MASTER'S THESIS | LUND UNIVERSITY 2015

Energy Efficient Geo-Localization for a Wearable Device

Robert Bagge, William Martinsson

Department of Computer Science
Faculty of Engineering LTH

ISSN 1650-2884
LU-CS-EX 2015-28

Energy Efficient Geo-Localization for a Wearable Device

(A Study On How To Combine Periodically Awoken Sensors to Produce an Accurate and Energy Efficient Geo-Localization Algorithm For a Wearable Device)

Robert Bagge
bagge.robert@gmail.com

William Martinsson
williammartinsson1@gmail.com

June 24, 2015

Master's thesis work carried out at
the Department of Computer Science, Lund University.

Supervisors: Elin Anna Topp, elin_anna.topp@cs.lth.se
Mikael Johansson, mikael@narrativeteam.com

Examiner: Jacek Malec, Jacek.Malec@cs.lth.se

Abstract

During the last decade there has been a surge of smart devices on markets around the world. The latest trend is devices that can be worn, so called wearable devices. As for other mobile devices, effective localization are of great interest for many different applications of these devices. However they are small and usually set a high demand on energy efficiency, which makes traditional localization techniques unfeasible for them to use.

In this thesis we investigate and succeed in providing a localization solution for a wearable camera that is both accurate and energy efficient. Localization is done through a combination of Wi-Fi and GPS positioning with a mean accuracy of 27 m. Furthermore we utilize an activity recognition algorithm with data from an accelerometer to decide when a new position estimate should be obtained. Our evaluation of the algorithm shows that by applying this method, 83.2 % of the position estimates can be avoided with an insignificant loss in accuracy.

Keywords: Localization, Energy Efficient, Accelerometer, Wi-Fi, GPS, Activity Recognition

Acknowledgements

First of all we would like to thank our supervisors *Elin-Anna Topp* and *Mikael Johansson* for the support they have given us during the work with this thesis.

We also want to mention *Narrative AB* that have provided us with equipment, office space and feedback whenever we needed external input.

Last but no least we want to thank our opponents *Antonina Tokarchuk*, *Felip Marti* and *Jonathan Bratel* for providing us with invaluable insights during the finishing of this report.

Contents

Glossary and Index

Accelerometer In our thesis, a low energy sensor that gathers information about the acceleration in three axes (X, Y, Z). The acceleration is often expressed as m/s^2. Earth's gravity induces a constant acceleration downwards of $9.82 m/s^2$.. 1, 10–13, 19, 35–39, 42, 45–47, 51, 53, 58, 59, 61, 70, 71, 73, 74

Android A mobile operating system based on the Linux kernel developed by Google. 10, 13, 28, 51, 59, 69–71, 73, 74

AP (Access Point), a Wi-Fi router of some kind. The access point periodically emits broadcast messages to notify the surroundings of its existence. 19, 22–28, 31, 32, 37, 51, 71–73

Bluetooth Is a wireless network technology to exchange data over small distances. Can also be used for localization. 10, 17

Cell A term that includes all techniques for mobile phones to call, send messages and transfer data, such as GPRS, UMTS, LTE etc. Can also be used for crude localization. 11, 22, 51

Cloud service A cloud service is designed provide easy, scalable access to applications, resources and services, and are fully managed by a cloud services provider. 10

FFT (Fast Fourier Transform), is an algorithm to compute the Discrite Fourier Transform (DFT). Fourier transforms are widely used in computer science for signal analysis. 36, 39

Fingerprinting Fingerprinting is a method to estimate the position of an entity. It is commonly used for Wi-Fi positioning. A fingerprint in Wi-Fi positioning consists of the received signal power to a number of Wi-Fi access points. This fingerprint is compared to other fingerprints in a database to obtain a position estimate. 22–24

GPS (Global Positioning System). 1, 10–13, 17, 19, 21, 22, 27, 28, 30, 31, 36, 49, 51, 59–61, 70, 71

iOS A mobile operating system for Apple mobile devices, developed by Apple. 10

Localization algorithm In this thesis, the algorithm which calculates a position estimate. 9, 11–13, 35, 49, 51, 52, 54, 56, 57, 59–62

Location System A system responsible for collecting and processing data from various sensors. Uses a localization algorithm to obtain a position estimate. 13, 15, 17–19, 21–25, 60, 61

MAC-address MAC-address or Media Access Control address is a unique identifier associated with network network interfaces. 22, 72

Multilateration Multilateration is a mathematical method to estimate the position of an entity. To do this you need to know the distance to at least three other positions. The position is then estimated through repeated triangulation over sets of three known positions. The number of sets is $\binom{n}{3}$ where n is the total number of known positions. 22, 23, 25–32, 51, 57, 58, 60, 61, 73

MyTracks Android application to montior the users movement over time. Developed by Google. 51, 54, 55

Path-loss exponent A variable that describes the loss in power as a radio signal travels. This is usually a value between 2 and 4, where it is set to 2 if there are no obstacles the signal has to pass through. 25, 31

Position estimate In this thesis, a geographical location estimate calculated by the localization algorithm. 10, 11, 17–19, 24, 31, 49, 50, 60

Position estimation In this thesis, the action of estimating a position . 12, 17–19, 24, 27, 28, 31, 49, 53, 56, 60, 73

Python Programming language, used mainly to simulate server functionality in this thesis. 13, 27, 28, 51, 69, 72, 74

SMA (Signal Magnitude Area), is a statistical measure. SMA is calculated by dividing area under the curve with the time interval. 39, 42, 45

SVM (Support Vector Machine), is a machine learning model which used to analyze and recognize data. 36

TTFF (Time To First Fix), the time before a GPS based location system obtain the first position estimate. 21

Wardriving When a user travels around and gathers data of nearby Wi-Fi networks. The data can then be uploaded to a database which holds information about: MAC-address, location, signal strength and transmitting frequency. 22–24, 30, 31, 57

Wi-Fi Is a term that include all WLAN (Wireless LAN) in IEEE 802.11 standards. Often falsely abbreviated to Wireless Fidelity. 1, 10–13, 17, 19, 21–25, 27, 28, 30, 36, 37, 49, 51, 59–61, 70, 71, 73, 74

Chapter 1

Thesis Scope and Approach

1.1 Introduction

Over the last couple of years a surge in wearable devices have appeared on the markets around the world. Smart watches, smart armbands and wearable cameras are just a few gadgets that are getting increasingly popular. At the same time the user expectations on these devices to function well and be able to do a lot of the same things that a mobile phone does today are sky high.

It is reasonable to think that, just as for mobile phones, localization services will play a major part in the applications of these devices. As wearable devices are much smaller than a mobile phone, the available battery resources are much more constrained. Since localization algorithms traditionally have been a power consuming operation, new smarter ways to do the same job have to be introduced if these wearable devices are to live up these high expectations. In this thesis we aim to explore the possibilities for such smart localization algorithms for the *Narrative Clip*, which is a small wearable camera further described in the section below.

1.2 The Narrative Clip

The *Narrative Clip* (The Clip) is a small wearable camera that can be attached to the clothes anywhere on the body. It is developed by *Narrative AB* (The Company) with headquarters in Stockholm, Sweden. The Clip, which is presented in Figure 1.1, automatically takes pictures without the wearer doing anything and thus enables the capturing of real authentic moments.

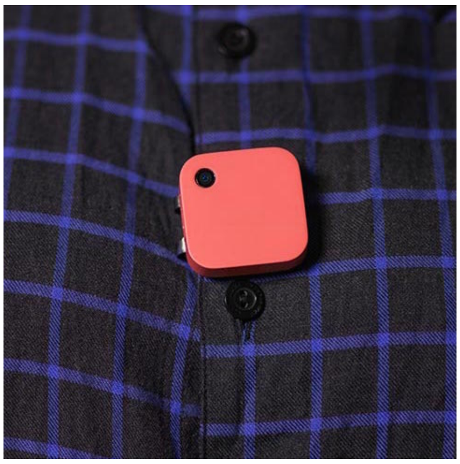


Figure 1.1: The Clip attached to the front of a shirt.

1.2.1 How it works

The capturing of pictures is done automatically every 30 seconds where the Clip goes to sleep in between. While in sleep, the major part of the Clip is shut down to save energy, but a few actions are still enabled. For example the Clip is still able to detect a double-tap gesture for user-induced capturing of a picture during this time. This method where the device just has to wake up for a brief moment for every picture is very energy efficient and enables the Clip's battery to last up to two days or over 5000 pictures. During this short window of time, other sensors on the device can record data and attach it to the picture. The pictures and the attached data are stored on the Clip until connected to a computer, whilst they are uploaded to the Company's cloud service. While in the cloud, the images as well as the data, are processed in different ways. When the processing is completed, the images are available to the user through web, Android or iOS applications. A visual presentation of the flow is presented in Figure 1.2.

The Clip consists of the components presented in Figure 1.3 where the accelerometer, Wi-Fi, Bluetooth and GPS sensors can be used to record data and attach it to the pictures as they are taken.

1.2.2 Positioning Algorithm

At the moment there is an embedded GPS-chip from a third party manufacturer that enables localization. Simultaneously with pictures being taken the GPS-chip records data. The GPS-data are encoded by the GPS-chip and stored together with the picture. When the Clip is connected to a computer the encoded GPS-data are sent to the third party manufacturer's servers, which decodes the data. Their servers utilizes a cached database of the positions of the GPS-satellites and can thus estimate the position based on the recorded GPS-data. There is a big limitation with using this GPS-chip though. The encoding and decoding of the data is a black box and the GPS-sensor thus has to be used in combination with a server solution from the same company. This closes the door on any modifications to be made. Furthermore, the current solution works poorly and only succeeds in providing a position estimate for a small quota of the pictures. This is further presented in section 1.5.

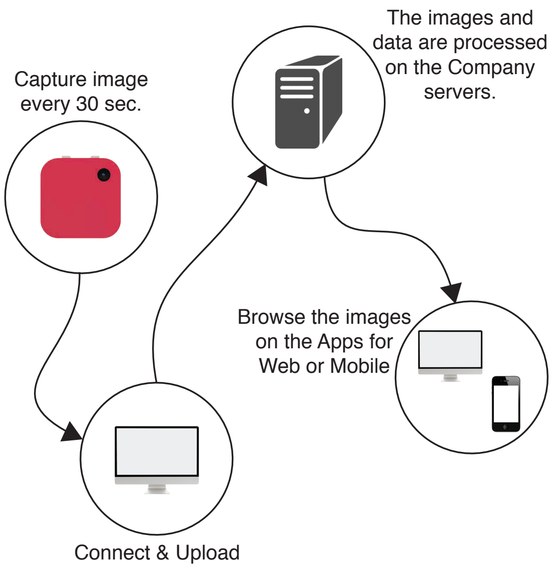

The images and data are processed on the Company servers.
Capture image every 30 sec.
Browse the images on the Apps for Web or Mobile
Connect & Upload

Figure 1.2: The flow of a picture from the moment it is captured until the user can browse it.

To improve the localization, other sensors can be used instead and the rest of the thesis is about how to use these sensors to get a better position estimate at a lower energy cost.

1.3 Problem Definition

How can periodically gathered data from the Wi-Fi, accelerometer and GPS sensors be combined to produce an accurate and energy efficient localization algorithm for a wearable device?

1.4 Motivation

Traditional localization algorithms that solely rely on a GPS-sensor fail for wearable devices for two reasons. First of all it is a very power consuming operation, something that is much more notable for a small wearable device with limited battery resources than on a mobile phone [6]. Secondly, these algorithms often fail to obtain a position fix in an urban environment. In mobile phones the problem is commonly solved by adding other sensors such as Wi-Fi and cell to the equation. An algorithm that continuously listens to all these three sensors and integrates a position over time succeeds in obtaining an accurate position in most environments. However this approach is problematic to use with a wearable device since it is very power consuming and often requires the device to be connected to the Internet. Many of these wearable devices are offline, meaning that they most of the time are not connected to the Internet. Although offline, it is still of a big interest to have a working localization solution for these devices.

Take the Clip for example. It wakes up every 30 seconds to take a picture and then goes back to sleep. It cannot keep track of data during the 30 seconds it is in sleep and is not connected to the Internet. However it is still of huge interest to know where the pictures are

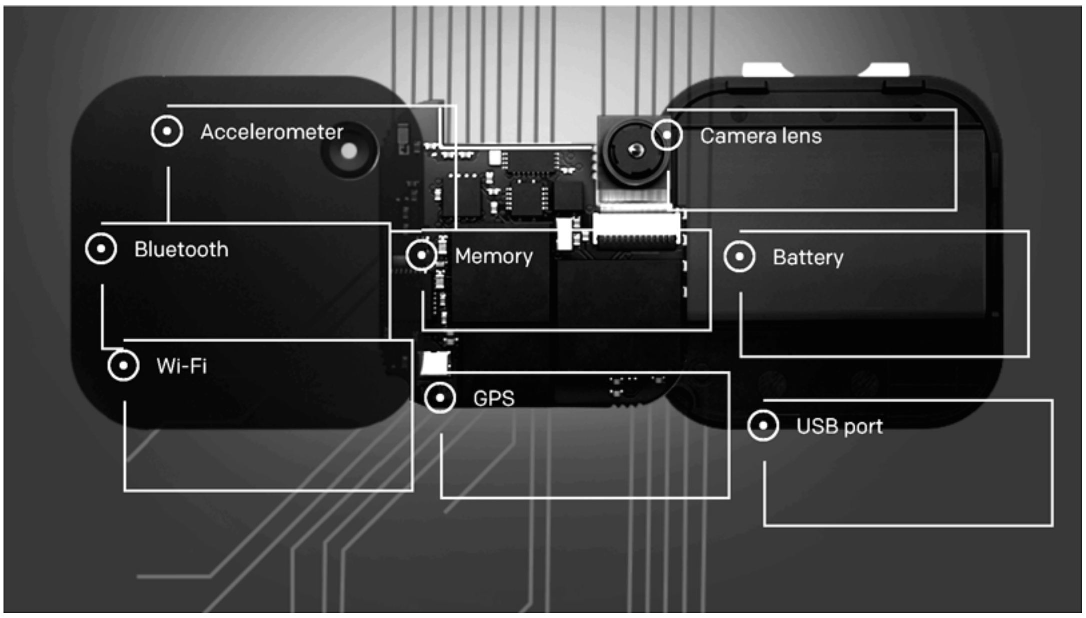


Figure 1.3: The different components of the Clip

taken. The location can be used to map out where the pictures are captured or be the part of an algorithm that group these pictures together. During the short window of waking up and capturing a picture there is a possibility to record data from a number of different sensors and then process it at a later time, server-side, when the pictures are being uploaded. Even though localization of this sort is not in real time and not as accurate as the method mobile phones use, it can provide wearable devices with a method of localization that is accurate enough for many applications at the same time as not making a huge mark in terms of energy consumption. To develop an localization algorithm that address this problem is thus the aim of this thesis.

1.5 Scope and Method

As of now the Clip records GPS-data on an embedded chip from a third party manufacturer. This data is sent from the servers of the Company to the servers of GPS-chip manufacturer for processing. Their servers then replies with a position in the cases where it can be obtained. The problem is that this is only done in about 5-10 % of the images and a big majority of the pictures are thus left without an estimated location. To remedy this problem a number of other sensors can be involved in the process of estimating a location:

- A Wi-Fi-sensor can be included to enhance the position estimation and make the algorithm more robust in urban environment.
- The accelerometer, magnetometer and gyrometer can possibly be used in a number of different ways to further improve the algorithm.
- A microphone can be used to make a decision whether the user is inside or outside [23].

To further add to the complexity of a localization algorithm, the images themselves can be analyzed with a number of different techniques to extract information about the location of the image.

Obviously the scope to analyze what implications all of these different sensors and techniques might have on a localization algorithm is far too grand for a thesis of this magnitude and thus a few limiting measures have been taken.

- The sensors that are included in the analysis are GPS, Wi-Fi and accelerometer.
- The accelerometer is used to enhance the energy efficiency of the location rather than the accuracy of the localization algorithm itself.
- Both the data gathering and the data processing on the servers have been simulated.
  - The data gathering is simulated with Android apps that are programmed to behave like the Clip.
  - The servers are simulated with Python scripts at a computer.
- The evaluation has been made with small datasets.

With the limitations listed above in mind, our solution should be viewed as a proof of concept rather than a finished product. During the work with the thesis the following process has been followed.

- A literature study is conducted.
- A solution is proposed and implemented.
- Finally, the solution is evaluated.

This process has been followed for the thesis as a whole but also for the different elements of the proposed algorithm. The algorithm we propose consists of two parts, one that handles the localization and one activity recognition part that aims to make the algorithm as a whole more energy efficient. These two parts are presented and evaluated separately from each other and then combined into one single algorithm. Finally, this algorithm is evaluated to provide a finite answer to the problem definition. An overview of the approach to come up with a working solution is presented in Figure 1.4.

1.6 Disposition

The structure of the report follows the same pattern as the process of work presented in Figure 1.4. A introduction to the problem and a overview of the Clip is presented in *Chapter 1: Introduction*. *Chapter 2: Background and Algorithm Outline* aims to provide a theoretical framework for location systems and describe the outline of the proposed algorithm. This section is followed by *Chapter 3* and *Chapter 4* that describes the localization and activity recognition parts of the solution respectively. These two entities are combined into one single solution that is proposed and evaluated in *Chapter 5: Final Solution*. Finally the solution is discussed together with some considerations of the thesis overall in *Chapter 6*, where also our conclusion of the thesis and a few items for future research are presented.

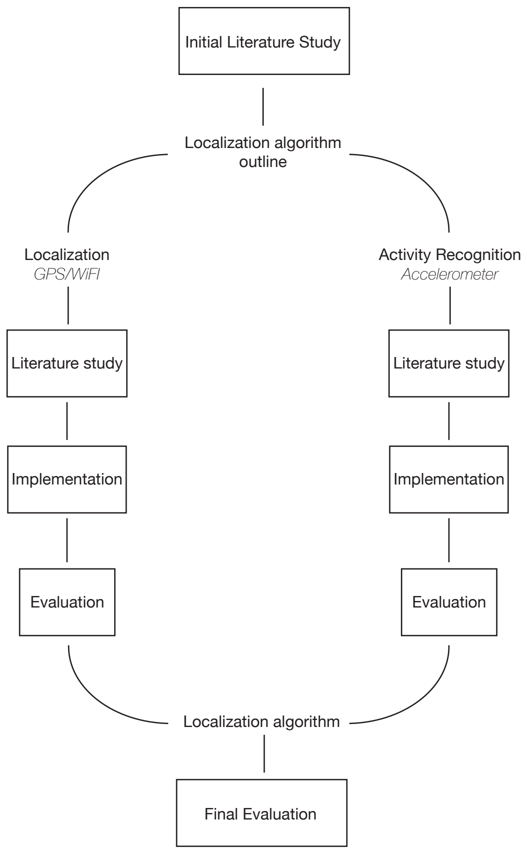


Figure 1.4: An overview of the approach we have had while working on this thesis.

1.7 Contributions

The thesis project has been worked upon in an agile manner, meaning that it has been developed iteratively. To do this it has been important that both of us have had a good theoretic base to stand on when discussing different solutions for the algorithms and both of us have thus conducting research into all the different areas of the thesis. The three Android applications and the Python server simulations have also been developed in a cooperative manner where both of us have had a part in at least a part of every application/program.

With this said there are parts of the project where one of us has been working individually.

Robert developed the UI for all the three Android applications presented in Appendix A. The application that simulates the Clip and records both Wi-Fi and accelerometer data was fully developed by Robert. It should be noted though that he used parts of the previously developed applications to do this. Furthermore Robert wrote most of the introduction and theory sections in the report. Chapter 6, Discussion, Conclusion and Future Research was fully written by Robert, but the contents was decided through discussion with William.

William on the other hand conducted the research behind trainable decision trees and also made the Python implementation CART that took the recorded accelerometer data and trained the decision tree we use in the activity recognition algorithm. Furthermore he implemented the server simulation that processes the accelerometer data as well as the file handling systems in both the Android applications and the Python server implementation. Furthermore, William developed a solution that takes KMZ-data and transforms it to CSV-data. This solution was used during the evaluation of the geo-location data in Chapter 3 and Chapter 5. In the writing process William wrote the evaluation sections of Chapter 4 and Chapter 5. It should be noted though that the evaluation itself was made in cooperation with Robert.

With this said, we have to stress that we have been working in a cooperative manner during the whole project and consider us both having played an equal part in the results the project have led to.

1.8 Summary

In this chapter we have presented the underlying idea for the thesis, the Clip we have been working with as well as defining the scope of the thesis and the disposition of this report. In the next chapter, Chapter 2, we present what a location system is, some background of the subject as well as our initial idea of the algorithm we have developed during this thesis.

Chapter 2

Background and Algorithm Outline

2.1 Background

The possibility to track one's position has become an expected service for most electronic devices in our everyday life but this has not always been the case. Take GPS, which is the most widely known technique for positioning. When created, GPS was mainly intended for U.S. military use but has over the years become the go to location system for the private sector as well.

With the history of GPS in mind, a vast number of new technologies for position estimation, so called location systems, are appearing in a rapid pace. It is still GPS that is the dominant force in the location technology industry, but one can only speculate about how younger technologies like Wi-Fi and Bluetooth will be used in a decade or two, when they have had the same time to evolve as GPS.

There are five key performance characteristics defined by LaMarca *et al.*[21] in the book *Location Systems* that can be used to asses different wireless location services.

Accuracy A metric that describes the average precision of the system. It is often a number such as median-error or mean-error, which is easy to use to compare different location systems.

Coverage Refers to the physical area where the systems can provide accurate position estimates.

Privacy The privacy the location system offers for its users. For instance, if the client-device itself calculates the position estimates, the privacy of the location system is generally good. On the contrary, it is generally bad if the calculations are made in the infrastructure of the system, since this forces the user to trust the system designers and managers that the information will not end up in the wrong hands.

Infrastructure Cost Refers to the cost in time and money that it takes to maintain the system. For instance this can often be calculated as the cost per square kilo-meter for deployment and maintenance for large-scale outdoor location systems and as the cost per square meter for smaller systems such as indoor location systems.

Per-client Cost The additional cost for adding an additional device or person to be located by the system.

These five characteristics are used to motivate certain choices that have been made during the work of the thesis as well as to evaluate the final solution.

2.2 Algorithm Outline

As previously suggested, the algorithm proposed for the location system of this thesis consists of two parts, one part that estimates the location of the Clip and one part that saves energy for the same.

The first part is present in all location systems and is responsible for estimating an as accurate position as possible for such a low cost as possible. The biggest constraint the Clip imposes on this part of the algorithm is that data are only recorded for a brief moment every 30 seconds. However the precision of the algorithm is not vital and estimation errors in the magnitude of 50-100 m are acceptable.

The other part of the algorithm is there to save energy for the device. The argument for including this part lies in the fact that people are stationary over 75 % of the time [23]. This means that 75 % of all the position estimates are unnecessary. If there is a way to find out when a person is moving before the position estimation is made there is a huge upside in energy saving. The solution idea is abstracted and presented in Figure 2.1

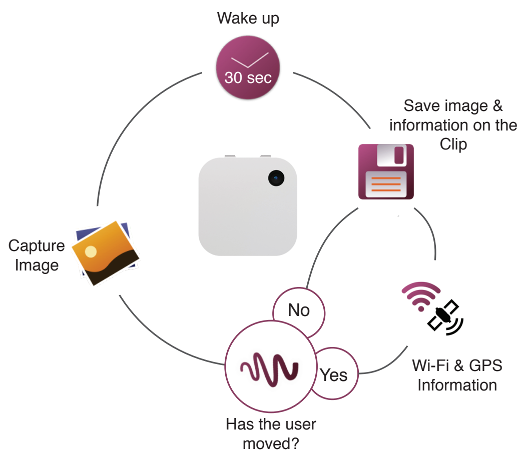


Figure 2.1: Solution idea where the Clip wakes up every 30 seconds to capture a picture and record data from the positioning sensors if the user has moved.

2.2.1 Localization

GPS is a powerful location system, however it often fails to provide a position fix indoors or in an urban environment. Luckily these environments often have a dense population of Wi-Fi Access Points (APs). This fact arguably makes GPS and Wi-Fi the perfect combination of sensors for a robust location system. The Wi-Fi can provide a location in an urban environment/indoors and GPS can provide a location in more remote areas. A full account of the localization part of the location system is presented in Chapter 3.

2.2.2 Activity Recognition

As previously mentioned people are stationary over 75 % of the day, something that makes 75 % of all the position estimates when the Clip captures an image unnecessary. If these unnecessary position estimations are avoided, huge energy savings can be won.

Luckily there is a kind of sensor out there that not only is very good at detecting movement, but also is extremely energy efficient, an accelerometer. Accelerometers are commonly used in step detectors and have also played a big part in the surge of exercise equipment we have seen the last few years. The FitBit armband and Apple Watch are just two of many devices that use the accelerometer a lot [2][13]. A full account of how we utilize the accelerometer to detect movement for optimizing the energy consumption is presented in Chapter 4.

2.3 Summary

In this chapter we present what a location system is, five criterias that can be used to evaluate a location system as well as the initial idea for the algorithm we have developed during this thesis. In the next-coming chapters, Chapter 3 and Chapter 4, we present the algorithm parts localization and activity Recognition respectively. We describe the area itself as well as proposing a solution that we implement in our proposed algorithm of this thesis.

Chapter 3
Localization

The Global Positioning System (GPS) is a powerful location system for mobile devices, it covers the whole globe, the infrastructure is already deployed and it is free to use. Furthermore it can calculate a position with a precision of up to a few meters [22]. However, positioning with GPS also has some major flaws. The Time to First Fix(TTFF) can be very long, from several seconds to several minutes. Power consumption is significant, which is a problem on devices with a limited battery life such as wearable devices. Another serious problem is that GPS relies on line of sight to satellites for accurate positioning and therefore the use of GPS as a positioning system for indoor and urban environments is limited [8][25][22][38]. This is very problematic since a lot of people, nowadays, live in these kinds of environments.

To address this problem, a number of new positioning systems have been developed, where the most popular one is based on Wi-Fi [38].

3.1 Related Work

In this section the articles we have used consequently through this chapter and there main findings are presented.

Zandbergen *et al.* conducts a study where they compare the same location system, *Skyhook*, when it is used on two different devices, an iPhone and a laptop. They find that the performance between the two devices are similar in terms of positional accuracy, median-error, but differs in terms of getting the same position estimate at the same time on the two devices [38].

Tippenhauer *et al.* studies the security of Wi-Fi based location systems. Besides providing a thorough explanation of how these systems works, they succeed in providing evidence to how vulnerable these systems are to location spoofing and database manipulation attacks [35].

Lui *et al.* makes thorough research on how the received signal strengths of Wi-Fi signals differs between devices. They find that the hardware and software of different Wi-Fi chipsets affect the measure and that different devices record different magnitudes of received signal power. They show that this even is true for chipsets from the same manufacturer. Their research implies that a fingerprinting database works best when it is recorded and used by the same kind of device [25].

Gezici *et al.* presents a solution that makes use of different characteristics of a Wi-Fi signal such as time of arrival, angle of arrival and signal strength to make a detailed map of an environment. They provide a thorough description of these characteristics and how they can be used. Finally the compare and investigate the theoretical limits of different Wi-Fi localization algorithms [15].

Lloret *et al.* investigates how public and private Wi-Fi networks can be used to make a global spanning location system. They explain the differences with using Wi-Fi signals opposed to GPS and cell signals in a location system. Finally they propose a system that uses the patterns of Wi-Fi signals to estimate the location of a device [22].

3.2 Our Work

In Wi-Fi localization, Wi-Fi APs are used to estimate a position. During the last decade a vast number of APs using the 802.11 Wi-Fi standard have been deployed all over the world. These APs broadcast a message that tells the surroundings about their existence. In areas where the density of APs are high enough the broadcast messages from different APs overlap each other. This feature can be used to calculate a location in a number of different ways [8][35]. This together with the fact that Wi-Fi localization does not require that a connection is established to every Wi-Fi network, makes Wi-Fi localization a very potent positioning system in terms of coverage, infrastructure cost, and per-client cost. In terms of accuracy and privacy it depends on the method used when constructing the location system [15][21][25][35].

3.2.1 Theory

Common methods for Wi-Fi positioning are a geometrical range-based approach as well as a method called fingerprinting. The geometrical range-based approach relies on the knowledge of the exact location of every AP, which are commonly stored in a database. Signal strengths of the broadcast messages are modelled as a function of distance from the location of an AP and thus multilateration between different APs and their signal strengths can be used to determine a location [35][38]. Fingerprinting uses another approach that is based on fingerprints of hardware addresses (MAC-adresses) and their signal strengths at a certain location to determine the position of a device. Fingerprints are usually gathered through a process called wardriving. When wardriving, a person or vehicle moves around an area and records the fingerprints of a huge amount of locations, these fingerprints are stored in a database. To estimate the position of a mobile device, the recorded fingerprint with signal strengths to different MAC-adresses is compared to the fingerprints in the database [25][38].

When deciding which method to use there are a number of factors that should be considered and both systems have their strengths and weaknesses. Depending on accuracy requirements and constraints of the positioning system, different methods can be used [15].

Geometrical range-based approach

As mentioned before a Geometrical range-based approach calculates a position through multilateration of the distances to a number of APs. To do this, a function that converts the received signal strengths to distances is needed as well as a database that stores the correct position of every AP. In a small-scale Wi-Fi positioning system it is feasible to get the accurate locations of all APs since network managers usually know the positions of them. However, when the Wi-Fi positioning system makes use of public APs as its reference points it is much harder to get accurate locations for the APs, since wardriving and similar techniques merely can give an estimate of the locations. Due to the changeable nature of an urban environment these techniques often get the position a little bit off and the median-error for the estimated positions when wardriving can be as much as 40 m [21]. Furthermore, converting received signal strengths to distances can cause large errors in urban environments since buildings, vegetation, vehicles etc. can cause the signal strengths to vary [7][38]. Despite all these limitations there are successful examples where this technique has been used. For example Bhasker *et al.* made an implementation where they reached an accuracy of roughly 20-30 m [4]. Although they knew the exact positions of every AP, it is reasonable to believe that a similar system based on a wardriven AP database could achieve acceptable accuracy.

Fingerprinting

The difficulty to model received signal strengths as distance in an urban environment has been a driving factor of other Wi-Fi positioning techniques being developed. Amongst them one of the most popular and powerful techniques is fingerprinting. The accuracy of a location system based on fingerprinting can be as good as a median-error of a few meters in a small-scale system and 20-50m in a large-scale public Wi-Fi localization system [29][38].

The fact that the accurate location for an AP is not needed when estimating a location with fingerprinting makes this a good candidate for a method when wardriving is used to create the database.

However, there are a few considerations to keep in mind when selecting fingerprinting as the go to method for a positioning system. For example, it has been proven that different Wi-Fi-devices record different fingerprints in the same location, which makes it unfeasible to use the same database for localization with different devices. There are even examples when the same Wi-Fi-device has recorded different fingerprints in the same location [25].

3.2.2 Security and Privacy

Tippenhauer *et al.* have proven that it is fairly easy to compromise a public Wi-Fi positioning system with three simple attacks - AP impersonation, AP Replacement and Database Manipulation, where for example a mobile device can be led to believe that it is in another position than it really is [35]. This shows that security also must be taken into account when designing a Wi-Fi positioning system.

As mentioned earlier, the privacy in a position system is generally fairly good when the position estimate is calculated at the client-device and fairly bad when it is calculated in the infrastructure. This is also true for Wi-Fi-positioning and both techniques can be used with either the geometrical range-based approach as well as with fingerprinting.

3.3 Proposed Solution

When designing the Wi-Fi location system for this thesis, a lot of effort has been put into analyzing what requirements and constraints the device of choice imposes on the system. The cost of the implementation and if it is viable within the scope of the thesis has also had a great impact on the solution.

Furthermore the offline nature of the device forces the position estimation to be done in the infrastructure instead of in the client-device. As previously stated this compromises the user privacy of the system, but is a necessity to make it work at all. Moreover the focus of the positioning system lies in client-side energy efficiency and global reach rather than in optimized accuracy.

3.3.1 System design

During the design process uttermost importance was given to the database. It had to be well-populated as well as the cost per query had to be very low. On these premises *WiGLE* was seen as the best alternative, as it is open-source and very well-populated. *WiGLE* gathers wardriving data and calculates the positions of the APs themselves instead of storing fingerprints of Wi-Fi-signals. This feature makes the use of a geometrical range-based approach for position estimation very feasible. Figure 3.1 below describes the flow within the proposed system is done.

1. A snap with AP hardware addresses (MAC addresses) and the received signal strengths are fed into the algorithm.

2. The geographical positions, in longitude and latitude, of the APs are queried from the *WiGLE*-database.

3. The distance from the mobile device that recorded the snap to every AP is modelled from the received signal strength indication (RSSI). This process is further described in Section 3.3.2 below.

4. The geographical position as well as the modelled distance for every AP is fed to an algorithm that estimates the position of the mobile device at the time of snap. The weighted centroid positioning algorithm proposed by Bin *et al.* that the system utilizes is further described below [5].

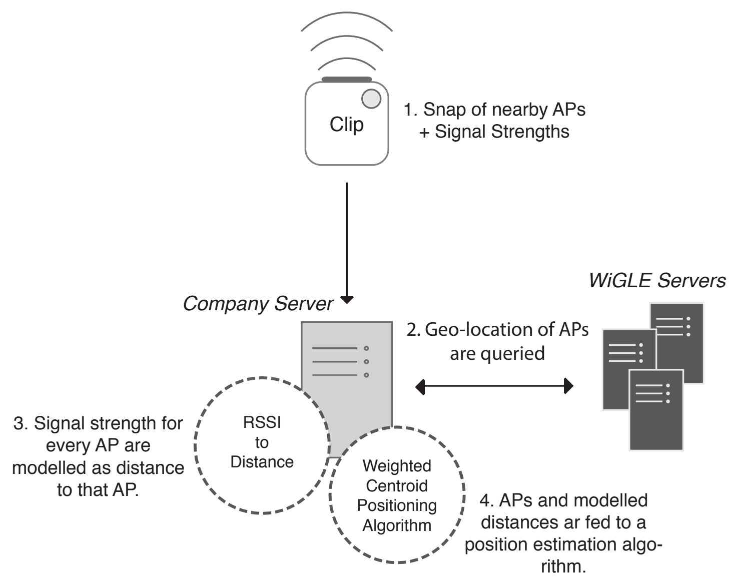

Clip
1. Snap of nearby APs + Signal Strengths
Company Server
WiGLE Servers
2. Geo-location of APs are queried
3. Signal strength for every AP are modelled as distance to that AP.
RSSI to Distance
Weighted Centroid Positioning Algorithm
4. APs and modelled distances ar fed to a position estimation algorithm.

Figure 3.1: Multilateration positioning algorithm for Wi-Fi.

3.3.2 Distance modelling

To model the distance from the Clip to an AP, the following formula proposed by Cook *et al.* is used [10].

$$P_r = P_t + 20\log\left(\frac{\lambda}{4\pi}\right) + 10n\log\left(\frac{1}{d}\right) \tag{3.1}$$

P_r denotes the power of the signal at the receiver (dBm), λ is the wavelength, n is the path-loss exponent and d is the distance in meters between transmitter and receiver. Due to the fact that an AP can be a broad number of different devices made by different manufacturers, the transmitting power (P_t) can vary a lot. This is not a problem in a closed localization system where the same type of device can be chosen for all APs. However it imposes a much bigger problem in a public setting where there is no way to keep track of all the different device types and their transmission power. Despite this P_t is set to the constant of 20dB due to the fact that the accuracy of the system is not critical and the additional estimation error this imposes is acceptable.

The wavelength of Wi-Fi-signals is derived from the formula for wavelength $\lambda = \frac{c}{v}$ and results in a wavelength of 0.125 m for 2.4 GHz transmission and 0.06 m for 5 GHz transmission. n is a path loss exponent and can be chosen arbitrarily to fit the designed system, but is affected by the radio propagation properties off the radio waves in a certain setting. Since the proposed location system is to operate in a public and global setting, it is impossible to choose a path loss exponent that is perfect in every situation. As previously explained, this is due to obstacles like cars, walls, trees and people affecting the radio propagation of the signal. In these kind of situations n can be set to a value between 2.7 and 3.6 and the proposed system uses a n of 3.5 [36]. With these simplifications the final formula used by the system looks like:

$$P_r = 20dB + 20\log(\frac{0,125m_{(2.4GHz)}}{4\pi}) + 10 * 3.5 * \log(\frac{1}{d}) \tag{3.2}$$

3.3.3 Position estimation

The weighted centroid positioning algorithm proposed by Bin *et al.* that is utilized in the proposed system, takes the position of all the APs and estimates the position of the mobile device through multiple multilateration [5]. It is basically multilateration over all different combinations of triangles within the set of APs. These multilaterated positions have a weight based on the modelled distance to the mobile device, which is included when a weighted mean position is calculated from all these multilaterated positions. The complete algorithm is described below.

1. Check the position of every AP with a lookup to *WiGLE* (or another database with similar features). Store them in a set.

2. Model the distance d for every AP to the position of the Clip with the formula previously described and store it together with the position of the AP. See Equation 3.2. d_x in equation 3.3-3.6 denotes the distance in meter from AP x to the Clip.

3. Create a triangle with every combination of AP-positions in the set. The number of triangle combinations is $\binom{n}{3}$, where n is the total number of APs.

4. Estimate the position $N(x_i, y_i)$ for every triangle i.

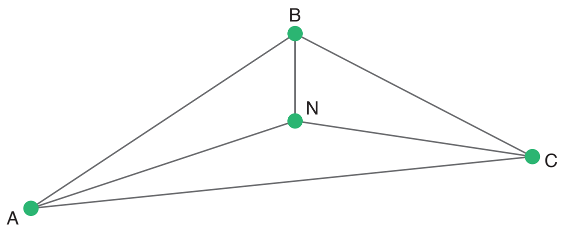


Figure 3.2: Weighted position N estimated for the triangle ABC

The coordinates x_i and y_i are calculated as presented in equation 3.3 and 3.4 below:

$$x_i = \frac{x_a \times (\frac{1}{d_a+d_b}) + x_b \times (\frac{1}{d_b+d_c}) + x_c \times (\frac{1}{d_a+d_c})}{\frac{1}{d_a+d_b} + \frac{1}{d_b+d_c} + \frac{1}{d_a+d_c}} \tag{3.3}$$

$$y_i = \frac{y_a \times (\frac{1}{d_a+d_b}) + y_b \times (\frac{1}{d_b+d_c}) + y_c \times (\frac{1}{d_a+d_c})}{\frac{1}{d_a+d_b} + \frac{1}{d_b+d_c} + \frac{1}{d_a+d_c}} \tag{3.4}$$

5. Calculate the ratio L_i between the longest and shortest side in the triangle and store it together with x_i and y_i.

6. When x_i and y_i have been calculated for all the triangles the final estimation x and y for the position can be calculated as proposed by Bin *et al* [5].

$$x = \frac{\sum_{i=1}^{k} \left[x_i \times (\frac{1}{d_{a(i)}} + \frac{1}{d_{b(i)}} + \frac{1}{d_{c(i)}} + \frac{1}{d_{L_i}}) \right]}{\sum_{i=1}^{k} \left[\frac{1}{d_{a(i)}} + \frac{1}{d_{b(i)}} + \frac{1}{d_{c(i)}} + \frac{1}{d_{L_i}} \right]} \tag{3.5}$$

$$y = \frac{\sum_{i=1}^{k} \left[y_i \times (\frac{1}{d_{a(i)}} + \frac{1}{d_{b(i)}} + \frac{1}{d_{c(i)}} + \frac{1}{d_{L_i}}) \right]}{\sum_{i=1}^{k} \left[\frac{1}{d_{a(i)}} + \frac{1}{d_{b(i)}} + \frac{1}{d_{c(i)}} + \frac{1}{d_{L_i}} \right]} \tag{3.6}$$

The initial testing showed that the general estimation error was good (about 20-30 m), however some of the positions were way off and had an estimation error of several hundred meters. Some research showed that this was due to some APs in the *WiGLE* database being placed in moving objects such as busses and taxis, which naturally means that the position stored in the database for this AP can be way off its actual position at a given moment. To solve this problem a simple solution to remove outliers among the set of APs can be added to the previously explained algorithm.

The solution takes advantage of the fact that Wi-Fi signals have a limited travel distance depending on the environment. In a complex environment such as an urban setting this is generally less than 150 m [30].

The median position in latitude and longitude from the set of APs is taken as a reference mark. Even though this is not the exact position of the mobile device, it will be close, due to the fact that it is quite a small area that the APs are positioned in. Threshold values for latitude and longitude are then calculated as follows:

$$\begin{aligned} MaxY(latitude) &= medianYi + 200m \\ MinY(latitude) &= medianYi - 200m \\ MaxX(longitude) &= medianXi + 200m \\ MinX(longitude) &= medianXi - 200m \end{aligned} \tag{3.7}$$

All the APs are compared to these thresholds and the ones that are outside are identified as outliers and removed from the set. These calculations are done before step (3) in the previously proposed algorithm. Further testing has showed that this approach reduced the errors of the outlier inferred estimates to the same magnitude as the other estimates of about 20-30m.

3.4 Evaluation and Discussion

Even though the thought and design effort has been put into the algorithm based on *WiGLE* and geometrical position estimation through multilateration, a second Fingerprint-based implementation has also been made. This implementation utilizes the powerful *Skyhook* database. This section aims to evaluate both of these implementations against each other as well as to the GPS-measurement and the real physical position of the measurement point.

The evaluation was done with the simulation setup explained in Appendix A, where the app *WifiScanner* on a *OnePlus One* is used for Wi-Fi measurements, the Clip for GPS measurements, *Google Maps* to pinpoint the real location and a Python-script to simulate the Company server.

The evaluation was made in the city of Lund in Sweden the 26 of February 2015 over the course of 1 hour where measurements were made at 24 different locations. To further test our solution we made sure to make some of these measurements at a big cemetery to simulate the situations where the density of Wi-Fi APs is low. Below in Figure 3.3 the location of the all the measurement points during the evaluation is presented.

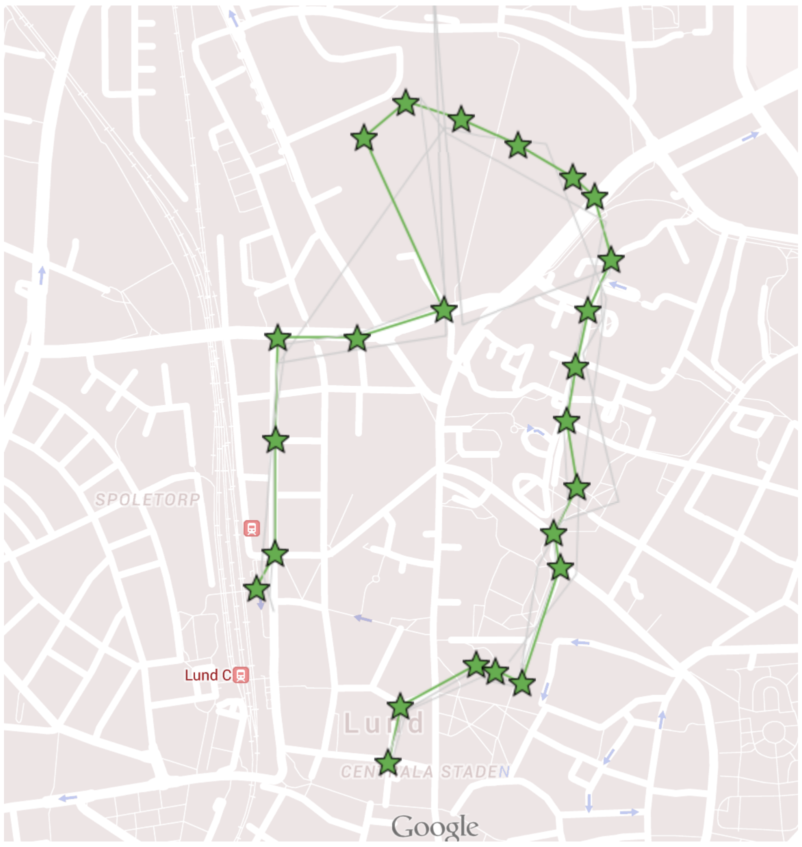


Figure 3.3: Location of measurement points during evaluation. A total distance of approximately 3.1 km was covered and measurements at 24 different location made during the data gathering.

At every measure point GPS data were recorded with the Clip and Wi-Fi data were recorded with the *WiFiScanner* Android application. The GPS data were then sent to the servers of the GPS-chip manufacturer for a position estimation while the Wi-Fi data were processed by two different Python-scripts. One Python script that first queries *WiGLE* for all the AP-positions and then estimates the position with multilateration as described in the section proposed solution. The other Python script sends all the fingerprints of APs to *Skyhooks* servers that answers with an estimated position. Below the results of these three processes are presented in Figure 3.4 - 3.6 in the order - GPS, Multilateration, *Skyhook* estimation.

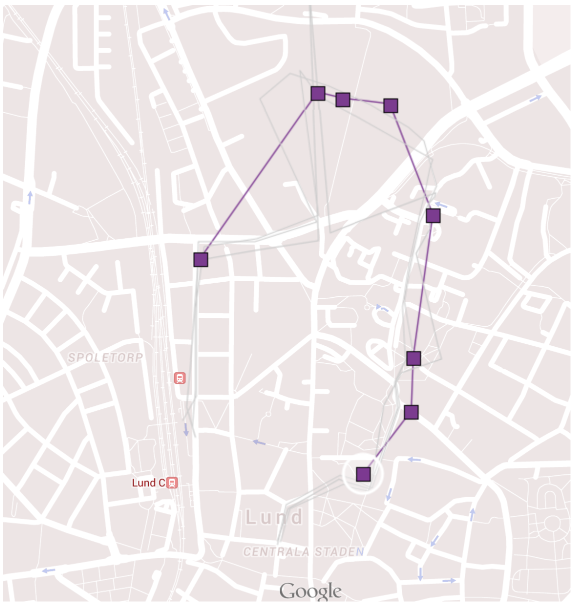


Figure 3.4: Positions that were estimated by the 3rd party GPS, mentioned in 1.2.2. 8 out of 24 measurement points obtained a position fix.

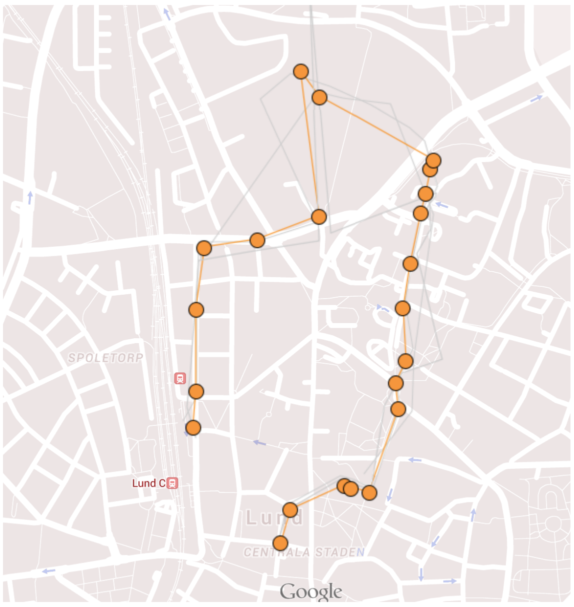


Figure 3.5: Positions that were estimated with the Multilateration implementation proposed in Section 3.3.3 where the AP-positions were queried from the *WiGLE* database. 22 out of 24 measurement points obtained a position fix.

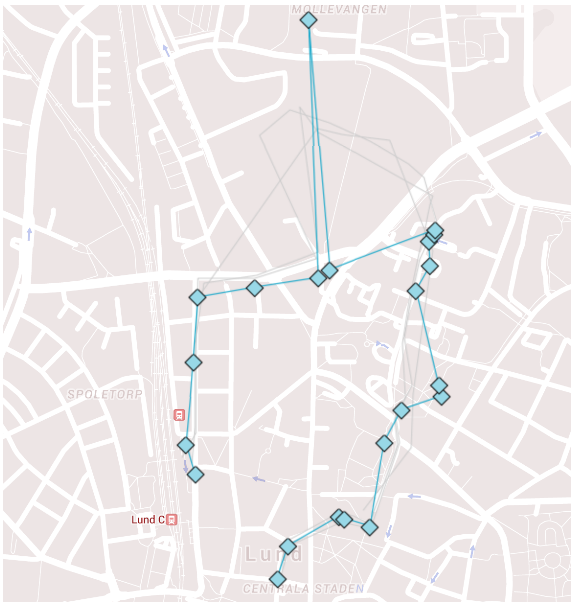


Figure 3.6: Estimated position from *Skyhook*. 22 out of 24 measurement points obtained a position.

In Table 3.1 the complete statistics for the evaluation session can be seen. Just as the literature study suggests, the evaluation shows that GPS localization performs very poorly in an urban environment while Wi-Fi positioning performs well regardless of the method of choice. Surprisingly our own implementation using multilateration and the *WiGLE*-database outperforms *Skyhook's* implementation a lot with a matching result of 22 positions obtained, but with a mean error of just 27.2 m opposed to 80.3 m for *Skyhook*. However it should be noted that we, ourselves, have been wardriving a lot for *WiGLE* in Lund and the database therefore possibly is more accurately updated there than in other places.

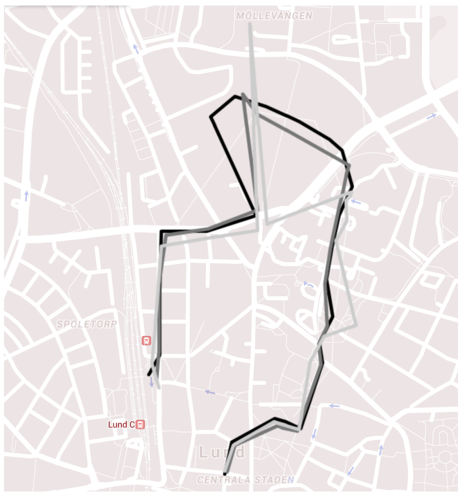


Figure 3.7: The real walking route in black, estimated walking route with our implementation in dark gray and the estimation by *Skyhook* in light gray.

Figure 3.7 shows a representation of the interpretation the Company's backend can make of the route the person who made the measurements has been taking based on the position estimates made with *WiGLE* and *Skyhook* compared to the actual route.

Measurement Point	**GPS**	**Multilateration**	**Skyhook**
1(14:24:33)	No fix	11 m	10 m
2(14:26:52)	No fix	15 m	13 m
3(14:30:18)	No fix	28 m	13 m
4(14:32:04)	32 m	24 m	27 m
5(14:33:58)	No fix	15 m	9 m
6(14:37:48)	No fix	8 m	43 m
7(14:40:07)	93 m	4	28 m
8(14:41:58)	25 m	37 m	88 m
9(14:45:06)	No fix	18 m	162 m
10(14:47:16)	No fix	4 m	28 m
11(14:48:46)	42 m	28 m	33 m
12(14:51:50)	No fix	36 m	54 m
13(14:55:20)	No fix	76 m	157 m
14(14:56:50)	No fix	111 m	195 m
15(14:59:44)	81 m	No fix	No fix
16(15:02:20)	77 m	40 m	405 m
17(15:04:42)	17 m	30 m	256 m
18(15:07:16)	No fix	No fix	No fix
19(15:13:51)	No fix	19 m	51 m
20(15:16:55)	No fix	9 m	23 m
21(15:19:18)	46 m	22 m	46 m
22(15:22:00)	No fix	17 m	29 m
23(15:25:11)	No fix	15 m	41 m
24(15:27:43)	No fix	31 m	56 m
Obtained position fixes	8/24	22/24	22/24
Mean error	48.5 m	27.2 m	80.3 m

Table 3.1: Statistics for the entire evaluation session.

As can be seen by the result, both the interpreted routes are fairly good. The *Skyhook* implementation fails to obtain good results in the cemetery we walked through. Since *Skyhook's* implementation and wardriving methods are somewhat of a black box it is difficult to say why their algorithm performs worse than our implementation under these circumstances (measure point 16 and 17 in Table 3.1). However, one can speculate that this is the symptom of wardriving the database with powerful equipment on a car instead of walking around. This method would build a database where APs far away from a car road have a big error and thus also impose a big error on the position estimation the database is utilised for in these places.

In all the above multilateration calculations the path-loss exponent n is set to 3.5. The reason for this is that we conducted a study with n between 2 and 4 and n=3.5 gives a small mean error as well as being in the range of 2.7-3.6 suggested by Torlak suggests [36]. The results of this study is presented in Figure 3.8. What can also be seen is that the difference between the *n*=2 and *n*=4 is very small, even though this in fact is a huge difference. This suggests that the formula for converting RSSI to distance in use is suboptimal for the circumstances and that the results could be further improved if a better formula is used.

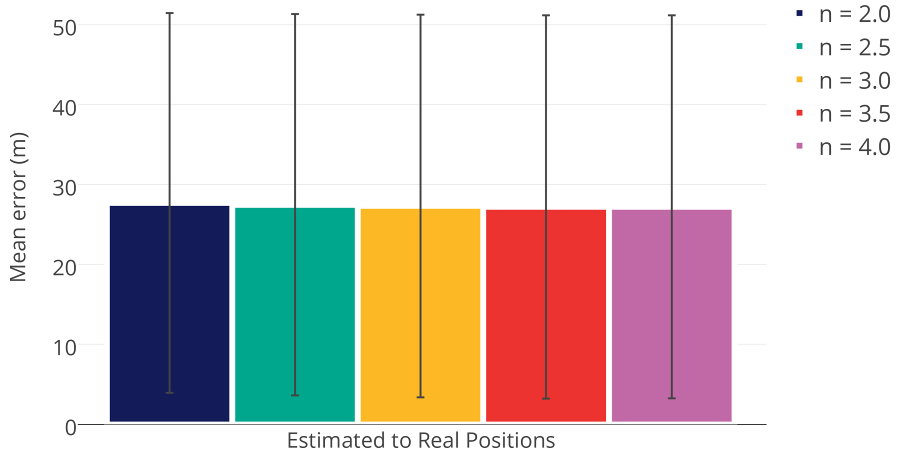


Figure 3.8: Comparison between what signal propagation constant, n, that gives the best mean error.

In Figure 3.9 below the mean error and the error standard deviations for multilateration and *Skyhook* are presented. Also a third implementation that utilises the median position for a set of APs to estimate the position is showed. The AP positions for this implementation is also queried from *WiGLE*. Compared to the multilateration the mean error of an implementation like this is very close, however the standard deviation of the estimation error is larger. For an implementation like this the computational complexity is much lower than for the multilateration approach and what method that should be used is thus a trade-off between accuracy and needed computational resources.

All the results presented in this chapter show that the multilateration approach using AP positions queried from *WiGLE* outperforms *Skyhook's* solution a lot. This might not always be the case though and a discussion about the usage of the different databases is presented in Section 6.1.1.

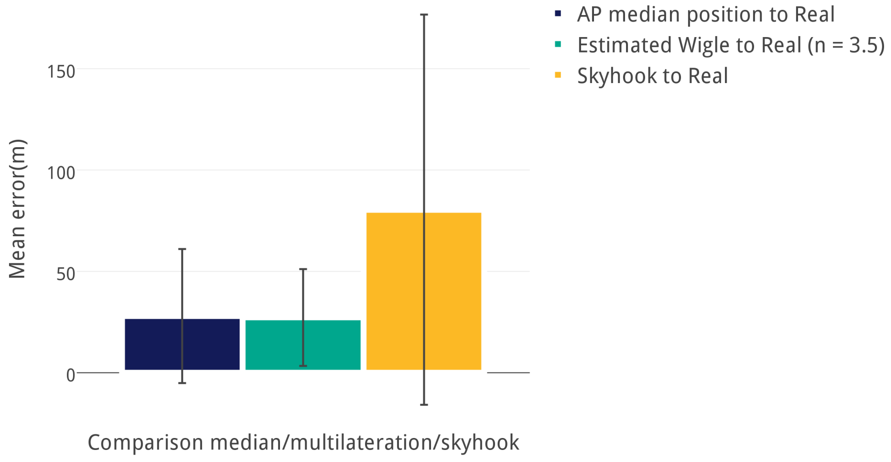


Figure 3.9: Comparison of mean errors between the three different algorithms.

3.5 Summary

In this chapter we have explored two different methods for Wi-Fi localization, Multilateration and Fingerprinting. Furthermore, we propose a Multilateration solution that uses WiGLE as a provider of AP geo-locations. Following this, an evaluation of Fingerprinting and Multilateration opposed to GPS and actual locations was presented where the Multilateration approach outperformed the Fingerprinting approach with a mean-error of 27 m opposed to 80 m.

In the next chapter, Chapter 4, we explore how to use an accelerometer for Activity Recognition and propose an algorithm to distinguish between when a user has moved and when she has not.

Chapter 4
Activity Recognition

In the previous decade there have been major improvements in Communication Technology and sensor miniaturization. This has enabled a new wave of technologies aiming to monitor human-behaviour, not the least through mounted tri-axial accelerometers that can be used for example to distinguish between activity and rest, what activity that is performed as well as what mode of transportation a person is currently travelling with [18][20][27]. Since accelerometers are very energy efficient, the utilization of them is of great interest to us during this thesis project. We look into how acceleration data can be used to make a localization algorithm more energy efficient and in some cases even enhance the precision. Our basic idea is that if a distinction can be done between activity and rest, many of the previous position measurements can be avoided and thus also save a lot of energy.

4.1 Related Work

Karantonis *et al.* present a solution how a single tri-axial accelerometer mounted on the hip with a very limited buffer-size can be used to decide if the subject is active or resting and even classify between a few different activities (walking, sitting, lying, falling). They do the processing and classification of the activities online, meaning that it is done on the device itself and not later on a server. They used an accelerometer with a buffer size of 2KB which was enough to store 1 second of data [20].

Hemminki *et al.* aim and succeed with using a single tri-axial accelerometer in the form of a smartphone mounted anywhere on the body to classify between different transportation modes of a subject [18].

Mizell shows how some trivial estimations of the gravity constant can be used together with some basic vector mathematics to estimate the gravity constant of the three axes of a tri-axial accelerometer as well as to project the acceleration of an axis on a horizontal and a vertical vector [28]. Something that many researchers have been using to help in classification of activities [18][23].

Figo *et al.* stress multiple uses for activity classification. They evaluate how different parameters such as FFT (Fast-Fourier Transform), max, min, median, range etc. of the acceleration data can be used to detect such activities. The areas they look at are efficiency when recognizing activities as well as the computational complexity of using the different parameters [12].

Lu *et al.* present the design, implementation and evaluation of the *Jigsaw Sensing Engine*, which is an underlying mobile phone service that uses acceleration data from an accelerometer as well as sound data from a microphone to optimize the utilization of a GPS-sensor. They propose a solution to the problem of the possibility that a phone is used in different ways by different subjects and thus need a robust technique for processing the data. Furthermore they solve the problem with acceleration data being device specific through the process of device calibration before using the data [23].

Siewiorek *et al.* utilise a tri-axial accelerometer taped to the abdomen to classify user activities by three different levels, low, medium and high. They do this through looking at parameter thresholds for different activities [32].

Veltink *et al.* deep-dive into the possibilities of a seismic accelerometer. Although the paper is a bit outdated and several bi-axial accelerometers are used, they propose some interesting ways to distinguish between activities. They apply a threshold on the processed accelerometer signal to distinguish between static and dynamic activities. The parameters they investigate are mean value, standard deviation, cycle time and morphology of the accelerometer signal [27].

Godfrey *et al.* uses the x, y and z - accelerations as well as pitch, yaw and roll on a tri-axial accelerometer mounted on the chest for detection between lying, sitting, standing and walking as well as of the postural transitions from standing to sitting and sitting to standing [16].

Unlike a lot of earlier research where the accelerometer must be placed at a fixed position, Sun *et al.* recognize the rather varying use of mobile devices and propose a solution for classification between seven different physical activities. The main feature they look at is SVM (Support Vector Machine) and they succeed to do a fairly accurate classification. They also look into the behavioural patterns of mobile device users and provides a detailed description of how the feature extraction is done [34].

4.2 Our Work

As previously stated the Clip uses two operating systems, one custom made OS to run low-power components as the accelerometer and the camera as well as one Linux-based OS to run heavier components such as the GPS-sensor, the Wi-Fi sensor as well as software for transferring pictures to an external device. Up to now both of these OSs have been run to ensure that a position measurement is made. However this is far from optimal since the energy consumption is far greater than if only the trimmed OS for running the camera and the accelerometer are used. If the research Lu *et al.* conducted is to be believed a modern day user might be stationary as much as 77.3 % of the time during the work-week and up to 38.1 % of the time during weekends. Even if the usage pattern of the Clip is not identical to that of a mobile phone, this data still implies that a big portion of the position measurements can be avoided with the application of an algorithm that uses acceleration data to determine what activity is being performed at the moment. If the user is stationary, the previously made position estimate can be used instead of starting the Linux-based OS and performing a new one. Without any reasonable doubt this has the potential to extend the battery-life of the portable device a lot.

During the research a lot of effort has been put into fully understanding the area of Activity Classification through the analysis of accelerometer data. Furthermore, focus has been on finding a fast (computation complexity) and cheap (energy consumption) way to classify the activities. The reason for this is because even if a position measurement has to be obtained, the portable device still only should be on for a maximum of 500 ms from the point that it wakes up until it goes in to sleep again. As previously stated a Wi-Fi-AP usually broadcast its announcement message every 100ms, which means that the Wi-Fi-measurements has to be run at least 101ms to capture all APs around. Add the boot-time for the Linux-based OS and the time for taking a picture and we have a very limited time-window to process and classify the accelerometer data within. During the research some exploration has also been conducted into the area of what continuously gathered acceleration data can be used for if it besides being a powerful way to save energy on the portable device, also was to be saved together with the pictures and processed on the Company's servers where a lot more computational power can be utilized and energy consumed.

The activities that have been identified as the activities that should be classified by the algorithm are activities with cyclic movements such as walking, running and biking. Also activities where the user despite being stationary still moves, such as going by car, going by bus and going by train, is important when a decision is made if a position measurement should be obtained. As a minimum the algorithm must be able to distinguish if the user is performing one of these activities or not. Ideally it should be able to identify what specific activity was performed.

All state of the art activity classification algorithms follow the same step by step formula during the process of classification [18][20][23]. First the raw data are sampled from the accelerometer and are then preprocessed to remove noise and artefacts. After that features are extracted from the preprocessed data, and the classification is then made by analyzing these features [18][20][23].

4.2.1 Theory

Sampling and preprocessing

When classifying body movements it is important to sample the accelerometer data at a frequency that covers all the frequencies where bodily movements are performed. Karantonis *et al.*'s research shows that all bigger bodily movements are contained within 20Hz. This even holds for walking and running where 99 % of the energy are contained within 15 Hz [1]. When Nyqvist's theorem, which states that sampling must be done at the double rate for no information to be lost, is applied this means that sampling has to be done at 40 Hz to capture all the bodily movements that need capture for this algorithm; walking, running, biking, sitting, standing. However, this algorithm also needs to make a distinction between standing/sitting and standing/sitting on a motorised transport such as a bus, a train and a car. This can possibly be done by capturing smaller bodily movements caused by the vibration of the mode of transport. To capture detailed information like this a higher sampling rate is needed. For example Hemminki *et al.* samples at the rate of 100Hz [18].

After sampling the raw accelerometer data, the data are preprocessed to make it more suitable for feature extraction [12][18][20][23][34]. The reason for this is that accelerometer data generally have a lot of noise caused by the small vibrations within the accelerometer as well as extraneous activities such as user-interactions. This is commonly done through applying a frequency filter such as a low-pass filter, a high-pass filter or a median filter and some researchers even use combinations of these filters [18][20]. For example Karantonis *et al.* first applies a median filter of magnitude 3, which is a filter that takes every data point and makes it the median-value of the data point itself, the previous value and the next value, and then adds a low-pass filter on that to further smooth out the data [20].

In this stage of preprocessing the data, the signal is commonly divided into frames, a process called framing [18][20]. This is to enable estimation of the gravity component of the signal as well as to make feature extraction more feasible.

Following framing, the gravity component of the accelerometer signal is usually extracted and the resulting signal is then projected on a global coordinate system [12][18][20] [34][23]. Traditional activity classification generally assumes that the used accelerometers have a fixed position and orientation [3][11][32][27]. Although this might have been feasible when doing professional research, it becomes unfeasible when classifying activities with a commercial device. The reason for this is that there, unlike in a controlled environment, is no way to control how the device is used. To enable the classification of activities when the position and rotation of the device are unkown, the acceleration data are projected on a global coordinate system in vertical and horizontal direction [18][23][28]. Mizell *et al.* propose a well-cited solution of how to do this.

For a chosen interval, typically a few seconds, an estimate of the gravity component of each axis is obtained. Lu *et al.* uses an interval of four seconds with success for this [23]. This process of averaging all the readings of an axis in an interval produces a fairly accurate measurement of the gravity component of the signal. The gravity component vector is denoted as $v = (v_x, v_y, v_z)$, where v_x, v_y and v_z are the gravity component of the axis' x, y and z during the interval. Then let $a = (a_x, a_y, a_z)$ be the raw acceleration data (motion imposed acceleration + gravity imposed acceleration). The dynamic acceleration can then be denoted as $d = (a_x - v_x, a_y - v_y, a_z - v_z)$. The vertical projection of the dynamic acceleration is then obtained through applying the dot product on d and v as $p = \left(\frac{d \cdot v}{v \cdot v}\right) v$. With other words p is the vertical component of d. Since a three-dimensional vector is the sum of its horizontal and vertical components, the horizontal projection h is thus $d - p$ [28].

However, there is some criticism towards this approach. Hemminki *et al.* states that this method does not capture the slower acceleration such as the acceleration/deceleration a train makes when starting or stopping at a station. This is due to the intervals for gravity estimation being short, typically a few seconds. Slower acceleration thus gets falsely included in the gravity estimation [18]. They propose a solution for this problem, however it is not included in this report.

With the accelerometer signal projected on the global coordinate system it is possible to extract features from the data and thus enable classification of activities regardless position and orientation of the accelerometer [23].

Feature Extraction

When processing a sensor signal, three different domains of features can be considered: the time domain, the frequency domain and the discrete representation domain [12]. The time domain features are popular in activity classification algorithms due to their low computation complexity as well as they can be obtained directly from raw accelerometer data [12].

The frequency domain and discrete representation domain require more computational power, but are better to recognize some types of activities. For example the fast fourier transform (FFT), which is a representation of the frequencies in a time signal, is good at capturing cyclic activities such as walking, running and biking. This is due to a frequency spike occurring at the frequency of the periodic activity. An activity with a periodic pattern repeated every 0.5 seconds shows a frequency spike at 2 Hz [12]. Since walking is generally done in a repetitive manner of 1-3 seconds, the occurrence of a frequency spike within the range of 0.33-1 Hz implies that the subject has been walking [18]. Running and biking obviously has a frequency spike at a higher frequency. Veltink *et al.* contradictory found that the cycle time of the signal rather had to do with the speed of the performed activity than the classification itself, but it has to be noted that their paper was written in 1996 and that they make use of 5 carefully placed bi-axial accelerometers rather than one arbitrarily placed tri-axial accelerometer.

Another time-domain feature that is very effective for activity classification when the position and orientation of the accelerometer are not know is the Signal Magnitude Area (SMA), which is a measure that includes the signal variations in all three axes [20]. Sun *et al.* propose a classification algorithm where they employ Mean-value, Variance, Correlation, FFT Energy and Frequency domain entropy to classify between seven different physical activities, while Veltink *et al.* propose a solution that combines the Mean-value with the signal morphology for the classification.

As can be seen by the previous research a number of different large variety of different features and combinations of features can be used for the classification and in the end it comes down to the following factors proposed by Karantonis *et al.* when choosing what features that should be used by the classification algorithm:

- Knowledge of the environment in which the classification should work.
- How much data that can be buffered by the accelerometer.
- The amount of processing time that is available and how much energy the algorithm is allowed to consume [20].

Activity Classification

When classifying activities it is common to use a decision tree [18][20][23][24][26]. A decision tree is an algorithm that makes big classifications such as if the subject is active or stationary in the higher levels of the tree and more detailed classifications such as what specific activity that is performed in the lower levels. Many of the researched papers are focusing on health-related questions like *How many calories is the subject burning?* or *How can a health-care person get an automatic notification if this elderly person falls?* [9][14][16][17][24]. For this kind of classification it is important to be able to classify transitions from for example sitting to standing, standing to sitting, standing to lying etc. This is also in line with the decision tree Mathie *et al.* proposed in 2004 [26].

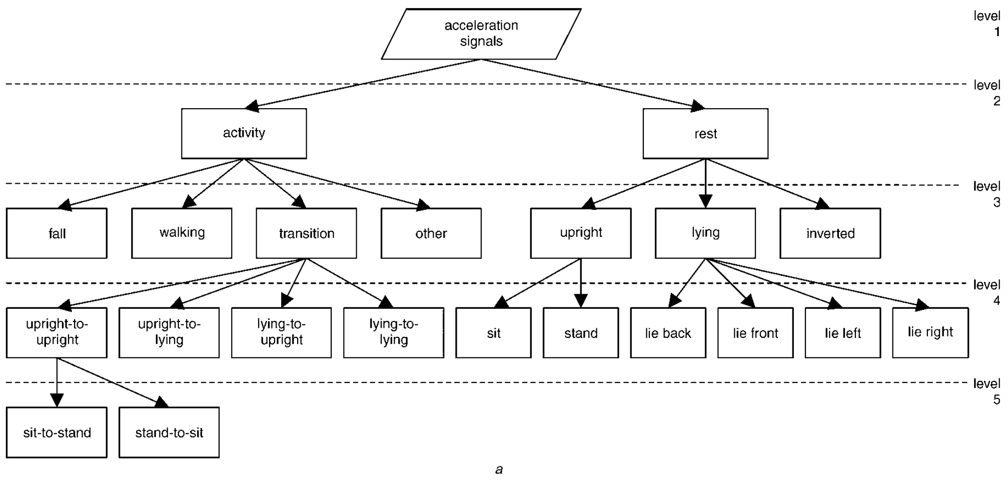


Figure 4.1: Decision tree used by Mathie *et al.* [26].

Karantonis *et al.* instead uses a binary decision tree to distinguish between activities without the need to classify the transitions. The aim of their paper is to propose a solution for metabolic energy expenditure [20].

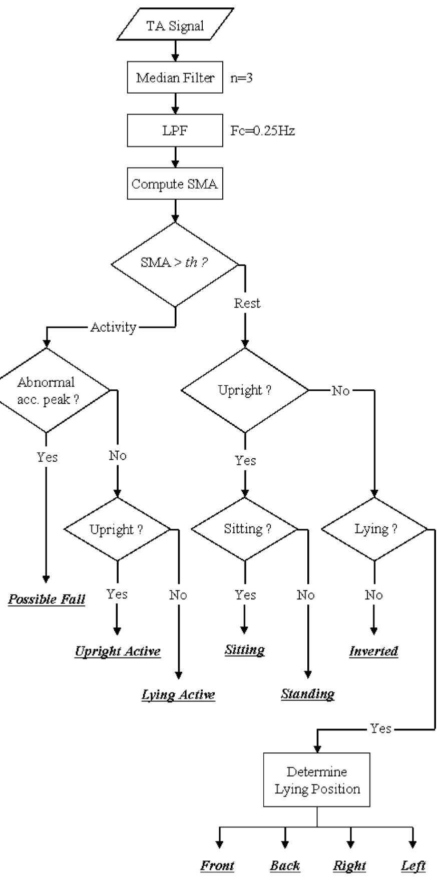


Figure 4.2: Binary decsion tree used by Karantonis *et al.* [20].

It should be noted that both of these decision trees utilize the angular position of the axes to distinguish if the subject is upright, sitting or lying. This can be done either through calculating of the angle between the estimated gravity vector and the measured acceleration vector or through the use of a gyrometer, a sensor that measures the change in angular direction of the different directions.

A more difficult task when constructing the decision tree is encountered by Hemminki *et al.* who not only aim to classify dynamic and stationary activities performed by a human being, but also try to distinguish between standing/sitting and standing/sitting in a motorised vehicle. They do this with success and their proposed decision algorithm is depicted in figure 4.3.

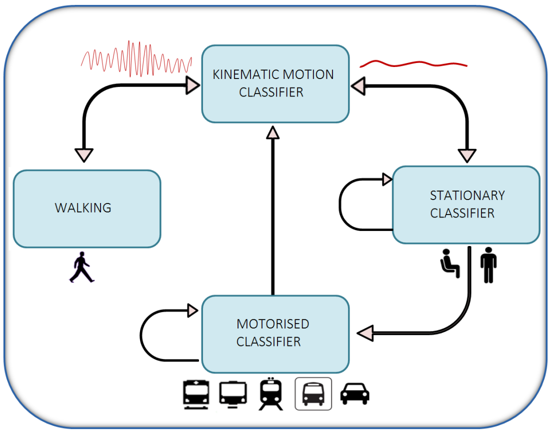


Figure 4.3: Proposed decision algorithm by Hemminki *et al.* [18].

Just like Mathie *et al.* and Karantonis *et al.*, Hemminki *et al.* have a kinematic motion classifier that distinguishes between activity and rest at the root of the algorithm. If an activity is classified as stationary a distinction between "normal stationary" and motorised stationary is done. The motorised classifier then makes the distinction between mode of transport. This classifier uses the acceleration and deceleration patterns of the horizontal acceleration vector to make the distinction [18].

4.2.2 Proposed Solution

In this section we explain our approach to constructing a classification algorithm that succeeds with fulfilling the previously stated requirement of being able to distinguish between dynamic activity (walking, running, biking), rest (standing, sitting) and travelling by a motorised vehicle (train, bus, car). An aim has also been to do the classification between the activities themselves, but this has been more of a wishful goal than something mandatory for the algorithm to work. Just like the previous section, theory, this section is divided into the sections *Sampling and Preprocessing*, *Feature Extraction* and *Activity Classification*. In *Sampling and Preprocessing* and *Feature Extraction* we describe how we do these activities, while we in *Activity Classification* propose two different decision trees that, in theory, can be used to make the desired classification. For the solution to work, the accelerometer data have to be recorded continuously. This can be done either through using a long buffer for the accelerometer or letting a small part of the chip run continuously even when in sleep.

Sampling and Preprocssing

Just as Hemminki *et al.* who aims to classify motorised transport, we sample with the frequency of 100Hz. As previously stated this frequency captures all the bodily movements of a human being.

During the preprocessing the same model as Karantonis *et al.* is used. A median-filter of magnitude 3 is applied to the data followed by a low-pass filter.

The filtering is followed by segmenting the data into frames of 256 samples. Every frame thus consists of 2.56 seconds of data. The reason for this window length is that it is at a suitable length for gravity estimation as well as it divides the 30 second accelerometer sample into approximately 12 frames [23].

After framing the data, the previously presented method by Mizell is used to estimate gravity and project the accelerometer samples on a global coordination system.

The vertical projection of every axis is summed together to create the total vertical acceleration. The same is done for the horizontal projection.

The result of the preprocessing is thus a series of frames with the total dynamic vertical acceleration as well as a series of frames with the total dynamic horizontal acceleration. These two series of frames are used for feature extraction and activity classification.

Feature Extraction

The goal of this section has been to analyze different features that, at minimum, can be used to distinguish between activity and no activity, but more desirably enable the classification of the following activities: walking, running, biking, sitting, standing, going by train and going by bus.

The features we have chosen are all frame-based features (a value that is computed for a whole frame), since they are very effective when capturing high-motion activities. Although it should be noted that these kind of features fail to capture slow movement like acceleration and deceleration of a train. However, as previously explained, the method for gravity estimation we use make detection of that kind of movement unfeasible anyway. All the features we have considered are time-domain features. The reason for this is that they have a low computational complexity compared to discrete-domain and frequency domain features. Since energy and speed are of uttermost importance when processing, the accelerometer data on the wearable device time-domain features suit this algorithm perfectly.

For every frame we consider the following features: Mean-value, Standard Deviation, Variance, Max-value, Min-value, SMA, Zero-crossings and Median. Veltink *et al.* uses thresholds to distinguish between activity and rest and our goal has been to find unique thresholds for a feature or the combination of features that can be used to distinguish between the different activities.

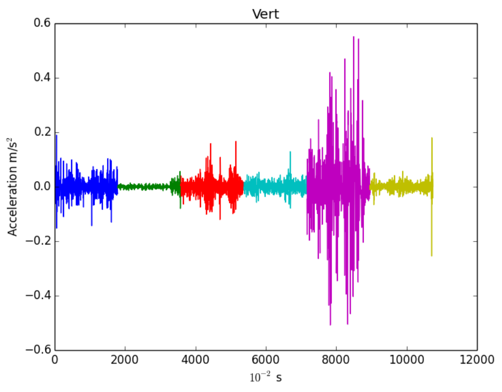
To find these unique thresholds a pilot study with three test subjects was conducted. The test subjects were two men measuring 1.80-1.90 m and a woman measuring 1.60 m. The subjects were chosen to get varying data for each activity and in that way pave the road for a more robust decision tree. The data from the pilot-study were used to train a decision tree that can distinguish between the different activities.

Activity Classification

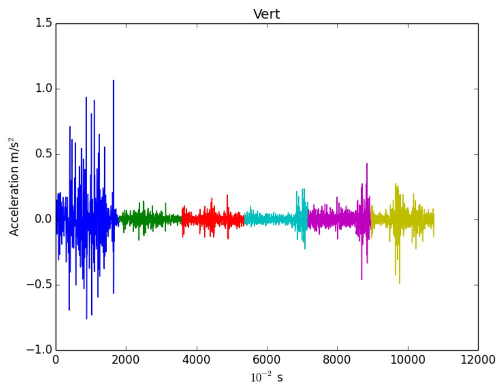
As previously mentioned we have chosen to use a decision tree when classifying our activites. The reason for this is that it is very good in terms of efficiency at the same time as it reaches the same level of performance as other classification algorithms [23]. The decision tree is created from the pilot-study data with help of the decision tree training algorithm CART [31][37].

The decision tree is built upon the vertical projection of the acceleration data. This is due to the data varying more distinctly between the different activities as well as being more trustworthy for the vertical projection than the horizontal projection [18].

As can be seen below in Figure 4.4-4.7 the gathered data from the pilot-study were varying a lot. In all the figures six different sample-series have been concatenated. Sample series 1 corresponds to the measurements of test person 1 when the wearable device was mounted at a chest position, series 2 corresponds to the thigh mount of the same person. Series 3-4 are measurements for test person 2 and series 5-6 are the measurements for test person 3. Both with the same order for Chest and Thigh as person 1. The horizontal axis corresponds to the acceleration (m/s^2) and the vertical axis to index of the sample. Every series consists of 7 frames where every frame includes 256 measurement samples. With a sampling rate of 100 Hz this means that each sample series corresponds to 18 seconds of activity.



(a) Vertical projection of acceleration data while sitting.

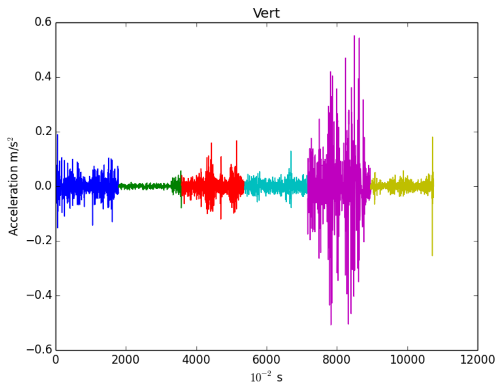


(b) Vertical projection of acceleration data while standing.

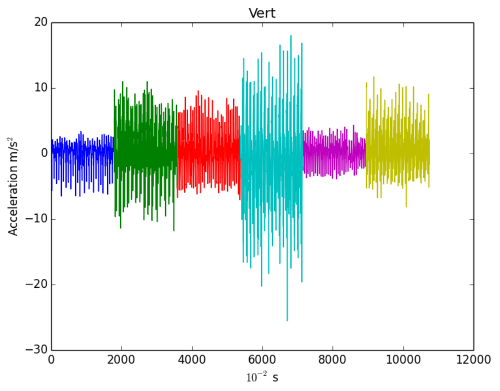
Figure 4.4

As can be seen in Figure 4.4a, 4.4b, 4.5a, 4.5b, 4.6a, 4.6b and 4.7 there is at least one sample series that differs a lot from the other ones, a so called outlier, for every activity. The appearance of these outliers makes us very confident that the trained decision tree is robust.

The series from the pilot study was used as input to the CART-algorithm to create a decision tree. The input consists of a matrix X where the columns represent the values for each feature and a vector Y that denotes the activity corresponding to the values in matrix X, the structure of X and Y can be seen in 4.1 and 4.2. When all features were considered and the algorithm produced a decision tree with no depth boundaries we ended up with an overly complex decision tree, the tree can be seen in Appendix B, Figure B.2.

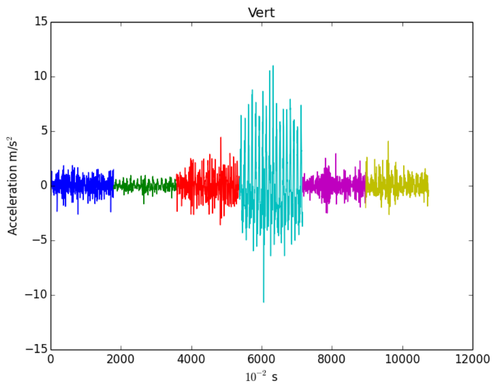


(a) Vertical projection of acceleration data while running.

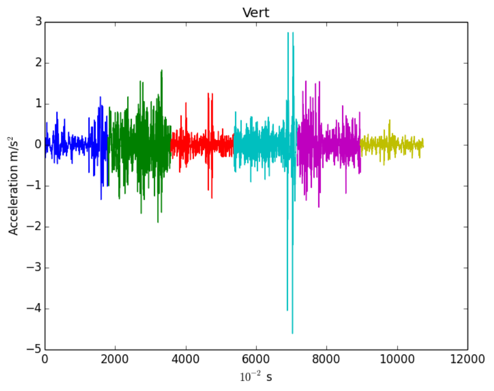


(b) Vertical projection of acceleration data while walking.

Figure 4.5



(a) Vertical projection of acceleration data while cycling.



(b) Vertical projection of acceleration data while travelling by bus.

Figure 4.6

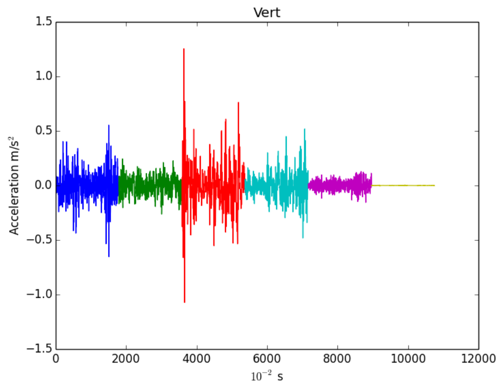


Figure 4.7: Vertical projection of acceleration data while travelling by train.

In order to make the decision tree more manageable and applicable to different data than the pilot-study data, the maximum depth was set to 4. As Table 4.1 shows, the number of features used to determine which activity that corresponds to the data were also reduced when a maximum depth was introduced. The final decision tree can be seen in Appendix B, Figure B.1 and the importance of different features are presented in Table 4.1a and Table 4.1b. Each leaf of the decision tree represents an activity classification group, denoted 0-7 where 0 is considered highest activity and 7 lowest activity. More specifically classes 7-4 is considered as stationary activities and 3-0 as activities where the subject has moved. Due to the Clip capturing an image every 30 seconds, the activity recognition algorithm analyzes 30 seconds of accelerometer-data that is divided in windows of 256 samples each. Each window is fed to the decision tree and given an activity classification of 0-7. If more than half of the windows in the 30 second period get an activity classification of 3 or less(low to high activity), the user is considered to have moved during 30 second period.

Feature	**Utilization (%)**
SMA	54.4
Zero Crossings	1.5
Median	9.0
Range	4.3
Min	6.0
Max	2.2
Var	1.0
Std	0.8
Mean	20.8

(a) Feature utilization when infinite depth is used

Feature	**Utilization (%)**
SMA	75.1
Zero Crossings	0
Median	0
Range	0
Min	1.9
Max	0
Var	0
Std	0
Mean	23

(b) Feature utilization when a maximum depth of 4 is used

Table 4.1: Feature utilization

$$X = \begin{pmatrix} SMA_0 & ZC_0 & Median_0 & Range_0 & Min_0 & Max_0 & Var_0 & Std_0 & Mean_0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ SMA_n & ZC_n & Median_n & Range_n & Min_n & Max_n & Var_n & Std_n & Mean_n \end{pmatrix} \tag{4.1}$$

$$Y = \begin{pmatrix} Activity_0 \\ \vdots \\ Activity_n \end{pmatrix} \tag{4.2}$$

4.3 Evaluation and Discussion

This section evaluates how well the previously proposed activity recognition algorithm can distinguish between different activities when they are performed by a test subject who did not participate in the pilot study. As earlier described, the decision tree has been trained with seven different activities, our expectation is that the final accelerometer algorithm, at a minimum, can distinguish between when the subject is moving (running, walking and biking) or not moving (bus, train, standing and sitting).

The evaluation was carried out in Lund on the 9th of May 2015 while the test subject, a 1.90 m tall man, was equipped with a Sony Xperia Z3 in his chest pocket and the earlier described application, *SimulationActivity*. The test subject was told to walk, run, sit, stand, bike and go by train. After the data were gathered, it was preprocessed and analyzed as described in Appendix A. The results of how the algorithm classified the test data are presented in Figure 4.8.

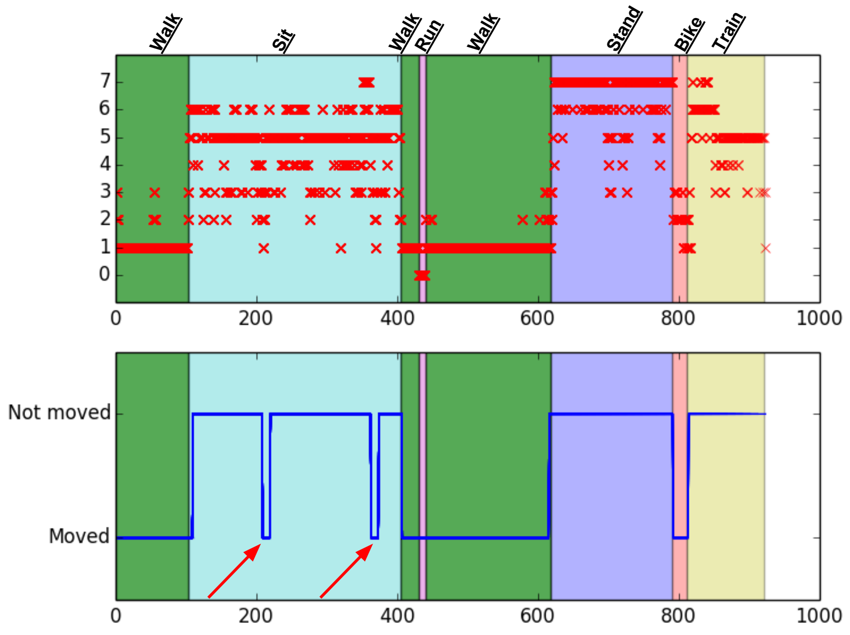


Figure 4.8: The results from our evaluation. Each activity is shown in different colors and labeled above the plot. The two arrows marks false positives.

Our results clearly shows that the accelerometer algorithm performs well. Only two periods of 30 seconds, where the test subject was stationary (sitting), were wrongly classified as if the subject was moving. These false positives were possibly caused by the test-subject making extraneous movements, such as flexing with the arms or reaching for an item. These can also be seen in Figure 4.8

What our results also show is that the algorithm fails to make a clear distinction between if the subject is going by train or standing/ sitting. Even if the evaluation showed a little bit more promise for distinguishing between going by bus and sitting/standing, this classification is also far from certain. Since these two activities just like walking, running and biking are means of movement for a person, an optimal algorithm should be able to capture them as well.

There are a lot of other approaches when coming to classify accelerometer data. We choose to use a simple decision tree based on the CART algorithm and achieved acceptable results for our application. Lu *et al.* compare four different classifier systems (Decision tree, Support Vector Machine, Naive Bayes and Multivariate Gaussian Mode) where the different systems have their own advantages to distinguish between different activities [23]. We think our algorithm would benefit from a similar comparison and ultimately we may be able to distinct different activities better than with our implemented decision tree.

4.4 Summary

In this chapter we have presented a thorough literature study of how to do activity recognition with a accelerometer. Based on the research we proposed an activity recognition algorithm that can classify user movement. Following, this an evaluation was made that shows that the algorithms shows promise, but that it still needs a bit work before it is good.

In the next chapter, Chapter 5, we propose the final algorithm of this thesis. The algorithm that combines both the solution for localization and activity recognition that we have proposed in this chapter and in Chapter 3.

Chapter 5
Final Algorithm

In Chapter 3 we propose a positioning algorithm that utilizes the dense population of Wi-Fi access points in an urban environment to provide a localization algorithm. In this chapter we propose a way to combine this localization algorithm with the activity recognition algorithm we present in Chapter 4 to produce a robust and energy efficient localization solution. This proposed solution is then compared to another solution where position estimates are obtained for every taken picture.

5.1 Proposed Solution

The basic idea is to utilize the fact that people are stationary a big majority of the day to avoid measuring unnecessary positions [23]. An unnecessary position measurement is a measurement for a position that is already known, i.e., when the person using the Clip has not moved since the last position measurement. As a contradiction, a position estimate should of course be obtained when the person has moved, i.e., is not stationary. In Chapter 4 we provide an algorithm that, even though not as pristine so it can identify what type of activity that the person is doing, still is very good at identifying when movement has occurred (travelling by train excluded). The solution makes use of this algorithm as the answer to exactly this problem, to identify if someone has moved or not. When movement is identified a new position measurement is obtained.

Ultimately the localization algorithm should make a sophisticated combination of the estimated position through Wi-Fi localization and the estimated position with GPS. However, the fact that the scope of the thesis does not allow for this combination to be made together with difficulties to synchronize Clip's GPS-measurements with the rest of the simulation has made us settle with a solution that just utilizes the Wi-Fi sensor for position estimation. This approach is good enough for evaluating the energy savings the activity recognition algorithm imposes on the final algorithm.

The final algorithm is thus very simple and elegant, as depicted below and in Figure 5.1.

1. Turn on device
2. Acquire position estimate
   (a) Save the estimated position
3. After 30 seconds - Wake up and take a picture
4. Has the person moved?
   (a) Yes - acquire new position estimate, attach to image and save it
   (b) No - Use the previously estimated position.
5. Repeat step 3-4 until the Clip is turned off or the battery runs out.

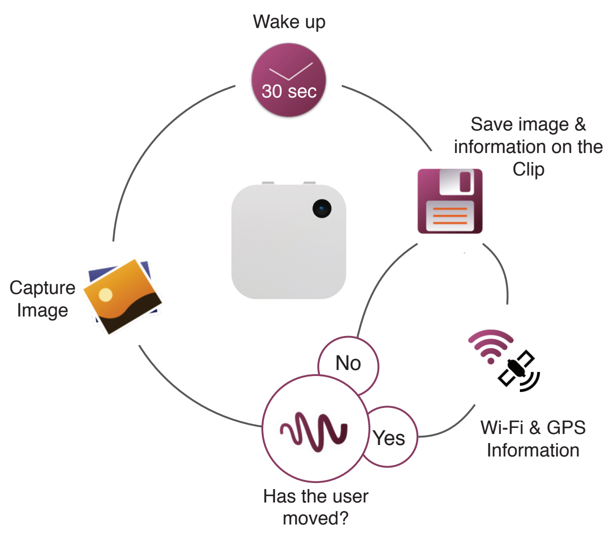


Figure 5.1: Illustration of the process described above.

5.2 Evaluation

To evaluate the proposed solution, a simulation of the Clip was run on the Sony Xperia Z3 as an Android application. A full recollection of the simulation details is accounted for in appendix A, but basically it is just an application that records accelerometer data continuously as well as snaps of nearby Wi-Fi APs and their signal strengths every 30 seconds. The collected data are processed with a Python script on the Macbook Pro. The same Python script was used in Chapter 4 to analyzes the accelerometer data. The activity recognition algorithm analyzes the accelerometer data in between two Wi-Fi snaps and if no movement is detected the latter Wi-Fi snap is deemed unnecessary.

The evaluation took place the 13th of May 2015 in Lund, Sweden during a period of 2 hours around lunch. This was just a normal day of work with a casual pop out to buy lunch. The data were processed by both the localization algorithms - *Skyhook* and multilateration - and compared to positions recorded by the powerful localization algorithm on the OnePlus One (GPS/Wi-Fi and cell combined) with the application *MyTracks*. The positions recorded with *MyTracks* are not the exact pattern of movement, but are very close.

During the session a total of 256 Wi-Fi snaps was taken, 2 every minute. The data from the evaluation session are presented in the Figures 5.2-5.4.

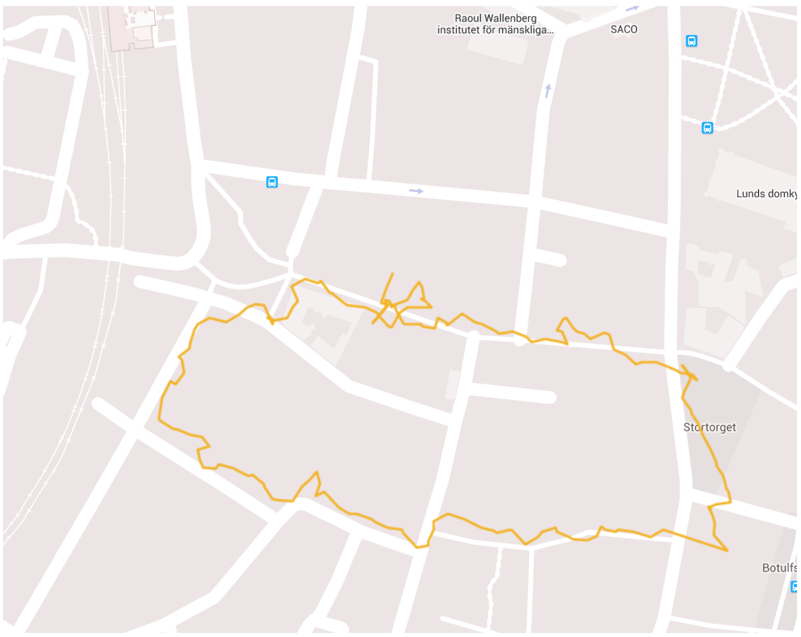


Figure 5.2: The estimated pattern of movement with MyTracks on the OnePlus One.

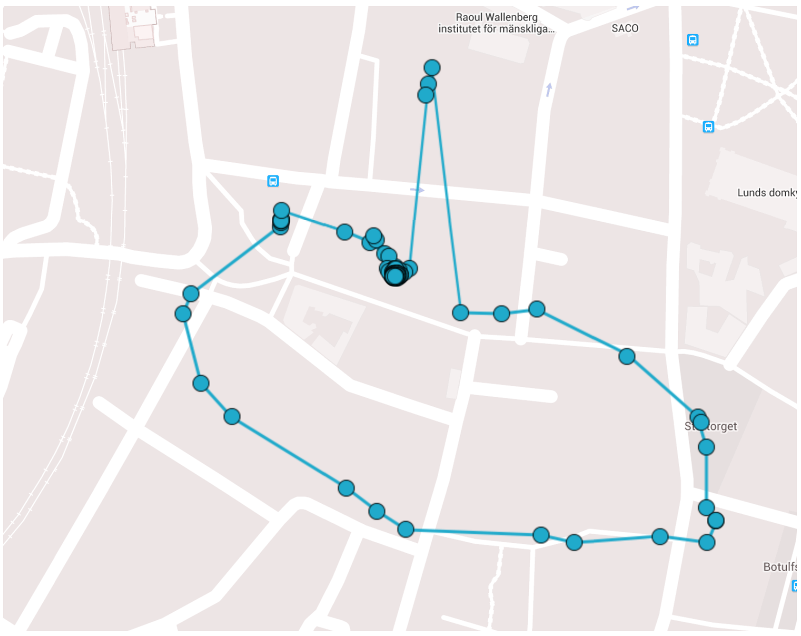


Figure 5.3: The dots are all the 256 positions estimated with *Skyhook's* positioning algorithm. The line is the interpolated path between the 256 position estimates.

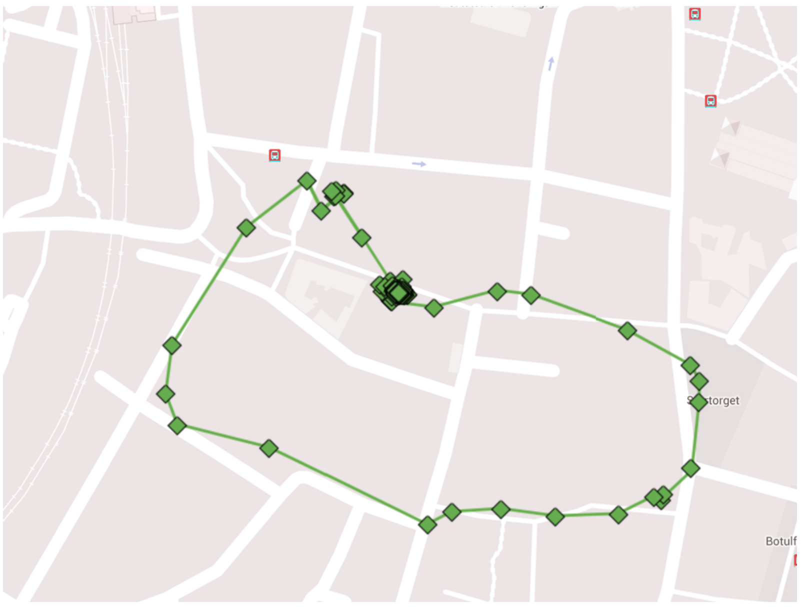


Figure 5.4: The diamonds are all the 256 positions estimated with our proposed localization algorithm. The line is the interpolated path between the 256 position estimates.

When the pattern of movement between the positions estimated by *Skyhook's* localization algorithm, Figure 5.3, and the real path estimated by the OnePlus, Figure 5.2, is compared it is obvious that although the *Skyhook* estimated positions are reasonable, they are still a bit off. The localization algorithm we propose in this thesis, Figure 5.4, gives a better estimate, but is also a little bit off. This is due to the estimation error already explained in Chapter 3. What is more interesting is how many of these 256 measurements that can be avoided if the accelerometer data are added to the equation and run through the activity recognition algorithm. In Figure 5.5 below the activity classification of the accelerometer data is presented. As can be seen it classifies a big majority of the 30-second intervals as stationary (Not moved), meaning that the position estimation is unnecessary.

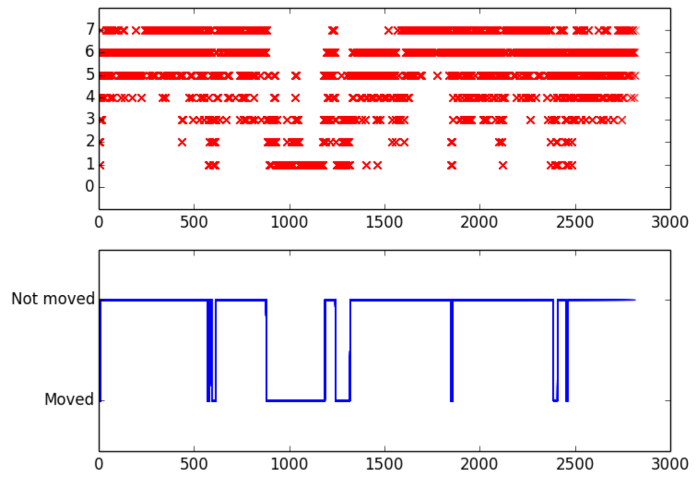


Figure 5.5: Activity classification of the accelerometer data.

If these position estimates are removed we get the pattern presented in Figure 5.6 and 5.7.

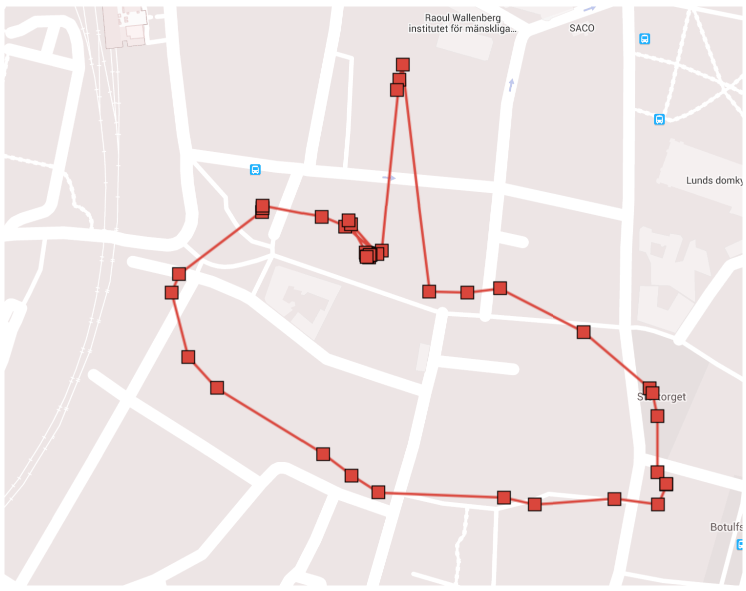


Figure 5.6: Estimated positions with *Skyhook*. 213 unnecessary position estimates removed.

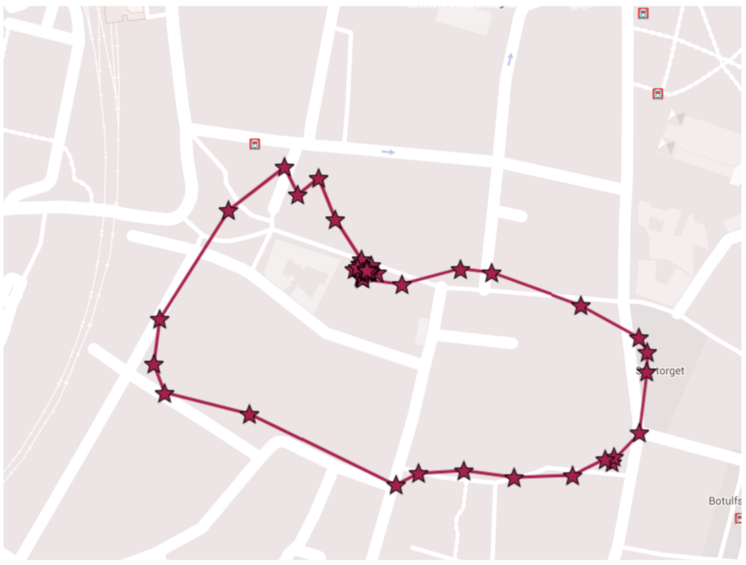


Figure 5.7: Estimated positions with our proposed localization algorithm. 213 unnecessary position estimates removed.

Even though 213 position estimates, 83.2 % of the samples, are removed the interpolated pattern of movement in 5.6 look almost identical to the one presented in Figure 5.3 and the interpolated pattern of movement in 5.7 looks almost identical to the one presented in Figure 5.4. To further demonstrate the resemblance between the plots we present them on top of each other in Figure 5.8 (*Skyhook*) and in Figure 5.9 (Our localization algorithm).

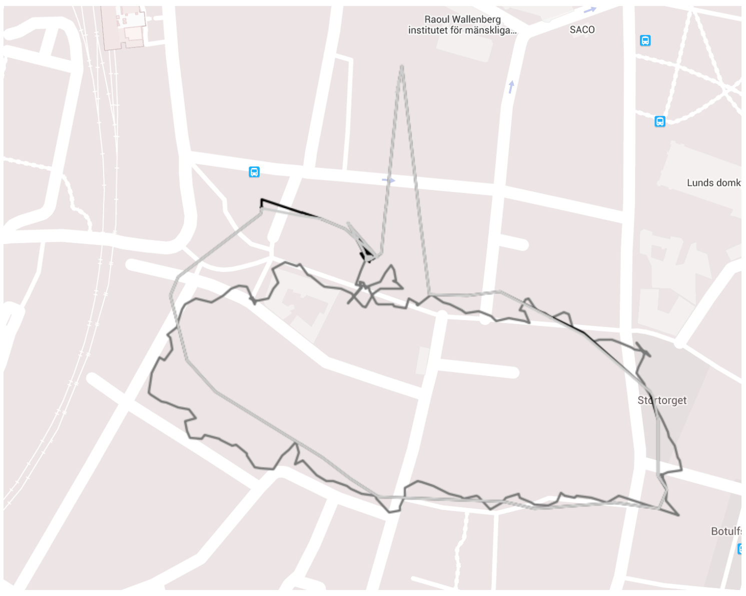


Figure 5.8: Black - All *Skyhook* positions, Light gray - unnecessary measurements removed, dark gray - MyTracks path.

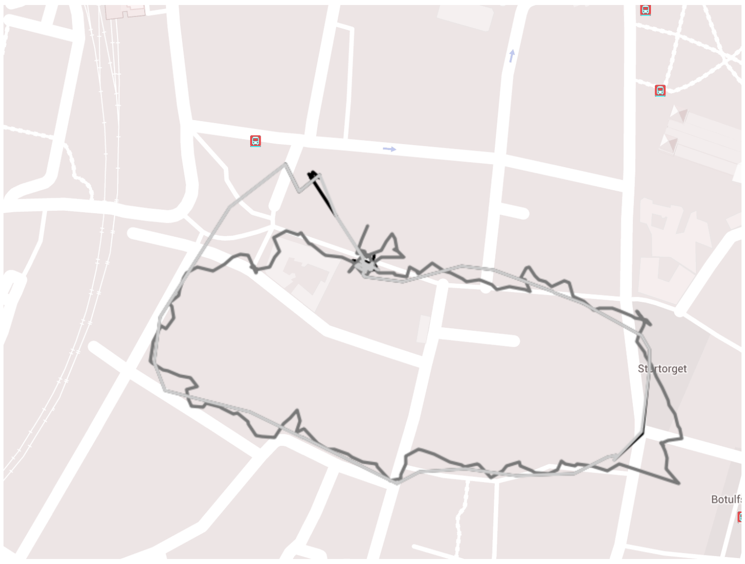


Figure 5.9: Black - All multilaterated positions, Light gray - unnecessary measurements removed, dark gray - MyTracks path.

As can be seen it is just in one place, up to the left, where the pattern of movement differs between when all measurement points are included and not. This is true for both the *Skyhook* estimates as well as for the estimates made by our proposed solution. Furthermore, when zooming in on the location of our office where a lot of the stationary activity was done, it is clear that it is in this area where most of the optimization is made. The difference in this area is presented in Figure 5.10 and in Figure 5.11.

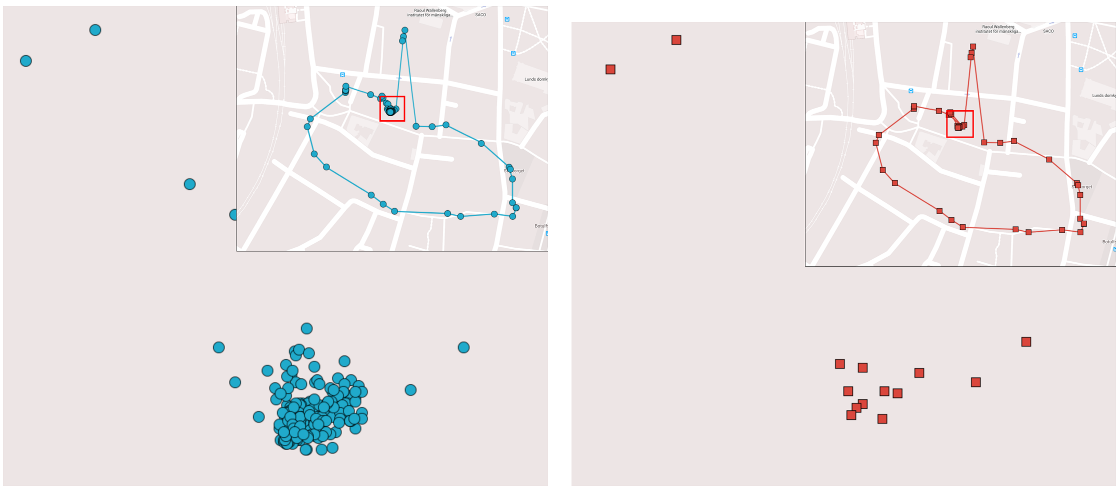


Figure 5.10: Close up at the area of the office where a lot of the stationary activity happened. Positions estimated with *Skyhook's* algorithm.

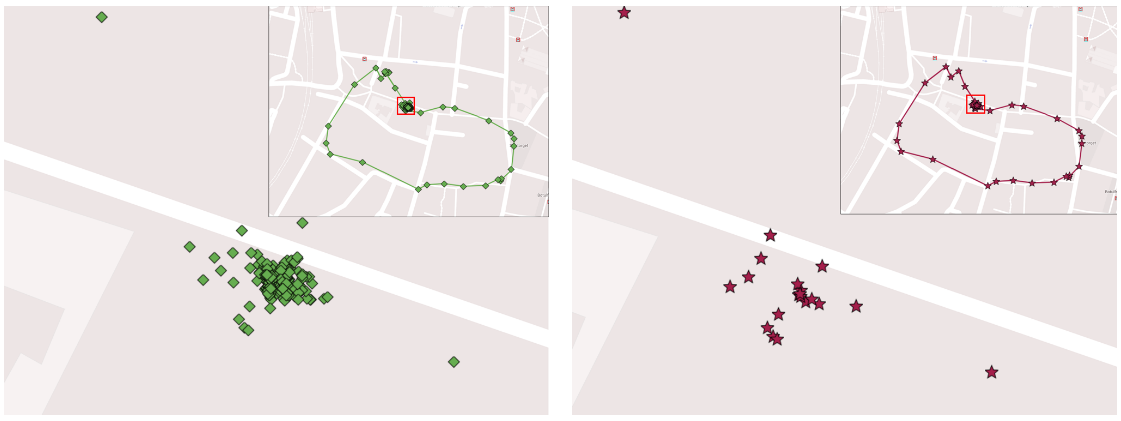


Figure 5.11: Close up at the area of the office where a lot of the stationary activity happened. Positions estimated with our localization algorithm.

Result in numbers:

- 2 hours and 8 minutes of simulation data.
- 256 measurement points with an interval of 30 seconds in between.
- 213 measurement points (83.2 %) that the activity recognition algorithm deemed to be unnecessary.
- 43 measurement points (16.8 %) where the activity recognition algorithm detected that movement had been made and thus that they are necessary.
- The two different patterns of movement, both the one with 256 measurement points and the one with 43 measurement points, are nearly identical.

5.3 Summary

In this chapter we have proposed an algorithm that utilizes the fact that people, nowadays, moves less than 75% of the time to avoid doing unnecessary position estimations. This is done through the combination of an activity recognition algorithm and a localization algorithm. We present the results of an evaluation that shows that the algorithm shows great promise, but also that the activity recognition algorithm needs a bit work before it works and every situation.

In the next chapter, Chapter 6, we provide a discussion of the individual parts of the final algorithm as well as for the thesis as a whole. Furthermore, we present our conclusion of the research we have conducted during this thesis as well as what future research we think can build upon the work we have presented in this report.

Chapter 6
Discussion, Conclusion and Future Research

6.1 Discussion

This chapter consists of a discussion about the two different parts of the solution, Localization and Activity Recognition, as well as a holistic discussion that aims to discuss the solution from their perspective.

6.1.1 Localization

In the evaluation of the localization algorithm, Section 3.4, it is obvious that our solution based on *WiGLE* outperforms the *Skyhook* implementation with a mean-error of just 27 m compared to *Skyhook* 80 m. As previously mentioned this number is probably not representative for all the world, since we ourselves have been wardriving a lot in Lund. Thus the *WiGLE* database is most certainly better updated there than in a lot of other places and the results should be looked upon more like something that could be with a good database, rather than something that is true in every situation. Since *WiGLE* is an open-source database that anybody with a smartphone can contribute to, it has a lot of potential to be well-populated as well as well-updated and this at a very low cost. This setup does however raise some questions about the certainty of population and robustness of the database. When somebody is wardriving for *WiGLE* that person has the power to decide if her results can be used for commercial use or not. These two facts together with the *WiGLE* API for code-induced queries being very limited puts a big question mark on how feasible *WiGLE* is for commercial use. Although *WiGLE* succeeds in providing good enough results to prove the concept of this thesis, it is most certainly better to look around for another more robust database if the multilateration approach is to be used in a commercial setting.

Even though the evaluation in Section 3.4 shows a much larger mean-error for the *Skyhook* implementation than for the multilateration approach it could be argued that this solution is still better for large-scale commercial use. They have a database, which they have full control over and that is well-populated in most urban areas in the world. Moreover, Apple was a big customer of their location services in the first generations of the iPhone, which proves that the solution works at a very large-scale. Furthermore, *Skyhook* holds a long list of patents related to geo-location which implicates a position as a pioneer in the field for over 10 years [19].

6.1.2 Activity Recognition

The proposed activity recognition algorithm performs well and succeeds in recognizing dynamic activities, in other words running, walking and cycling. However, the evaluation shows that it falls short in other areas. One area in which it fails is in recognizing stationary activities where the user is moving, such as motorised transport. Another area in which it fails is in making a clear distinction between the different activities. It rather succeeds in doing the minimum, making the broader distinction between dynamic and stationary activity.

It can be argued why the algorithm fails in these areas. For example we only include time-domain metrics in the algorithm to make computations less complex. Many of the other algorithms that were included in our research make use of frequency and discrete-domain metrics as well. Of course the inclusion of more different features would make it possible to say more about the recorded data and it is possible that this inclusion would solve a few of these areas where the algorithm fails at the moment.

Another shortcoming of the algorithm is that it only includes the vertical projection of the accelerometer data in its analysis. This was chosen because it says more about the movement than the horizontal projection. Although less important, the horizontal projection of the data still doubles the information available for analysis for every time-window. This would possibly solve a few problems that the algorithm faces now.

A third area where improvement can be made is in the estimation of the gravity component of the acceleration in the different direction, x, y and z. As previously mentioned the gravity estimation that is done today does not take slow acceleration and deceleration into account and thus makes it impossible for the algorithm to identify for example the acceleration when a train leaves a station. However it should be noted that for this information to be interesting a much longer signal of data than 30 seconds has to be analyzed and the algorithm thus has to be modified a lot to take this information into account.

This discussion might seem loaded with criticism against our own solution, but then one has to put in mind that all of these decisions were made for a reason. Either they were made because a smaller computational complexity in the Clip was strived for or because the research needed to include such a solution simply would not fit within the scope of the thesis.

6.1.3 Holistic Discussion

This section aims to provide a discussion for the big, holistic questions of the project such as if we provide an answer to our problem definition, if the results can be trusted and how the solution stands according to different criteria presented in Section 2.1 .

Significance of Results

As already mentioned in Section 1.5, the thesis should be more viewed as a proof of concept than a fully tested implementation that gives the found results in every single case. Above everything, there are two different areas that make our results to be more of an indication than results that are fully reliable.

First of all, the implementation is a simulation. Even though the Android applications are constructed to simulate the Clip as closely as possible, it is still a totally different hardware. A solution like this should thus be implemented on the Clip and then thoroughly tested before the results can be validated.

Secondly, the data-sets for both the construction of the algorithms and the validation of the same are small. This means that the results can't be given the statistical significance needed to validate the solution. To do this, much bigger data-sets would have to be used and should be when a implementation on the Clip is validated. This is especially true for validating the activity recognition algorithm since different body sizes, movement patterns and behaviours play into how the acceleration data looks.

With this said, both these shortcuts had to be made for us to fit the whole algorithm within the scope of the thesis and we are confident in the thesis as a means to prove the concept of a localization algorithm where position measurements are periodically induced by an activity recognition algorithm.

Problem Definition

"How can periodically gathered data from the Wi-Fi, accelerometer and GPS sensors be combined to produce an accurate and energy efficient localization algorithm for a wearable device?". How well we succeed in providing an answer to this question can be argued. Sure the proposed solution use the Wi-Fi, accelerometer and GPS-sensors to provide an, for the application, accurate and energy efficient algorithm, so in that sense we provide a thorough answer for the problem definition. However, the question could also be answered in a much broader way, where different approaches and algorithms are compared and evaluated. Unfortunately the scope of the thesis has not left us with enough time and resources for this kind of analysis and that broad view of the question thus goes unanswered.

Although the evaluation of the proposed solution shows that it is energy-efficient and a large majority of the position measurements can be avoided, it is more ambiguous how energy-efficient it actually is. Since all parts of the algorithm are simulated it is hard to say how much energy a solution like this would save the Clip since there are a lot of variables playing into this. A solution using the Clip and the Company servers would have to be implemented and tested with a big dataset to give a thorough answer to this question.

Location System Parameters

In Section 2.1 we present 5 criteria that can be used to evaluate how good a locations system is. In the following we discuss our proposed localization algorithm in respect to these criteria.

Accuracy The mean-error of our localization algorithm is 27 m for multilateration and 80 m for *Skyhook*. We have constantly written that this error magnitude is pretty good, but how good is it?

When making a location system the first thing one should think of is what applications it will have. Obviously a precision of 27-80m is not good enough for a navigation system in a driver-less car or for tracking how far someone has run during exercise. However, the Clip just needs a location system that can tell in which area a picture has been taken. Our proposed solution clearly succeeds with this task and can thus be seen as a success in terms of accuracy. Although successful in this matter, it should be noted that the applications for the proposed location system are limited by the accuracy of the estimates.

Coverage A combination of location estimates based on Wi-Fi and GPS has a very good coverage. In urban areas where GPS falls short of providing a position estimate, a Wi-Fi based approach is very good. On the other hand GPS is very good out in the nature or at the sea where there aren't a lot of things to obscure the GPS-sensor from the satellites. Of course it is hard to tell exactly how good the coverage of our proposed solution is and a thorough study would have to be made just for that. What is certain is that the coverage is much better than an approach that just relies on GPS (23/24 measure points with position estimates compared to 8/24 with only GPS).

Privacy In a location system with very good privacy the position should be estimated on the device itself. In this way the location information does not need to be sent anywhere and is thus kept private for the user. It is in terms of privacy, our proposed solution falls short the most. Both the GPS- and Wi-Fi- localization solutions have an infrastructure where the device is just used as a means for gathering data. The data are then sent to servers that handle all the position estimation themselves. This means that the data are not kept private for the user and that she thus has to rely on the infrastructure and the system operators to not use the data for any malicious purposes. Obviously this is not optimal, but the approach enables the Clip to be offline and thus saves a lot of energy.

Infrastructure Cost Since the proposed system is based on simulations and relies on other solutions such as *Skyhook* and *WiGLE* it is hard to say what the final infrastructure cost for an implemented solution would be.

Per-client Cost Just as for the infrastructure cost it is hard to say what the per-client cost for an implemented solution would be.

6.2 Conclusion

The scope of the thesis has been very broad, which has forced us to make a lot of prioritizations along the way and we have not had the time to deep dive into every area as much as desired. Even with these prioritizations the proposed solution succeeds in providing an answer to the problem definition. Also when considered from a wider point of view, as a location system as a whole, it holds up in terms of coverage and accuracy. The user privacy of the system is however not the best, but necessary if the energy consumption is to be kept low.

The proposed location system does however leave a few areas where improvement can be made. First and foremost, as explained in the discussion, the activity recognition algorithm can be enhanced by analyzing more metrics, optimizing the decision tree as well as by including the horizontal projection vector of the acceleration data in the calculations and improving the gravity estimation. Doing this would certainly make the activity recognition more reliable at the same time as it can produce more detailed data that can be analyzed by the Company's backend servers.

The localization algorithm performs very well and the question there is first and foremost which database for Wi-Fi positioning that should be used. If the multilateration approach is chosen, the enhancements that can be made is to find a better way to model distance from the RSSI property of received Wi-Fi signals. This will improve the accuracy of the location estimates even more.

Finally we think that solutions with compromises and combinations of sensors, like the ones we propose in this thesis, are necessary if wearables are to live up to all the high expectations that are put on them today.

6.3 Future Research

Even though we succeed in providing a solution for the defined problem definition there are still improvements to be made.

First and foremost the activity recognition algorithm can be further enhanced. A solution where more features, such as the frequency spectrum features and the horizontal projection features, are included in the analysis are to be implemented. Following this, the activity classification decision tree can be trained with accelerometer data from a large data-set including data from subjects of all different ages and body compositions. These two actions has the possibility to make the algorithm more accurate and more robust at the same time.

As of now this thesis is more of a proof of concept study than a solution that works in every situation. To prove that it is more than that, the evaluations made during this thesis project have to be re-done with much bigger data-sets. For example the localization algorithm should be tested in a wide range of different places on the globe and the activity recognition algorithm should be evaluated with data from a wide range of different subjects, with different body compositions, nationalities and ages.

The thesis fails in providing data on how big the actual energy savings would be if an implementation like this are made for the Clip. To show this, a study that compares the energy consumption between acquiring a GPS/Wi-Fi location and running the accelerometer continuously can be conducted. Since this is a very central part of the solution we suggest a study like that to be conducted before an actual implementation is made.

During the research of this thesis we have come across interesting solutions that uses other sensors, not included in this thesis, for similar applications. For example Zhou *et al.* 2012 includes light intensity from a light sensor and a magnetic measurement from a magnetometer in a solution that can tell if the user is residing indoors or outdoors [39]. Lu *et al.* uses the ambient sound from a microphone and Johan Sjöberg does RGB analysis on the Clip's pictures themselves to do the same thing [23][33]. Since the mean-error of the localization algorithm is 27 m, information like this can provide additional valuable information of where the user actually is located.

Finally we would like to see an actual implementation of the solution we have proposed on the Clip and Company servers themselves. This is necessary to eliminate differences caused by hardware and would facilitate some of the research mentioned above.

Bibliography

[1] Erik K Antonsson and Robert W Mann. The frequency content of gait. *Journal of biomechanics*, 18(1):39–47, 1985.

[2] Apple. `https://www.apple.com/watch/apple-watch-edition/18-karat-yellow-gold-case-black-classic-buckle/`, 2015. [Online; accessed 27-April-2015].

[3] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. *Pervasive computing*, pages 1–17, 2004.

[4] Ezekiel S Bhasker, Steven W Brown, and William G Griswold. Employing user feedback for fast, accurate, low-maintenance geolocationing. In *Pervasive Computing and Communications, 2004. PerCom 2004. Proceedings of the Second IEEE Annual Conference on*, pages 111–120. IEEE, 2004.

[5] Li Bin, Dou Zheng, Ning Yu, and Lin Yun. An improved weighted centroid localization algorithm. *International Journal of Future Generation Communication and Networking*, 6(5):45–52, 2013.

[6] Aaron Carroll and Gernot Heiser. An analysis of power consumption in a smartphone. In *USENIX annual technical conference*, pages 1–14, 2010.

[7] Eddie CL Chan, George Baciu, and SC Mak. Wi-fi positioning based on fourier descriptors. In *Communications and Mobile Computing (CMC), 2010 International Conference on*, volume 3, pages 545–551. IEEE, 2010.

[8] Yih-Shyh Chiou, Chin-Liang Wang, Sheng-Cheng Yeh, and Ming-Yang Su. Design of an adaptive positioning system based on wifi radio signals. *Computer Communications*, 32(7):1245–1254, 2009.

[9] PYM Chung and GYF Ng. Comparison between an accelerometer and a three-dimensional motion analysis system for the detection of movement. *Physiotherapy*, 98(3):256–259, 2012.

[10] Byron Cook, Graham Buckberry, Ian Scowcroft, John Mitchell, and Tony Allen. Indoor location using trilateration characteristics. In *Proc. London Communications Symposium*, pages 147–150, 2005.

[11] Jonny Farringdon, Andrew J Moore, Nancy Tilbury, James Church, and Pieter D Biemond. Wearable sensor badge and sensor jacket for context awareness. In *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*, pages 107–113. IEEE, 1999.

[12] Davide Figo, Pedro C Diniz, Diogo R Ferreira, and João MP Cardoso. Preprocessing techniques for context recognition from accelerometer data. *Personal and Ubiquitous Computing*, 14(7):645–662, 2010.

[13] Fitbit. `https://www.fitbit.com/uk/charge`, 2015. [Online; accessed 27-April-2015].

[14] Emma Fortune, Vipul Lugade, Melissa Morrow, and Kenton Kaufman. Validity of using tri-axial accelerometers to measure human movement–part ii: Step counts at a wide range of gait velocities. *Medical engineering & physics*, 36(6):659–669, 2014.

[15] Sinan Gezici. A survey on wireless position estimation. *Wireless personal communications*, 44(3):263–282, 2008.

[16] A Godfrey, AK Bourke, GM Olaighin, P Van De Ven, and J Nelson. Activity classification using a single chest mounted tri-axial accelerometer. *Medical engineering & physics*, 33(9):1127–1135, 2011.

[17] Jennifer Healey and Beth Logan. Wearable wellness monitoring using ecg and accelerometer data. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on*, pages 220–221. IEEE, 2005.

[18] Samuli Hemminki, Petteri Nurmi, and Sasu Tarkoma. Accelerometer-based transportation mode detection on smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, page 13. ACM, 2013.

[19] Skyhook Wireless Inc. `http://www.skyhookwireless.com/patents`, 2015. [Online; accessed 20-May-2015].

[20] Dean M Karantonis, Michael R Narayanan, Merryn Mathie, Nigel H Lovell, and Branko G Celler. Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):156–167, 2006.

[21] Anthony LaMarca and Eyal De Lara. Location systems: An introduction to the technology behind location awareness. *Synthesis Lectures on Mobile and Pervasive Computing*, 3(1):1–122, 2008.

[22] Jaime Lloret, Jesus Tomas, Alejandro Canovas, and Irene Bellver. Geowifi: A geopositioning system based on wifi networks. In *The Seventh International Conference on Networking and Services, ICNS*, pages 38–43, 2011.

[23] Hong Lu, Jun Yang, Zhigang Liu, Nicholas D Lane, Tanzeem Choudhury, and Andrew T Campbell. The jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 71–84. ACM, 2010.

[24] Vipul Lugade, Emma Fortune, Melissa Morrow, and Kenton Kaufman. Validity of using tri-axial accelerometers to measure human movement—part i: Posture and movement detection. *Medical engineering & physics*, 36(2):169–176, 2014.

[25] Gough Lui, Thomas Gallagher, Binghao Li, Andrew G Dempster, and Chris Rizos. Differences in rssi readings made by different wi-fi chipsets: A limitation of wlan localization. In *Localization and GNSS (ICL-GNSS), 2011 International Conference on*, pages 53–57. IEEE, 2011.

[26] MJ Mathie, Branko G Celler, Nigel H Lovell, and ACF Coster. Classification of basic daily movements using a triaxial accelerometer. *Medical and Biological Engineering and Computing*, 42(5):679–687, 2004.

[27] Ruth E Mayagoitia, Anand V Nene, and Peter H Veltink. Accelerometer and rate gyroscope measurement of kinematics: an inexpensive alternative to optical motion analysis systems. *Journal of biomechanics*, 35(4):537–542, 2002.

[28] David Mizell. Using gravity to estimate accelerometer orientation. In *2012 16th International Symposium on Wearable Computers*, pages 252–252. IEEE Computer Society, 2003.

[29] Esmond Mok and Günther Retscher. Location determination using wifi fingerprinting versus wifi trilateration. *Journal of Location Based Services*, 1(2):145–159, 2007.

[30] Ernesto Navarro-Alvarez, Mario Siller, and Kyle O'Keefe. Gps-assisted path loss exponent estimation for positioning in ieee 802.11 networks. *International Journal of Distributed Sensor Networks*, 2013, 2013.

[31] Scikit-learn. `http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier.feature_importances_`, 2014. [Online; accessed 19-April-2015].

[32] Daniel Siewiorek, Asim Smailagic, Junichi Furukawa, Andreas Krause, Neema Moraveji, Kathryn Reiger, Jeremy Shaffer, and Fei Lung Wong. Sensay: A context-aware mobile phone. In *2012 16th International Symposium on Wearable Computers*, pages 248–248. IEEE Computer Society, 2003.

[33] Johan Sjöberg. Segmentation of image sequence into scene-coherent parts. Master's thesis, LTH, 2015.

[34] Lin Sun, Daqing Zhang, Bin Li, Bin Guo, and Shijian Li. Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations. *Ubiquitous intelligence and computing*, pages 548–562, 2010.

[35] Nils Ole Tippenhauer, Kasper Bonne Rasmussen, Christina Pöpper, Srdjan Capkun, Srdjan Capkun, and Srdjan Capkun. iphone and ipod location spoofing: Attacks on public wlan-based positioning systems. 2009.

[36] Prof. Murat Torlak. `http://www.lcs.poli.usp.br/~ablima/grad/tfm/slides/Torlak/lectureradio.pdf`, 2015. [Online; accessed 27-April-2015].

[37] Wikipedia. `http://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees`, 2015. [Online; accessed 2-April-2015].

[38] Paul A Zandbergen. Comparison of wifi positioning on two mobile devices. *Journal of Location Based Services*, 6(1):35–50, 2012.

[39] Pengfei Zhou, Yuanqing Zheng, Zhenjiang Li, Mo Li, and Guobin Shen. Iodetector: A generic service for indoor outdoor detection. In *Proceedings of the 10th acm conference on embedded network sensor systems*, pages 113–126. ACM, 2012.

Appendices

Appendix A

Simulation Setup

To save time in the work with the thesis we have chosen to simulate both how the server and the Clip behave in different situation. The server has been simulated with different Python scripts on a Macbook and the Clip has been simulated by Android application on two Android phones, a Sony Xperia Z3 and a Oneplus One.

A.1 Devices

A.1.1 Macbook Pro Retina 15"

Figure A.1: Macbook Pro Retina 15" mid 2014

Model name	Macbook Pro
Model Identifier	MacbookPro11,2
Processor Name	Intel Core i7
Processor Speed	2,2 GHz
Number of Processors	1
Total Number of Cores	4
L2 Cache (per Core)	256 KB
L3 Cache	6 MB
Memory	16 GB
System Version	OS X 10.10.3
Kernel Version	Darwin 14.3.0

A.1.2 Oneplus One

Figure A.2: OnePlus One

Operating System:	Cyanogen 11S based on Android 4.4
CPU:	Qualcomm Snapdragon 801, 2.5 GHz Quad-core
GPU:	Adreno 330, 578 MHz
Storage:	64 GB eMMC 5.0
Sensors:	accelerometer, Gyroscope, Proximity and Ambient Light
Connectivity:	GSM: 850, 900, 1800, 1900 MHz
	WCDMA: Bands: 1/2/4/5/8
	LTE: Bands: 1/3/4/7/17/38/40
Wi-Fi:	Dual-band Wi-Fi (2.4G/5G) 802.11 b/g/n/ac
Positioning:	Internal GPS antenna + GLONASS, digital compass

A.1.3 Sony Xperia Z3

Figure A.3: Sony Xperia Z3

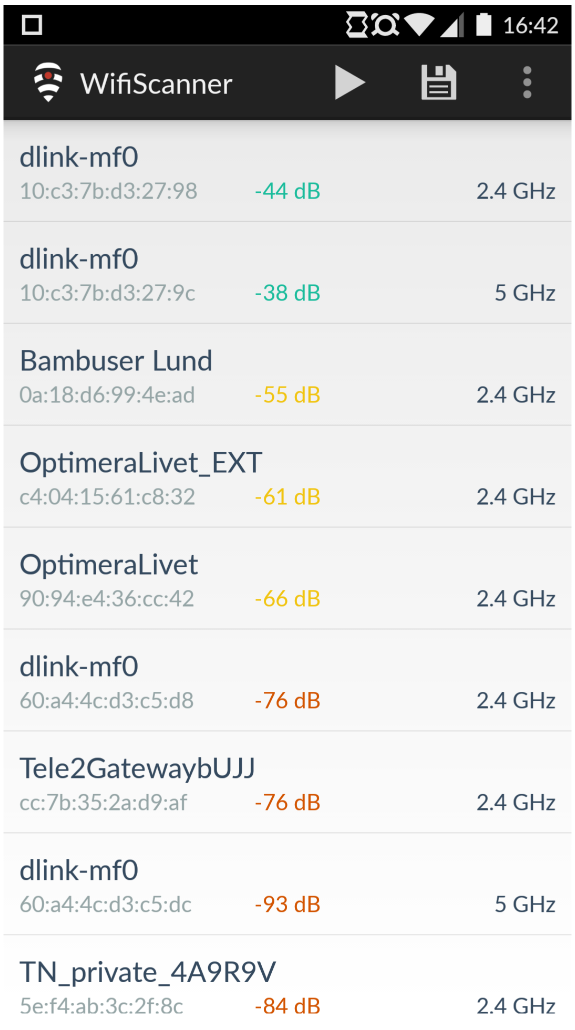
Operating System	Cyanogen 11S based on Android 4.4
Processor	Qualcomm Snapdragon 801, 2.5 GHz Quad-core
GPU	Adreno 330
Memory	3 GB RAM
Storage	16 GB + 128 GB microSD slot
Connectivity	(2G) GSM/GPRS/EDGE
	(3G) UMTS/HSPA
	(4G) LTE
Wi-Fi	Dual-band (2,4G/5G)
Positioning	Assisted GPS + GLONASS
Sensors	accelerometer, Gyroscope, Proximity and Ambient Light

A.2 Simulation

A.2.1 Wi-Fi simulation

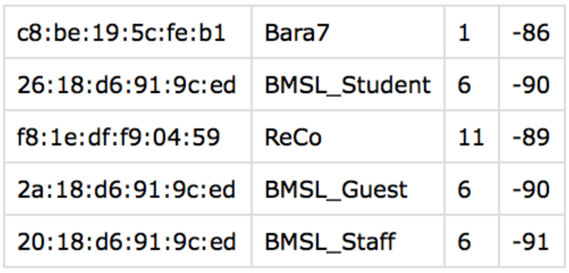
Data gathering

The data gathering during the Wi-Fi-evaluation was done with both the Oneplus One and Sony Xperia Z3. An Android application named WiFiScanner was developed to enable the gathering of data. The functionality of the application is very simple. It consists of a simple UI, depicted in Figure A.4 that enables one to see what Wi-Fi APs that are nearby and what signal strength is received from each and every one of them. This set of APs and signal strengths can then be saved to a textfile that the server simulation can process. A screenshot of such a snap is presented in Figure A.5.



dlink-mf0	10:c3:7b:d3:27:98	-44 dB	2.4 GHz
dlink-mf0	10:c3:7b:d3:27:9c	-38 dB	5 GHz
Bambuser Lund	0a:18:d6:99:4e:ad	-55 dB	2.4 GHz
OptimeraLivet_EXT	c4:04:15:61:c8:32	-61 dB	2.4 GHz
OptimeraLivet	90:94:e4:36:cc:42	-66 dB	2.4 GHz
dlink-mf0	60:a4:4c:d3:c5:d8	-76 dB	2.4 GHz
Tele2GatewaybUJJ	cc:7b:35:2a:d9:af	-76 dB	2.4 GHz
dlink-mf0	60:a4:4c:d3:c5:dc	-93 dB	5 GHz
TN_private_4A9R9V	5e:f4:ab:3c:2f:8c	-84 dB	2.4 GHz

Figure A.4: A screenshot of WifiScanner.



c8:be:19:5c:fe:b1	Bara7	1	-86
26:18:d6:91:9c:ed	BMSL_Student	6	-90
f8:1e:df:f9:04:59	ReCo	11	-89
2a:18:d6:91:9c:ed	BMSL_Guest	6	-90
20:18:d6:91:9c:ed	BMSL_Staff	6	-91

Figure A.5: A screenshot of a snap with APs and signal strengths. The column order is MAC-address - SSID - Channel - Power(dBm)

The full sourcecode for the application can be found at GitHub/WifiScanner

Server program

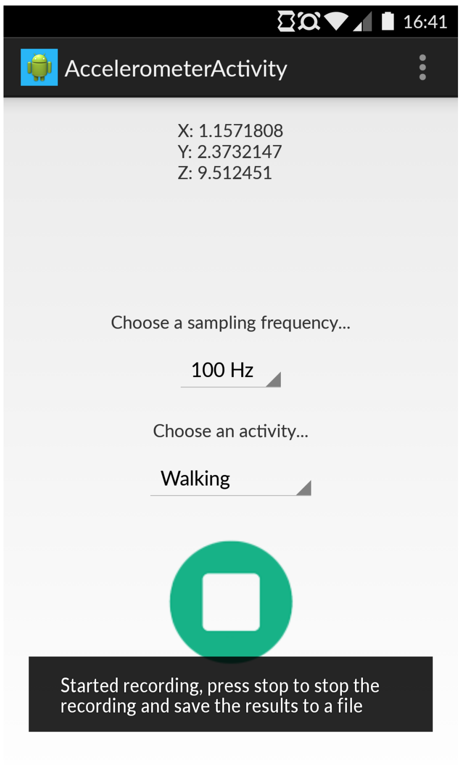
The server simulation consists of a Python script that is runs on the Macbook. The script takes the files with APs and signal strengths as input. For every AP in the file, *WiGLE* is queried to get the geo-location of that AP. These locations are saved in a local database and are then used to make a position estimation through multilateration for every set of APs with signal strengths, the same files that were recorded with the WifiScanner app. The server simulation also queries *Skyhook* with these snapshots/fingerprints to get an estimated location from them as well.

A.2.2 Accelerometer simulation

Data gathering

Just as with the Wi-Fi simulation the data gathering was simulated with an Android application. However, only the Sony Xperia Z3 was used for the data gathering. The reason for this is that accelerometers are such fine-grained sensors that different manufacturers and devices may result in slightly different measurements. To make the data gathering as close as possible to the Clip it was necessary to use the same device for all the gathering of accelerometer data.

The application that was developed for this purpose is AccelerometerActivity. It is a simple application with a simple UI, depicted in Figure A.6. When using it, one chooses the sampling ratio and a label that the data record should be marked with. After pressing play the application records data until stopped whilst it automatically saves the data to a textfile. A textfile with data from the application is presented in Figure A.7.



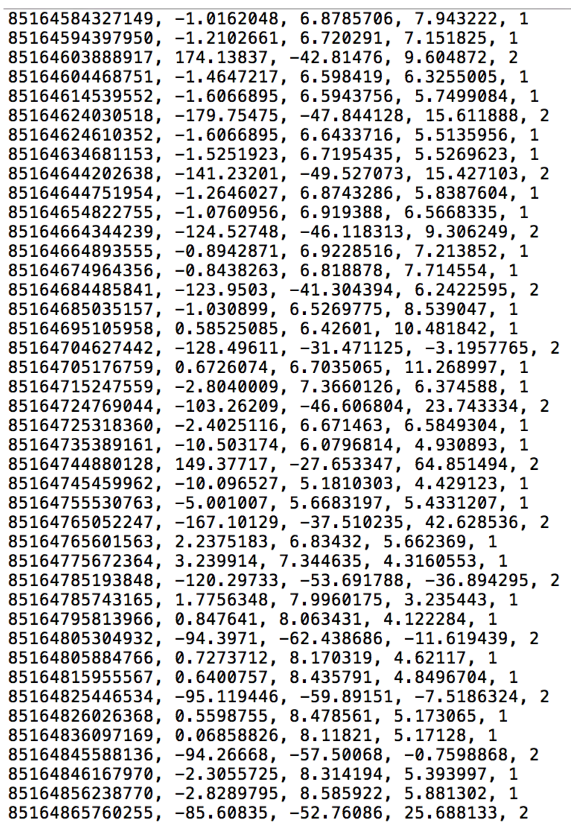
X: 1.1571808
Y: 2.3732147
Z: 9.512451

Choose a sampling frequency...
100 Hz

Choose an activity...
Walking

Started recording, press stop to stop the recording and save the results to a file

Figure A.6: A screenshot of AccelerometerAnalyzer.



```
85164584327149, -1.0162048, 6.8785706, 7.943222, 1
85164594397950, -1.2102661, 6.720291, 7.151825, 1
85164603888917, 174.13837, -42.81476, 9.604872, 2
85164604468751, -1.4647217, 6.598419, 6.3255005, 1
85164614539552, -1.6066895, 6.5943756, 5.7499084, 1
85164624030518, -179.75475, -47.844128, 15.611888, 2
85164624610352, -1.6066895, 6.6433716, 5.5135956, 1
85164634681153, -1.5251923, 6.7195435, 5.5269623, 1
85164644202638, -141.23201, -49.527073, 15.427103, 2
85164644751954, -1.2646027, 6.8743286, 5.8387604, 1
85164654822755, -1.0760956, 6.919388, 6.5668335, 1
85164664344239, -124.52748, -46.118313, 9.306249, 2
85164664893555, -0.8942871, 6.9228516, 7.213852, 1
85164674964356, -0.8438263, 6.818878, 7.714554, 1
85164684485841, -123.9503, -41.304394, 6.2422595, 2
85164685035157, -1.030899, 6.5269775, 8.539047, 1
85164695105958, 0.58525085, 6.42601, 10.481842, 1
85164704627442, -128.49611, -31.471125, -3.1957765, 2
85164705176759, 0.6726074, 6.7035065, 11.268997, 1
85164715247559, -2.8040009, 7.3660126, 6.374588, 1
85164724769044, -103.26209, -46.606804, 23.743334, 2
85164725318360, -2.4025116, 6.671463, 6.5849304, 1
85164735389161, -10.503174, 6.0796814, 4.930893, 1
85164744880128, 149.37717, -27.653347, 64.851494, 2
85164745459962, -10.096527, 5.1810303, 4.429123, 1
85164755530763, -5.001007, 5.6683197, 5.4331207, 1
85164765052247, -167.10129, -37.510235, 42.628536, 2
85164765601563, 2.2375183, 6.83432, 5.662369, 1
85164775672364, 3.239914, 7.344635, 4.3160553, 1
85164785193848, -120.29733, -53.691788, -36.894295, 2
85164785743165, 1.7756348, 7.9960175, 3.235443, 1
85164795813966, 0.847641, 8.063431, 4.122284, 1
85164805304932, -94.3971, -62.438686, -11.619439, 2
85164805884766, 0.7273712, 8.170319, 4.62117, 1
85164815955567, 0.6400757, 8.435791, 4.8496704, 1
85164825446534, -95.119446, -59.89151, -7.5186324, 2
85164826026368, 0.5598755, 8.478561, 5.173065, 1
85164836097169, 0.06858826, 8.11821, 5.17128, 1
85164845588136, -94.26668, -57.50068, -0.7598868, 2
85164846167970, -2.3055725, 8.314194, 5.393997, 1
85164856238770, -2.8289795, 8.585922, 5.881302, 1
85164865760255, -85.60835, -52.76086, 25.688133, 2
```

Figure A.7: Data from AccelerometerAnalyzer. Depending on if the data comes from the accelerometer or gyrometer, the data are structured in the following order: Accelerometer - Timestamp(nanoseconds from device startup), X-axis value(m/s^2), Y(m/s^2), Z(m/s^2), 1(id for accelerometer). Gyrometer - Timestamp, Roll(degrees), Pith(degrees), Yaw(degrees), 2(id for gyrometer)

The full sourcecode for the application is found at GitHub/AccelerometerAnalyzer.

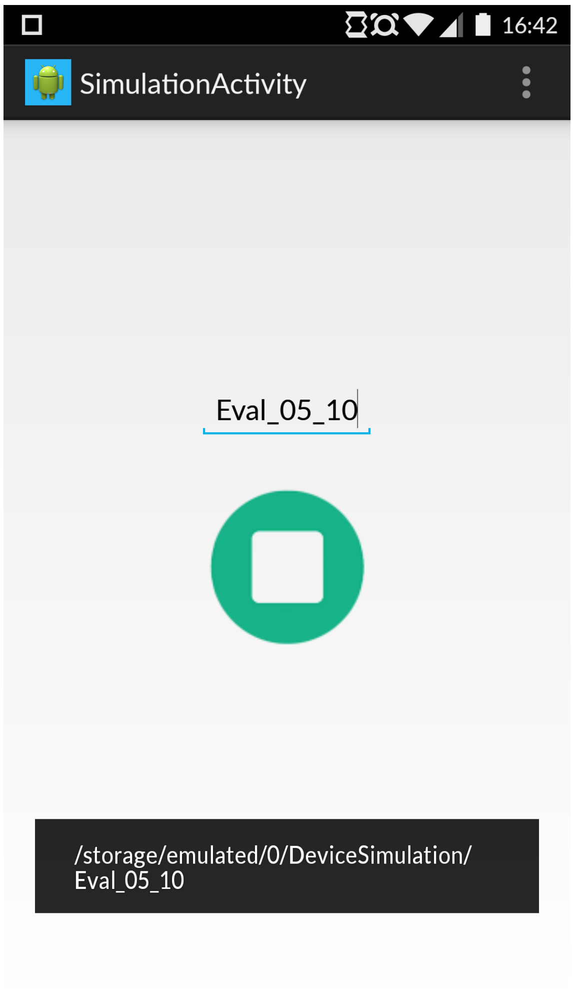
Server program

Just as for the Wi-Fi simulation the server simulation consists of a Python program. The program does all the steps for processing accelerometer data described in Chapter 3.

A.2.3 Simulation for final evaluation

Data gathering

For the final evaluation yet another Android application was developed. The goal of this application is to simulate the process of waking up every 30 seconds to record a Wi-Fi-snapshot as well as to record accelerometer data in between these snapshots. To do this parts of the two applications AccelerometerActivity and WifiScanner were merged into one application. An event-timer to trigger Wi-Fi-measurements and attach accelerometer data to them was also added. The files with data recorded by this applications are structured the same way as the ones depicted in Figure A.5 and A.7.



/storage/emulated/0/DeviceSimulation/
Eval_05_10

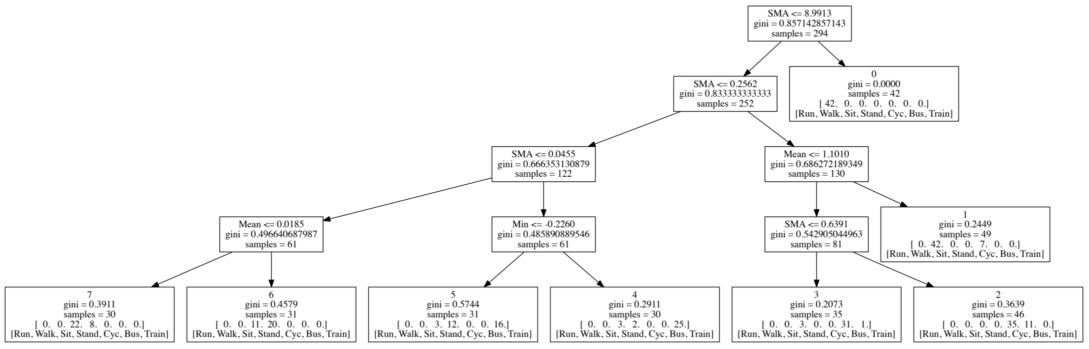
Figure A.8: A screenshot of the Simulation application

The full sourcecode for the application can be found at GitHub/Simulation

Server program

To simulate the server for the evaluation both of the previous presented server programs are used.

Appendix B
Decision trees



SMA <= 8.9913 gini = 0.857142857143 samples = 294
SMA <= 0.2562 gini = 0.833333333333 samples = 252
0 gini = 0.0000 samples = 42 [42. 0. 0. 0. 0. 0. 0.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
SMA <= 0.0455 gini = 0.666353130879 samples = 122
Mean <= 1.1010 gini = 0.686272189349 samples = 130
Mean <= 0.0185 gini = 0.496640687987 samples = 61
Min <= -0.2260 gini = 0.485890889546 samples = 61
SMA <= 0.6391 gini = 0.542905044963 samples = 81
1 gini = 0.2449 samples = 49 [0. 42. 0. 0. 7. 0. 0.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
7 gini = 0.3911 samples = 30 [0. 0. 22. 8. 0. 0. 0.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
6 gini = 0.4579 samples = 31 [0. 0. 11. 20. 0. 0. 0.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
5 gini = 0.5744 samples = 31 [0. 0. 3. 12. 0. 0. 16.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
4 gini = 0.2911 samples = 30 [0. 0. 3. 2. 0. 0. 25.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
3 gini = 0.2073 samples = 35 [0. 0. 3. 0. 0. 31. 1.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]
2 gini = 0.3639 samples = 46 [0. 0. 0. 0. 35. 11. 0.] [Run, Walk, Sit, Stand, Cyc, Bus, Train]

Figure B.1: Decision tree with depth 4.

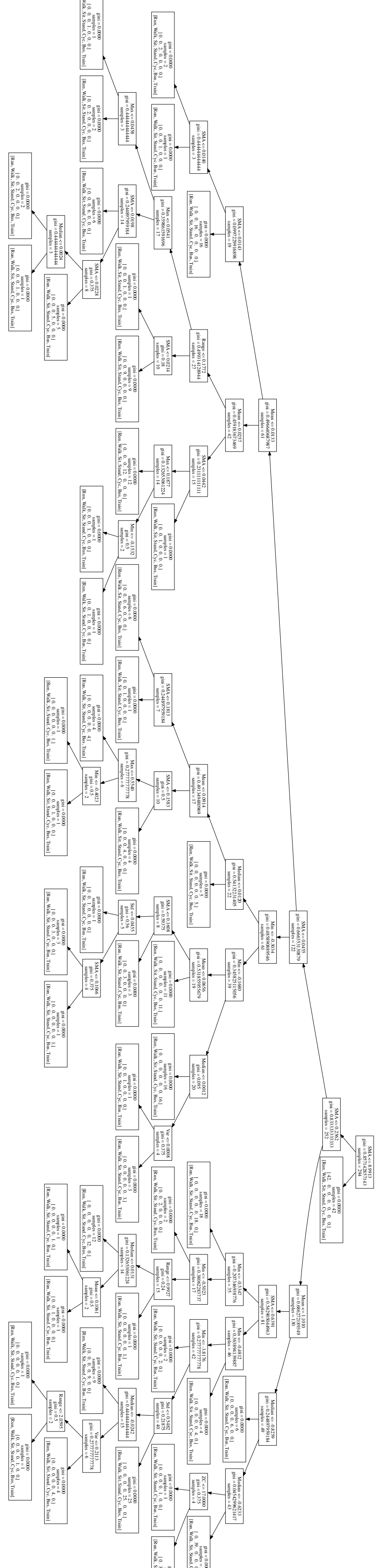


Figure B.2: Decision tree when no depth limitations is considered.

INSTITUTIONEN FÖR DATAVETENSKAP | LUNDS TEKNISKA HÖGSKOLA | PRESENTATIONSDAG 2015-06-04

EXAMENSARBETE Energy Efficient Geo-Localization for a Wearable Device
STUDENTER Robert Bagge, William Martinsson
HANDLEDARE Elin Anna Topp (LTH), Mikael Johansson (Narrative)
EXAMINATOR Jacek Malec (LTH)

Positionering som fungerar för wearables

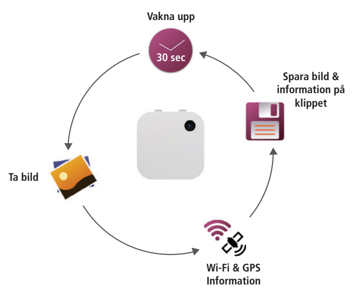
POPULÄRVETENSKAPLIG SAMMANFATTNING **Robert Bagge, William Martinsson**

Wearables förväntas trots sin ringa storlek ha liknande funktionalitet som betydligt större och kraftfulla enheter. Detta kräver att man utnyttjar deras styrkor för att till exempel få ut en korrekt position utan att påverka batteritiden.

Under det knappa decennium som gått sedan den första iPhonen kom ut på marknaden i juni 2007 har det skett en enorm utveckling bland smarta enheter. I skrivande stund finns det smarta klockor, glasögon, armband och kameror som håller på att ta över världen. Förväntningarna på dessa så kallade wearables är enorma, inte minst bland företag som ser en helt ny massmiljardmarknad torna upp sig vid horisonten. Trots det mindre formatet förväntas de kunna göra mycket av det arbete som våra telefoner och surfplattor utför idag. Det är ungefär som att slänga in en fotbollsspelare från Allsvenskan i en Champions League-final och förvänta sig att hen håller jämna steg med de andra.

Den mindre storleken innebär mindre plats för beräkningskraft, kameror och dylikt, men framförallt betydligt mindre plats för batteri. På grund av det mindre batteriet uppstår det en balansgång mellan batteritid och det arbetet som ska utföras. Det är i denna djupa dalgång av utmaningar och möjligheter som vi har vandrat in i för att söka svaret på frågan om det går att göra en positioneringsalgoritm som är så pass energisnål att den lämpar sig även för dessa små enheter.

Vi har arbetat med The Narrative Clip, vilket är en liten kamera som man kan fästa någonstans på kroppen.* Två gånger per minut vaknar klippet upp för att ta en bild på det som är framför personen för att på så sätt skapa ett fotografiskt minne över det personen upplevt. En viktig del i detta är att veta var bilderna är tagna rent geografiskt. De begränsade batteriresurserna gör det omöjligt att lyssna på signaler från GPS och Wi-Fi hela tiden för att få en så exakt position som möjligt. Det man kan göra istället är att spela in information från de olika sensorerna samtidigt som klippet vaknar upp och tar en bild för att sedan låta behandla datan i efterhand. Ett sådant tillvägagångssätt är väldigt energisnålt i jämförelse med att lyssna på sensorerna hela tiden.

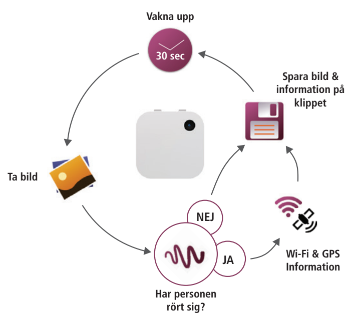


Uppgifter som klippet utför varje gång den vaknar upp för att ta en bild (var trettionde sekund).

Studier har visat sig att människor sitter still en större del av dagen, närmare 80 % av vår vakna tid för att vara exakt. Om det skulle kunna gå att urskilja när en person har rört sig eller inte kan 4 av 5 positionsmätningar undvikas och på så sätt spara energi. Lyckligtvis möjliggör den kroppsnära positionen av klippet detta på ett alldeles utmärkt sätt.

För att känna igen olika aktiviteter använder vi oss av accelerometern i klippet, vilket är en liten och strömsnål sensor som kan känna av små rörelser i klippet. Tiden mellan bilderna används för att samla in information från accelerometern som sedan körs genom en rörelseigenkännings-algoritm. Algoritmen letar mönster i accelero-meterdatan för att avgöra om personen har rört sig eller inte.

Genom att kombinera denna lösning med en positioneringsalgoritm baserad på Wi-Fi och GPS får vi ut en position som är korrekt med en precision på 28m samtidigt som ca 80 % av alla positionsmätningar kan undvikas. Det beskrivna tillvägagångssättet möjliggör positionering som är tillräckligt exakt för den smarta kameran samtidigt som den är väldigt energisnål.



Uppgifter som klippet utför varje gång den vaknar upp för att ta en bild om strömsnål positionering ska ske.

Vi tror att fler lösningar med kompromisser som dessa är det som krävs för att wearables ska kunna leva upp till de enorma förväntningar som finns på dem idag.

* *https://youtu.be/aUapwv8G0I8*