

September 2011
School of Health and Society
Department Computer Science
Embedded Systems

Indoor Positioning using Sensor-fusion in Android Devices

Authors

Ubejd Shala
Angel Rodriguez

Instructor

Fredrik Frisk

Examiner

Kamilla Klonowska

School of Health and Society
Department Computer Science
Kristianstad University
SE-291 88 Kristianstad
Sweden

Authors, Program and Year:

Ubejd Shala, Master's Program - Embedded Systems, 2011
Angel Rodriguez, Master's Program - Embedded Systems, 2011

Instructor:

Fredrik Frisk, Dr, HKr

Examination:

This graduation work on 15 higher education's credits is a part of the requirements for a
Degree of Master in Embedded Systems

Title:

Indoor Positioning using Sensor-fusion in Android Devices

Abstract:

This project examines the level of accuracy that can be achieved in precision positioning by using built-in sensors in an Android smartphone. The project is focused in estimating the position of the phone inside a building where the GPS signal is bad or unavailable. The approach is sensor-fusion: by using data from the device's different sensors, such as accelerometer, gyroscope and wireless adapter, the position is determined. The results show that the technique is promising for future handheld indoor navigation systems that can be used in malls, museums, large office buildings, hospitals, etc.

Keywords:

Sensor fusion, accelerometer, gyroscope, compass, INS, GPS, Wi-Fi, indoor navigation, smartphone, Android, Nexus S

Language:

English

Approved by:

Kamilla Klonowska, PhD, HKr
Examiner

Date

SUMMARY

In open air environment a GPS receiver is able to determine its position with very high accuracy. Inside a building, where the GPS signal is bad or unavailable, the position estimation from the GPS receiver is very erroneous and of no practical use.

Main *objective* of this thesis was to examine the level of accuracy that can be achieved in indoor positioning by using built-in sensors in an Android smartphone.

The device used for this project was a Nexus S, a smartphone co-developed by Google and Samsung. Its large number of built-in sensors was the main reason of choosing it for this project.

The *contribution* is the implementation of a sensor-fusion algorithm, where measurements of acceleration and orientation from the device's accelerometer, magnetometer and gyroscope are used to detect user's movement, pace and heading. Signal strength measurements of the Wi-Fi are used to estimate the position based on the known locations of the wireless Access Points using location fingerprinting.

In order to use advantages of one sensor to compensate for the drawbacks of the other, the algorithm includes a weighting system where the most reliable and stable source of information for the moment is promoted.

The *results* showed that the level of accuracy was quite satisfactory. The accomplished average deviation between the estimated and real position was less than two meters.

There are a number of ideas for *further development*. Increasing the accuracy of the position estimation is identified as the most essential. One interesting approach could be to implement the sensor fusion on a Kalman filter based model.

ACKNOWLEDGEMENTS

This thesis was carried out partly at Epsilon Embedded Systems in Malmoe, and partly at the Department of Computer Science at Kristianstad University.

First we would like to give our gratitude to Epsilon for providing us with equipment and for welcoming us to do the testing in their premises. Thanks also to our colleagues at Epsilon for sharing their ideas with us.

We would like to thank our supervisor Fredrik Frisk, who suggested the subject of this thesis, and our examiner Kamilla Klonowska for their feedback.

Finally we would like to thank our families and our close ones for their patience and support throughout the project.

*Ubejd Shala
Angel Rodriguez
Malmoe, September 2011*

TABLE OF CONTENTS

1 INTRODUCTION.....	1
1.1 BACKGROUND	1
1.2 STRUCTURE	1
1.3 AIM	1
1.4 TERMINOLOGY	2
2 THEORY	3
2.1 Wi-Fi.....	3
2.2 GPS.....	3
2.3 BLUETOOTH	4
2.4 INS.....	4
2.5 ERROR SOURCES	6
2.6 SENSOR FUSION	6
2.7 MOBILE PLATFORM	7
2.8 EQUIPMENT	8
3 DATA ACQUISITION AND ANALYSIS	10
3.1 INERTIAL SENSORS TEST	10
3.2 DEVICE STATIONARY ON THE TABLE	11
3.3 WALKING TEST.....	16
3.4 TROLLEY TEST.....	22
3.5 Wi-Fi TEST	23
3.6 BLUETOOTH	26
4 PROTOTYPE IMPLEMENTATION.....	27
4.1 FUNCTIONALITY	27
4.2 SYSTEM DESIGN.....	29
4.3 MAPPING	30
5 CHOICE OF SOLUTION.....	31
5.1 CHALLENGES.....	31
5.2 ESTIMATED POSITION USING INS SENSORS	35
5.3 ESTIMATED POSITION USING Wi-Fi.....	38
6 FINAL RESULTS.....	42
6.1 SENSOR-FUSION - INTEGRATION OF INS AND Wi-Fi	42
7 CONCLUSIONS	46
8 RECOMMENDATIONS FOR FURTHER WORK.....	47
REFERENCES	48
APPENDICES	50
APPENDIX A	50
APPENDIX B	51

LIST OF FIGURES

Figure 2-1: Android architecture diagram [12]	7
Figure 2-2: Nexus S smartphone	8
Figure 2-3: WAP54G and DIR-615.....	9
Figure 3-1: Coordinate system of the device [12]	10
Figure 3-2: Acceleration output (x-, y, z-axes)	12
Figure 3-3: Angular velocity	12
Figure 3-4: Calculated angles	13
Figure 3-5: Angles after correction (device stationary on the table).....	13
Figure 3-6: Angle around z-axis while rotating 2π radians	14
Figure 3-7: Angle around z-axis after compensating for the drift.....	14
Figure 3-8: Measured magnetic field (in micro-Teslas, uT)	15
Figure 3-9: Reference coordinate system of orientation [12].....	16
Figure 3-10: Rotation around the z axis (in radians).....	16
Figure 3-11: Acceleration output (x-, y, z-axes)	17
Figure 3-12: Linear acceleration on y-axis.....	18
Figure 3-13: Calculated velocity	18
Figure 3-14: Calculated distance	19
Figure 3-15: Illustration of gravity component	19
Figure 3-16: Android's linear acceleration and calculated linear acceleration.	20
Figure 3-17: Orientation output (around x-axis, i.e. pitch)	20
Figure 3-18: Orientation output (around z-axis, i.e. azimuth).....	21
Figure 3-19: Calculated angle around z-axis (azimuth, the walking direction)	21
Figure 3-20: Calculated angle around x-axis (pitch).....	22
Figure 3-21: Linear acceleration output (y-axis).....	22
Figure 3-22: Calculated velocity	23
Figure 3-23: Calculated distance	23
Figure 3-24: Part of the 5-th floor plan of the Epsilon building.....	24
Figure 3-25: Signal strength variation recorded over time.....	24
Figure 3-26: Measured signal strength at different testing points/nodes (from AP3)	25
Figure 4-1: Application state chart	28
Figure 4-2: System design	29
Figure 4-3: Epsilon map as displayed on the smartphone.....	30
Figure 5-1: The effect of constant acceleration on speed and position	32
Figure 5-2: Estimated position using inertial sensors only (device on trolley)	36
Figure 5-3: Step and motion detection	37
Figure 5-4: Estimated position using inertial sensors (walking test).....	38
Figure 5-5: Estimated position with Euclidian distance (instantaneous Wi-Fi readings)	40
Figure 5-6: Potential error in measurements using the average signal strength.....	41
Figure 5-7: Estimated position using Euclidian distance method (averaged Wi-Fi readings) .	41
Figure 6-1: Estimated position with average of linear acceleration, step detection and Wi-Fi	43
Figure 6-2: The sensor-fusion model.....	44
Figure 6-3: Estimated position using weighted average, Wi-Fi update at stops	45

LIST OF TABLES

Table 2-1: The Nexus S specification sheet [26].....	8
Table 2-2: Wi-Fi Access Points.....	9
Table 3-1: Built-in sensors in Nexus S.....	10
Table 3-2: Sample frequencies (ms).....	11
Table 3-3: Accelerometer output statistics	12
Table 3-4: Gyroscope statistics (angular speed).....	13
Table 3-5: Magnetometer output statistics	15
Table 3-6: Azimuth statistics.....	16
Table 3-7: Statistics of Wi-Fi signals over time	25
Table 3-8: Bluetooth scanning statistics.....	26
Table 6-1: Pros and cons of the information sources	42

1 Introduction

This chapter introduces the background, the structure, and the aim of the project.

1.1 Background

In the early days of smartphones, everything that could be done with a smartphone was possible to do from day one, without having the user install additional programs or applications. While the yesterday's smartphones barely supported access to the internet, today's smartphones can't do all that much without the internet and without downloading applications.

The high degree of customization and adaptation to the user's needs and desires, together with the large number of built-in sensors has helped boost their development.

One of the most appreciated features built in many smartphones is the access to maps and location information. Today's smartphones are often equipped with Global Positioning System (GPS) receivers that are used to determine a user's position. This information is then used by a navigation application to give the user directions to a desired destination. In an unknown environment the information from the GPS can help a user get to a destination with minimal support from the surroundings within a relatively small period of time. Since this feature depends on the GPS signal, it is of no practical use during periods with no GPS signal available.

1.2 Structure

This report begins with a short introduction (this chapter) where mainly the aim and purpose are presented together with a short presentation of previous research done in the area. Chapter 2 describes the information channels considered in this thesis (the built-in sensors, Wi-Fi, GPS and Bluetooth), their functioning and error characteristics. Also the development platform and equipment used for the project is described. The groundwork of this thesis where the data acquisition, testing and error analysis is done is found in chapter 3. The equations for calculating speed, distance angles etc, are also included in this chapter. Chapter 4 describes the implementation of the prototype software. Problems, methods and solutions are discussed in chapter 5, which also includes individual testing of the INS and Wi-Fi positioning sub-systems. Chapter 6 contains the description of the sensor fusion model, and presents the final results. Finally chapter 7 – Conclusions, and 8 – Future work, conclude this thesis.

1.3 Aim

The aim of this project is to examine the level of accuracy that can be achieved in precision positioning by using built-in sensors in an Android smartphone. The focus of the project is in estimating the position of the phone inside a building where the GPS signal is bad or unavailable. The purpose is to use all means of available information to make an as accurate estimation of the position of the phone as possible. The main problem to be solved is the implementation of a system realized from the sensor fusion of the Android inertial sensors, such as the accelerometer, magnetometer, and the gyroscope, with the signal strength measurements of the Wi-Fi.

The project involves also research of the value of the information given by the different information channels, as well as research of possible applications of the technology. In

addition, a software prototype with the researched functionality, in form of an Android application is developed.

There is a great deal of research done previously in the field of positioning inside buildings, and there are also different approaches and solutions for this problem. Most of the approaches make use of external transmitters/receivers installed in the building such infrared, VHF radio or Bluetooth, or make use of Wi-Fi and GSM networks, measuring signal strength to calculate the estimated position.

1.4 Terminology

Throughout the report, unless otherwise stated, the terms below are defined according to:

Device:	The Nexus S smartphone
Inertial sensors:	Accelerometer, gyroscope and magnetometer
Compass:	Digital compass; a software method provided by Android which uses the accelerometer and the magnetometer to compute the orientation
Trolley:	Carriage (with wheels)
Tilt/tilted:	Device not aligned to Earth's coordinate system

2 Theory

This chapter describes the information channels considered and used during this project, as well as their functionality and error characteristics. Also the development platform and equipment used for the project is described.

2.1 Wi-Fi

Wi-Fi (Wireless Fidelity) is the common name for the IEEE 802.11 standard. A Wi-Fi enabled device, such as a personal computer, video game console, smartphone, or digital audio player connects to the Internet via a wireless network Access Point [15].

Communication across a wireless network is like two-way radio communication:

- A device's wireless adapter converts data into radio signals and transmits it via an antenna.
- A wireless router receives the signal and decodes it, and then it sends the information to the Internet using a, physically wired, Ethernet connection.

The process works the same way in reverse; with the router receiving data from the Internet, converting it into radio signals, and sending it to the device's wireless adapter.

Each wireless router broadcasts a signal that is received by devices in the area. These devices have the capability to measure the strength of the signal. This strength is converted to a number, known as received signal strength indicator (RSSI). Wi-Fi devices, such as smartphones, typically perform this conversion automatically in order to provide signal strength information to applications running on it.

2.2 GPS

2.2.1 Overview

The Global Positioning System (GPS) is a satellite-based navigation system developed by the U.S. Department of Defence for military purposes. The system was declared fully operative in 1994 [14]. Today the GPS is used also for civilian purposes, such as surveying, map design, tectonics and obviously navigation.

The GPS system consists of three segments; the space segment (satellites), the control segment and the user segment (receivers) [13].

The space segment consists of 24 satellites orbiting in six orbits which have an inclination of 55° relative to the Equator. The orbits are arranged in such way that at least six satellites are always visible from everywhere on the Earth's surface.

GPS satellites send navigation messages continuously at a rate of 50 bits per second; the main information in the message is the time when the message was sent, exact orbital information ("ephemeris"), and the general system health and rough orbits of all satellites (the "almanac"). The control segment is composed of a number of stations and antennas, they are used to control and monitor the health of the satellites, and do necessary corrections when needed, for instance adjust the satellites' clocks.

The user segment consists of military users of the GPS Precise Positioning Service and the civil users of the Standard Positioning Service. The GPS receiver is mainly composed of an antenna, a very stable internal clock, the software for calculating the user's location and speed, and usually a display for providing the information to the user.

2.2.2 Position Calculation

GPS receivers use geometric trilateration to combine the information from different satellites to predict the correct location. As mentioned above, the GPS message contains information about the time when message was sent, precise orbital information, health of the system and rough information about the orbits of other satellites.

The receiver uses the message to calculate the transfer time of each message and computes the distance to the satellite. With the aid of trilateration [14] [16], the distances to the satellites together with the satellites' locations are used to calculate the position of the receiver.

The advantage of satellite systems is that receivers can determine latitude, longitude, and altitude to a high degree of accuracy. However, line of sight (LOS) is required for the functioning of these systems. This leads to an inability to use these systems for an indoor environment where the LOS is blocked by walls and roofs.

2.3 Bluetooth

Bluetooth is a wireless technology standard used by two (or more) devices for communication over short distances. The big advantages of Bluetooth are that it is wireless, inexpensive and automatic.

Compared to infrared technology - another way of wireless communication (commonly used in television remote control systems) - Bluetooth has a number of advantageous qualities. Although infrared communications are fairly reliable and don't cost very much to build into a device, they have a couple of drawbacks. First, infrared is a line of sight technology. For example, you have to point the remote control at the television to change the channels. The second drawback is that infrared is almost always a one-to-one technology.

Bluetooth is intended to get around the problems that come with infrared systems. It uses a technique called spread-spectrum frequency hopping, which allows connection between several devices simultaneously, without them interfering with one another. In addition Bluetooth doesn't require line of sight between communicating devices, which makes the technology useful for controlling several devices in different rooms.

Class 1 Bluetooth supports a maximum transfer speed of 1 megabit per second (Mbps), while class 2 Bluetooth, the common standard on handheld devices, can manage up to 3 Mbps, with a maximum communication distance of up to 10 meters.

2.4 INS

INS (Inertial Navigation System) is based on a self-contained navigation technique in which measurements provided by accelerometers and gyroscopes (and/or compasses) are used to track the position and orientation of an object relative to a known starting point, orientation and velocity [4].

INS can be divided in two categories which differ in the frame of reference in which the sensors operate. The navigation system's frame of reference is defined as the *body frame*, and the frame of reference in which the navigating occurs as the *global frame*.

The first category includes Stable Platform Systems, where the inertial sensors are mounted on a platform which is isolated from any external rotational motion, i.e. the platform is aligned with the global frame.

The second category, in which this project is based, includes Strap-down Systems: the inertial sensors are mounted rigidly onto the device, and therefore output quantities measured in the body frame rather than the global frame [4].

2.4.1 Accelerometer

An accelerometer is a device that measures the *proper acceleration* of the device. This is not necessarily the same as the coordinate acceleration (change of velocity of the device in space), but is rather associated with the phenomenon of weight experienced by a test mass that resides in the frame of reference of the accelerometer device.

Generally, accelerometers measures acceleration by sensing how much a mass presses on something when a force acts on it.

An accelerometer at rest relative to the Earth's surface will indicate approximately 1 G upwards, because any point on the Earth's surface is accelerating upwards relative to the local inertial frame (the frame of a freely falling object near the surface). To obtain the acceleration due to motion with respect to the Earth, this "gravity offset" must be subtracted [18].

2.4.2 Gyroscope

Generally, a gyroscope is a device for measuring or maintaining orientation, based on the principles of conservation of angular momentum [17].

A conventional (mechanical) gyroscope consists of a spinning wheel mounted on two gimbals which allow it to rotate in all three axes. An effect of the conservation of angular momentum is that the spinning wheel will resist changes in orientation. Hence when a mechanical gyroscope is subjected to a rotation, the wheel will remain at a constant global orientation and the angles between adjacent gimbals will change. A conventional gyroscope measures orientation, in contrast to MEMS (Micro Electro-Mechanical System) types, which measure angular rate, and are therefore called rate-gyros [4].

MEMS gyroscopes contain vibrating elements to measure the Coriolis effect. A single mass is driven to vibrate along a drive axis, when the gyroscope is rotated a secondary vibration is induced along the perpendicular sense axis due to the Coriolis force. The angular velocity can be calculated by measuring this secondary rotation.

The Coriolis effect states that in a frame of reference rotating at angular velocity ω , a mass m moving with velocity v experiences a force [4]:

$$F_c = -2m(\omega * v) \quad (2-1)$$

An important note to be made is that whereas the accelerometer and the magnetometer measure acceleration and angle relative to the *Earth*, gyroscope measures angular velocity relative to the body.

2.4.3 Magnetometer

A magnetometer is an instrument used to measure the strength and/or direction of the magnetic field in the surrounding area of the instrument.

Magnetometers can be divided into two basic types: *scalar* magnetometers that measure the total strength of the magnetic field to which they are subjected, and *vector* magnetometers (the type used in this project), which have the capability to measure the component of the magnetic field in a particular direction, relative to the spatial orientation of the device [19].

2.5 Error sources

2.5.1 Accelerometer

The most important source of error of an accelerometer is the bias. The bias of an accelerometer is the offset of its output signal from the true value, in m/s^2 . A constant bias error of ε , when double integrated, causes an error in position which grows quadratically with time. The accumulated error in position is [4]:

$$s(t) = \varepsilon * \frac{t^2}{2} \quad (2-2)$$

where t is the time of the integration.

It is possible to estimate the bias by measuring the long term average of the accelerometer's output when it is not undergoing any acceleration. Uncorrected bias errors are typically the error sources which limit the performance of the device [4].

2.5.2 Gyroscope

The bias of a rate gyro is the average output from the gyroscope when it is not undergoing any rotation (i.e: the offset of the output from the true value), in $^\circ/\text{h}$. A constant bias error of ε , when integrated, causes an angular error which grows linearly with time [4]:

$$\theta(t) = \varepsilon * t \quad (2-3)$$

The constant bias error of a rate gyro can be estimated by taking a long term average of the gyro's output whilst it is not undergoing any rotation. Once the bias is known it is trivial to compensate for it by simply subtracting the bias from the output [4].

Another error arising in gyros is the 'calibration error', which refers to errors in the scale factors, alignments, and linearities of the gyros. Such errors tend to produce errors that are only observed whilst the device is turning. Such errors lead to the accumulation of additional drift in the integrated signal, the magnitude of which is proportional to the rate and duration of the motions [4].

2.5.3 Magnetometer

The two main sources of measurement errors are magnetic contamination in the sensor, errors in the measurement of the frequency and ferrous (iron containing) material on the operator and in the instruments. If the sensor is rotated as the measurement is made, an additional error is generated [19].

2.6 Sensor fusion

Sensor fusion is the combining of sensory data derived from sensory data from disparate sources such that the resulting information is in some sense better than would be possible when these sources were used individually. The term better in this case can mean more accurate, more complete, or more dependable, or refer to the result of an emerging view, such as stereoscopic vision (calculation of depth information by combining two-dimensional images from two cameras at slightly different viewpoints) [11].

2.7 Mobile platform

2.7.1 Android Operative System

Android is an open-source platform for mobile devices. It is developed and managed by Google Inc, and it includes operating system, middleware and key applications. Recently it has become the world's most used platform for smartphones. The increasingly high popularity together with the fact of being an open-source project has given the platform a very large community of developers. The mentioned characteristics are the main reasons for choosing Android as the development environment for this thesis rather than any other mobile platform like iPhone OS, Symbian or Windows Phone.

Android is a Linux-based OS software; its stack is divided in four different layers which include five groups [12]:

- Application layer: This layer is the one used by end phone users. Applications can run simultaneously (multitasking) and they are written in Java language.
- Application framework: The software framework used to implement the standard structure of an application for the Android OS.
- Libraries: The available libraries are written in C/C++ and they are called through a Java interface.
- Android runtime: The Android runtime consists of two components. First a set of core libraries which provides most of the functionality in the Java core libraries. And second the virtual machine Dalvik which operates like a translator between the application side and the operating system.
- The kernel: This Linux based kernel is used by Android for its device drivers, memory management, process management and networking.

The Android architecture is illustrated in the picture below:

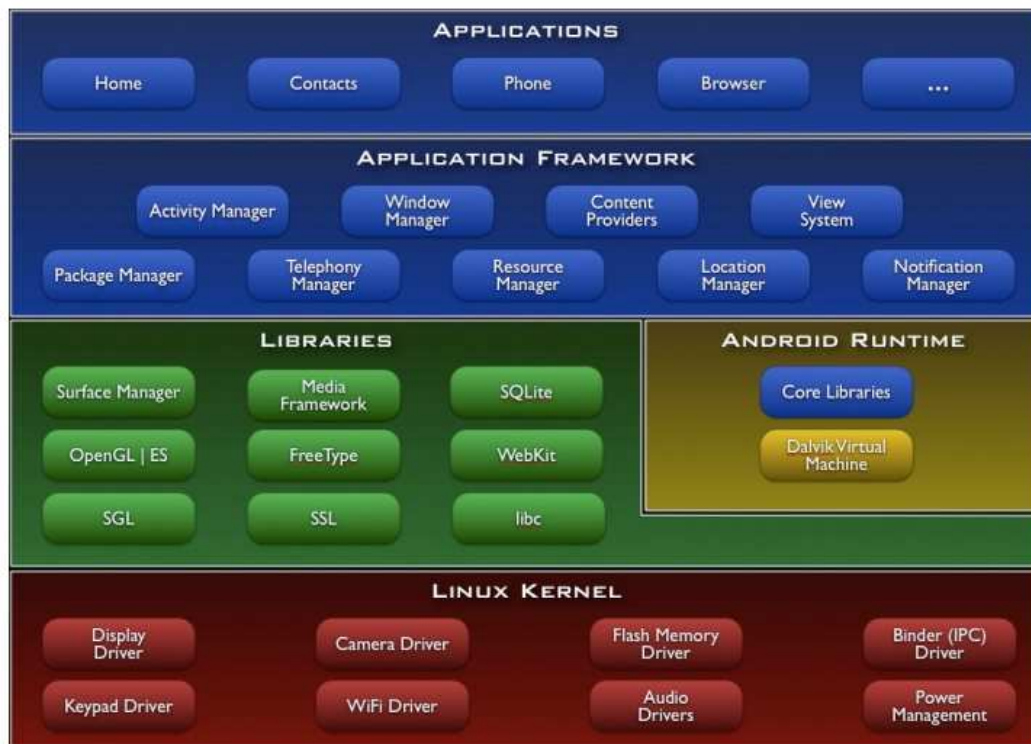


Figure 2-1: Android architecture diagram [12]

2.8 Equipment

2.8.1 Nexus S

Nexus S is the next generation of Nexus devices co-developed by Google and Samsung. It is the first smartphone to use the Android 2.3 “Gingerbread” OS. [23].

The most significant reason of choosing this device is the large number of built-in sensors into it. At the project start Nexus S was the only Android smartphone available on the market that included and supported a gyroscope.

Table 2-1: The Nexus S specification sheet [26]

Nexus S	
Physical dimensions	Height: 123.9mm Width: 63mm Depth: 10.88mm Weight: 129g
Processor	Cortex A8 (Hummingbird), 1 GHz
Operating System	Android™ 2.3.4 (Gingerbread)
Memory	16GB iNAND flash memory
Display	4.0” WVGA (480x800)
Connectivity	GSM: 850, 900, 1800, 1900 HSDPA (7.2Mbps) HSUPA (5.76Mbps) Assisted GPS (A-GPS) Wi-Fi: 802.11 n/b/g Bluetooth 2.1+EDR microUSB 2.0 Near Field Communication (NFC)
Sensors	Accelerometer, 3-axes gyroscope, sensor, digital compass, proximity sensor, ambient light sensor



Figure 2-2: Nexus S smartphone

2.8.2 Access Points

The AP:s used in this project are listed in the table below. The reason of choosing them during the project is no other than they were available at Epsilon.

Table 2-2: Wi-Fi Access Points

Name	Vendor	Model ID
AP1	Cisco Systems	WAP54G
AP2	Cisco Systems	WAP54G
AP3	D-Link	DIR-615
AP4*	Netgear	WNR834B

*AP4 is only used in the Norra station, the university building in Hässleholm.



Figure 2-3: WAP54G and DIR-615

3 Data acquisition and analysis

In this chapter data acquisition, evaluation and error analysis of the all considered information channels is performed. The equations for calculating speed, distance angles etc, are also included. First the inertial sensors are tested, and then the Wi-Fi and Bluetooth.

3.1 Inertial sensors test

The sensors are tested for errors and to consider the need for calibration before the outputs is used. The built-in sensors in the Nexus S are listed in Table 3-1.

Table 3-1: Built-in sensors in Nexus S

Sensor	Name	Vendor	Range
3-axes accelerometer	KR3DM	STMicroelectronics	19,61
Gyroscope	K3G	STMicroelectronics	34,91
Magnetometer	AK8973	Asahi Kasei Microdevices	2000,0

To analyze the accuracy and the behavior of the sensors, numerous tests are performed repeatedly. The testing software, implemented in java on the Android OS, takes samples from the accelerometer, magnetometer and gyroscope at the highest rate allowed by Android, which is approximately 20, 16 and 1.2 milliseconds per sample for the accelerometer, magnetometer and gyroscope respectively. The output of the sensors is relative to the device's orientation, here referred to as the device's coordinate system.

The device's coordinate system is defined relative to the screen of the phone in its default orientation. The axes are not swapped when the device's screen orientation changes.

The x axis is horizontal and points to the right, the y axis is vertical and points up, and the z axis points towards the outside of the front face of the screen [12].

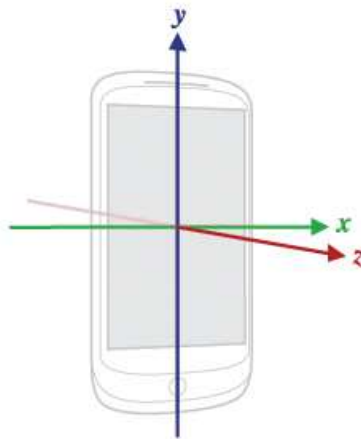


Figure 3-1: Coordinate system of the device [12]

The tests are performed in three different methods. The device is:

1. Stationary on the table
2. Stationary on a moving trolley
3. Held in the hand while walking

Each method is tested four times, with the phone top (its y-axis) pointing at different directions: North, West, South and East. The report includes only the measuring done in the North direction.

3.1.1 Sampling frequency

Ideally it should be possible to choose a high and constant sampling frequency in order to miss as little data as possible. In Android it is not possible to choose the sampling frequency and read directly from the inertial sensors. Instead a sensor event is generated every time the sensor values changes.

Since the sample frequency on the device is limited, some information will be missed.

A test was done to analyse the sampling frequency of each sensor. The measurements were done both with the device being stationary and moving. The results of both tests were similar, and are presented on Table 3-2.

Table 3-2: Sample frequencies (ms)

Sensor	Sampling rate (avg.)	Max	Min	Standard deviation
Gyroscope	1,154713	3,512	0,589	0,2714452
Magnetometer	16,81256	19,300	14,380	0,4401434
Accelerometer	20,57853	23,703	18,256	0,4601378

3.2 Device stationary on the table

In the first method the samples are recorded during a period of approximately 20 seconds with the phone laying flat with its back on the table. The phone is immobile in order to prevent any force other than gravity from affecting the output.

3.2.1 Accelerometer

The accelerometer measures the acceleration in three axes in m/s^2 . It outputs the acceleration applied to the device by measuring forces applied to the sensor. The measured acceleration is always influenced by the force of the earth's gravity [12]:

$$a_d = -g - \sum F / m \quad (3-1)$$

where a_d is the acceleration applied to the device, g the force of gravity, F the force acting on the device, and m the mass of the device. The sign Σ represents the sum of the x-, y- and z-axis.

As a result, when the device is in free fall and therefore accelerating towards the ground at 9.81 m/s^2 , its output will generate 0 m/s^2 for all three axes.

Thus, when the device is put on the table (and obviously not accelerating), the accelerometer output from the three axes should read:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \quad (3-2)$$

the g being earth's gravity of 9.81 m/s^2 .

Figure 3-2 shows the phone's actual acceleration output for the x, y and the z axes while being stationary on the table.

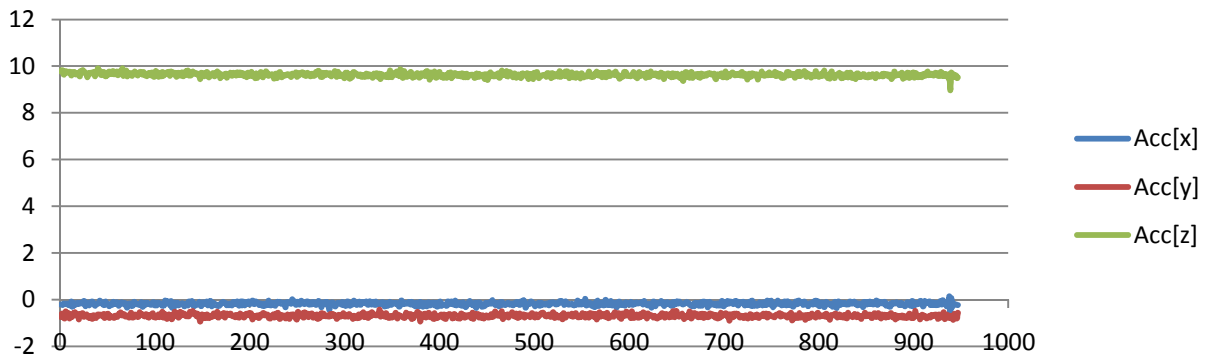


Figure 3-2: Acceleration output (x-, y, z-axes)

Table 3-3: Accelerometer output statistics

Axis	Average	Max	Min	Std deviation
Acc _x [m/s ²]	-0,17324	0,13407	-0,65122	0,06874
Acc _y [m/s ²]	-0,68445	-0,42138	-0,93853	0,07291
Acc _z [m/s ²]	9,62924	9,90241	8,96389	0,08333
Total Acc [m/s ²]	9,65561	9,92406	9,01541	0,08288

The test shows that the total acceleration measured at rest was on average about 9.66 m/s^2 and not the expected 9.81 m/s^2 . The standard deviation for the total acceleration, 0.08 m/s^2 , is equivalent to nearly one percent of the total acceleration, which, with time, could potentially generate a large error.

3.2.2 Gyroscope

The gyroscope values are in radians/second and measure the rate of rotation around the x, y and z axis. Rotation is positive in the counter-clockwise direction.

When the device is at rest on a table and not moving, the gyroscope values should read a magnitude of 0 radians per second.

Figure 3-3 shows the measured angular speed around the x, y, and z axis when the device is stationary on the table. (Values of the x axis are made invisible on the graph because of the offset from y and z axis).

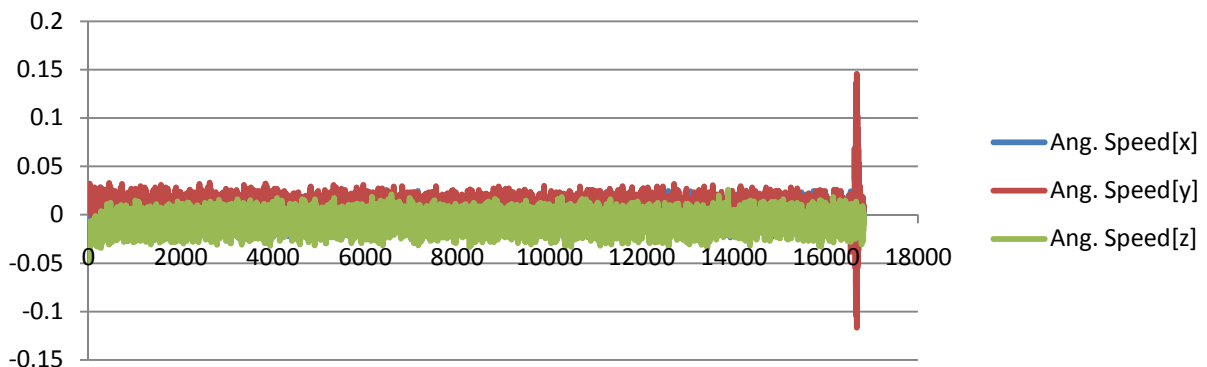


Figure 3-3: Angular velocity

Table 3-4: Gyroscope statistics (angular speed)

Axis	Average	Max	Min	Std deviation
ω_x [rad/s]	-0,000127	0,062308	-0,072082	0,007556
ω_y [rad/s]	0,006185	0,145385	-0,114842	0,008577
ω_z [rad/s]	-0,007724	0,025656	-0,047647	0,007891

Observing the results it can be seen that there exists an offset (the bias) on all of the three axes, but mostly on y (positive) and z axis (negative). To calculate angle α , the output of the gyroscope ω (angular speed), is integrated over time t [25]:

$$\alpha_n = \sum_{i=0}^n (\omega_i * \Delta t) \quad (3-3)$$

The graph below shows the results without compensating for the bias error introduced by the offset: during 20 seconds we get an angle of 0.12 and -0.15 radians around the y and z axis respectively. For example, the error on the y axis is:

$0.12/20 = 0.006$ radians/s, or **0.35 degrees/s**

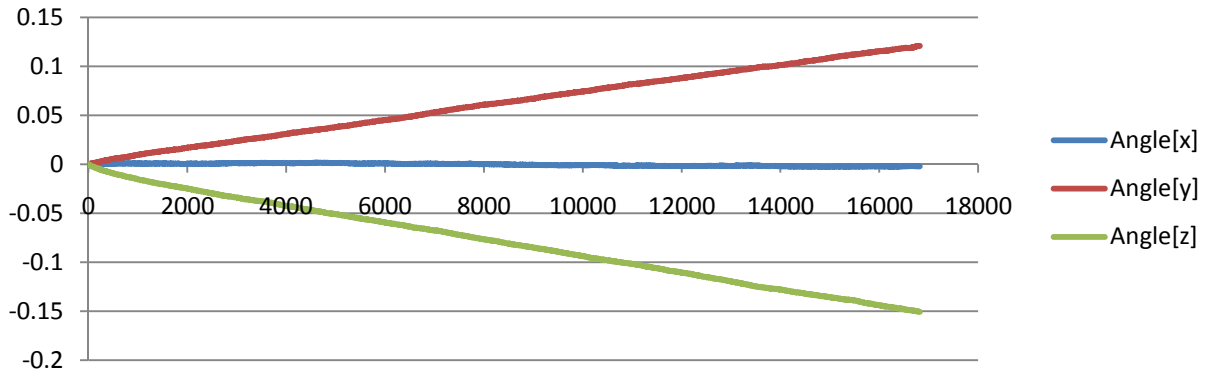


Figure 3-4: Calculated angles

Since the expected average is zero, the calculated average of the measured values is approximately equal to the offset error.

The graph below shows the results (calculated angles) after compensating for the bias error ϵ .

$$\omega'_i = \omega_i - \epsilon_{x,y,z} \quad (3-4)$$

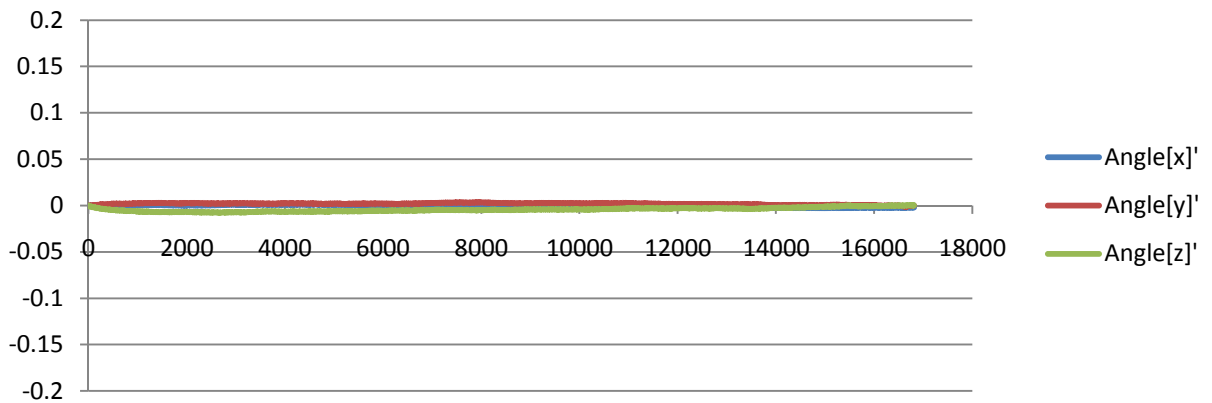


Figure 3-5: Angles after correction (device stationary on the table)

Besides the bias it was observed that the gyroscope introduces a drift in its measured angular speed. To analyze the drift one more test was done. During the test the device was rotated 360° around the z-axis to end exactly in the same position. The test was repeated rotating the phone in different tempos (5 seconds, 10 seconds and 20 seconds) and in different directions (clockwise and anti-clockwise).

All tests produced similar results. Figure 3-6 shows the integrated angular speed (angle around z-axis in radians) for one of the tests:

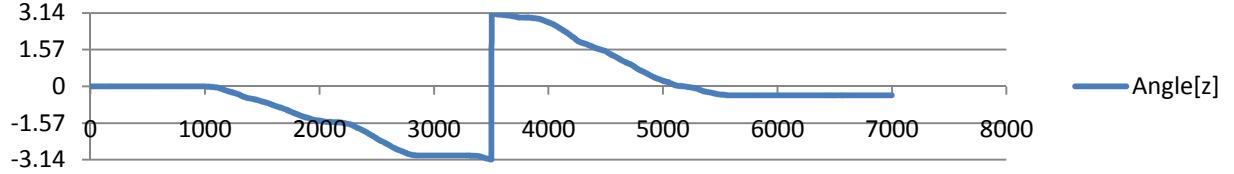


Figure 3-6: Angle around z-axis while rotating 2π radians

All of the six tests performed showed a similar output as the one displayed in the graph. After one complete rotation (2π radians) the drift was about 0.3 radians when rotating clockwise and -0.3 radians when rotating anti-clockwise. This error however was not completely constant, being slightly bigger when the rotation was done the faster.

To compensate for the drift a correction is applied. Equation 3-4 becomes:

$$\omega_i = \omega_i - \omega_i * (1 - \varepsilon'_{x,y,z}) \quad (3-5)$$

where ε' is the estimated error of the angular speed around each axis.

The drift error around the z-axis is calculated with the following formula (being 0.33 the average drift in the previous tests):

$$\varepsilon'_z = 0.33/2\pi = 0.053 \text{ (5.3\%)} \quad (3-6)$$

This correction significantly reduces the gyroscope drift which is accumulated over time in the angles. A similar test to the previous one was done, this time compensating for the drift. Results are illustrated in Figure 3-7.

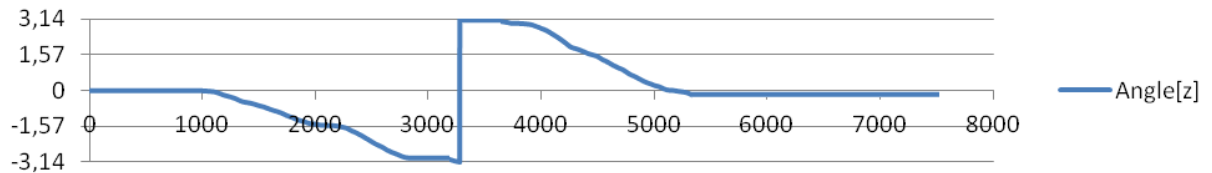


Figure 3-7: Angle around z-axis after compensating for the drift

The remaining error is very small, but as it accumulates in the angle due to the integration of the angular speed, an adjustment needs to be done regularly. The adjustment is done in such way that when the magnetometer readings are stable, the gyro azimuth is set to the azimuth given by the magnetometer.

3.2.3 Magnetometer

The magnetometer measures the strength of the ambient magnetic field in micro-Tesla (uT), in the x, y and z axes. The magnetometer output together with accelerometer values are passed to a function to get the orientation (see next section).

Figure 3-8 shows the measured magnetic field in the x, y and z axis respectively.

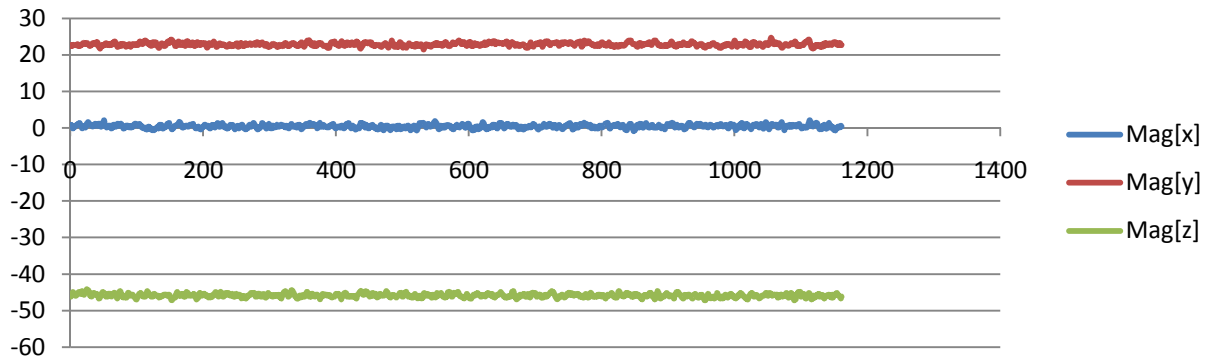


Figure 3-8: Measured magnetic field (in micro-Teslas, uT)

Statistics from the test are displayed in the table below:

Table 3-5: Magnetometer output statistics

Axis	Average	Max	Min	Std deviation
Mag _x [uT]	0,49209	2,375	-0,875	0,47814
Mag _y [uT]	22,87351	24,4375	21,4375	0,43761
Mag _z [uT]	-45,85589	-44,1875	-47,1875	0,47514

One interesting observation is the sensitivity of the sensor, which results in a rather high standard deviation at above 0.40 uT for each axis.

3.2.4 Orientation

The orientation sensor (digital compass) is a “fused sensor”, i.e. it is a software method provided by Android which uses readings from the accelerometer and the magnetometer to compute the phone’s orientation based on the rotation matrix.

The values returned by the method are in radians and are positive in the counter-clockwise direction:

- values[0]: *azimuth*, rotation around the z axis.
- values[1]: *pitch*, rotation around the x axis.
- values[2]: *roll*, rotation around the y axis.

The reference coordinate system is defined according to:

- x is defined as the vector product $\mathbf{y} \cdot \mathbf{z}$ (it is tangential to the ground at the device's current location and roughly points West).
- y is tangential to the ground at the device's current location and points towards the magnetic North Pole.
- z points towards the center of the Earth and is perpendicular to the ground.

The main functionality that the magnetometer provides to this project is the ability to calculate the magnetic Azimuth, i.e. the relative direction or angular distance from the magnetic north.

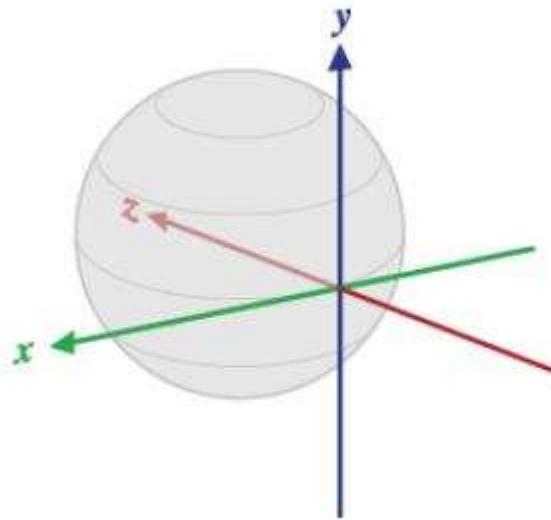


Figure 3-9: Reference coordinate system of orientation [12]

The graph and the table below show the device's measured (actually calculated) rotation, in radians, around the z axis (of the coordinate system defined and illustrated above), device stationary on the table, with its top pointing to north.

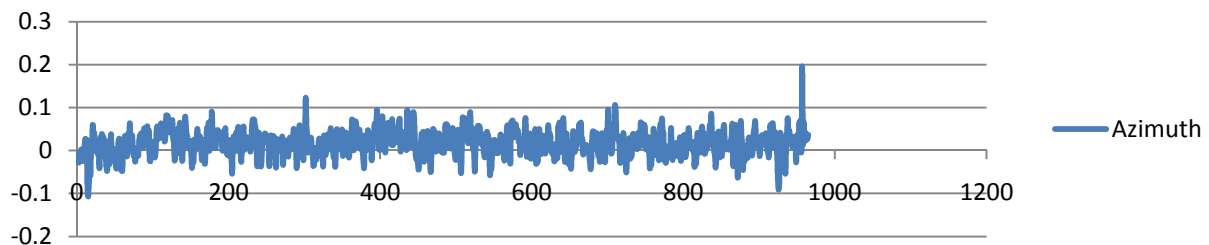


Figure 3-10: Rotation around the z axis (in radians)

Table 3-6: Azimuth statistics

Rotation	Average	Max	Min	Std deviation
Azimuth [rad]	0,01705	0,19625	-0,10671	0,02980

The expected result is a stable value around zero radians. The actual results from the sampled data show the opposite; the signal is rather unstable with a rather high standard deviation. This demonstrates the effect of the errors from the accelerometer and sensitivity of the magnetometer.

3.3 Walking test

A second test was done to analyze the sensors' outputs while moving. Here the samples are recorded during a period of approximately 20 seconds. The first 5 seconds standing still, and then walking for about 15 steps and after standing still again for 5 seconds. The test was performed 2 times, one walking with direction North and another walking with direction South. The report includes only the measuring done in the North direction.

3.3.1 Raw acceleration

Figure 3-11 shows the raw acceleration values on the device's three axes during the walking test.

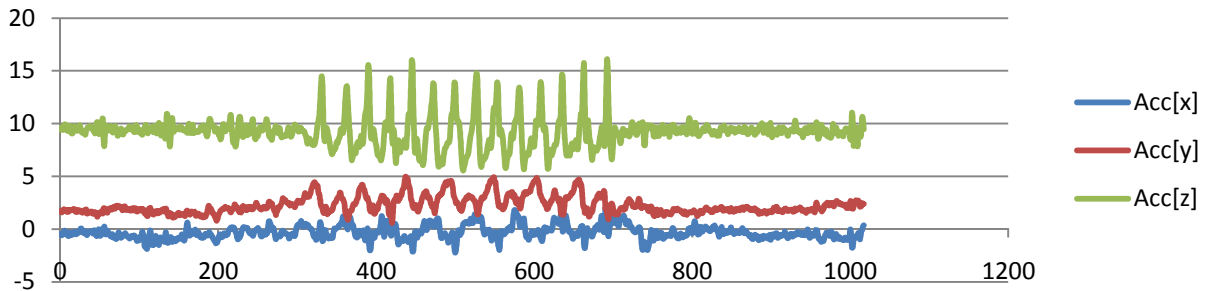


Figure 3-11: Acceleration output (x-, y, z-axes)

As expected, most of the activity is sensed on the z-axis of the device. This is because the device is held in the hand with its back facing the surface of the earth and having the gravity affecting mostly the z-axis. The force applied from the step impact has the same direction as the gravity which explains the sharp peaks detected on z-axis.

Some of the activity sensed on the y-axis is the acceleration detected in the walking direction caused by the actual moving. The rest is the gravity component acting on it, together with the acceleration caused by the tilt (the device is not held perfectly horizontal in the hand).

The acceleration sensed on the x-axis is mostly caused by the gravity component and tilt. Since the walking is done “straight” along the y-axis, no (very little) acceleration is expected to be sensed on the sides.

3.3.2 Linear acceleration

The linear acceleration is calculated by using the magnetometer and accelerometer outputs to get the orientation of the device and deduct the gravity components on respective axis. The result is the acceleration along the axes on the device's coordinate system. The interesting component in this context is the forward (and backward) acceleration, which is obtained by reading the linear acceleration on the y-axis.

The output is a three dimensional vector indicating acceleration along each device axis, not including gravity. All values have units of m/s^2 . The coordinate system is the same as is used by the acceleration sensor.

The output of the accelerometer, gravity and linear-acceleration sensors must obey the following relation [12]:

$$\text{Total acceleration} = \text{Gravity} + \text{Linear acceleration} \quad (3-7)$$

The linear acceleration is treated by Android as a pseudo sensor retaining the outputs in the same way as the other three sensors. The implementation is done differently depending on the device and the exact code is not public. However, Google states that for the Nexus S only the output from the compass and accelerometer is used to calculate the linear acceleration [22].

Figure 3-12 shows only the obtained linear acceleration component on the y-axis (forward acceleration) as is the acceleration to be considered for calculating the velocity and walked distance.

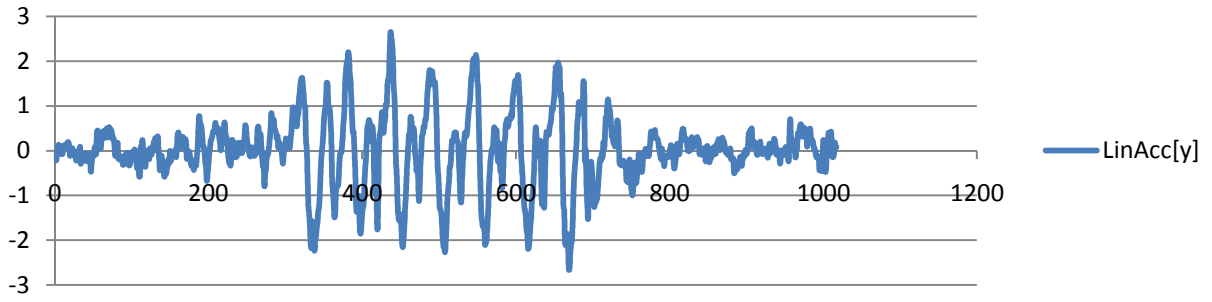


Figure 3-12: Linear acceleration on y-axis

To calculate the velocity v , the linear acceleration a , is integrated over time t [25]:

$$v_n = \sum_{i=0}^n (a_i * \Delta t) \quad (3-8)$$

Figure 3-13 shows the velocity resulted from integrating the forward acceleration. The form of the curve seems correct. And it is possible to identify the steps while walking (15 steps) and also that the increment/decrement of speed is stronger with the step of one foot than with the step of the other foot.

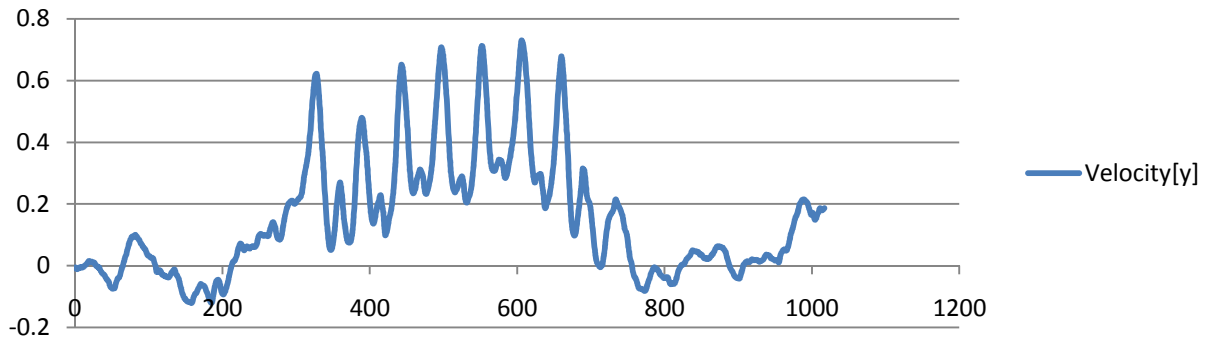


Figure 3-13: Calculated velocity

To calculate the distance d , the velocity v , is integrated over time t [25]:

$$d_n = \sum_{i=0}^n (v_i * \Delta t + \frac{1}{2} a_i \Delta t^2) \quad (3-9)$$

Figure 3-14 shows walked distance calculated by integrating the velocity over time (previous figure).

While the form of the curve is very likely as expected, the distance constantly increases while walking, the values are not similar to the real walked distance. They are approximately 3 times smaller as the curve shows that we walked a total distance of 3 meters).

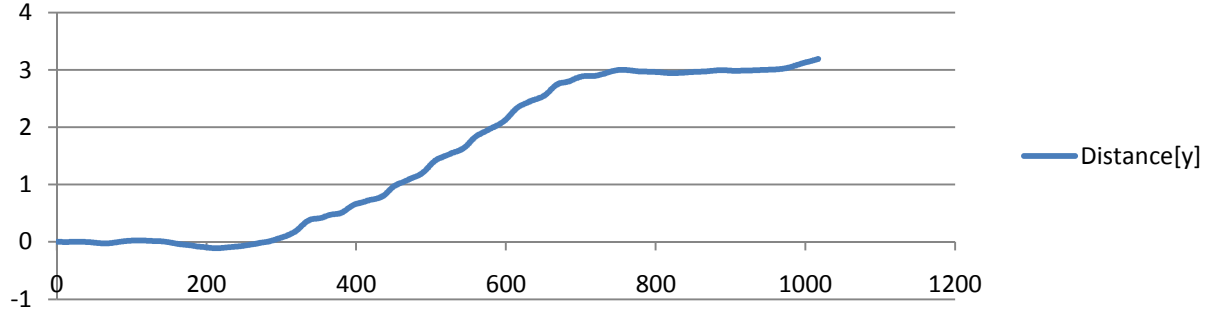


Figure 3-14: Calculated distance

As the results of integrating the linear acceleration weren't the desired, another test was done to check if the linear acceleration that android calculates could introduce errors (due to the use of the compass orientation to deduct the gravity component). For calculating the linear acceleration only the raw acceleration data and the gyroscope were used.

This test aims to compensate for the gravity component caused by rotations around the x-axis only (as this affects the linear acceleration on walking direction most). It is therefore assumed that the gravity component caused by rotations on the y-axis is zero. Figure 3-15 shows the principle.

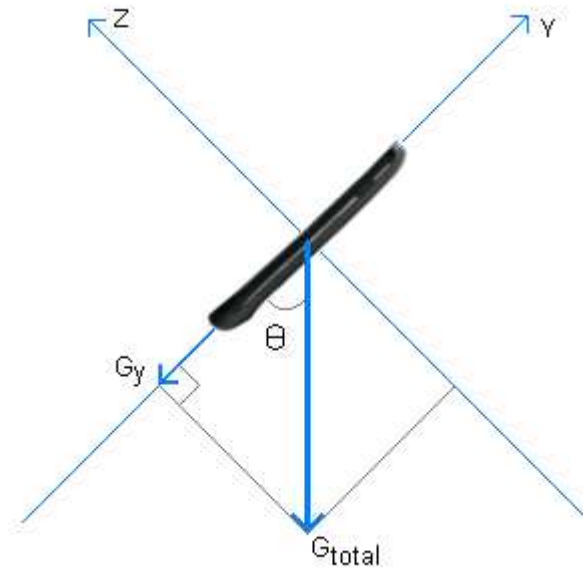


Figure 3-15: Illustration of gravity component

The first issue here is to calculate the initial value of the angle around the x-axis. This angle would be the initial value to add to the integrated angular velocity. The solution to this problem was to start with the device at rest on the table and average the acceleration on the y-axis. As the only acceleration at rest is the gravity, that value is the projection of the gravity vector on the axis. With the gravity and the projected value on the axis, it is possible to calculate the rotation angle as follows [12]:

$$\theta_0 = \arccos\left(\frac{G_y}{G_{total}}\right) - \frac{\pi}{2} \quad (3-10)$$

Where θ_0 is the initial rotation angle around the x-axis and G_y is the projection of G_{total} , the total gravity along the y-axis. As the acceleration values fluctuate, an average of 500 values at

rest was taken for both G_y and G_{total} . The reason to calculate the total gravity and not use 9.81 is that as shown in the resting test the total acceleration at rest is slightly lower than expected. The following formula was used and also the average over 500 values was taken [12]:

$$G_{total} = \sqrt{G_x^2 + G_y^2 + G_z^2} \quad (3-11)$$

Having the correct rotation angle (θ_y) around the x-axis calculated by integrating the angular speed output of the gyroscope, the forward acceleration (along the y-axis) is calculated as follows [12]:

$$LinAcc_y = Acc_y - G_{total} * \cos\left(\frac{\pi}{2} + \theta_y\right) \quad (3-12)$$

Where the projection of the gravity vector is deducted from the acceleration measured on the y-axis.

Figure 3-16 compares the *calculated* linear acceleration along the y-axis (red) with the linear acceleration that Android returns (green).

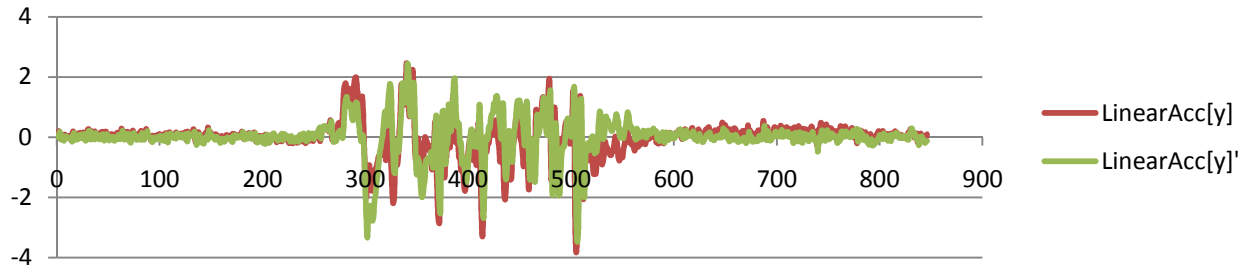


Figure 3-16: Android's linear acceleration and calculated linear acceleration.

As the graph shows, there are no major differences from the two outputs. According to these results it can not be concluded that the values returned by the linear acceleration in Android were incorrect (Figure 3-12). The poor values could be caused by the low accuracy of the sensors.

3.3.3 Orientation

Figure 3-17 and Figure 3-18 show the orientation of the device in radians during the walking test generated by the digital compass. The first one is the azimuth (rotation around the z-axis) or in other words the direction towards north while walking. The second figure shows the pitch (rotation around the x-axis).

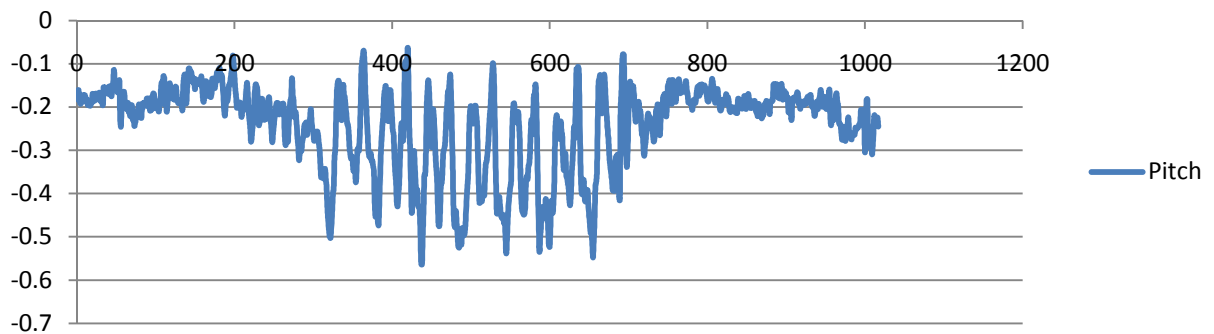


Figure 3-17: Orientation output (around x-axis, i.e. pitch)

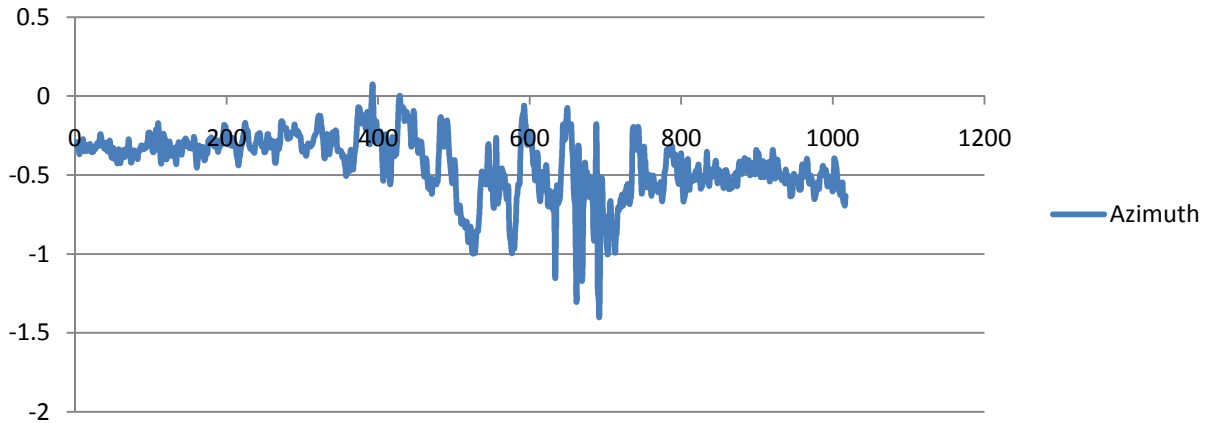


Figure 3-18: Orientation output (around z-axis, i.e. azimuth)

The digital compass uses besides the magnetometer values also the accelerometer values (to know how the device is oriented). Here it is possible to appreciate that the peak values in the accelerometer (steps) affect the calculated orientation in the compass resulting into peak values in the orientation graphs. This is further confirmed after analyzing the gyroscope values during the same test.

3.3.4 Orientation using the gyroscope

Figure 3-19 shows the results of the calculated angle (direction) during the walking test. The initial value of the angle is zero and the current angle α is calculated by adding the sum of all calculated angles changes from initial value (integrating the angular speed), equation 3-3.

The results can be compared to Figure 3-17 above showing the orientation from the compass around the z-axis (azimuth).

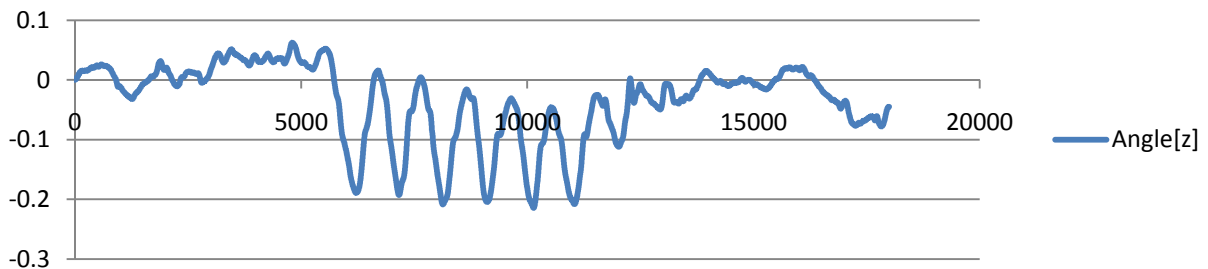


Figure 3-19: Calculated angle around z-axis (azimuth, the walking direction)

Figure 3-20 shows the results of the calculated angle around the phone's x-axis. This can be compared with the pitch output from the compass.

It is noticed that the initial azimuth and pitch is not zero. So a starting value based on the compass values could be added here to the angles calculated by integration of the gyroscope angular speeds.

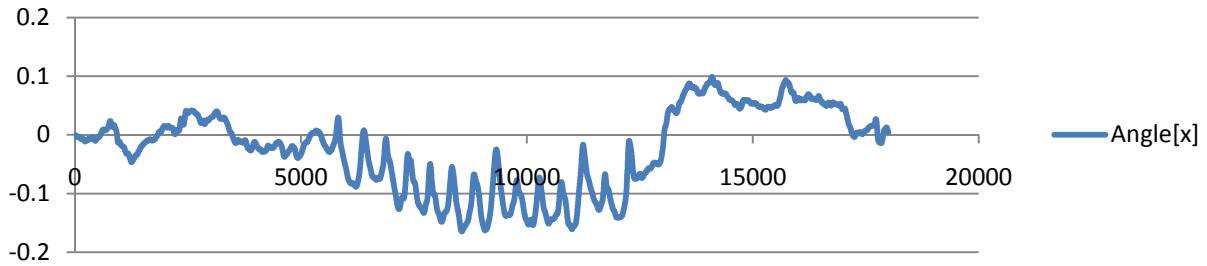


Figure 3-20: Calculated angle around x-axis (pitch)

This results show that angle change calculated by integrating the angular speed looks much smoother than the angle change from the magnetometer due to the peaks caused by the accelerometer use. Angles calculated using the gyroscope may be best fitted to solve the direction problem. However, since the gyroscope introduces a small error which accumulates over time, and needs to be corrected for continuously. A possible approach is to fuse both magnetometer and gyroscope values to get the orientation. More on this later.

3.4 Trolley test

To further study the linear acceleration it was decided to do another test where the readings of the linear acceleration are affected by noise as little as possible. The test was performed in the same way as the walking test, but this time the device was put on a moving trolley.

The goal is to measure linear acceleration without the gravity component for a long period.

The acceleration on the y-axis is measured at rest and an average is made (gravity component at rest). Then during the test the linear acceleration is simply calculated by deducting the gravity component from the acceleration on the y-axis:

The actual test was: no movement during approximately five seconds - walk 15 steps while pushing the trolley – no movement for a few seconds.

The results are presented in the graphs below.

3.4.1 Linear acceleration

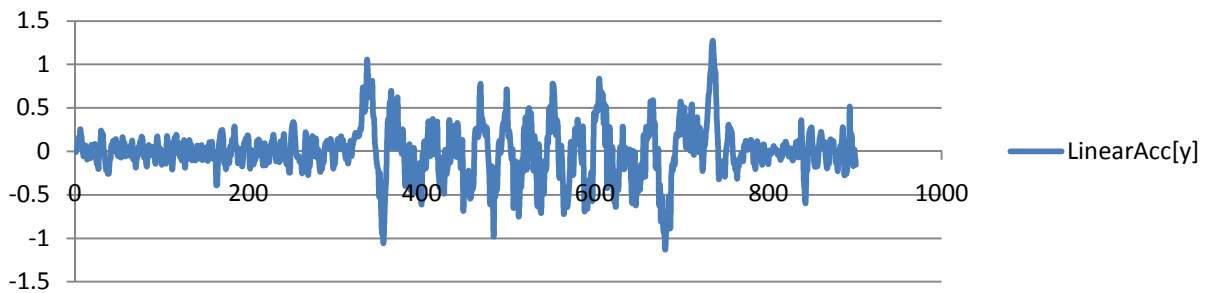


Figure 3-21: Linear acceleration output (y-axis)

As before, the velocity is calculated by integrating the acceleration over time (equation 3-8). Except that now, the linear acceleration is the acceleration on the y-axis minus the average acceleration at rest on that axis (gravity component):

$$LinAcc_y = Acc_y - AvgGRest_y \quad (3-13)$$

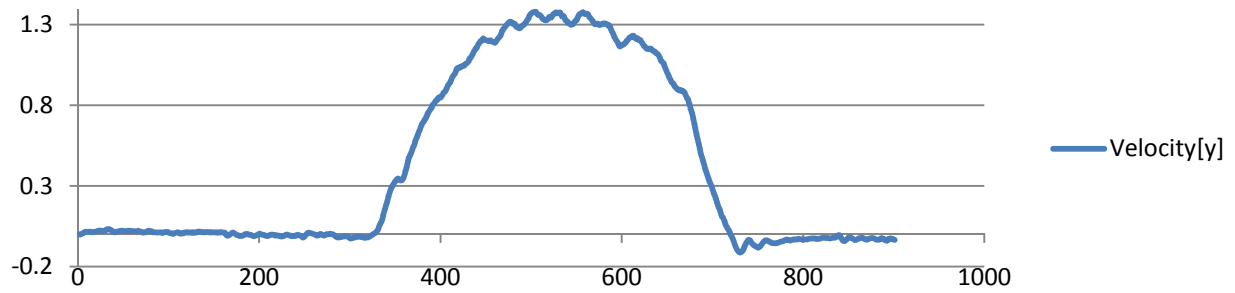


Figure 3-22: Calculated velocity

The distance is calculated by integrating the velocity according to equation 3-9.

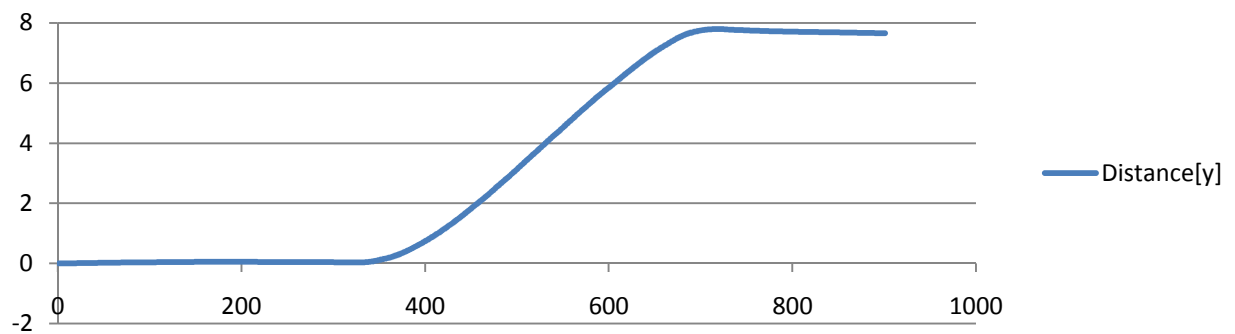


Figure 3-23: Calculated distance

The 15 walked steps are close to 8 meters; the obtained results are fairly accurate.

3.5 Wi-Fi test

3.5.1 Test environment

The image below represents the part of the 5-th floor plan of the Epsilon building used during the testing. The floor contains mainly of offices, conference rooms, electronic labs, a dining room, and toilets. Green marks represent the location where the Access Points are positioned.

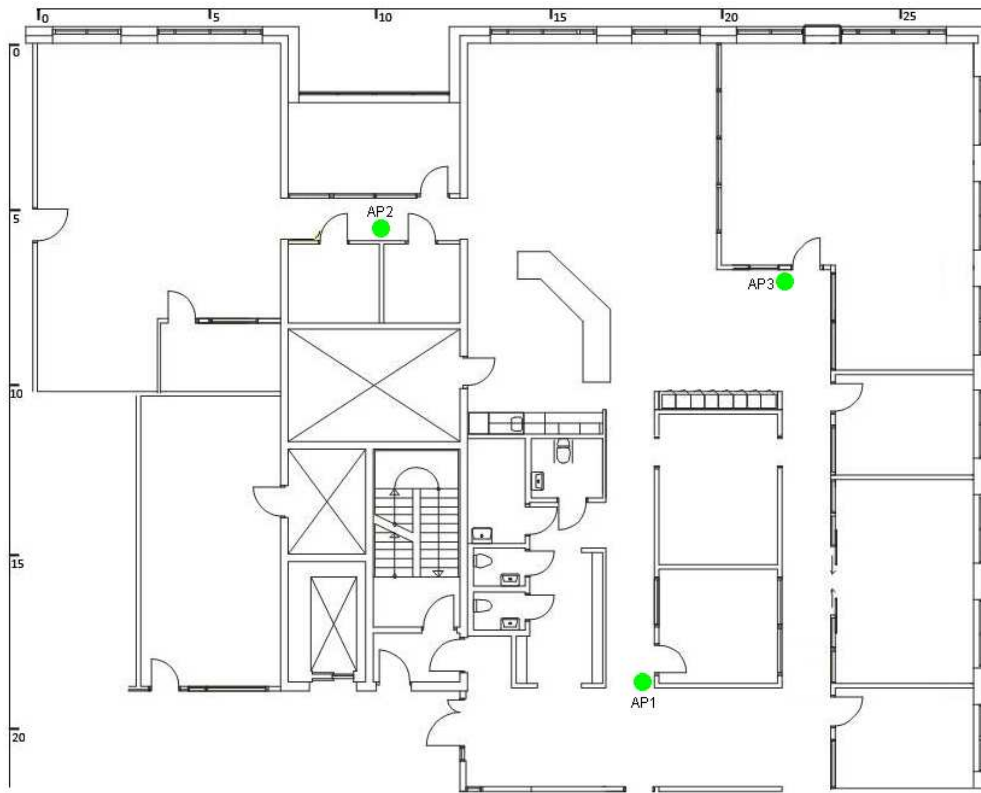


Figure 3-24: Part of the 5-th floor plan of the Epsilon building

3.5.2 Wi-Fi signal

The Android Wi-Fi Manager returns a list of the scan results. Each element in the list includes the following information:

- **SSID** (Service Set Identifier): The network name
- **BSSID** (Basic Service Set Identifier): The MAC-address of the Access Point
- **RSSI** (Received Signal Strength Indicator): The detected signal level in dBm (dBm = the power ratio in decibels (dB) of the measured power referenced to one milliwatt)

3.5.2.1 Signal strength variation over time

Figure 3-25 illustrates the signal strength variation recorded over a period of approximately one minute. The measurement is done with the phone placed on the table.

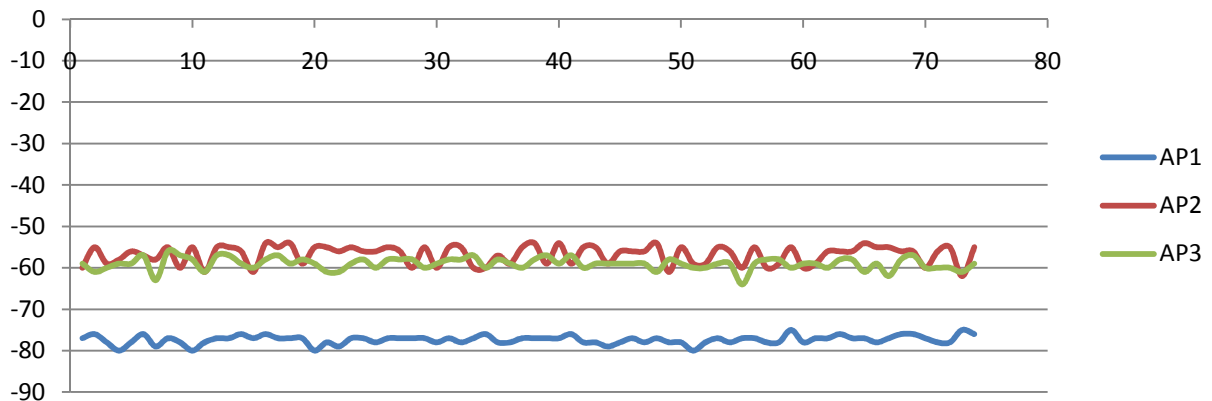


Figure 3-25: Signal strength variation recorded over time

Table 3-7: Statistics of Wi-Fi signals over time

Access Point	Mean	Max	Min	Standard Deviation
AP1	-77,37837838	-75	-80	1,042950277
AP2	-56,82432432	-54	-62	2,247422317
AP3	-59,02702703	-56	-64	1,452187514

The standard deviation decreases when the signal is weaker (AP2 is closest, AP1 farthest).

3.5.2.2 Measured signal strength

Figure 3-26 shows the measured Wi-Fi signal strength for the AP3 (marked green). The measuring is done with the user holding the phone in the hand, and pointing at four different directions (towards each side/wall of the building). The values represent the measuring at each test point (red mark) for each direction respectively.

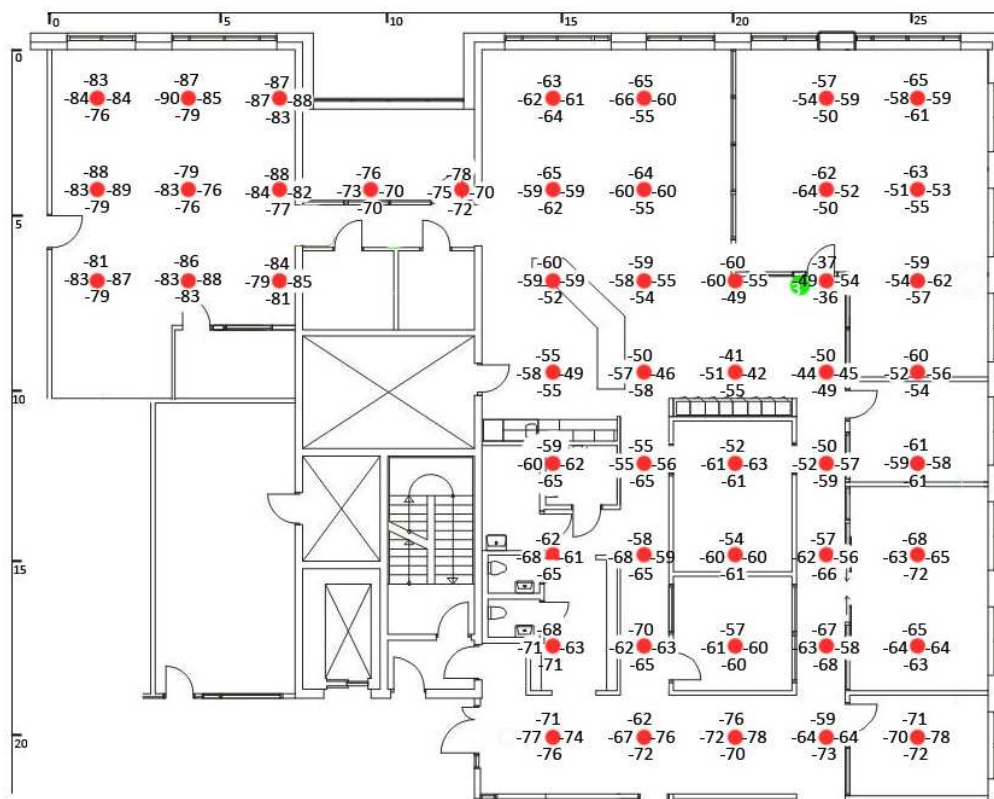


Figure 3-26: Measured signal strength at different testing points/nodes (from AP3)

3.5.2.3 Analysis

Studying the values it can be seen that depending on the user's position with reference to the AP, the signal is stronger or weaker. If the user is positioned between the phone and the AP, i.e. in the line of sight, then the signal strength is generally noticeably lower. Similar results are noticed when there is a wall between the phone and the AP.

It was also noticed that the signal strength varied with the number of people present or moving on the building. In this case the signal is either reflected or blocked, resulting in different signal strength measured in the same point.

The signal strength varies also due to reflections from the walls, floor and ceiling, or other structures in the building [10].

3.6 Bluetooth

In a similar way as the Wi-Fi Manager the Android's Bluetooth Device scans and retrieves a list with the nearby Bluetooth devices. Each element in the list includes the following information:

- NAME (device's friendly name)
- ADDRESS (MAC address)
- TYPE (PC, mobile phone, etc)

The difference compared to Wi-Fi is that the information does not include anything about the signal strength. This information by itself cannot be used to estimate an exact position in the building. However, it can be used to detect in which part of the building the user is located: as the Bluetooth range is limited to max 10 meters, knowing which devices should be visible at each area and the list of visible devices could locate the user at a certain part of the building.

Therefore the aim with the Bluetooth test was to study the scan period. Table 3-8 shows the results (unit seconds).

Table 3-8: Bluetooth scanning statistics

Nr of scans	Average scan period	Max scan period	Min scan period
7	12.0	17.2	7.4

4 Prototype implementation

This chapter describes the implementation of the software prototype in the form of an Android application developed in this project. The description includes the functionality, system design, user interface and the mapping algorithm. Appendix A contains screenshots of the application.

4.1 Functionality

The following functionality is included in the developed prototype:

Data sampling

The sensor's values are sampled and collected during a desired period using the highest sampling frequency for each of the sensors. Then those samples are stored in a file together with extra information like the event time-stamp and processed values (e.g. integrated acceleration). The samples are stored in separate files for each sensor with the purpose of facilitating the data analysis with external tools (e.g. MatLab or Excel).

Display raw data

For each of the sensors the current values are displayed on the screen. This was used mainly on early stages of the development and to analyse the behaviour of the sensors.

Calibration

Before using the application needs to calibrate the sensors. This is done with the user pointing to the defined north of the building and remaining still for a few seconds. The calibration consists in the following: averaging the compass azimuth (to set the stating angle), measuring linear acceleration and gyroscope bias (values at rest) to correct the readings afterwards.

Record RSSI fingerprints

This allows the user to create the RSSI fingerprint records that are used for the Wi-Fi position estimation channel. At the current point the average RSSI value of at least three measurements is saved for every visible AP. The measurements are repeated facing the four directions: north, east, south and west. For each record the coordinates can be introduced via a dialog. The records together with their coordinates are stored in different files for each AP.

Load fingerprints

Here the previously created fingerprint files are loaded into memory. The RSSI record values are placed into matrices to permit fast access time. This step needs to be done before using the Wi-Fi channel for the positioning in the map view. It also facilitates easily usage and comparing different fingerprints sets during the research.

Map selection

The user can select the map structure to be used in the map view. It is limited to Epsilon's office map and the east section in Norra station at the University of Kristianstad.

Display map view

The previously selected map is displayed and the position estimation engine is started with the default channels. The estimated position is displayed as a trace of dots. Being the thickest point the last or current position.

Channel selection

While the map view is shown the user is able to activate and deactivate the different positioning channels/sources merged to calculate the estimated position. The possible sources are: Wi-Fi, step detection + pattern, integrated linear acceleration and integrated acceleration.

Display traces

By default only the merged trace of positions is shown. Via the menu it is possible to make visible other traces of positions calculated using only one source (e.g. only Wi-Fi trace)

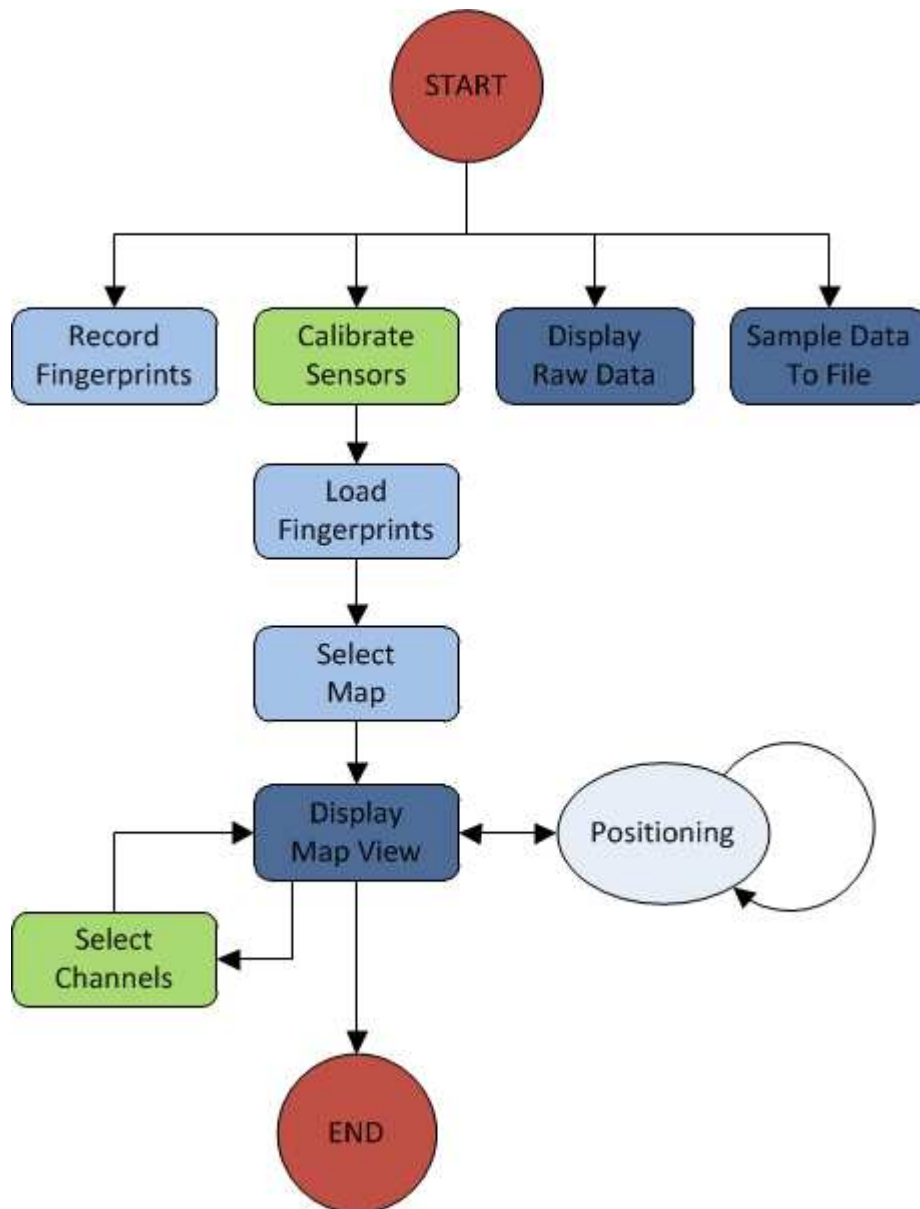


Figure 4-1: Application state chart

The above graphic shows the application state flow chart. It displays the possible states during a normal run and also it specifies the sequences between those states. From start the user can move to the main four functionalities. For the map display and position estimation there are several in-between states that needs to be done for the optimal functioning (e.g. calibration)

4.2 System design

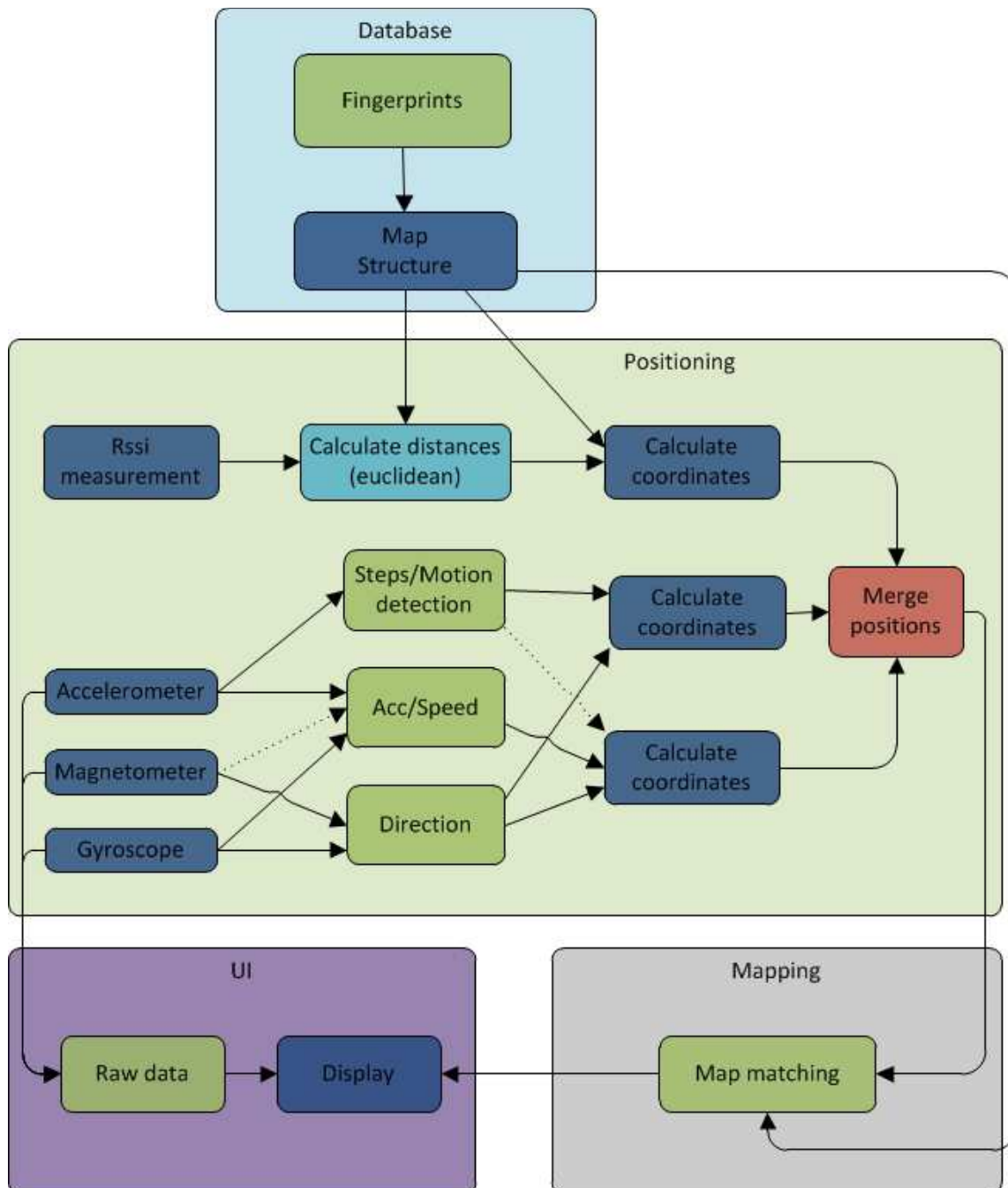


Figure 4-2: System design

The image above shows the application's structure and system design. The application is structured as a single activity with different layouts (views). The main processing line takes care of the sensors reading which is handled by an event listener. When a new value from the sensors is sampled depending on the mode it will be processed by the positioning engine, displayed on the screen or stored in a sampling array.

The UI is handled from an independent thread; the screen updates are controlled by a timer and by external calls from the positioning block. The mode changes are triggered either by the

user interaction (through the menu) or by special event in the block which handles the sensors (e.g. the calibration is completed).

The view's layouts are defined separately in an xml file and are loaded on the mode changes. The source is divided in classes. The main four classes correspond with the four blocks above. Other classes include file management, calculations, etc.

4.3 Mapping

In the mapping process the position returned from the fusion of the INS and Wi-Fi positioning is mapped to a point on the map displayed on the screen (Figure 4-3).

The position is calculated in cm while running the application. To show the correct location on the map shown on the screen, a conversion from cm to pixels is done. The map is an image, so the pixels per meter are measured (17.1 at the Epsilon building). Then the coordinates are multiplied by this factor and the map offset is added (the map offset is the coordinates on the screen where the map image starts).

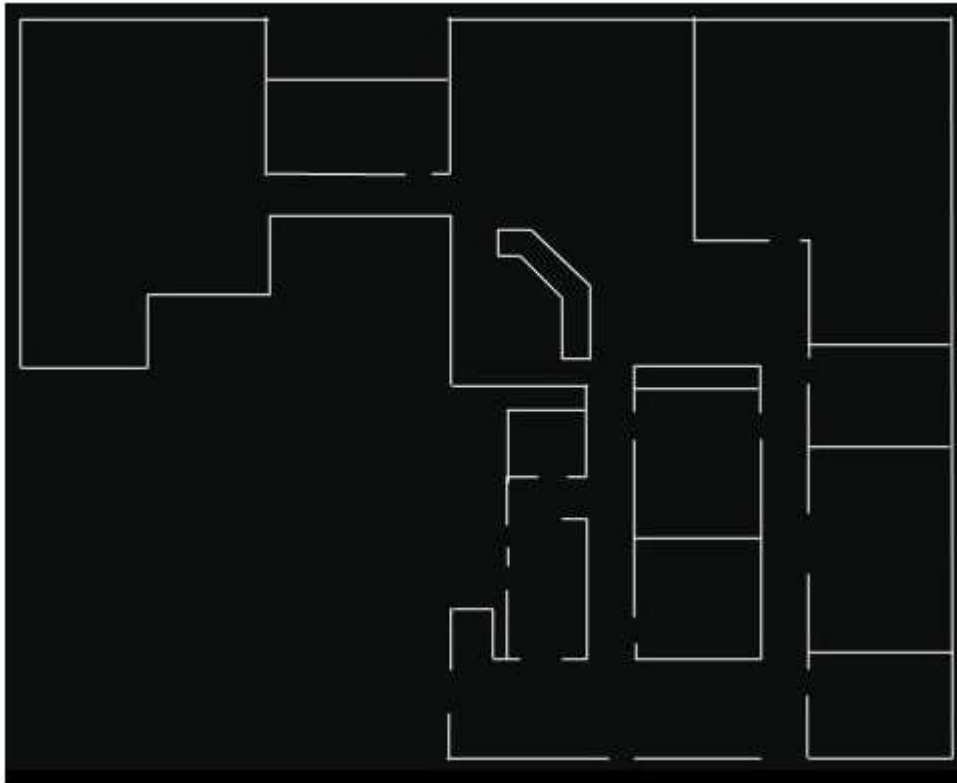


Figure 4-3: Epsilon map as displayed on the smartphone

5 Choice of solution

This chapter describes the methods considered and used in the attempt of achieving the main purpose of the project, i.e. to examine the results obtained in indoor positioning using available sensors in the smartphone.

First the identified challenges are presented together with description of available methods of how/if the challenges can be overcome. In the second part independent tests are done: position is estimated using inertial sensors and Wi-Fi separately in order to analyse the results and accuracy of each sub-system independently.

5.1 Challenges

5.1.1 Dead reckoning

Dead reckoning is the process of estimating one's current position based upon a previously determined position, and advancing that position based upon known or estimated speeds over elapsed time, and course [24]. When the direction is known and the occurred movement is detected, the new position is calculated based on an assumed or estimated speed.

The speed could be determined considering the accelerometer variations and/or a walking pattern of the user. The direction can be detected using the accelerometer, gyroscope (and magnetometer).

Due to the complexity of the human walking pattern and his freedom of movements, to assume a moving speed has as a result a high error rate in the new estimated position. Even if it is assumed that the detected direction is free of errors (which in the case of using only the magnetometer are very dependent on disturbances).

A disadvantage of dead reckoning is that since new positions are calculated only from previous positions, the errors of the process are cumulative, so the error in the estimated position grows with time. That's why just by themselves they are not an appropriate solution unless combining dead reckoning estimation with other channels estimation to correct and reduce the introduced error in the estimated position.

5.1.2 Calculate speed and distance

Physically, an accelerometer measures acceleration, or change in speed; speed is in its turn change in position. Mathematically, the integral of acceleration is speed, and the integral of speed is distance (or location). In general, the integral of accelerometer measurements can be obtained by simply adding up the samples. For instance, starting from standing still and accelerating at a constant speed of 1 m/s^2 , means that after one second the speed will be 1 m/s . After two seconds the speed will reach 2 m/s and so on. The acceleration goes from 0 to 1 m/s^2 during that first second. Assuming constant acceleration the traveled distance is 0.5 meters. In the 2nd second the acceleration goes from 1 m/s^2 to 2 m/s^2 , and 1.5 meters is added to the traveled distance, and so on. It is pretty straight forward to calculate this. The graph below illustrates the effect of constant acceleration on speed and position over time.

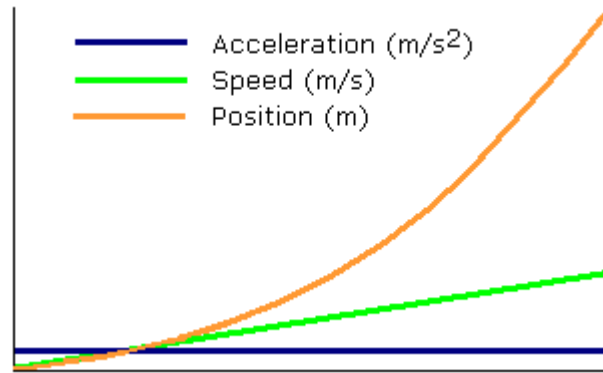


Figure 5-1: The effect of constant acceleration on speed and position

So far so good, but what makes things difficult is the gravity, which is always trying to pull everything towards the center of the Earth at $1G (= 9.8 \text{ m/s}^2)$.

The accelerometer will measure the gravitational force while the device is at rest. On the other hand, dropping the device off a cliff, besides from the air resistance slowing it (and the effects of spinning) the device will experience no noticeable acceleration as it falls and is accelerated at 9.8 m/s^2 . How is this possible? While stationary the accelerometer will measure acceleration, and while it is changing speed as it falls down it will measure nothing? This goes against the idea that the accelerometer measures change in speed. Even if the accelerometer will measure zero acceleration, of course the speed and position will change as it is falling towards the ground.

This can be adjusted for mathematically by subtracting out the effects of gravity. If the device is on Earth and the accelerometer reads $0G$, it can be assumed it is accelerating at 9.8 m/s^2 . In the same way, if it reads 9.8 m/s^2 down, it can be assumed it is stationary with gravity pulling it down. So we just subtract 9.8 m/s^2 out of the acceleration output.

But the problem with subtracting gravity is knowing which way is down. This can be achieved if the device's rotation is known (how it is tilted). To calculate the rotation when the device is not accelerating or is at rest is quite easy; however, when accelerometer output is affected by other forces of motion and is accelerating sideways (i.e. when carrying the phone while walking), the rotation calculations become much more complicated.

This brings us to centrifugal force. If we tie the device to the end of a string and swing it around in a circle, the accelerometer will measure acceleration of $1G$ down, and many G s away from us. The force applied from the string to turn the device around shows up in the accelerometer. This is centrifugal force that is basically fighting the device's natural tendency to want to continue in a straight line ("bodies in motion tend to stay in motion..."). From the accelerometer readings, the device is unable to tell the difference between this spinning and true acceleration in a straight line.

The same applies when the device is turned while held in the hand and walking; with the accelerometer only it is impossible to distinguish this type of acceleration from the type that causes its position to change (linear acceleration). Gyroscopes, which measure orientation, can help sense out this effect and subtract it from the calculations.

Error accumulation

To calculate position from acceleration a double integration is required. This means that encountered errors will accumulate. If the accelerometer readings are off even a little, the calculated speed will be off, which will make the position even further off for that estimate and all future estimates. The next reading may also be off slightly and get another slightly

incorrect speed which will then be used to calculate a distance which is applied to the already incorrect position to get a new position that will now be even further off.

The AD converter that reads the accelerometer is reading instantaneous values from the accelerometer. The software then must “interrupt” between these readings by assuming a relatively smooth change from one reading to the next. The available sample frequency on the device is limited, so it is certain that some acceleration will be missed, and then the readings will be off. Additionally, noise will also affect the accuracy of the results.

Since these errors are cumulative, it will not take long for the position calculations to be much off. Therefore, to *accurately* calculate distance from the accelerometer only is very difficult.

To overcome this problem and *estimate* the position some assumptions must be made. For instance:

- the orientation of the device does not change: rotational changes would not be mixed with the “true” (linear) acceleration;
- changes in acceleration are somewhat smooth: to reduce the sampling error
- continuously get an absolute reading (Wi-Fi): to adjust for the cumulative errors

5.1.3 Determine direction

Previous researches about pedestrian navigation systems show that one of the main sources of error in position comes from the errors in the determination of the azimuth of walk [3].

To determine the direction (orientation), a normal approach is to use the functionality of a magnetometer. The magnetometer provides to the smartphone the ability to calculate the magnetic azimuth (angular distance from the magnetic north).

The problem is that magnetometers are sensitive instruments and that makes them very vulnerable to magnetic disturbances in the environment when measuring direction. The magnetic disturbances are especially present inside buildings.

Gyroscopes on the other hand, which are also used to measure orientation, are not sensitive to magnetic disturbances. A gyroscope measures angular speed, which when integrated over time gives us the angle (relative to a starting direction).

The disadvantage with gyroscopes is that the output drifts over time (which is caused by the vibrating parts in the gyroscope [4]). If not compensated for, the drift error accumulates and the gyroscope output is useless.

Compasses and gyroscopes are often used in combination with each other. The ideal usage in an integrated system is when the gyroscope can correct the magnetic disturbances, and the compass can determine and compensate the bias of the gyros and the initial orientation.

Because of the difficulty in differencing the changes in the magnetic values caused by disturbances from the ones of the movements, a solution which integrates the use of the gyroscope is proposed. The advantage of one sensor can compensate the drawback of the other to increase the reliability of the measured direction.

5.1.4 Wi-Fi positioning

Wi-Fi signals are RF (radio frequency) signals. The RF signal propagation is strongly influenced by the medium encountered by the signal wave while expanding. Indoor environments are a very clear case of multipath environments (a term used to describe real world conditions) where the RF signals don’t emanate as a single wave, but as an expanding wave front.

At start a uniform wave front leaves the transmitting antenna (AP). Obstacles encountered by the wave front may create new RF signals. Some components will travel straight to the receiving antenna while other components could diffract, scatter or be reflected by obstructions.

- Diffraction: the radio waves are bent around sharp objects creating a new wave front.
- Scattering: RF energy is reflected by a non-uniform surface in multiple directions.
- Reflection: the wave contacts a uniform surface and is reflected with a predictable angle.
- Multipath fading: occurs when receiving the superposition of multiple copies of the transmitted signal, each traversing a different path. The results can be constructive (amplified) or destructive (attenuated) interference at the receiver.

These elements can have either positive or negative impact in the received signal; increasing or decreasing the RSSI measured value. For example without diffraction, scattering and reflection the signal cannot reach some places in indoor environments.

Previous research in wireless positioning show that Wi-Fi-enabled devices can be located by applying one of two types of location-sensing techniques, propagation based [10] and location fingerprinting [9]. Propagation-based techniques measure the received signal strength (RSS), angle of arrival (AOA), or time difference of arrival (TDOA) of received signals and apply mathematical models to determine the location of the device. The drawbacks of propagation-based method are the need to compute every condition that can cause wave signal to blend in order to achieve accurate localization.

Location fingerprinting (LF) is a two-phase process. First, a radio map of observed signal strength values from different locations are recorded during an offline calibration phase. Then, in real time, signal strength values observed at the user's mobile device are compared to the radio map values using proximity-matching algorithms in order to infer current user locations [9].

As the application environment of this research is indoor buildings (clearly a multipath environment) a solution using fingerprints appears the most suitable. An approach that uses a mathematical model based on Euclidean Distances for measuring likelihood between two set of signal values was tested. The Euclidean model is considered being one of the most widely approach according to previous research, as it gives very good results with a relatively low computation load. This last point is of great importance in this case as the measured signal is compared with every fingerprint in the database records. Each fingerprint consists in several signal values (one for each AP). Due to the high repetition frequency, low computation is necessary to permit parallel processing of the other signals (accelerometer, gyroscope and magnetometer).

Two problems with the LF approach are identified:

- Signal strength variation (see Figure 3-25)
- Changing environment: movement of people, varying arrangement of furniture, etc

Consequences of the above will result in different values obtained during the (offline) recording of the signal strength values compared to those observed during real-time running of the application. Averaged signal strength values can be used reduce the problem.

5.1.5 Bluetooth

As it was stated during the Bluetooth test in chapter three, because of the Bluetooth scan does not include any information about the signal strength, there is no possibility to estimate an exact position in the building.

Although the information can be used to detect and locate a user within a larger part of the building, because of the relative long scan period, that information will only be valid during a short time. The Bluetooth scan process is very costly (in CPU terms) and it takes in average 12 seconds per scan. As the distance the user can move in 12 seconds can be relatively high, even the obtained area location could be inaccurate.

The facts above verified that Bluetooth did not add any value to the project, it was therefore decided to exclude it from the sensors to be used in this research.

5.2 Estimated position using INS sensors

As the linear acceleration calculated by Android was not perfect, and seemed to be affected by the noise the steps introduce and the errors from the magnetic interference on the magnetometer (Android makes use of accelerometer and magnetometer to keep track of the orientation), the alternative was to try to calculate the linear acceleration using gyro and accelerometer.

It was found that the main issue here is to get rid of the *gravity component*.

To begin with, the initial orientation angles are needed; this can be calculated calibrating the device completely at rest measuring the gravity in each axis.

Next, the exact orientation angles are needed; this can be obtained from integrating the gyro. But the gyro accumulates an error over time (see Gyroscope section in chapter 3) which makes the results differ very much from the reality after a while.

Finally, it is needed to keep track of the gravity component (i.e. know which way is down) “online”, as the phone is held in the hand and the user is walking. This part involves somewhat more complicated calculations; it was appreciated to be unfeasible to be implemented correctly within the time scope of this project.

It was decided to go ahead with another test using the trolley. The gravity component was compensated for (see equation 3-13) and it was assumed that “down direction” does not change (device is stationary on the trolley). The results are illustrated in Figure 5-2. The green line is the actual path and the blue dots represent the estimated position.

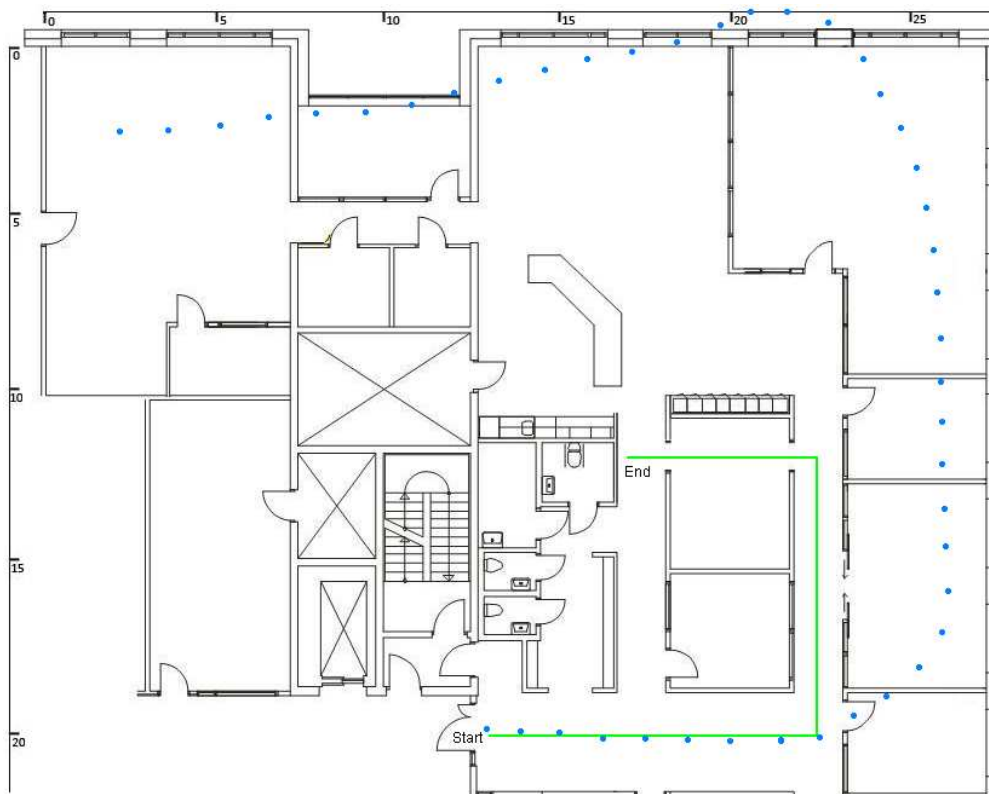


Figure 5-2: Estimated position using inertial sensors only (device on trolley)

The velocity (integral of acceleration) gets too high, and after a few seconds the position estimation is too much off from the reality.

Results improved if the speed was limited (see improvements below), but since it is not very useful to have a solution where the phone has to be on a trolley all the time (not real life), the decision to make use of the Androids linear acceleration was made with the argument of being the most stable although not very precise.

To minimize the errors caused by double-integral and make the linear acceleration have more useful results, following improvements suggestions were tested:

- Step detection: An algorithm that detects steps by analyzing the step pattern on the total acceleration and the gap between peaks. The steps are used to calculate an extra positioning source that uses a step pattern (length) and the current orientation. See next section for details.
- Motion detection: An algorithm that checks for peaks of the total acceleration is the last 15-35 values; when there is no movement the location is not updated (except correction from Wi-Fi when fusion is made). See next section for details.
- Limiting the walking speed: The accelerometer inaccuracy sometimes resulted in a too high calculated velocity (integrated acceleration); a limitation to maximum 1m/s was done. So the speed varies between -1m/s and 1m/s. Without the speed limitation, the integrated velocity gets too high and after a few seconds the position estimation is too inaccurate (out of the map).
- Reset speed when stopped. When the above algorithm indicates that we are not moving anymore the velocity is reset to 0. And then it starts to accumulate again when movement is detected.

5.2.1 Step and motion detection

The graph in Figure 5-3 shows the total acceleration measured by the accelerometer while walking. After several tests it was observed that every step was introducing a significant increase in the acceleration during a short instant. These acceleration increments are in the form of sharp peaks in the graph. An approach to identify peaks in the acceleration was taken. A peak is detected when the acceleration tend changes from incrementing to decrementing. The arrows in the graph (both blue and red) identify peaks. However a step normally introduces a peak that reaches at least 13m/s^2 , so a threshold of 12.5m/s^2 was introduced to discard small peaks and also be sure not to miss any step. Another phenomenon which was observed is that sometimes one step could generate more than one peak. This issue was solved adding the requisite that the previous step should have happened at least 350 ms before.

The same flow control of the total acceleration is used to detect motion. When during a period of at least 450 ms (25 samples) no peaks are detected, then the current state is set to no-motion. As soon as there is a peak the state is immediately set back to motion.

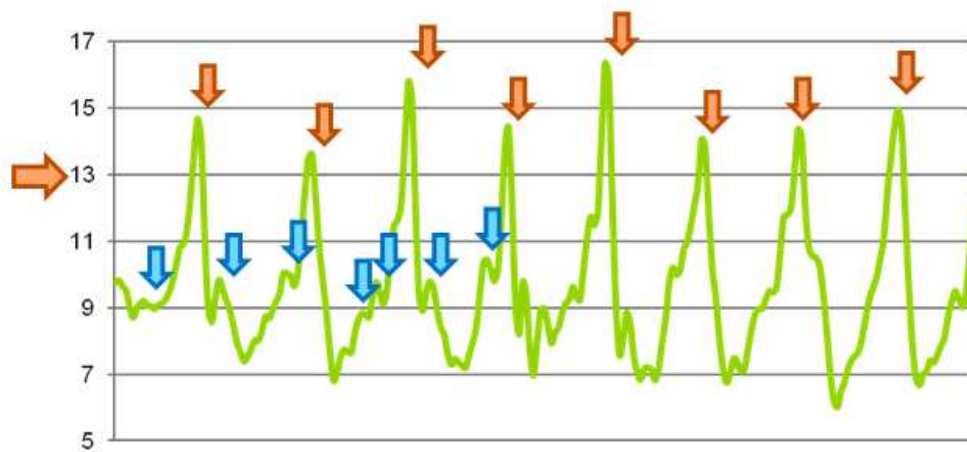


Figure 5-3: Step and motion detection

The two described procedures result into an acceptable performance. However possible improvements can be done, especially in the detection of false positives (steps or motion detected when the user is not moving but shakes the device).

A walking test was done where the position was estimated using the inertial sensors and the algorithms described above. The results are illustrated in Figure 5-4.

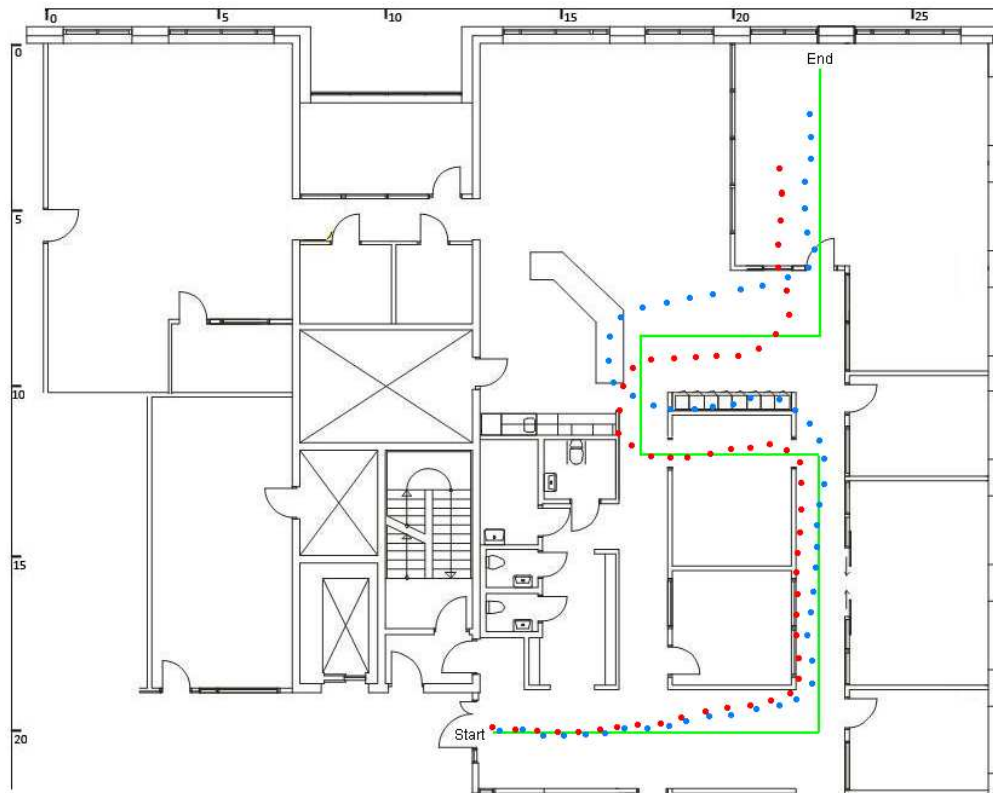


Figure 5-4: Estimated position using inertial sensors (walking test)

Green line is the actual walked path. Blue dots represent the position estimation using integration of linear acceleration returned by Android (speed limited to max 1m/s, no movement = 0m/s). The red dots show the position estimation using the step detection algorithm (only).

As can be seen, with the speed limitation, the results are not too far from the reality. Of course, the results will deviate from the reality as the actual speed exceeds the limitation.

Even though the step detection algorithm alone seems to give best results in this particular test, one should keep in mind that the results will get worse with time, as the stride (step length) varies, and the detection of false steps increases.

The facts mentioned above further motivate the need to continuously adjust for the errors by updating the position from more long-term reliable channels.

5.3 Estimated position using Wi-Fi

Due to the observations made during the tests (previous chapter), and studies done previously it was considered unfeasible to implement signal propagation model indoors. An approach using the fingerprints was tested. The idea is to measure the signal in specific points on the floor in a grid like pattern with the device pointing at four different directions. The grid and the measured signal strength (of AP3) for each direction can be seen in Figure 3-26.

The measured values, together with the corresponding point for the actual measurement expressed in x/y-coordinate system, are stored into a .txt file. When the testing software is executed the file is loaded in a database. As the software runs the Wi-Fi is scanned with on average 500 ms interval and the measured signal strength is compared to the records in the database. The fingerprint points with the most similar values to the current measurement are then selected to estimate the current coordinates of the phone's position.

A common approach to determine how much a set of measured signals resembles a set of reference signals is to use the Euclidian distance.

5.3.1.1 Euclidian distance

The Euclidean distance or Euclidean metric is the "ordinary" distance between two points that one would measure with a ruler, and is given by the Pythagorean formula.

The Euclidean distance between point p and q is the length of the line segment connecting them [21]:

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^n (q_i - p_i)^2} \quad (5-1)$$

In our case q is the measured "point" and p is the reference "point"; n is the number of APs, and q_i and p_i are the measured and reference values respectively (in this case RSSI).

The calculation in (4-1) is done as many times as the number of nodes.

The x and y coordinates of the phone's current position are estimated by calculating the weighted mean of the coordinates of the *closest* m nodes (fingerprint points) [21]:

$$x = (\sum_{i=1}^m 1/d_i)^{-1} * [\frac{x_1}{d_1} + \frac{x_2}{d_2} + \dots + \frac{x_m}{d_m}] \quad (5-2)$$

$$y = (\sum_{i=1}^m 1/d_i)^{-1} * [\frac{y_1}{d_1} + \frac{y_2}{d_2} + \dots + \frac{y_m}{d_m}] \quad (5-3)$$

where x_i and y_i are the stored coordinates of the selected fingerprint points, and d_i is the distance (to each fingerprint) calculated using Euclidian method according to equation 5-1. The m is the number of chosen nodes. The chosen nodes are the nodes with the lowest values of calculated d (i.e. closest to the current measured position).

A walking test was done where the position was estimated using the algorithm described above. Four nodes are chosen to calculate the estimated position. The result of the test is illustrated in the Figure 5-5. The green line represents the actual path and the blue dots represent the estimated position.



Figure 5-5: Estimated position with Euclidian distance (instantaneous Wi-Fi readings)

As it was seen earlier (Figure 3-25), the signal strength is not constant at a fix point; it varies over time. The fingerprint values taken for this test are only one epoch; as a result, two instant measurements at the same point can differ so that the maximum potential error is:

$$\text{error}_{\max} = |\text{signal}_{\max} - \text{signal}_{\min}| \quad (5-4)$$

The simple application of position determination based on Euclidean distances leads to feasible results without the need for extended computing capacities. If records of more than one epoch are available it is possible to determine supplementary information like the average, median, maximum, minimum or the standard deviation of the signal strength measurements. It is possible to improve the results with a fuzzy logic based approach [7].

By using the average signal strength value at each point the *potential* error is reduced according to:

$$\text{error}_{\max1} = |\text{signal}_{\max} - \text{signal}_{\text{average}}| \quad (5-5)$$

or:

$$\text{error}_{\max2} = |\text{signal}_{\text{average}} - \text{signal}_{\min}| \quad (5-6)$$

The effect is illustrated in the Figure 5-6.

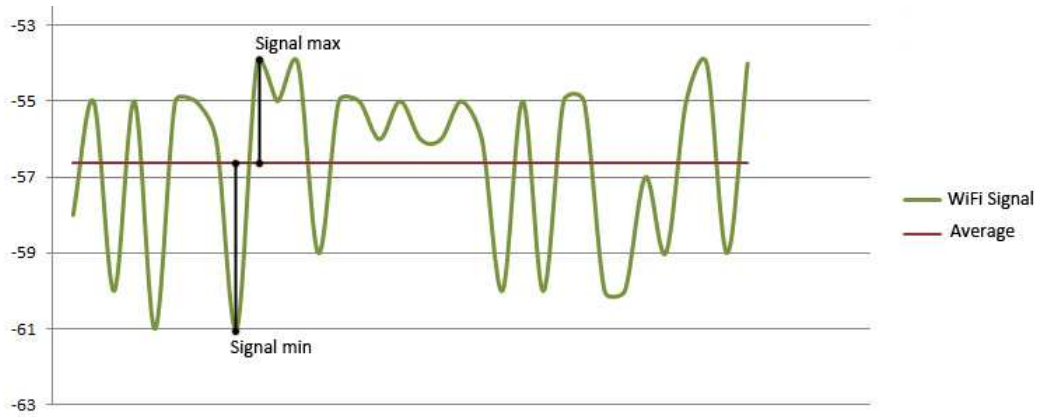


Figure 5-6: Potential error in measurements using the average signal strength

Another test was performed, now with the average approach. At each test point at least three values per AP in each direction were taken and averaged. The results, presented on the Figure 5-7, show a quite good improvement compared to the previous test. Because of the improved results, the average approach was a candidate algorithm to be implemented for the demo application.

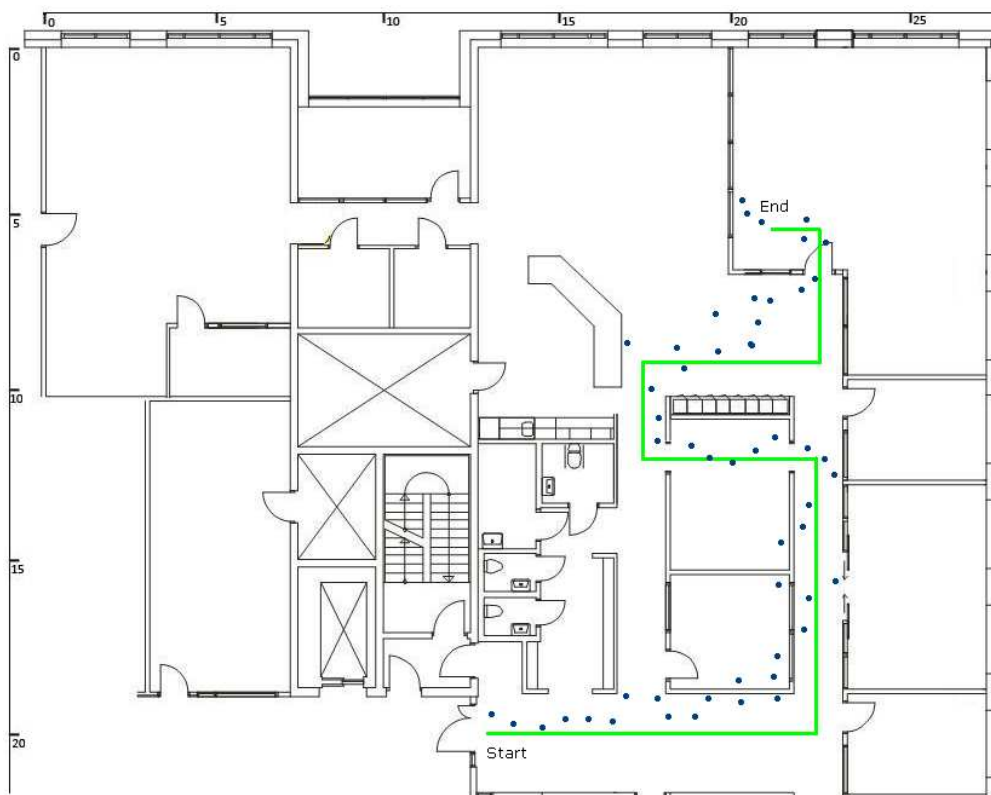


Figure 5-7: Estimated position using Euclidian distance method (averaged Wi-Fi readings)

6 Final results

This chapter presents implementation of the sensor fusion model, and presents the final obtained results. The level of accuracy acquired is quite satisfactory. The average deviation between the estimated and real position was less than two meters.

6.1 Sensor-fusion - integration of INS and Wi-Fi

There are numerous sensor fusion algorithms. The Kalman filter is one of the most mentioned and used in real time systems where multiple sequential measurements (rather than a single measurement) are produced to describe the state of the system.

Kalman filters are often used with motion dynamics. If the way the device is moving is known Kalman filter is a powerful tool to model the dynamics. For instance car movement is very constraint, i.e. it does not accelerate straight up or down. The filter is then used to constrain the error using a model of how the object moves.

To build and implement such model with Kalman filter is considered out of this project's scope, and is left as a suggestion for future improvement.

An approach using the average of the sub-systems was tested. All information channels were evaluated. Tables below show pros and cons of the information sources available and the additional methods identified during the tests so far.

Table 6-1: Pros and cons of the information sources

Information channel	Pros	Cons
Linear acceleration	Acceptable accuracy at lower velocity	Bad accuracy at high velocity
Wi-Fi	Acceptable estimation when averaged	Jumpy at instant readings
Step detection	Short term accuracy	Detecting false steps; Pre-defined step length
Orientation using magnetometer	Absolute azimuth Long term stable accuracy	Unpredictable external disturbances
Orientation using gyroscope	No external disturbances Short term accuracy	Relative azimuth drift
Motion detection	Position is not updated when not moving	Last estimated position could be incorrect
Speed limit	Increases linear acceleration accuracy at lower velocity	Losses when really moving above the limit
Speed reset at stop	Not affected by false accelerations (device tilted)	Speed not updated when walking really slowly

First, for the position estimation the average value of all three sources (linear acceleration, Wi-Fi, step detection) was used. The test results, shown on Figure 6-1, were acceptable, but did not produce the desired results; position estimation did not seem to improve when Wi-Fi readings were added. This was caused by the effect of the sometimes “jumpy” Wi-Fi signal strength.

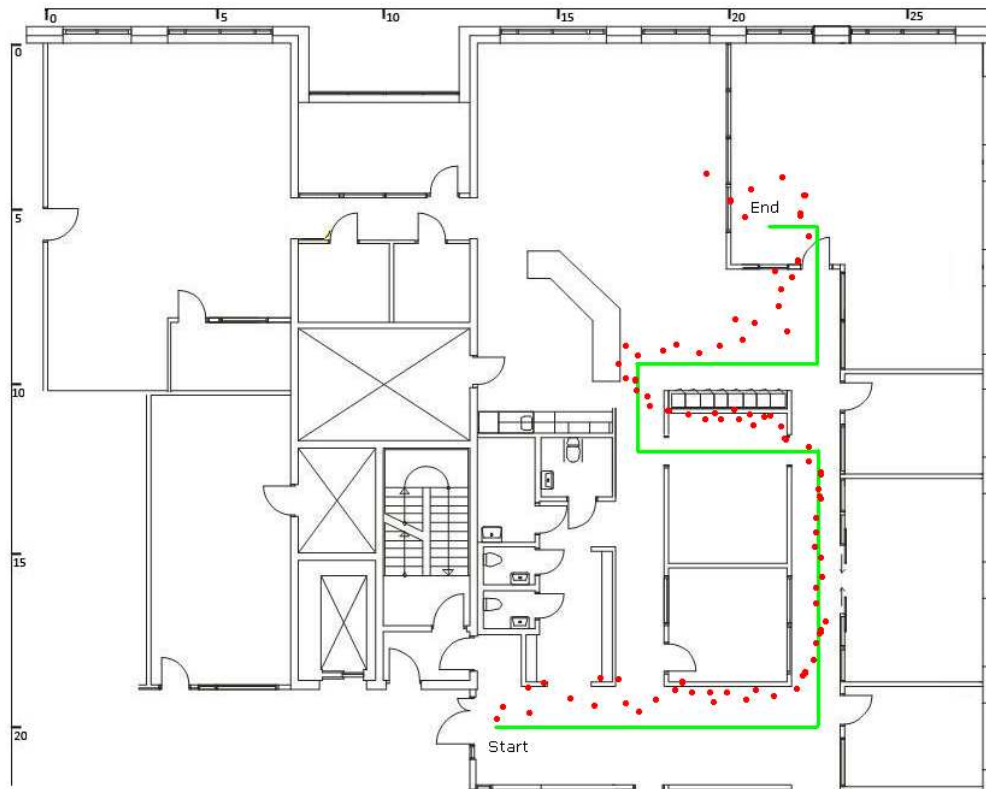


Figure 6-1: Estimated position with average of linear acceleration, step detection and Wi-Fi

It was then decided to add the following in an attempt to further improve the results:

- Amplify the currently most stable sub-system by using a weighted system; for instance, if the Wi-Fi signal is jumpy, decrease its “reliability” (multiply values with a factor smaller than 1).
- Redo the motion detection algorithm: when no motion is detected by the inertial sensors, the position should be updated using the average of the last Wi-Fi readings.

This resulted in the final sensor-fusion model illustrated in Figure 6-2.

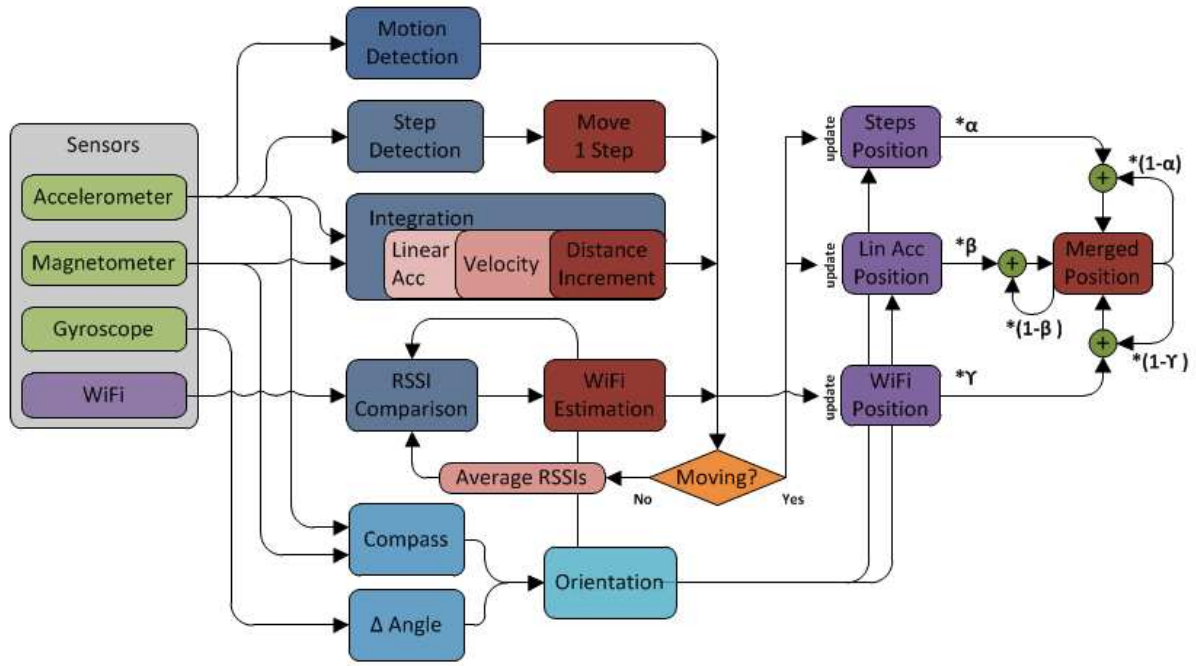


Figure 6-2: The sensor-fusion model

The graph describes the sensor fusion design detailed and how the merged position and independent channels are updated. The two inertial channels (linear acceleration and steps detection) are only updated when motion is detected. When no motion is detected the Wi-Fi channel updates the position estimation using the average of the RSSI measured since stopped. This last process functions as a correction of the off error introduced while moving. The Wi-Fi channel gets the new estimated position directly from the fingerprint comparison algorithm; it uses the current orientation to select the set of fingerprints to compare. The other two channels calculate a distance increment and calculate the new estimated position as follows:

$$X_i = X_{i-1} + d * \sin(\alpha - bearing + \pi) \quad (6-1)$$

$$Y_i = Y_{i-1} + d * \cos(\alpha - bearing + \pi) \quad (6-2)$$

In the above formulas d is the distance increment (the step pattern for the step detection channel). The current orientation angle is α , and bearing is the orientation of the building. The correction of π radians to the angle is made because the coordinates increase down and to the right. The convention taken is that the (0,0) point is on the upper left corner of the map.

The orientation is calculated fusing the compass (which uses the magnetometer and accelerometer data) together with the integrated gyroscope data. As the compass is much more unstable its output is only used for the angle initialization and for correcting gyroscope drift. Being the gyro's integrated data used to calculate the angle change. Every minute, if the compass azimuth is stable (standard deviation lower than 0.10 in last 300 readings) the angle is reset. Because the Nexus S compass doesn't work as good as in other smartphones this fusion can sometimes (especially in areas with big magnetic turbulences) cause the drift to increase.

Each of the channels fuses themselves to get the merged position. A weighted average is used with different weight factor for each channel. As the Wi-Fi is more unstable and updates with a high frequency a low weight factor is used to merge (0.05). The inertial channels both use the same weight (0.5) as the entire test set done showed a similar accuracy for both channels. To balance the update frequency, the linear acceleration channel only merges every 1 second (50 samples).

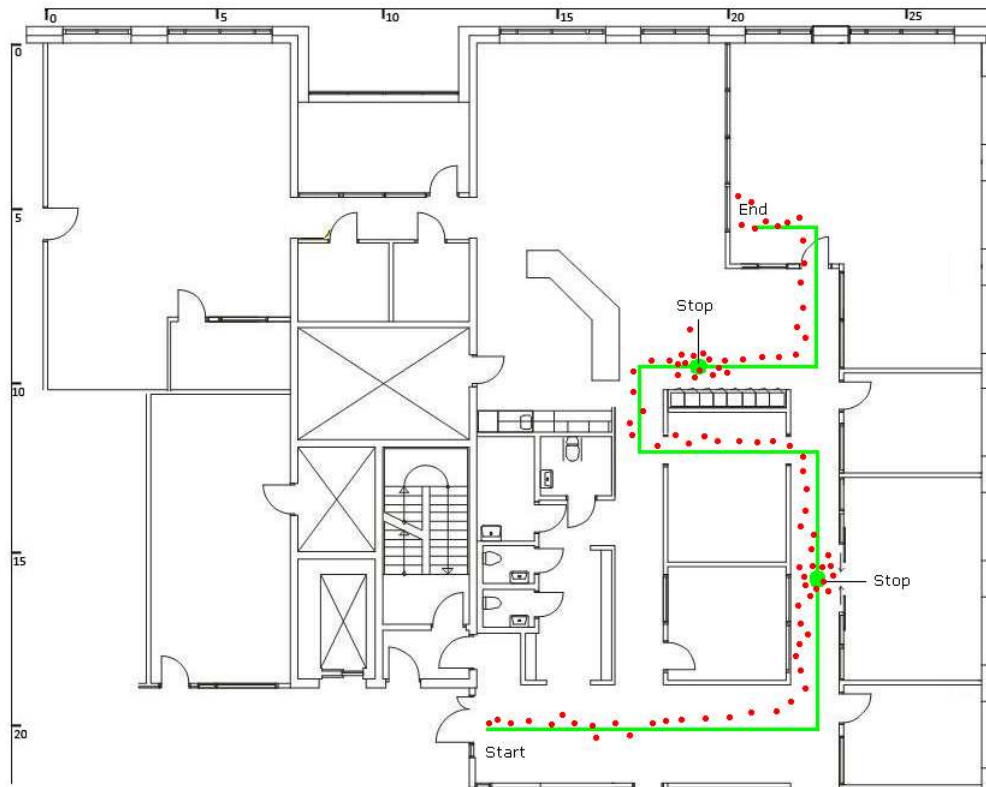


Figure 6-3: Estimated position using weighted average, Wi-Fi update at stops

As can be seen in Figure 6-3 the results improved significantly. On the path, two stops are made to let the position be updated by Wi-Fi only. The position is updated based on the average of the last 5 readings of the Wi-Fi signal. When motion is detected again the position is estimated using the information from all channels as before. The average distance between the actual path and the estimated position is now below meter. For indoor navigation the achieved accuracy is sufficient.

7 Conclusions

The main objectives of this thesis, as described in the beginning of the report, were: research of the area; examine the level of accuracy that can be achieved in indoor positioning by using available sensors in an Android device; implementation of a prototype software, and research of possible applications of the technology.

In the research stage possible solutions were identified. Previous research in the field of indoor-positioning showed that the problem had different approaches and solutions. The common factor of them was the need of external infrastructure (Wi-Fi, GSM network, infrared) to adjust for the cumulative errors from the built-in sensors (accelerometer, gyroscopes, etc) to estimate the position.

In order to meet the second objective different approaches were considered. It was decided to implement a system where advantages of each technique could be used. In the INS sub-system the linear acceleration is double-integrated to calculate the distance, and the gyroscope keeps track of the orientation. These information sources combine the information from previous samples to estimate the position. To reduce the effect of the errors the linear acceleration method was complemented with additional functions: movement detection, walking speed limit and step detection. This improved the INS position estimation significantly.

The INS sub-system was integrated with signal strength measurements of the Wi-Fi, which was used to estimate the position based on the known locations of the wireless Access Points. The measured signal is compared to the previously stored fingerprints in the database. The fingerprint points with the most similar values to the measurement are selected to estimate the current position.

Because of the Wi-Fi signal strength variations, sometimes the estimated position was jumpy. To compensate for this, a weighting system was tried, where the most reliable and stable source was “promoted”. Another feature which improved the accuracy was added: when no motion is sensed by the inertial sensors, the position is updated by Wi-Fi only; last Wi-Fi readings make an average and update the position. When motion is sensed again the position is estimated using all channels as before.

The prototype software was implemented in Java on the Google Android OS. It holds the fingerprint database for the map structure, takes samples from the accelerometer, magnetometer and gyroscope, scans the wireless network, calculates and merges the coordinates from the different sub-systems, does the map-matching and displays the results (estimated position) on a map on the screen.

The results showed that the sensor fusion positioning technique accomplished an average deviation between estimated and real position of less than two meters.

The amount of research in the field shows that this type of navigation technique has various potential application areas, such as shopping malls, large hotels, museums, etc. It is considered a high probability that a similar system can be developed into a commercial product.

8 Recommendations for further work

There are a number of different aspects of further developing this project. The most significant is to increase the accuracy of the position estimation. That can be approached in various ways:

- Improve linear acceleration calculation by using rotation matrix calculations;
- Implement a model with Kalman filter to integrate the information from various sensors.

Kalman filter's recursive nature of error estimation can be used to improve the accuracy of the position estimation. With increased computational power on handheld devices, a propagation-based technique could be implemented to determine the position of the device using the wireless network.

References

Books

- [1] Meier, R. (2010). *Professional Android 2 Application Development* (2:nd edition). Wiley Publishing, Inc., ISBN 978-0-470-56552-0.

Scientific articles/papers

- [2] Bruns, E. (2007). Enabling Mobile Phones to Support Large-Scale Museum Guidance. *Multimedia, IEEE, Volume 14*, P.16-25.
- [3] Ladetto, Q., & Merminod, B. (2002). Digital Magnetic Compass and Gyroscope Integration for Pedestrian Navigation. *9th Saint Petersburg International Conference on Integrated Navigation Systems, Saint Petersburg, Russia, 2002*
- [4] Woodman, O. J. (2007). An introduction to inertial navigation. *Technical Report, University of Cambridge, Computer laboratory*.
- [5] Hiyama, A. (2005). Position Tracking Using Infra-Red Signals for Museum Guiding System. *University of Tokyo*.
- [6] Takeshi, I. (2008). ComPass system: low power wireless sensor network system and its application to indoor positioning.
- [7] Teuber, A. & Eissfeller, B. (2006). WLAN Indoor Positioning Based on Euclidean Distances and Fuzzy Logic. *University FAF, Munich, Germany*.
- [8] Welch, G & Bishop, G (2006). An Introduction to Kalman Filter. *Department of omputer Science, University of North Carolina at Chapel Hill*.
- [9] Taheri, A & Singh, A (2004). Location Fingerprinting on Infrastructure 802.11. *IEEE Computer Society Washington, DC*.
- [10] Prasithsangaree, P., Krishnamurthy, P. & Chrysanthi, P. (2002). On indoor position location with wireless LANs. *Personal, Indoor and Mobile Radio Communications, 2002*.
- [11] Elmenreich, W. (2002). Sensor Fusion in Time-Triggered Systems, PhD thesis. *Vienna University of Technology*.

www

- [12] Official Android Developer's site (Electronic) Available:
<http://developer.android.com/index.html> 2011-05-27
- [13] U.S. Government. GPS (Electronic) Available:
<http://www.gps.gov/systems/gps/> 2011-08-29
- [14] Wikipedia. GPS (Electronic) Available:
<http://en.wikipedia.org/wiki/Gps> 2011-05-02
- [15] Wikipedia. Wi-Fi (Electronic) Available:
<http://en.wikipedia.org/wiki/Wi-Fi> 2011-05-03
- [16] Wikipedia. Trilateration (Electronic) Available:
<http://en.wikipedia.org/wiki/Trilateration> 2011-05-02
- [17] Wikipedia. Gyroscope (Electronic) Available:
<http://en.wikipedia.org/wiki/Gyroscope> 2011-05-02
- [18] Wikipedia. Accelerometer (Electronic) Available:
<http://en.wikipedia.org/wiki/Accelerometer> 2011-05-03
- [19] Wikipedia. Magnetometer (Electronic) Available:
<http://en.wikipedia.org/wiki/Magnetometer> 2011-05-03
- [20] Turner, Glenn. Gyroscopes (Electronic) Available:
<http://www.gyroscopes.org> 2011-05-03
- [21] Wikipedia. Euclidean Distance (Electronic) Available:
http://en.wikipedia.org/wiki/Euclidean_distance 2011-05-05

- [22] Mail Archive. Nexus S sensors (Electronic) Available:
<http://www.mail-archive.com/android-developers@googlegroups.com/msg148671.html>
2011-05-15
- [23] Google. Nexus S (Electronic) Available:
<http://www.google.com/phone/detail/nexus-s> 2011-08-15
- [24] Wikipedia. Dead reckoning (Electronic) Available:
http://en.wikipedia.org/wiki/Dead_reckoning 2011-05-06
- [25] Wikipedia (Electronic) Available:
http://en.wikipedia.org/wiki/Equations_of_motion 2011-05-05




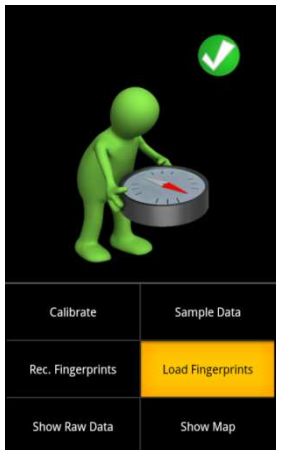
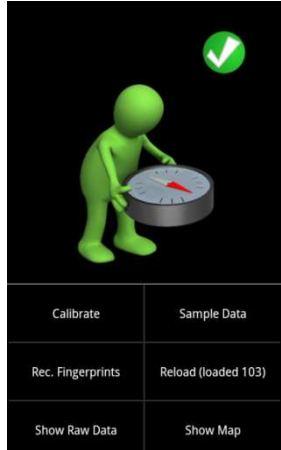
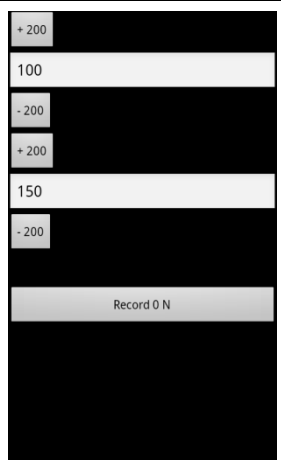
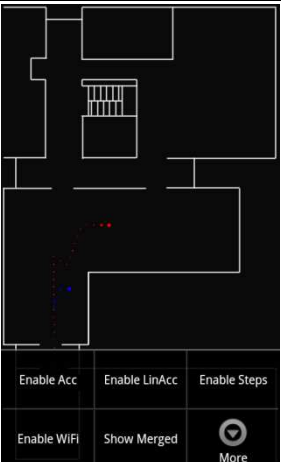
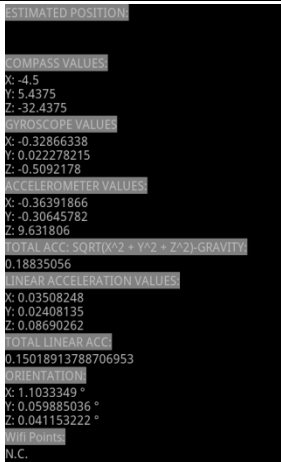
Other

- [26] Nexus S Owners Guide (2010-12-15), *NSOG-1.0-104*

Appendices

Appendix A

Screenshots

		
Start screen	Select Calibrate	Calibrating
		
Calibration complete	Load fingerprints	Fingerprints loaded
		
Record fingerprints	Show map (enable channels)	Display raw data

Appendix B

Norra station map with AP locations

