper quartile and sample maximum.

6.5.2 Second Set of Test

The second set of tests were divided into the following stages:

1. quality control
2. data import
3. randomization
4. data training
5. calculation/testing of algorithms
6. statistical analysis

All stages were performed individually for the Symbian dataset and the Android dataset.

The seconds set of tests was performed on a sub-set of all of the gathered data. The results of the first set of tests was used to determine what data was to be used for the second set.

We discovered several sources of garbage data. The first source of error was erroneous measurements. Such errors are generally caused by the GPS or the GPS API. The accuracy of GPS varies based on topography, location of the GPS antenna and the GPS satellite constellations. GPS reports its estimated accuracy, and our data gathering software logs all GPS data, such errors are normally easily detected.

Unfortunately errors caused by the GPS API are not detectable using accuracy measurements. Such errors are normally caused by the API erroneously returning a cached GPS measurement when the GPS comes back online after being paused for a period of time.

When dealing with WLAN data we detected a second source of error: non-unique BSSIDs. Even though the standard states that BSSID shall be a unique identifier, this is not always the case. We have observed WLAN networks on opposite sides of a country with the same BSSID. This type of error is due to non-conforming hardware or individuals changing their hardwares BSSID intentionally.

Since we know that the maximum range of a GSM BTS is 35km (see appendix A page i), an observation error of a single cell higher than 35km is a clear indication of bad data. The average range of a WLAN is much lower, thus detecting bad WLAN data is even easier. The highest discrepancy found in our tests was 328.030 meters.

A second, and in our data the largest, source of garbage data did not introduce faulty data, but rather introduced large amounts of useless data. The source was human error: neglecting to shut down or pause logging devices when ones location was static over a long time, resulting in hundreds or thousands of fingerprints from the exact same location.

The last source of garbage data does not involve erroneous or useless data, but the algorithms themselves. All of the algorithms have a threshold of training data needed before they function properly. The system must be trained with a certain amount of data from an area before it starts calculating reliable estimations for said area. Some of our data was gathered while traveling through areas a single time. We did not measure the threshold of the individual algorithms, but determined, using the first set of tests, that for all algorithms to function properly one generally needed at least two log-files from any given area.

Both qualitative and quantitative methods were used to uncover garbage data to be removed. Any outlier-files discovered in the first set of tests were manually inspected to determine the cause of imprecision. Generally the whole file was discarded when the imprecision was determined to be caused by an error. Large log-files were examined to determine if their size was due to neglecting to terminate the data gathering tool when stationary for longer periods of time. Such files were either discarded, or the data gathered in a stationary location was removed from

Algorithm	Higest score	Lowest score
Algorithm 3 on the Symbian dataset	22 000	750 000
Algorithm 3 on the Android dataset	19 500	38 000
Algorithm 4	1 700	20 000
Algorithm 5	1 700	49 000

Table 6.2: Highest and lowest fingerprint-score assigned to individual candidate fingerprints when running the DCM-algorithms on the datasets

the file. Finally we manually determined that removing the outlier-files also removed any areas below the data threshold needed for all algorithms to function properly.

The new Android dataset consisted of 21.885 fingerprints and the Symbian dataset 127.672 fingerprints. To speed up the testing process the Symbian dataset was split into three parts, each consisting of approximately 40.000 fingerprints. Each dataset was imported into a blank database, resulting in one Android database and three Symbian databases.

Each database was then cloned ten times. The fingerprints in each cloned database was then randomly split into two equal parts. Finally each cloned database was trained. The first half of the fingerprints was used to train the system, the second half was reserved for testing the algorithms. The result was 10 databases based on the Android dataset and 30 databases based on the Symbian dataset, each trained on a random selection of fingerprints, each with a set of fingerprints not included in the training ready for testing the algorithms.

Each algorithm was run using the fingerprints reserved for testing as input and the trained data as background. The estimated location, the true location of the tested fingerprint, the difference between the two, and any other data relevant to the algorithm being tested was logged. The results were catalogued based on algorithm and database. The process was repeated once for every algorithm on every database. Hence each and every algorithm was tested ten times over the same data randomly split into training and testing sets each time.

In addition the time spent training each database was logged. Furthermore, the time spent testing each set of fingerprints, with each algorithm, on each database was recorded.

6.6 Determining Optimal Penalty Values for Algorithms 3, 4 and 5

The DCM algorithm chosen for the tests, described in section 2.3.5 page 10, uses a penalty value. No description of the penalty value, only the use of a penalty value, is presented in Laitinen et al[5]. To correctly test the algorithms based on this method, the correct usage of the penalty term needed to be determined.

The DCM algorithms function by comparing the query fingerprint of interest to a set of stored fingerprints with known locations. The calculations are based on the difference in observed signal strength of the same radio access points between the fingerprint of interest and the stored fingerprints. Each stored fingerprint is assigned a score based on these differences, and the highest scoring fingerprint is used as the estimated location. The penalty term is used to penalize stored fingerprints for radio access points not present in both the query fingerprint of interest and the stored fingerprint being assigned a score. The question is how many points

should be removed from the final score for one missing access point?

The optimal penalty score will vary based on the overall score the algorithm assigns: A too low penalty value will have no impact on the final result, while a too high penalty will void the calculations resulting in only the penalties impacting the final result.

The overall score the algorithm assigns will vary based on several different factors: the dataset the algorithm is trained with, the type of area, the type of network the algorithm is analyzing measurements of and the network equipment used to gather the data the algorithm is trained with. How received signal strength is measured varies from network type to network type. Different network equipment measures received signal strength in different ways. Radio implementation and antenna configuration affects received signal strength. Finally the overall score will vary based on the surrounding network topography and the density of wireless networks. Table 6.2 page 80 illustrates this by showing the highest and lowest assigned score when running the DCM algorithms on the datasets.

Figure 6.5 page 82 compares the mean and standard deviation when running algorithm 3 on five randomly selected log-files with a penalty value of 1 and a penalty value of 10. The difference in penalty value has a huge impact on the result from *log-file 1* and *log-file 3* where we observe a difference in standard deviation of over 50 meters, a minor negligible impact on the result from *log-file 18* and *log-file 47* with a difference in standard deviation of a fraction of a meter and no impact whatsoever on the result from *log-file 4*.

To find an estimation of the optimal penalty value we ran the different relevant algorithms on a small random selection of files through a wide range of different penalty values. The mean and standard deviation was recorded for each log-file, and the mean and standard deviation over all of the log-files was recorded for each penalty value tested.

The choice of optimal penalty value was based on the mean and standard deviation over all of the log-files at each penalty value. The mean and standard deviation of the individual log-file was then used as a control. If the mean or standard deviation for a single, or a few, log-files were offset to a great degree from other log-files in the same set, or from the measurements over the whole set, this would warrant a closer examination.

Since the optimal penalty value varies based on not only the data in the database but the log-files (or live fingerprints) being analyzed, the findings are an estimation. We were simply interested in finding a good value for testing the performance of the algorithms coming as close as possible to the performance we would expect in the wild. The granularity was determined during testing and set at 500. We determined that variations below 500, though measurable, would only make sense if we were interested in finding the optimal value in a static system.

Figure 6.6 page 83 illustrates how we found the optimal penalty value for algorithm 1 when run on Symbian Series 60 data. We see that penalty value 1 results in a very bad result, drops quickly over values 10, 100 and 1000 and then becomes fairly stable. The zoomed in view shows where we found our optimal value by first finding the global minimum of the mean over all of the log-files and using the global minima of the individual log-files as a control.

If we look at the second graph which presents the measured standard deviation on the same files, run through the same algorithm, with the same penalty values we see that the graphs are quite similar. We have the same rapid drop over the low values, and more prominent stability over the rest of the measurements.

Note that the stability presented in these graphs is not absolute. All of the measurements vary slightly from penalty value to penalty value. However the variation around the optimal penalty value with the granularity we have chosen is in general only a fraction of a meter and not visible in this graph.

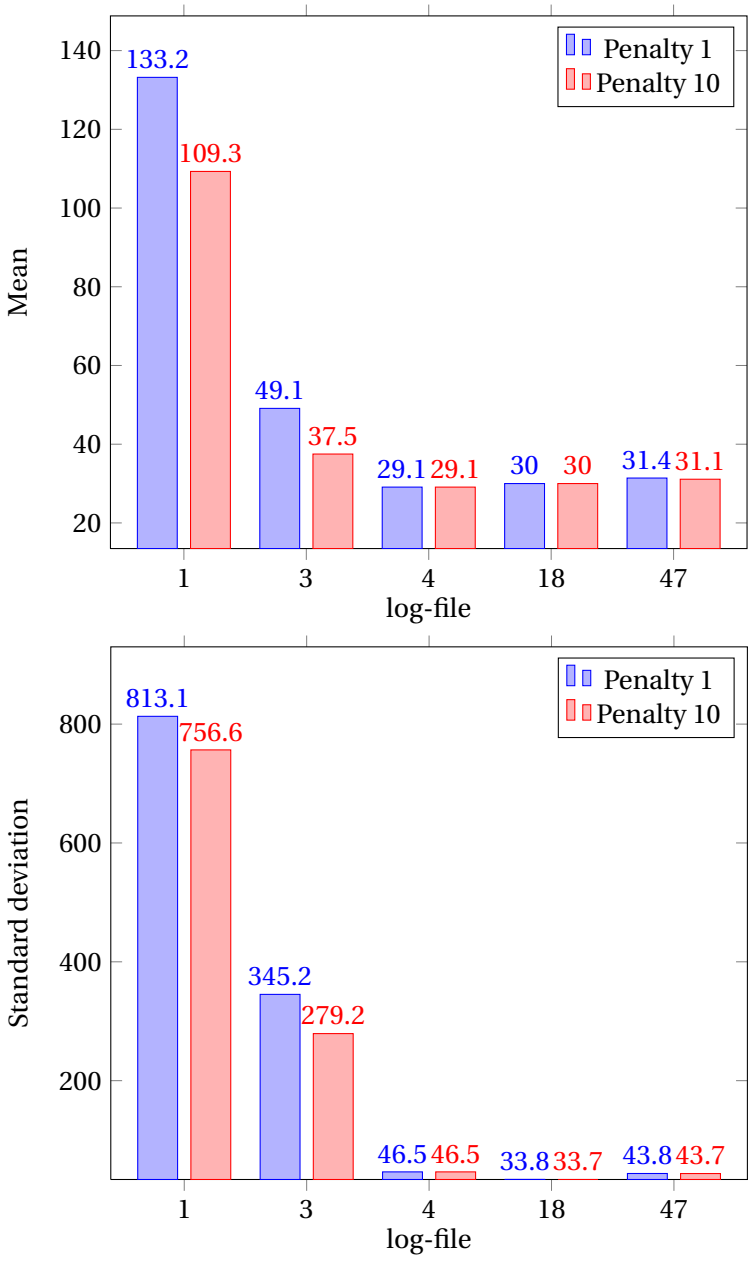


Figure 6.5: The influence of the DCM penalty value on different log-files

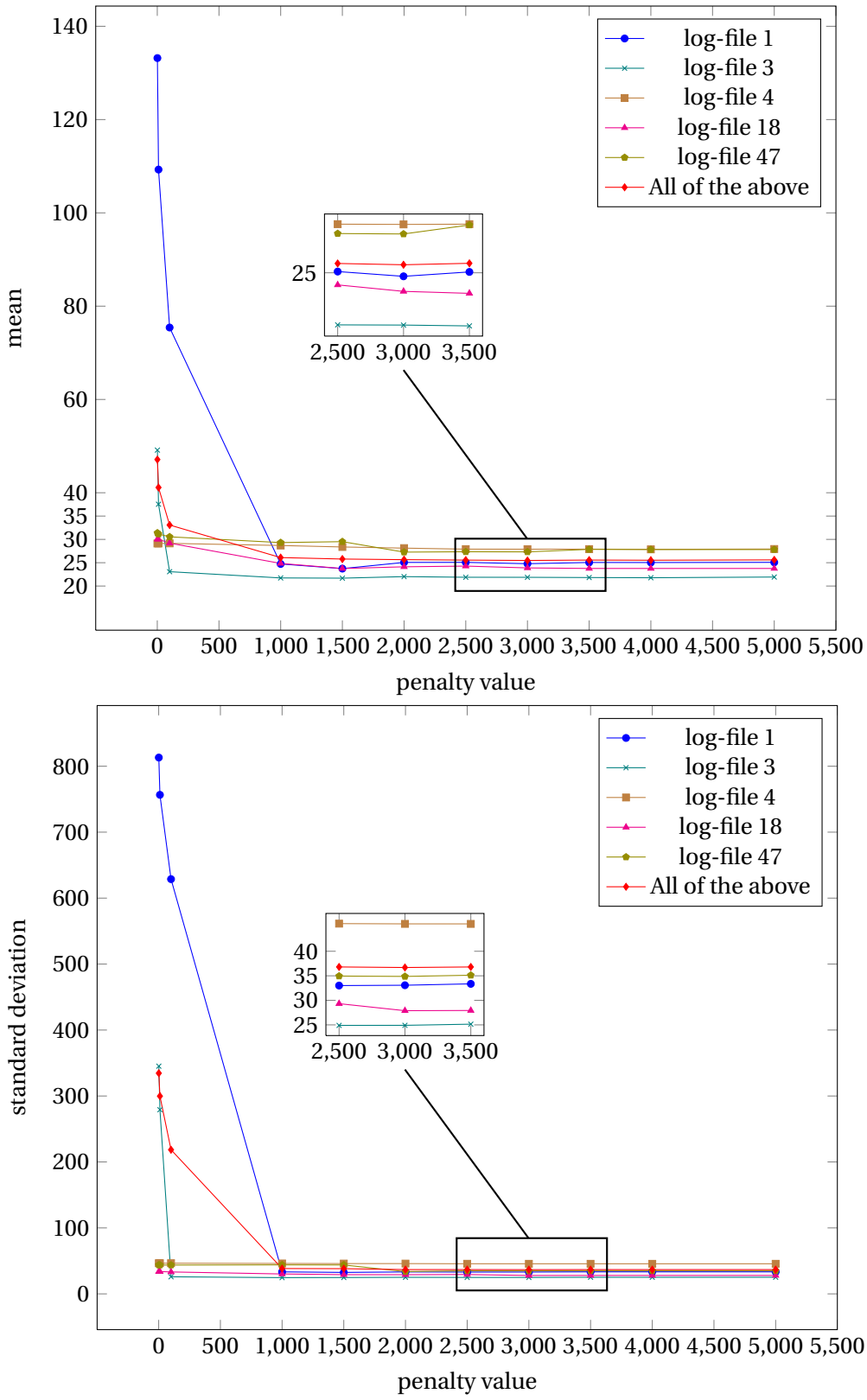


Figure 6.6: How the optimal penalty values were found

Algorithm	Dataset	Optimal penalty value
3	Symbian	3000
3	Android	9500
4	Android	7500
5	Android	8000

Table 6.3: Optimal penalty values for DCM methods on the datasets

Table 6.3 page 84 lists the optimal penalty values we found for each DCM algorithm on each dataset. Note that algorithm 3 is the only DCM algorithm suitable for the Symbian dataset.

We suggest that in a live production system using DCM fingerprinting methods such tests to find optimal penalty values should be performed on a regular basis to ensure that the value in use is always optimal since the optimal value varies based on the data in the database.

In addition we suggest using determining optimal penalty values for individual mobile station models or classes of models. By maintaining such individual penalty values and continuously updating them, better performance is achievable for the individual mobile stations. This suggestion is backed up by the estimated optimal penalty values for algorithm 3 on the Android and Symbian dataset seen in table 6.3 page 84.

Naturally we do not recommend manual methods, such as the method described above, when continuously tuning a live DCM based mobile location estimation system. Instead a suitable mathematical optimization algorithm should be used.

6.7 Results

6.7.1 Performance

In this section we first discuss the performance or success rate of the algorithms in the second set of tests presented in figure 6.7 page 85 for the Android dataset and figure 6.8 page 86 for the Symbian dataset. The figure 6.9 page 87, compares the algorithm performance between the Android and Symbian datasets. Finally we compare the performance of the different algorithms between the first and second set of tests.

Algorithms 1, 3, 4, 5, 6a, 7a, 8a match our expectations and are not discussed further. We predicted algorithm 2 would have the same performance as algorithm 1. Algorithm 2 performed lower than expected. This is due to the difference in granularity of the two, and the lack of training data for an area. When training for algorithm 2, a serving cell–received signal strength pair must be observed three times before training is successful (one observation is a point, two observations is a line, three observations is an area). The same goes for a single serving cell for algorithm 1. Since the training observations for a serving cell in algorithm 1 is the sum of the observations of all of the received signal strength areas for the same cell in algorithm 2, the frequency of areas with less than three observations will be higher in algorithm 2 than algorithm 1, which explains the lower performance.

The performance of algorithm 6b was predicted to be equal to the performance of algorithm 2. Algorithm 6b is slightly higher than predicted. The reason is that while algorithm 2 only relies on serving cell and received signal strength, algorithm 6b falls back on serving cell and received signal strength if no WLAN measurement is available. In some instances where algorithm 2 fails

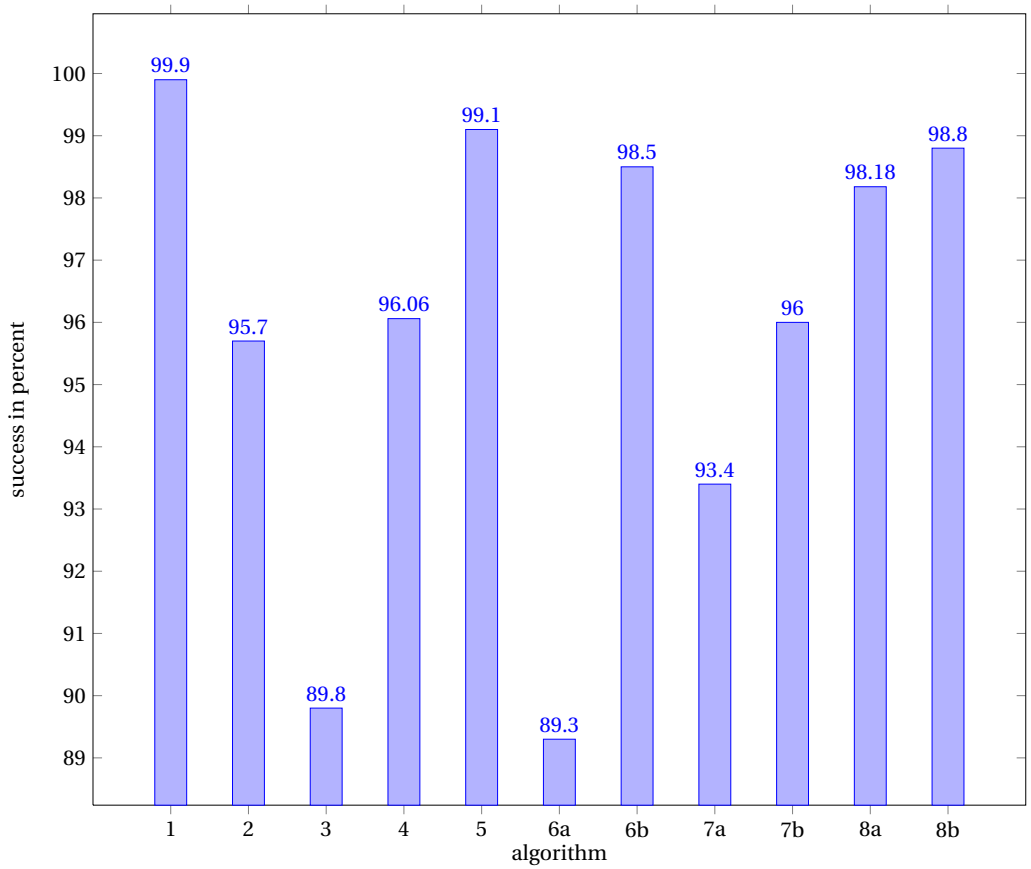


Figure 6.7: The success rate in percent of algorithms in the second set of tests on the Android dataset.

due to lack of training measurements, algorithm 6b will succeed because it has a well trained WLAN area to base an estimation on. The same goes for algorithm 7b. We predicted a performance equal to algorithm 2, but the performance is slightly higher. The cause is the same as with algorithm 6b except that it is well trained neighboring areas, rather than WLAN areas, that boosts the performance. Finally, the same goes for algorithm 8b. A well trained neighboring area or WLAN area will cause the algorithm to succeed, even if a well trained serving cell–received signal strength area does not.

Notice that the performance of algorithm 3 and 6a on the Symbian dataset is extremely low when compared to all the other algorithms. Their performance on the Android dataset is also lower, but to a much lesser extent. The reason for this is their dependency on WLAN data to function. Algorithm 3 and 6a cannot successfully analyze a fingerprint containing no WLAN measurements (6b on the other hand, can, since it falls back to standard E-CGI if the fingerprint does not contain WLAN measurements). Remember that the Symbian dataset contains less data gathered in sub-urban areas than the Android dataset, and that while the Symbian dataset contains some rural measurements, no such measurements are present in the Android dataset. Generally the density of WLAN networks is higher in sub-urban areas than in rural areas, and much higher in urban areas. Both the lower performance of these two algorithms, and the difference between the performance between the two datasets is as expected.

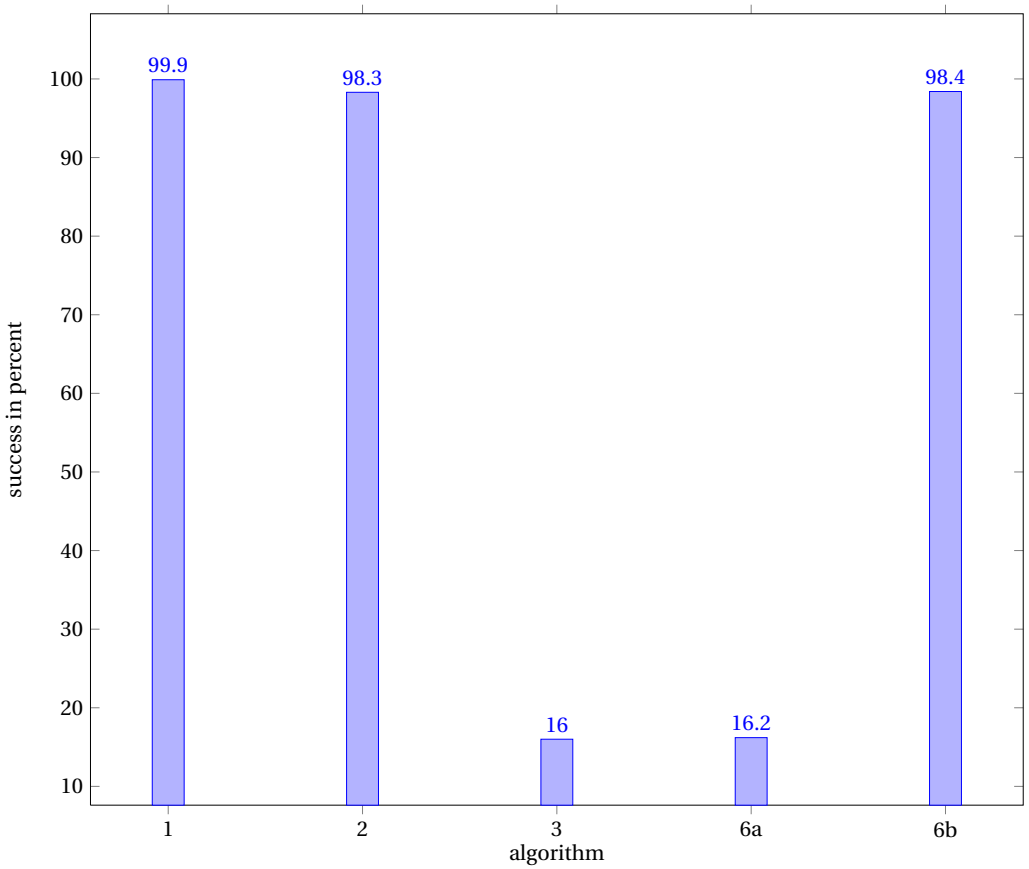


Figure 6.8: The success rate in percent of algorithms in the second set of tests on the Symbian dataset.

Finally, comparing the results of the two datasets show approximately the expected result: the relationships between the different algorithms are for the most part the same, while the actual performance is either the same or better in the second set of tests. This is illustrated in figure 6.10 page 88 which compares the results of the Android dataset between the two sets of tests and figure 6.11 page 89 which compares the Symbian dataset between the two sets of tests. The lower the threshold for an algorithm failing due to lack of training data in an area, the greater the difference between the datasets, since areas with a very low training density were removed before the second set of tests.

We also see that, between the two sets of tests, the performance of algorithm 4 and 5 have switched places. This is due to the fact that parts of the sub-urban data was removed from the second set of tests as part of removing areas with too few measurements for accurate training, hence less data lacking WLAN measurements is present in the second set of tests. This is in line with our prediction that the performance of the two vary based on the availability of WLAN measurements, and that algorithm 4 outperforms algorithm 5 in areas with low WLAN density and vice versa.

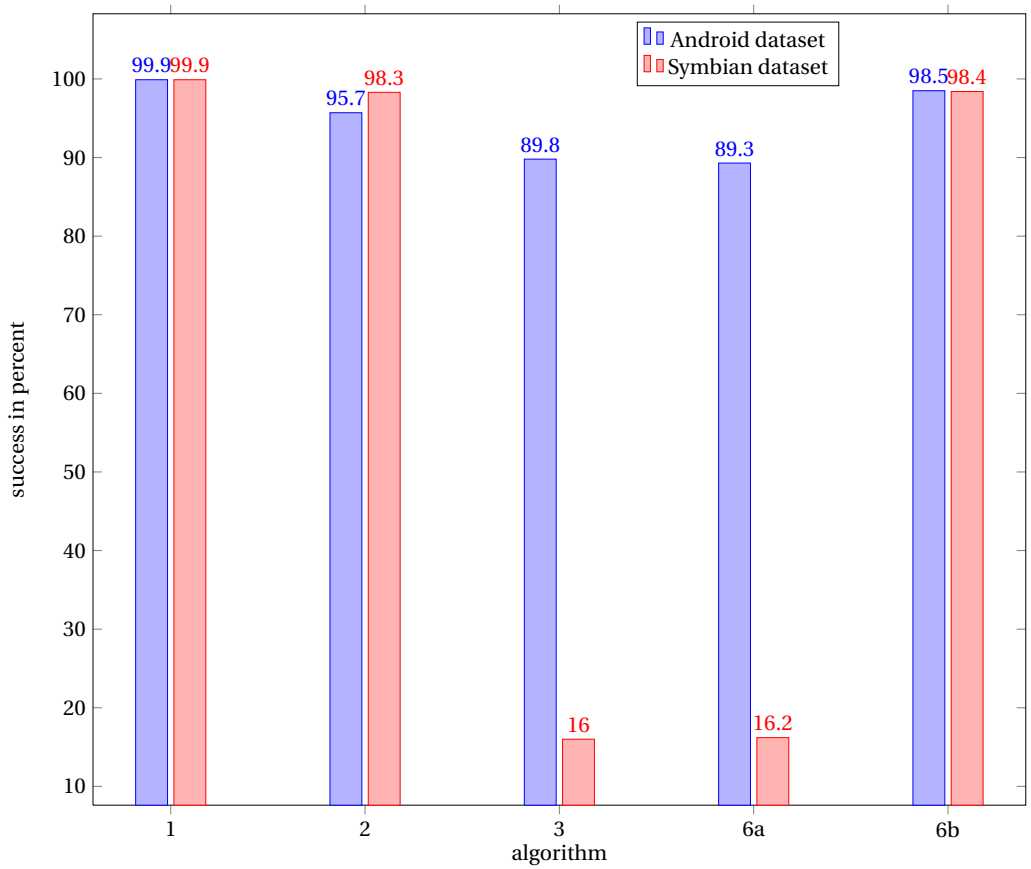


Figure 6.9: The success rate in percent of algorithms in the second set of tests between the Symbian and Android datasets.

6.7.2 Precision

In this section we investigate the precision of all of the tested algorithms on both of the datasets and both of the sets of tests. The findings are illustrated using box-and-whisker diagrams, which present data as quartiles.

Quartiles divide statistically ranked members of a data-set into four groups, each representing a fourth of the distributed population. The first quartile, Q1 cuts off the lowest 25% of the data. The third quartile, Q3 cuts off the highest 25% of the data. The second quartile, Q2 cuts the data in half, thus Q2 is the median of the data-set and is sometimes denoted M.

The interquartile range, IQR is a measure of statistical dispersion and equal to the difference between Q3 and Q1. In a data-set points outside of 3/2 of the IQR are considered outliers.

There is no universal agreement on how to calculate quartiles, and at least five different methods are commonly in use[2]. We selected the method known as the "Tukey"-method[47, pages 9,54,61,223] as Tukey invented[48, pages 39–43] the box-and-whisker graphs we use to present our data. Where n is the sample size the position of Q1 is calculated by $\frac{n+3}{4}$ if n is odd and $\frac{n+2}{4}$ if even. The position of Q3 is calculated by $\frac{3n+1}{4}$ for odd n and $\frac{3n+2}{4}$ for even n. If the position of Q1 and Q2 are not whole numbers the mean of the value on each side of the position

[2] http://mathforum.org/library/drmath/view/60969.html

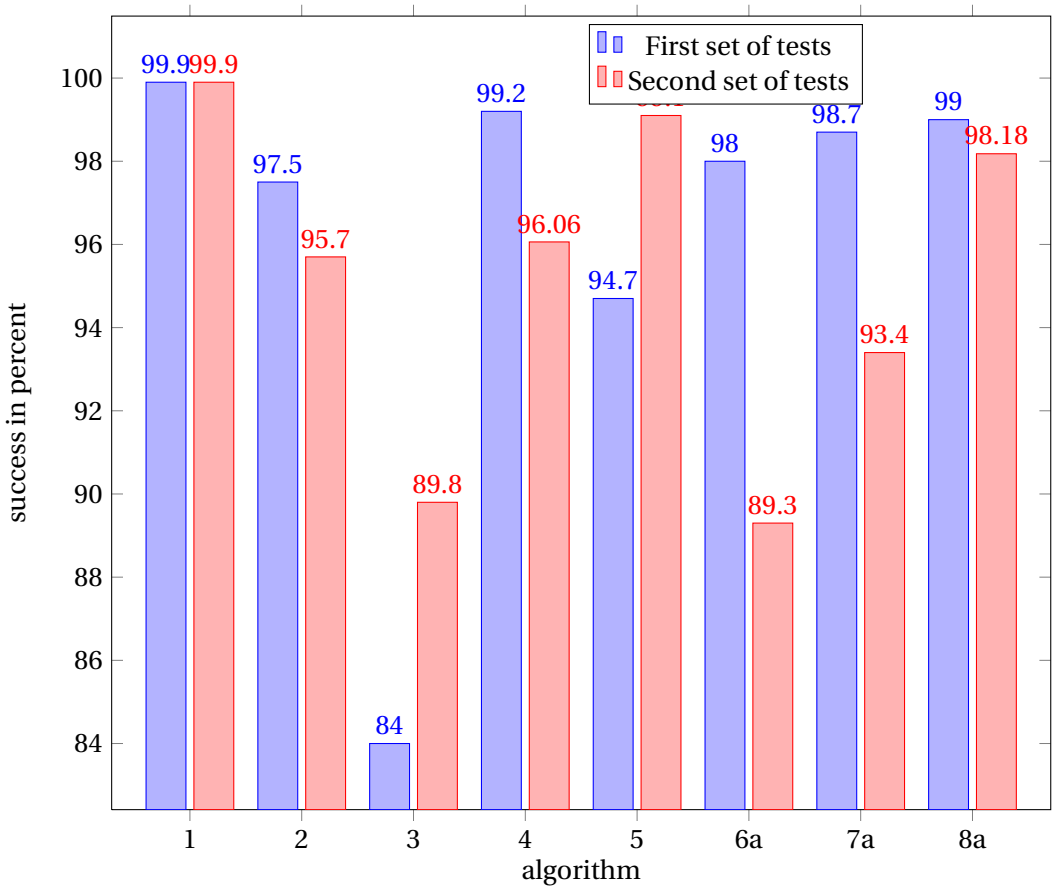


Figure 6.10: The success rate of algorithms in percent between the two datasets on the Android data

is calculated.

The quartiles are used when comparing the algorithms using box-and-whisker diagrams. Figure 6.12 page 88 shows an example diagram. The box in the center starts and ends at Q1 and Q3 thus it represents the middle 50% of the data. The line dividing the box in two represents Q2, the statistical median. The vertical lines above and below the box extend to 2/3 of the IQR and illustrate the non-outliers above and below the middle 50% of the data. Note that in our diagrams we cut off the lower vertical line at 0. Outliers are not plotted in our box-and-whisker diagrams.

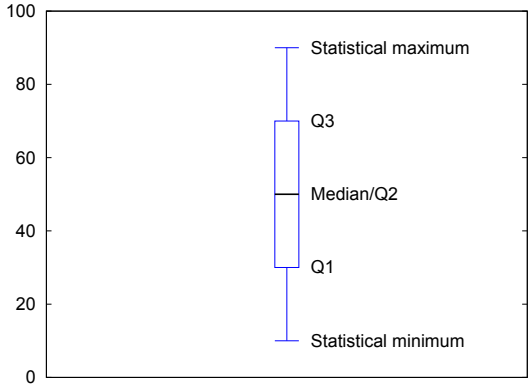


Figure 6.12: Example of box-and-whisker diagram

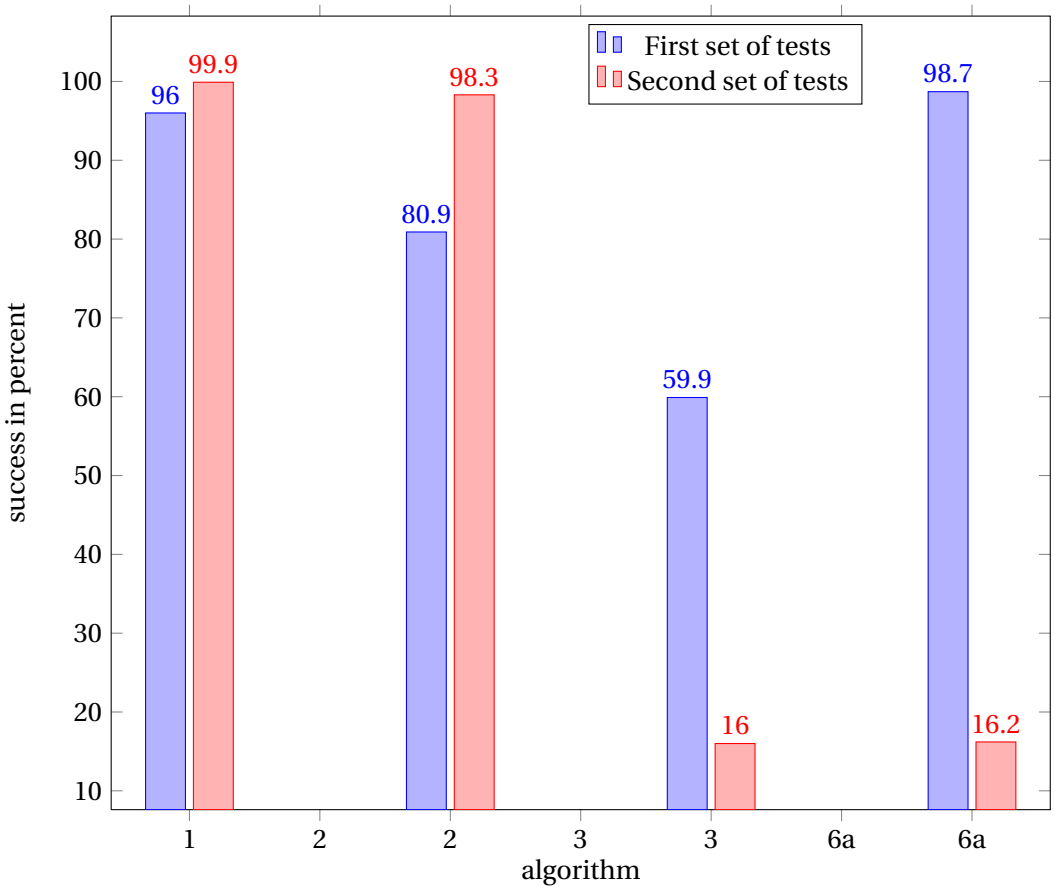


Figure 6.11: The success rate of algorithms in percent between the two datasets on the Symbian data

Figures 6.13, 6.14, 6.15, and 6.16 pages 90, 92, 91 and 93 illustrate the precision of the algorithms on the first and second set of tests divided by datasets.

In this section we compare the predictions in section 6.3 pages 69– 76 of the precision of each algorithm to the actual precision found through our tests.

First, however, we must address the issue of neighboring cells. The initial expectations were that using neighboring cell information when estimating locations would improve precision to a large extend. The tests invalidate this assumption: The increased precision due to neighboring cells is much lower than expected, and in some instances, using neighboring cells actually decrease precision. This is best illustrated in figure 6.16 page 93. Here we see that algorithms 7a and b as well as algorithms 4

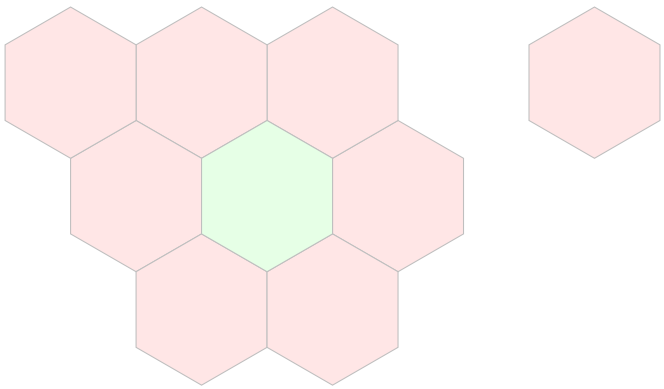


Figure 6.17: Example of neighboring cell broadcast

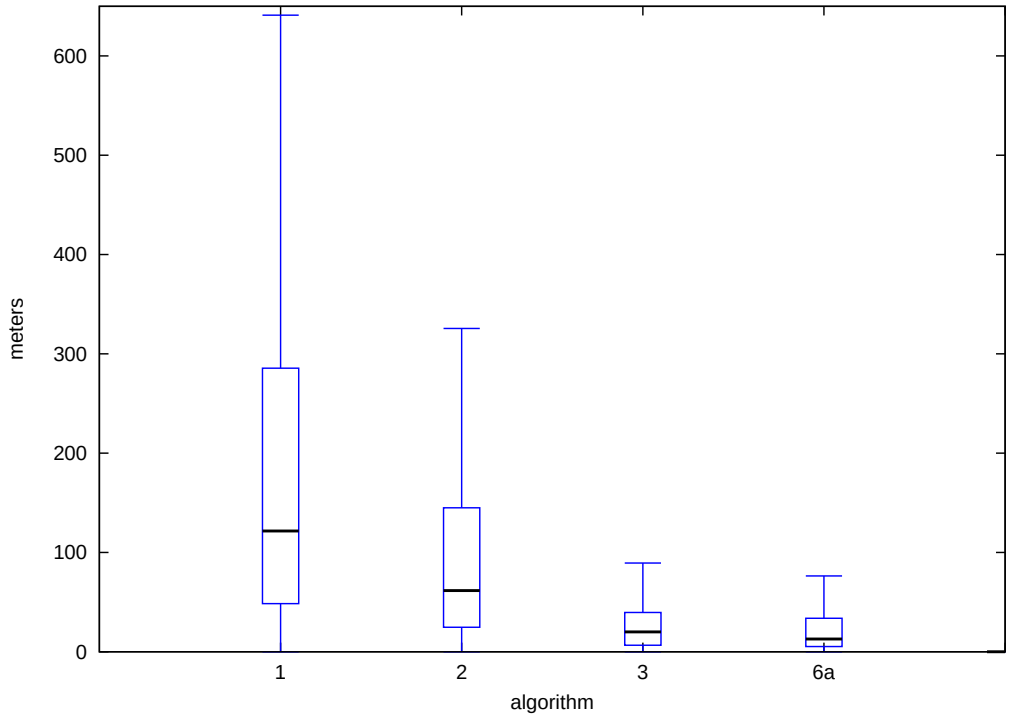


Figure 6.13: Comparing algorithms on the Symbian data-set with the first set of test

and 5 have a lower precision than expected. The precision of algorithms 4, 7a and 7b is no better than the precision of simple E-CGI.

The reason lies in how the neighboring cell information is used: neighboring cell lists are maintained by BTSs so that mobile stations can determine when to perform cell re-selection. Each BTS maintains a list of other BTSs close by and broadcasts it to any mobile stations listening. The mobile stations then measure the received signal strength of each of these BTSs to determine if a cell re-selection is needed. This results in the area a single cell is observed as a neighboring cell being much larger than the same cell is observed as a serving cell. Exactly how many BTS and over what areas a cell is broadcast as a neighbor varies based on network topography and up to each network operator. Figure 6.17 page 89 illustrates an example where the green serving cell is broadcast as a neighboring cell by the red cells.

The *Intersecting Areas* method combining the use of neighboring cells and WLANs (algorithm 8a and b) is largely unaffected by the effect of the large neighboring areas when compared to DCM. *Intersecting Areas* mitigates the effect by relying on *intersecting* the areas. Since WLAN areas are much smaller than neighboring cell areas, intersecting the two will effectively remove most of the neighboring cell area.

We predicted algorithm 1 to show the lowest precision in the tests, followed by algorithm 2 with the second lowest precision. This holds true in both sets of tests on both the Android and the Symbian dataset aside from the mentioned problem with algorithms relying on neighboring cells.

Algorithm 3 we predicted to have the second best precision, only surpassed by algorithm

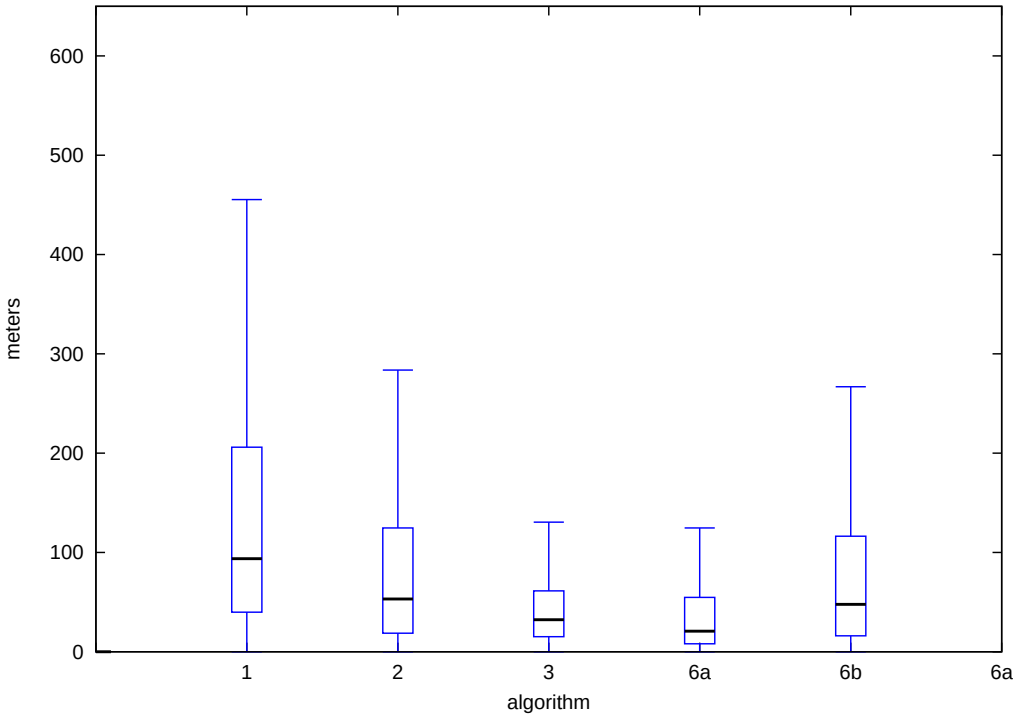


Figure 6.14: Comparing algorithms on the Symbian data-set with the second set of test

5. This was false due to the mentioned problem of neighboring cell areas. Algorithm 3 outperformed algorithm 5 in terms of precision. On the positive side, the *Intersecting Areas* algorithms, except algorithms 7a and b, outperformed algorithm 3 in terms of precision.

Algorithm 4 was predicted to show the lowest precision of the DCM algorithms in the test. This held true. However, the precision was lower than expected, and the precision of algorithm 4 was close to the precision of simple E-CGI.

The prediction that algorithm 5 would have the highest precision of the DCM algorithms, thus having the highest precision of all of the algorithms was false, again due to the neighboring cell area problem. Instead algorithm 5 had the second lowest precision of all the algorithms.

The prediction that the precision of algorithm 6b lie between algorithm 2 and 6a holds true, as does the prediction that algorithm 7a lies close to algorithm 4, though algorithm 4 lies lower than expected.

Again algorithm 7b was predicted to have a precision between algorithm 2 and 7a, which it does. The precision of algorithm 7b is, however, lower than we initially expected, due to the problem of large neighboring areas.

Finally we predicted the precision of algorithm 8a to lie close to the DCM algorithm based on the same data, algorithm 5, and algorithm 8b to lie somewhere between algorithm 2 and 8a. The precision of algorithm 8a is much better than algorithm 5. This is due to algorithms 8a and b handling the problem of large neighboring cell areas much better than the DCM-based algorithms. Algorithm 8b has a bit higher precision than expected.

When comparing the two sets of tests we see the exact same trend over all of the algorithms

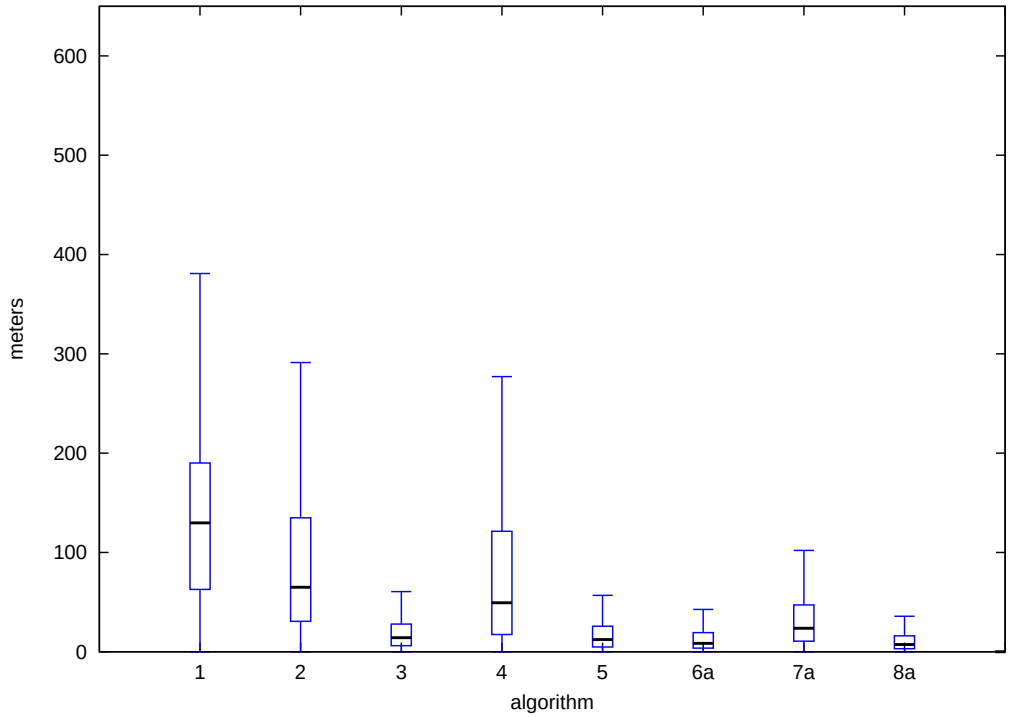


Figure 6.15: Comparing algorithms on the Android data-set with the first set of tests

for both the Android and the Symbian datasets. When comparing the two datasets we see the same overall trend with one exception: The precision of algorithm 6b is quite a bit lower on the Android dataset. This is due to a higher concentration of samples from sub-urban areas in the Symbian dataset causing a higher frequency of fall-backs to E-CGI due to the lack of WLAN networks.

6.7.3 Time

As mentioned we timed the calculations performed for each algorithm, in addition to the training, in the second set of tests. The tests were performed on two different sets of multicore machines. All data was loaded to RAM, and none of the tests used more than a small percentage of the RAM available on the machines. The tests were not multithreaded so a single test was run on a single core. The first set of machines had AMD Opteron 8218 1GHz processors, the second Intel Xeon CPU L7555 1.87GHz processors.

Table 6.4 page 93 lists the training times per fingerprint listed by training type, while table 6.5 page 94 lists the training times per fingerprint listed by algorithm. Note that several of the training types are common for multiple algorithms, and need only to be trained once even if used by several algorithms: the results of the *cell* training is used by algorithms 2, 6a, 6b, 7a, 7b, 8a and 8b; the results of the *wlan* training is used by algorithms 6a, 6b, 8a and 8b; the result of the *neighbors* and *neighbors3g* training is used by algorithms 7a, 7b, 8a and 8b. The *simplecell* training is used solely by algorithm 1.

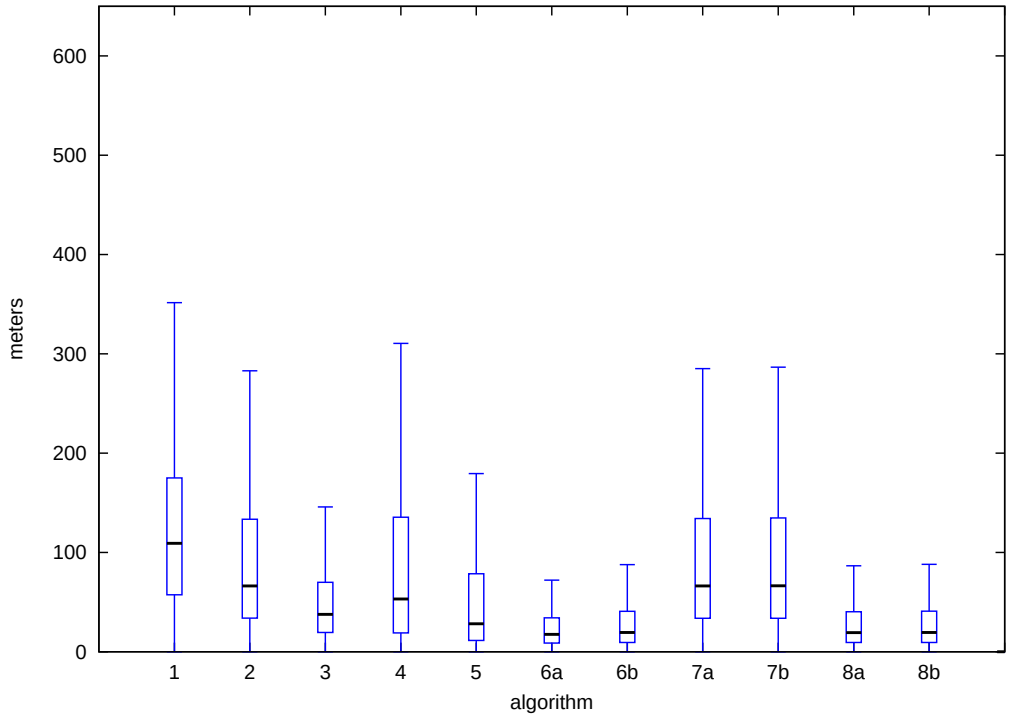


Figure 6.16: Comparing algorithms on the Android data-set with the second set of tests

Training type	Time on 8218	time on L7555
simplecell	.000050	.000019
cell	.000071	.000038
wlan	.047279	.017133
neighbors	.027417	.023901
neighbors3g	.000498	.000400

Table 6.4: Average training time per trained fingerprint in seconds listed by type

When comparing these numbers keep in mind that training only happens once. In the test setup, training happens once before the algorithms are tested, while in a live system training would be an ongoing process.

The time spent training the system is as expected: The time is correlated to the density of observations. Hence training neighboring cell information measurements takes longer than training serving cell information, since a fingerprint only contains a single serving cell, but up to eight neighboring cells. Similarly training WLAN measurements takes much longer, since there is no upper limit to the amount of WLAN measurements in a single fingerprint. Finally note

Algorithm	Time on 8218	time on L7555
1	.000050	.000019
2	.000071	.000038
6	.047350	.017171
6.1	.047350	.017171
7	.027986	.024339
7.1	.027986	.024339
8	.075265	.041472
8.1	.075265	.041472

Table 6.5: Average training time per trained fingerprint in seconds listed by algorithm

Algorithm	Time on 8218	time on L7555
1	0.005383	0.008095
2	0.005621	0.008930
3	0.749295	11.088524
4	16.210149	17.802947
5	18.815485	14.625443
6	0.021477	0.023968
6.1	0.037931	0.008301
7	0.003671	0.003938
7.1	0.003632	0.004676
8	0.005185	0.005112
8.1	0.005067	0.006172

Table 6.6: Average algorithm processing time per fingerprint in seconds

the difference between neighboring cell information and neighboring cell information on 3G. This difference is due to the fact that the networks we gathered data from generally present less neighboring cells on their UMTS networks than on their GSM networks.

Table 6.6 page 94 lists the average time each algorithms spent processing a *successful* location estimation. Only the number of fingerprints the algorithms successfully estimated a location for were included when calculating the spent time. All of the algorithms were implemented such that the time spent on non-successfully estimated fingerprints is negligible compared to the time spent on successfully estimated locations.

Comparing the average time spent analyzing a fingerprint by the different algorithms shows an extreme difference between the DCM methods and the other methods. We actually see a difference of over 3000% between the *intersecting areas* methods and the DCM methods. Even when taking the time spent training the system into consideration, the differences are huge. The *intersecting areas* methods are on average comparable to the simple CGI and E-CGI methods. One must, however, keep in mind that these measurements are implementation specific. The database, database access, and algorithms themselves were not optimized beyond normal

indexing. Generally the system was tuned toward doing as much as possible in code rather than in database. In a live system, finely tuned towards the algorithms in use, we would not expect to see such extreme differences. We would still expect to see the DCM methods to perform slower and require higher CPU and memory usage.

One last, and rather surprising, observation remains: The average time difference between the two different processors running on algorithm 3. The 8218 core is on average more than 11 times faster processing fingerprints with algorithm 3. The only explanation is differences between the two architectures. We have run multiple extra tests and implementations to ensure that the difference is not due to the data, database or implementation of the algorithm.

6.7.4 The Need for a Standard Dataset

Initially we wanted to compare the results of our tests with other results found in the literature, expecting minor discrepancies due to samples from different areas and different network equipment. Unfortunately, data worth comparing with was near impossible to find. As examples the article presenting the DCM method we choose[5] gathered fingerprints without using GPS but relied on average speed and manually recording locations to calculate fingerprints true locations. In addition they used grids to meticulously map out areas. Zimmermann et al[7] based their estimations on data obtained from a network operator. Kahlaf-Allah and Kyamakya[15] based their work on existing propagation prediction models. LaMarca et al based their *place lab*[49] on gathering data from the Internet provided by groups of war driving enthusiasts. Everywhere we looked we found results that were not compatible for comparison with each other. This was a huge motivational factor to create a flexible test system, with the possibility to create large test-sets and make them freely available.

6.7.5 Conclusions

The most exiting results of the tests are the success rate and precision of the *Intersecting Areas* location estimation method. On its own the *Intersecting Areas* method has a slightly lower success rate compared to the DCM methods, but when allowing the method to fall back on E-CGI the situation changes. Algorithm 6b, based on serving cell and WLAN only, has a much higher success rate than algorithm 3, the DCM method based on the same data. Algorithm 7b compares directly to its DCM peer, algorithm 4, with a .06% difference, and algorithm 8b with its DCM peer, algorithm 5, with only a .3% difference.

CGI (algorithm 1) has, not surprisingly, the highest success rate. However, as the precision of the CGI method is too low for most types of LBSs, the CGI method should only be used when no other option exists.

E-CGI (algorithm 2) improves precision, but is still outperformed in terms of precision by most of the other algorithms. In addition, E-CGI has a lower success rate than many of the other algorithms in the test. In conclusion E-CGI should be reserved for instances when more advanced methods are not available due to lack of samples, either in the system or on the querying device. Generally this means devices not able to measure WLANs or neighboring cell information, or devices lacking neighboring cell information presiding in areas without any WLANs to measure. E-CGI should, for this reason, not be disregarded as a method. Furthermore due to the similarities between E-CGI and *Intersecting Areas*, they both rely on areas, one of the benefits of *Intersecting Areas* is that it allows seamless fall back to E-CGI. Not only does this boost the per-

formance of the *Intersecting Areas* method, but also allows it to successfully calculate location estimations in areas and for mobile stations with limited samples.

In terms of precision the *Intersecting Areas* method showed the best results for all of the different types of network samples: WLAN, neighboring cells and the combination of the two. Neither DCM methods or the *Intersecting Areas* method can improve the precision of CGI or E-CGI when only a single serving cell, or serving cell–received signal strength is available. However, the *Intersecting Areas* method provides fall-back to E-CGI with negligible increases of processing time, and without the need for additional data storage. The fall-back mechanism is trivial, and does not require any changes to the algorithm or it's implementation. In addition, fall-back to CGI when the data needed for E-CGI is unavailable, can be implemented. Implementing CGI fall-back does require a small amount of additional data storage.

In terms of processing time and speed the *Intersecting Areas* method naturally requires more than the simple CGI and E-CGI methods, as it handles larger sets of samples. When compared to standard DCM, *Intersecting Areas* offers a huge improvement.

Besides the comparison of the algorithms this tests shows one interesting result: the effect of using neighboring cell information. We see that in urban and sub-urban areas, or more specifically areas with a high concentration of WLAN networks, neighboring cell information not only does not improve precision, but actually reduces precision when using DCM. Thus, when using DCM, neighboring cell information should not be used at all when enough WLAN data is available.

The *Intersecting Areas* method is a location estimation method that rivals standard DCM in terms of successful results. It offers better precision, less data storage, is quicker and less processing intensive, and offers better accessibility by successfully working on a wider range of network samples. In addition the training data used by the algorithm is stored in such a way that it is free of any data linkable to individuals or contributors, and is safe to use in a freely available training set. Finally *Intersecting Areas* can easily be extended to include new types of networks samples, making it flexible towards new types of networks and network equipment.

Our tests show significant benefits of using *Intersecting Areas* in urban and partially sub-urban areas. They cannot, however, be used to draw conclusions about the overall achievements of the algorithms. The tests do not show how the algorithms would perform in rural areas, though we have theorized about this. Further tests should be conducted to conclude how *Intersecting Areas* performs compared to other algorithms in rural areas, in other cities, and on other mobile networks before concluding on the overall achievements.

6.8 Summary

In this chapter we selected a set of different mobile location estimation methods, tested them, and compared them to the *Intersecting Areas* method using the test system presented in chapter 5.

We demonstrated that the *Intersecting Areas* method, when used in urban and partially sub-urban areas, is superior to the standard DCM method in terms of speed, precision, data storage and that *Intersecting Areas* and standard DCM are comparable in terms of success rate.

We saw how the *Intersecting Areas* method relies on falling back to CGI or E-CGI to be available to the widest possible range of network equipment, and that the *Intersecting Areas* method easily can be extended to function on future networks and network equipment.

We also found that while we expect basing algorithms on neighboring cell information can

improve their precision in rural areas, neighboring cell information not only does not improve precision in urban areas, but actually reduces it for some algorithms.

Finally we saw how *Intersecting Areas* as opposed to DCM can successfully be used to create a freely available training database without endangering the privacy and anonymity of the contributors.

CHAPTER 7

CONCLUSION

The outcome of this thesis can be divided into three major parts: a suggested design for a privacy preserving and open mobile location estimation system; a location estimation method providing speed, high precision, and simplicity, focused on privacy preservation; a test system created to ease gathering large sets of data and testing multiple location estimation methods. Since designing a location estimation method was an integral part of designing a privacy preserving and open mobile location estimation system, it will be presented first.

This chapter summarizes the results of our efforts, our main contributions, and investigates some unanswered issues and shortcomings. Finally we present the issues we consider worth while investigating that could not be included due to time restraints.

7.1 The Intersecting Areas Location Estimation Method

The location estimation method used in a privacy preserving and open mobile location estimation system must not store any data that can compromise its users. Our goal was to create a location estimation method that could compete with the simplicity of the simple CGI and E-CGI methods, compete with the precision of more advanced DCM methods, while removing any privacy issues entangled in the data stored on the back-end, and minimizing the privacy issues of data-transfer.

In most aspects *Intersecting Areas* is a success. The method rivals, and when used only on WLAN far outperforms, the DCM methods tested in terms of success rate.

We implemented *Intersecting Areas* to fall back on E-CGI to allow it to rival E-CGI in terms of success rate and availability on mobile stations and in areas lacking samples. Falling back to E-CGI requires no extra storage or calculations on the back-end. *Intersecting Areas* can be extended to allow a second fall back to standard CGI when E-CGI fails. This would, however, require an additional set of trained areas, and a minimal amount of extra calculations on the server. On the other hand, this allows *Intersecting Areas* to rival CGI in terms of success rate and availability.

Also in terms of precision the *Intersecting Areas* method outperforms the implemented DCM methods. The *Intersecting Areas* method based on WLAN has the highest precision of all of the methods in the test, and rivals each of the DCM implementations when based on the same types of samples.

Furthermore *Intersecting Areas* can easily be extended to include any type of network and network sample easily. The method needs not be changed. One only needs to create a new list of areas in the database, and instruct the algorithm to train on, and utilize the new type of samples. This renders the *Intersecting Areas* method flexible and adaptable to new types of networks and methods introduced in the future.

In terms of time and processing the *Intersecting Areas* method is more comparable to CGI and E-CGI but *much* much less demanding than the DCM methods in the test. Also in terms of storage space needed the *Intersecting Areas* outperforms DCM. While the DCM methods rely on storing as much data as possible, the *Intersecting Areas* method has a fairly low upper limit of needed space. In addition to actual space, this also allows quicker and simpler database access. We expect *Intersecting Areas* to greatly lighten the server-side load both in terms of storage, memory and CPU-usage compared to DCM. Furthermore, the complexity of determining optimal penalty values found in some DCM methods is not present in *Intersecting Areas*.

In terms of privacy *Intersecting Areas* is a success when regarding the storage of data. The method does not rely on storing unlimited amounts of fingerprints as DCM does. Instead the system is only updated when a network sample datum is detected in a location outside of the area surrounding previous observations. In addition, since the data is stored as areas, it is not linkable to any user. No data about or linkable to the contributing user is required by the *Intersecting Areas* method.

Thus the *Intersecting Areas* method can be used to create a privacy preserving, crowd sourced, free location estimation services, as the main goal of this project specified. The data gathered and used by the *Intersecting Areas* method can be stored in such a way that one can freely release said data to the public without compromising any contributers' privacy or anonymity. It is worth noting that the same goes for data stored for use with the CGI and E-CGI methods, a feature we have used to enhance the performance of *Intersecting Areas* by providing fall-back to these methods.

One of the original goals of creating a new mobile location estimation method, however, is not reached by the *Intersecting Areas* method: preserving privacy and anonymity of data-transfer. When compared to the DCM methods, the only impact on privacy preservation methods *Intersecting Areas* has is to reduce the frequency of crowed sourced updates needed by the system. When compared to CGI and E-CGI, *Intersecting Areas* improves nothing. In fact, has a negative impact, by increasing the amount of data in each network sample, each query and each reply. The result is that a system using *Intersecting Areas* must apply other means of protecting users' anonymity and privacy such as encryption, onion-routing over peer-to-peer networks, or the like.

7.2 The Privacy Preserving Mobile Location Estimation System

Our main interest in creating a new mobile location estimation system and a test bench was to be able to create a privacy preserving, and open mobile location estimation service. Our main motivation for creating such a system was threefold: we feel existing service providers do not do enough towards ensuring their users' privacy and anonymity; we feel users should be able to be informed of their own location without having to give up anything, such as anonymity, in return; we want to ameliorate the conflict of interest between location estimation services and cloaking services and their users.

In chapter 3 we suggested a design for such a system. We feel that this design when used together with our suggested *Intersecting Areas* mobile location estimation method can achieve these goals. By removing the common bundling of location estimation system and LBS and replacing it with a stand-alone location estimation system one can, not only ameliorate the conflict of interest between cloaking services and location estimation services, but return to the users control over knowledge of their location. Users would be free to determine their location without having to pay for it with their freedom or anonymity, and would be free to decide how and for what their location should be used.

Furthermore, using *Intersecting Areas* with fall-back to E-CGI and CGI it is possible to create a location estimation system that not only outperforms standard DCM in terms of precision and speed, but relies on crowd sourced data unlinkable to any user or individual providing users with anonymity while allowing the data itself to be freely released. We also suggested a method of encouraging users to contribute to the system, and a method to protect the system from saboteurs by determining users trust. The system should, on the other hand, keep track of users contributions to protect the integrity of submitted data, through a trust wallet-based system. The trust system does not need to keep track of what data users contribute, but rather count how many valid updates they have contributed. This is a feature of the system itself and not the location estimation method. The suggested method cannot, however, protect the system from malicious users sabotaging the system by first gaining the systems trust.

Besides the problem of trusted users turning saboteurs, two issues remain unresolved: The system, as suggested, does not ease the burden of protecting a users anonymity and privacy when data is transfered between the user and the system. Instead the system must rely on existing methods such as encryption or onion-routing. In addition we have yet to determine a good method to protect the system from random noise in updates. We have, however, suggested that this can be solved by finding a mathematical optimization algorithm suitable for the task. This would in turn also solve the problem of trusted users turning saboteurs if implemented correctly.

Two final challenges, outside of the scope of this thesis, must be addressed after such a sys-

tem has been implemented: Creating the system is, on its own, not enough. One must convince people to use the system. This is a challenge for two reasons: Many people, in our opinion, do not care about their privacy and anonymity. They will gladly use existing systems, and have no incentive to switch to a privacy preserving system. Convincing such potential users, involves informing and educating them. Second, deploying this type of system may prove to be a challenge. Existing location estimation systems are generally tightly entangled with existing mobile stations, producers, and service providers. Modern mobile stations usually are delivered fully equipped with a location estimation service provided by the producer, the supplier of the software, or a third party contracted by the producer or supplier. Gaining access to such a market is complicated, and providing users with an incentive to switch from the default provider even more difficult. To succeed creating our suggested service, these issues must be addressed.

7.3 The Mobile Location Estimation Method Test System

To design and test a new mobile location estimation method a well designed test system is needed. Not only must the method be tested in terms of precision, success rate, and processing speed, but all of these aspects must be compared to existing methods. One of our initial concerns was how difficult it was to find comparisons of the precision of existing location estimations methods in the literature. Finding comparisons based on large sets of data was even harder, and finding measurements of the success rate of different methods nearly impossible. Finally, most datasets used were not available for further testing by other parties.

This lead us to the conclusion that a large location estimation test bench, flexible enough to compare any type of mobile-based location estimation method, was needed. Furthermore, we determined that the system should be able to gather and use field-measurements from a widest possible range of different devices. Based on this we designed and implemented a test system that proved invaluable when designing a new mobile location estimation method.

The test bench allowed us to test several different existing location estimation methods over a wide range of data, on a large set of measurements, and compare them to our proposed estimation method. Doing a similar experiment based on the results found in the literature alone would have been difficult, if not impossible.

The system was, to a certain degree, extended and adapted during the tests, and some work therefore needs to be put into refactoring the system. The system has its weaknesses which should be addressed, mainly the lack of a mathematical optimization based module to detect and remove bad data as early in the process as possible.

In the future we would like to see the system used to create a large standard dataset for testing new and existing location estimation methods made freely available to the public. Making a rich dataset and the test system freely available would make testing future location estimation methods less time consuming. Furthermore, this would ease testing of existing location estimation methods, in addition to allowing such tests to be performed by different groups at different times with comparable results.

7.4 Future Work

Naturally in a project as expansive as this, the time constraints have graciously provided us the opportunity to compile a list of things we want to investigate in the future. A list is presented for each of the three main parts of this thesis.

7.4.1 Suggested Mobile Location Estimation System

- A mathematical optimization method should be found and tuned to the system. This will give the system a much better means of protecting the integrity of the data location estimations are based on. Not only will it be able to handle random noise in the crowd sourced updates, but should also be able to filter data from malfunctioning devices and malicious users who manage to bypass the systems trust-based protection.
- More work should be done to investigate the relations between quality control of data, protection of the integrity of the system and users privacy and anonymity. A specific digital wallet system should be chosen for the suggested protection and incentive system, it should be implemented and finally tested compared to other similar systems.
- More work needs to be done in regards to protecting users' privacy and anonymity when data is transfered between the system and it's users. Existing suggestions such as encryption and onion-routing should be tested. Furthermore, work should be put towards enhancing existing, or developing new location estimation methods that accommodate privacy preservation of data transfer. This includes work on alternatives to the traditional server—client model with storage, calculations and observations done on the network equipment or on the network itself.
- Finally we naturally want to see the suggested location estimation system implemented, released, and used.

7.4.2 The Intersecting Areas Mobile Location Estimation Method

- The logger hardware should be extended to log UMTS networks.
- The software needed to lock the logger hardware to a single cell should be implemented, allowing one to log and investigate the actual layout of cells.
- An alternative implementation using concave hulls should be done, and tests performed to investigate if the use of concave hulls improves precision enough to warrant the increased complexity.
- Alternative storage methods should be tested to determine if the suggested *limiting areas* storage method improves the precision enough to warrant the extra storage and complexity. This should be tested on both concave and convex hulls.
- The precision and performance of the method should be tested when used on TA and other NMR measurements.

7.4.3 The Location Estimation Test System

- The same, or similar, mathematical optimization suggested for the location estimation system should be implemented in the test system to allow the removal of garbage data at an earliest possible stage.
- A module should be developed for the test system that allows one to measure the sample density needed in an area for each individual algorithm to function, and the density needed for each individual algorithm to perform at an optimal level.

- Separate large sets of data gathered with different devices in different areas should be collected and used to compare location estimation methods. Specifically separate data should be collected in rural, sub-urban and urban areas, and separate data should be collected traveling at low and high speeds, since neighboring cell layouts often are created as to ensure mobile stations traveling at high speeds listen to large cells, while mobile stations stationary or traveling at low speeds listen to smaller cells.
- Finally the test system should be polished and released freely, together with the above mentioned datasets, to allow easier testing of new location estimation methods and allow accurate testing of methods on the same data performed by different groups.

APPENDIX A

CALCULATING DISTANCE FROM TIMING VALUES

Here we illustrate how the granularity of TA translates to distance in GSM and UMTS networks.

GSM networks use Gaussian Minimum Shift Keying modulation with a baud rate of 270.833 kb/s[50, page 4]. From the baud rate we can determine the time needed to transfer a single bit:

$$\frac{1s}{2708333bits} \times 1000000 = 3.69\mu s$$

In GSM networks the TA is calculate by measuring by how many bits a message is delayed[51]. We already calculated the transfer time of a single bit to be $3.69\mu s$, hence an increase of one TA represents a propagation delay of $3.69\mu s$. TA has a granularity of $3.69\mu s$. This is the round trip delay. The propagation delay one way between the mobile station and BTS is:

$$\frac{3.69\mu s}{2} = 1.845\mu s$$

Using the speed of light we translate the delay into distance:

$$300m/\mu s \times 1.845\mu s = 553.5m$$

Hence the granularity of TA in GSM networks is $553.5m$.

TA in GSM is a value between 0 and 63[51]. Table A.1 page ii shows how we now can translate TA to a distance from the BTS.

UMTS uses Quadrature Phase Shift Keying modulation[52] and more complex and granulated timing measurements[53] than GSM. Translating timing measurements to distance in UMTS follows the same basic principals and gives a granularity of 80 meters[54].

TA	Min distance from cell	Max distance from cell
0	0 m	553.5 m
1	553.5 m	1107 m
2	1107 m	1660.5 m
...	...	...
63	34.87 km	35.42 km

Table A.1: Example of translating Timing Advance values to distance from BTS in GSM networks

APPENDIX B

THE CODE

The code and documentation produced as part of this thesis is available at
http://www.opengsmloc.org/thesis/code.tar.gz

APPENDIX C

THE DATABASE

This is an illustrated overview of the database described in section 5.3 page 62. The database overview is divided into three parts: the *main database*, the *algotestground*, and the *norwegiancells*. The *algotestground* is an area used for converting existing fingerprints in the database to other formats used by some location estimation methods. The *norwegiancells* contains the true locations of cells in Norway gathered from the official governmental online collection of BTSs[1]. The database runs on PostgreSQL 8.0[2] software. A SQLite[3] clone of the database is maintained to be used on mainframes and clusters with no access to PostgreSQL software.

All GPS coordinates stored anywhere in the database are stored in the WGS84 Decimal Degrees format. Yellow boxes indicates database tables while green boxes indicate database views.

[1] http://finnsenderen.no (2011-01-03)
[2] http://postgresql.org (2011-01-03)
[3] http://sqlite.org (2011-01-03)

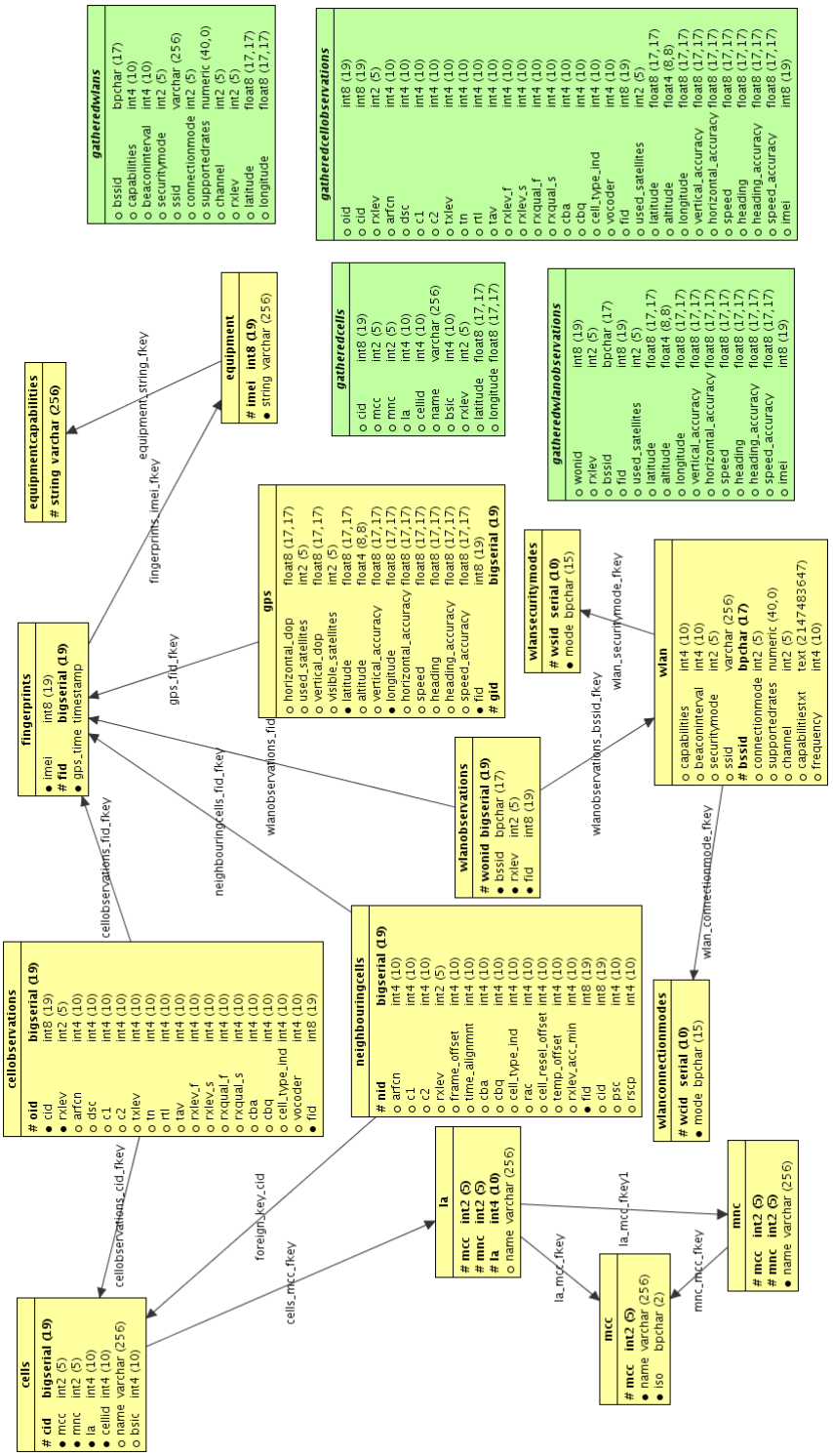


Figure C.1: The main database layout. Yellow boxes are database tables, green boxes are database views.

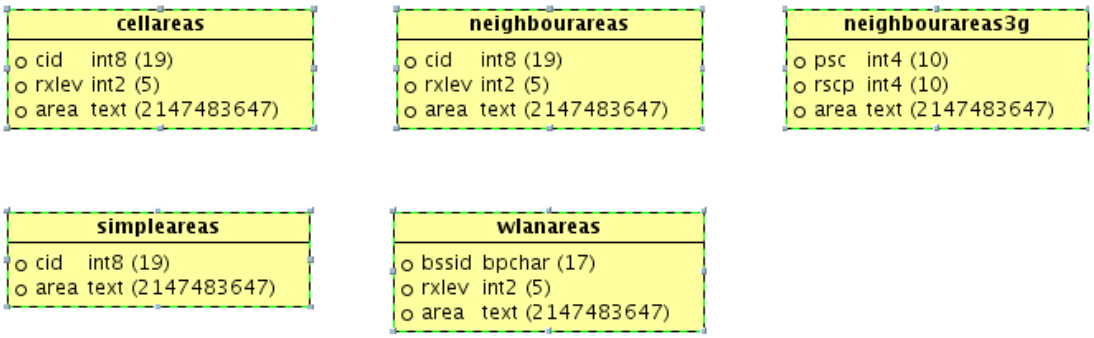


Figure C.2: The *algotestground* section of the database

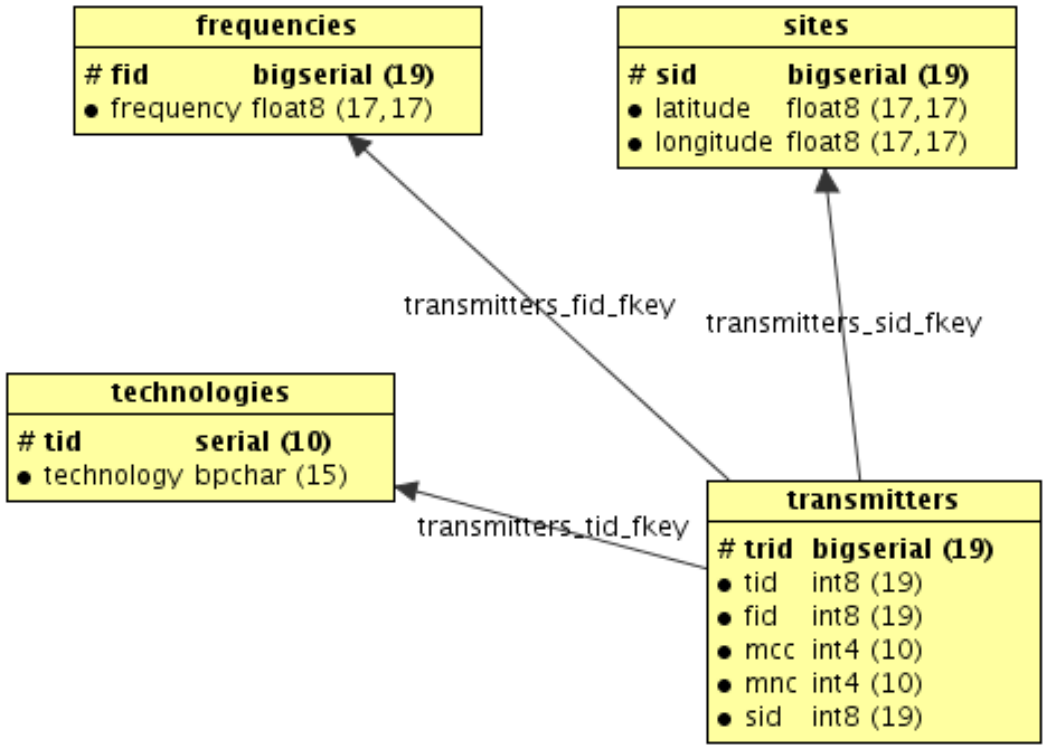


Figure C.3: The *norwegiancells* section of the database

APPENDIX D

DATA

All of the data, databases, and field-measurements used in this thesis are available at
http://www.opengsmloc.org/thesis/data.tar.gz

BIBLIOGRAPHY

[1] M. Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE Transactions on Vehicular Technology*, 29(3):317–325, August 1980.

[2] Y. Okumura, E. Ohmori, T. Kawano, and K. Fukuda. Field Strength and Its Variability in VHF and UHF Land-Mobile Radio Service. *Review of the Electrical Communication Laboratory*, 16(9-10):825–873, 1968.

[3] A. Medeisis and A. Kajackas. On the use of the universal Okumura-Hata propagation prediction model in rural areas. *VTC2000-Spring. 2000 IEEE 51st Vehicular Technology Conference Proceedings (Cat. No.00CH37026)*, pages 1815–1818, 2000.

[4] C.E. Cohen, H.S. Cobb, D.G. Lawrence, B.S. Pervan, J.D. Powell, B.W. Parkinson, G.J. Aubrey, W. Loewe, D. Ormiston, B.D. McNally, D.N. Kaufmann, V. Wullschleger, and R. Swider. Autolanding a 737 using GPS and Integrity Beacons. *Proceedings of 14th Digital Avionics Systems Conference*, 42(3):474, 1995.

[5] H. Laitinen, J. Lahteenmaki, and T. Nordstrom. Database correlation method for GSM location. *IEEE VTS 53rd Vehicular Technology Conference, Spring 2001. Proceedings (Cat. No.01CH37202)*, 4:2504–2508, 2001.

[6] H. Koshima and J. Hoshen. Personal locator services emerge. *Spectrum, IEEE*, 37(2):41–48, 2000.

[7] D. Zimmermann, J. Baumann, M. Layh, F. Landstorfer, R. Hoppe, and G. Wolfle. Database correlation for positioning of mobile terminals in cellular networks using wave propagation models. *IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall. 2004*, 7:4682–4686, 2004.

[8] H. Schmitz, M. Kuiperst, K. Majewskit, and P. Stadelmeyers. A new method for positioning of mobile users by comparing a time series of measured reception power levels with predictions. *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, 3:1993 – 1997, 2003.

[9] S. Ahonen and P. Eskelinen. Performance estimations of mobile terminal location with database correlation in UMTS networks. *3G Mobile Communication Technologies, 2003. 3G 2003. 4th International Conference on (Conf. Publ. No. 494)*, pages 400–403, 2003.

[10] M. Najar and J. Vidal. Kalman tracking for mobile location in NLOS situations. *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications*, pages 2203–2207, 2003.

[11] T. Nypan, K. Gade, and O. Hallingstad. Vehicle positioning by database comparison using the Box-Cox metric and Kalman filtering. *Vehicular Technology Conference. IEEE 55th Vehicular Technology Conference. VTC Spring 2002 (Cat. No.02CH37367)*, pages 1650–1654, 2002.

[12] M. Khalaf-Allah. Nonparametric Bayesian Filtering for Location Estimation, Position Tracking, and Global Localization of Mobile Terminals in Outdoor Wireless Environments. *EURASIP Journal on Advances in Signal Processing*, 2008:1–15, 2008.

[13] M. Khalaf-Allah. A Novel GPS-free Method for Mobile Unit Global Positioning in Outdoor Wireless Environments. *Wireless Personal Communications*, 44(3):311–322, September 2007.

[14] M. Khalaf-Allah and K. Kyamakya. Mobile location in GSM networks using database correlation with bayesian estimation. *11th IEEE Symposium on Computers and Communications (ISCC'06)*, (3), 2006.

[15] M. Khalaf-Allah and K. Kyamakya. Mobile Location using Database Correlation with Least-Squares and Bayes Filtering. *European Wireless 2006*, 2006.

[16] M. Khalaf-Allah and K. Kyamakya. Database Correlation using Bayes Filter for Mobile Terminal Localization in GSM Suburban Environments. *2006 IEEE 63rd Vehicular Technology Conference*, pages 798–802, 2006.

[17] C. Laoudias, P. Kemppi, and C. G. Panayiotou. Localization Using Radial Basis Function Networks and Signal Strength Fingerprints in WLAN. *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pages 1–6, November 2009.

[18] C. Hu, W. Chen, Y. Chen, and D. Liu. Adaptive Kalman Filtering for Vehicle Navigation. *Journal of Global Positioning Systems*, 2(1):42–47, 2003.

[19] C. M. Takenga, Q. Wen, and K. Kyamakya. On the accuracy improvement Issues in GSM Location Fingerprinting. *2006 IEEE 64th Vehicular Technology Conference*, pages 1–5, September 2006.

[20] C. Drane, M. Macnaughtan, and C. Scott. Positioning GSM telephones. *IEEE Communications Magazine*, 36(4):46–54, 59, April 1998.

[21] J. M. Borkowski and U. Lempiäinen. Practical Network-Based Techniques for Mobile Positioning in UMTS. *EURASIP Journal on Advances in Signal Processing*, 2006:1–16, 2006.

[22] G.M. Djuknic and R.E. Richton. Geolocation and assisted GPS. *Computer*, 34(2):123–125, 2001.

[23] S. Blackburn, editor. *The Oxford Dictionary of Philosophy*. Oxford University Press - Oxford Reference Online, 2008. http://www.oxfordreference.com/views/ENTRY.html?subview=Main&entry=t98.e2518 (2010-06-06).

[24] J. Law and E. A. Martin, editors. *A Dictionary of Law*. Oxford University Press - Oxford Reference Online, 2009.

http://www.oxfordreference.com/views/ENTRY.html?subview=Main&entry=t49.e3024 (2010-06-06).

[25] A. Pfitzmann and M. Köhntopp. Anonymity, Unobservability, Pseudonymity, and Identity Management - A Proposal for Terminology. *Citeseer*, pages 1–16, 2004.

[26] E. Snekkenes. Concepts for personal location privacy policies. *Proceedings of the 3rd ACM conference on Electronic Commerce - EC '01*, (1):48–57, 2001.

[27] U. Hengartner and P. Steenkiste. Protecting access to people location information. *Security in Pervasive Computing*, pages 222–231, 2004.

[28] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar. Preserving user location privacy in mobile data management infrastructures. *Privacy Enhancing Technologies*, 4258:393–412, 2006.

[29] C. Zhang and Y. Huang. Cloaking locations for anonymous location based services: a hybrid approach. *GeoInformatica*, 13(2):159–182, April 2008.

[30] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 384–393, 1998.

[31] L. Sweeney. k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 2002.

[32] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. *Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 31–42, 2003.

[33] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing Location-Based Identity Inference in Anonymous Spatial Queries. *IEEE Transactions on Knowledge and Data Engineering*, 19(12):1719–1733, December 2007.

[34] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: anonymous location-based queries in distributed mobile systems. *Proceedings of the 16th international conference on World Wide Web*, pages 371–380, 2007.

[35] B. Gedik and L. Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pages 620–629, 2005.

[36] M. F. Mokbel, C. Chow, and W. G. Aref. The new Casper: query processing for location services without compromising privacy. *Proceedings of the 32nd international conference on Very large data bases*, (1), 2006.

[37] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privcy Beyond k-Anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1):3–es, March 2007.

[38] P. Syverson, D. Goldschlag, and M. Reed. Onion Routing for Anonymous and Private Internet Connections. *NAVAL RESEARCH LAB WASHINGTON DC CENTER FOR HIGH ASSURANCE COMPUTING SYSTEMS (CHACS)*, 1999.

[39] M. Gruteser and B. Hoh. Protecting Location Privacy Through Path Confusion. *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 194–205, 2005.

[40] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. *Pervasive Computing*, 3468:152–170, 2005.

[41] B. Moon, H.V. Jagadish, C. Faloutsos, and J.H. Saltz. Analysis of the clustering properties of the Hilbert space-filling curve. *IEEE Transactions on Knowledge and Data Engineering*, 13(1):124–141, 2001.

[42] G Ghinita, P. Kalnis, and S. Skiadopoulos. Mobihide: A mobilea peer-to-peer system for anonymous location-based queries. *Advances in Spatial and Temporal Databases*, 4605:1–18, 2007.

[43] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. *Conference on Pervasive Services*, pages 88–97, 2005.

[44] A .R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1):46–55, January 2003.

[45] ISO/IEC IS 15408-2:2008. *Information technology – Security techniques – Evaluation criteria for IT security – Part 2: Security functional components*. International Organization for Standardization, Geneva, Switzerland.

[46] H. Edelsbrunner. *Alpha Shapes—a Survey*. Springer Verlag, 2010.

[47] D.C. Hoaglin, F. Mosteller, and J.W. Tukey. *Understanding robust and exploratory data analysis*. Wiley New York, 1983.

[48] J.W. Tukey. *Exploratory data analysis*. Addison-Wesley, 1977.

[49] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, et al. Place Lab: Device Positioning Using Radio Beacons in the Wild. In *Proceedings of Pervasive*, volume 3468, pages 116–133. Springer, 2005.

[50] G.L. Stüber. *Principles of mobile communication*. Springer Netherlands, 2001.

[51] ETSI TS 145 010. *3GPP specification: 45.010; Digital cellular telecommunications system (Phase 2+); Radio subsystem synchronization*. European Telecommunications Standards Institute, Sophia Antipolis, France, 2011.

[52] ETSI TS 125 101. *3GPP specification: 25.101; Universal Mobile Telecommunications System (UMTS); User Equipment (UE) radio transmission and reception (FDD)*. European Telecommunications Standards Institute, Sophia Antipolis, France, 2011.

[53] ETSI TS 125 402. *3GPP specification: 25.402; Universal Mobile Telecommunications System (UMTS); Synchronization in UTRAN Stage 2*. European Telecommunications Standards Institute, Sophia Antipolis, France, 2010.

[54] J. Borowski, J. Niemelä, and J. Lempiäinen. Performance of cell ID+ RTT hybrid positioning method for UMTS radio networks. In *The 5th European Wireless Conference Mobile and Wireless Systems beyond 3G, February*, pages 24–27, 2004.

INDEX