Utilising Raw GNSS Data from Android Units to Improve Positioning

Karin Seifert
Utilising Raw GNSS Data from Android Units to Improve Positioning

Karin Seifert

Abstract

In this study it was examined if positioning can be improved by utilising raw Global Navigation Satellite System (GNSS) data from Android devices instead of using Android's existing Location API. If this is the case, the coworkers at BM System may modify their apps, one is called Loco, to develop one that calculates positions directly by utilising raw GNSS data instead of using Android's Location API which is done presently. There is an app called GNSS Compare that does just this, estimates positions with raw GNSS data. In addition, if the phone using the app has dual frequency capabilities it uses this for better calculations by eliminating one of the errors, the ionosphere delay, that occur in positioning.

To allow evaluation of the accuracy of the positions, the app had to be altered to write to .gpx files. In order to test the accuracy, field tests were performed. These were executed by going to certain points, geodetic control points, which have a precise position which is known prior and testing the accuracy. Two phones were included in this study, Xiaomi Mi 8 and Samsung Galaxy S9 with the former having dual frequency capabilities and two apps were similarly tested, GNSS Compare and BM's app Loco. At each control point, Xiaomi Mi 8 with GNSS Compare, Xiaomi Mi 8 with Loco, Samsung Galaxy S9 with GNSS Compare and Samsung Galaxy S9 with Loco were tested statically. These points were then compared by evaluating the median squared error from the control point as well as the minimum squared error. It was concluded that Xiaomi Mi 8 with GNSS Compare performed the best out of the four combinations of phones and apps with an average of its median values being 6.20 meters. It was also the second best when comparing the average minimum values, closely following Xiaomi Mi 8 with Loco with the former scoring at 1.27 meters and the Xiaomi with Loco 99 cm. The lowest best value for Xiaomi Mi 8 with GNSS Compare was 5 mm, which is a very small number, and as a result was the best combination of phone and app in that aspect. In conclusion, Xiaomi Mi 8 with GNSS Compare seems to produce the highest accuracy of positioning amongst the phones and apps tested. A reason behind this is that the app calculates the positions by utilising the raw data of the phone compared to Loco which uses the Android Location API. In addition, Xiaomi M 8 is a phone with dual frequency capabilities and since GNSS Compare uses this property when applying their ionosphere delay correction it may give this combination an advantage compared to the others. However, this needs to be further studied.
Acknowledgements

I would like to thank BM System and all their helpful coworkers for giving me the opportunity to carry out my bachelor project at their company. It allowed me to learn new and interesting areas of computer science as well as discovering what the environment is like working in real life.

I would like to thank my two supervisors, Kristoffer Forsgren and Ludvig Sjöbom, for their help and support throughout my project. I received help for the background, keeping me on track and deciding the size of the scope as well as reading through my thesis and giving me very useful feedback and insights.

I would like to thank my subject reviewer, Edith Ngai, for the advice and suggestions regarding my report.

I would like to thank my family for their support by reading my thesis and giving valuable feedback and their helpful conversations about the project.
## Contents

1. **Introduction** ................................................. 4  
   1.1 Project Description and Research Questions ............. 4  
   1.2 Problem Formulation .................................. 5  

2. **Background** ............................................ 5  
   2.1 Geodetic Control Point .................................. 5  
   2.2 Trilateration ........................................... 6  
   2.3 Global Navigation Satellite System ............. 8  
   2.4 Error Sources and Consequence .................. 9  
      2.4.1 Clock-Related Errors .......................... 10  
      2.4.2 Ionosphere Error ................................ 11  
      2.4.3 Troposphere Error .............................. 12  
      2.4.4 Shapiro Delay .................................. 12  
   2.5 Dual Frequency ........................................ 12  
   2.6 GNSS Compare ........................................... 12  
   2.7 Coordinate System: SWEREF 99 18 00 .................. 12  
   2.8 Phone Models ........................................... 13  
      2.8.1 Xiaomi Mi 8 ...................................... 13  
      2.8.2 Samsung Galaxy S9 .............................. 13  
   2.9 Why Would BM Benefit from Better Positioning? ... 13  

3. **Related Work** ........................................... 14  

4. **Methodology** .......................................... 15  
   4.1 Code .................................................. 16  

5. **Results** ................................................. 18  
   5.1 Control Point 10748 ............................... 20  
   5.2 Control Point 10634 ................................ 22  
   5.3 Control Point 10622 ............................... 24  
   5.4 Control Point 10629 ................................ 26  
   5.5 Control Point 2724 .................................. 28  
   5.6 Control Point 10563 ................................ 31  
   5.7 Control Point 10747 ................................ 34  
   5.8 Control Point 8074 .................................. 37  
   5.9 Control Point 10213 ................................ 39  

6. **Discussion and Analysis** .......................... 41  
   6.1 Xiaomi Mi 8 with GNSS Compare Versus Xiaomi Mi 8 with Loco .... 42  
   6.2 Samsung Galaxy S9 with GNSS Compare Versus Samsung Galaxy S9 with Loco .... 43  
   6.3 Xiaomi Mi 8 with GNSS Compare Versus Samsung Galaxy S9 with GNSS Compare .... 43  
   6.4 Xiaomi Mi 8 with Loco Versus Samsung Galaxy S9 with Loco .... 44
1 Introduction

BM System is one of Sweden’s leading companies in digitisation of transportation and service operations and provides their customers with intelligent management systems. Their systems use the latest techniques within positioning, mobility and geographical information systems with the purpose of achieving high operational reliability and usability.

There are several telephone models that support dual frequency, a technique where the receiver tracks more than one radio signal from every satellite on different frequencies. If this was utilised it could result in a much more exact and precise positioning, particularly in urban areas where exact location may be difficult to evaluate.

1.1 Project Description and Research Questions

- Is it possible to improve the positioning for an Android device by utilising the GNSS data on devices with dual frequency receivers? If so, by how much?
- In which terrain does this work the best? How do the various areas differ? Suburban, urban areas, etc

In order to provide their customers even more precise positioning, the coworkers at BM System are focusing on further extending their estimations of positioning. Currently there are some telephone models that support dual frequency and if this technique is utilised in their calculations, their estimations may be of even higher precision. Furthermore, the goal with this thesis is to investigate if it is achievable to utilise this technique together with evaluation of positions directly from raw GNSS data in an Android device instead of using the already existing Android Location API and if so, also explore the possibility of adding and incorporating this to BM’s already existing applications.

This shall be evaluated by performing tests on the accuracy of the positioning. There are two different phone models used in this project that are tested on the two apps GNSS Compare and an app developed by BM System, called Loco. One of the phones, Xiaomi Mi 8, supports dual frequency which the app GNSS Compare utilises to improve its calculations while the other phone, Samsung Galaxy S9, does not. Four different combinations of these phones and apps will be tested for each control point and compared to one another. The combinations are Xiaomi Mi 8 with the app GNSS Compare, Xiaomi Mi 8 with Loco, Samsung Galaxy S9 with the app GNSS Compare and lastly Samsung Galaxy S9 with Loco. The tests consist of finding the mean positional value, the median positional value, the best value (closest to the control point, estimated by the square root error) as well as the distance from the best value to the actual position of the control point.
This work provides the following contributions:

- Capture where the new positions are found in the app GNSS Compare Write these to a .txt file and a .gpx file
- Look at the positions from BM app
- Upload the files onto a map for visual comparison
- Go to different geodetic control points to find the accuracy
- Find mean, median and best values. Compare them to one another.

1.2 Problem Formulation

Presently the positioning for Loco can be up to 5 to 10 meters incorrect and this works due to the use of predefined routes and position matching. However, the coworkers at BM would like to have an even more precise positioning for their different intelligent management systems. The existing system for positioning works well and improved calculations may not be needed, although a highly efficient and exact system would both improve existing intelligent systems as well as generate new ideas for brand new and innovative areas with need of high accuracy positioning. For example, a precision of decimetres would improve their Road Service System, especially road maintenance like snow removal, in several ways. It would be highly beneficial both knowing if which lane has been plowed was known as well as whether the road or sidewalk has been worked on to the app as both the security of the roads would increase as well as less fuel wasted to the vehicles tending to the snow removal and time that could be spent on lanes not yet cared for. This may result in both less contribution to carbon dioxide emission as well as a decrease of wastage in the economic perspective. In addition, creating a system of more exact location calculations may enable new intelligent systems of high quality.

One way to possibly improve position calculations is to use a technique called dual frequency, a concept later explained more in depth. Shortly, it uses more than one signal of different frequencies from one satellite which can improve positioning as one part of the atmosphere treats signals of higher frequencies differently than those of lower frequencies. An app is applying this when used by phones that have dual frequency capabilities. If this property/feature was extracted, calculations of Loco may be improved and therefore examining if using this technique can increase accuracy of positions is of high significance.

2 Background

2.1 Geodetic Control Point

Geodetic control points are placed points that serve as a reference of the geodetic position with high precision. There is a network of geodetic control points
in Uppsala that the National Land Survey and Municipality of Uppsala has created and it is used in this thesis for the purpose of evaluating the accuracy of utilising raw GNSS data from phones to improve the positioning compared to Loco which uses Android’s Location API for location. If the former has a higher precision, then the conclusion of utilising raw GNSS data and calculating the position from that is more exact and therefore the preferred choice, can be drawn. In that case, an implementation of using raw GNSS data to calculate the current location of a phone would be highly beneficial and of huge incentive to be made.

2.2 Trilateration

Trilateration works by utilising lines between reference points, which in positioning is where the satellite that sent the signal is located. These lines have known lengths since they can be calculated by multiplying the time it took them to travel from the satellite to the receiver with the speed of light.

\[ \text{Distance} = \text{Speed of light} \times \text{Time} \]

Where Distance is the distance from satellite and Time is the time taken for signal to travel from satellite to receiver. The time taken for the signal to travel from the satellite to the receiver is known since the time stamp when the signal was sent is saved as well as the time stamp when it was received is known, the time is simply a subtraction of the two time stamps. With only one reference point it is only possible to pinpoint the receiver’s position to be on the perimeter of a circle with a radius of the length between the receiver and satellite. In Figure 1 in trilateration with one satellite is shown. The circle named "A" symbolises a satellite. Since the distance between the satellite and the receiver is the only known and no direction, the receiver may be positioned on the perimeter around the satellite of the distance between them.
If another satellite is added it narrows down the position to two possible locations, which is similarly illustrated in Figure 2. These two points are positioned where the perimeters of the two satellites' signal lengths intersect. In this figure, the two circles named "A" and "B" are the satellites and P1 and P2 are the possible points where the receiver may be positioned.
Adding a third satellite will result in only one location being the true location and is shown in Figure 3. This position is where the perimeter of all the satellites’ signal length intersect. Since this only happens at one point, the location of the receiver can be pinpointed to that point. In the figure, the three circles named "A", "B" and "C" symbolise the satellites and P shows the position of the receiver. It is important to note that this is only applicable in two dimensions. For three dimensions an extra satellite would be needed to find the position due to the extra dimension. However, the principle is very much the same.

Figure 3: A figure explaining how trilateration with three satellites works.

2.3 Global Navigation Satellite System

The Global Navigation Satellite System (GNSS) is a system of satellites that deliver/supply signals to its GNSS receivers which in turn use the collected data in order to estimate the location of the receiver. There are different examples of GNSS systems, like Galileo which is Europe’s system and GPS which is the USA’s counterpart, GLONASS in Russia and BeiDou in China.
A simple explanation of how a GNSS receiver calculates your location would be to mention that first it calculates how long it takes for the signal from every satellite to reach the receiver. As explained earlier, the time taken for the signal to travel from the satellite to the receiver is known, since the time stamp when the signal was sent is saved, as well as the time stamp when it was received is known, the time is simply a subtraction of the two time stamps.

It then multiplies that by the time of speed of light to compute the distance to every satellite at the time when the signal was sent. Next step is to calculate the position to at least four satellites using trilateration. Since the exact location of each satellite when the signal was sent is known to the receiver, it can calculate its own position relative to those satellites.

2.4 Error Sources and Consequence

There are several errors that occur in Global Navigation Satellite System (GNSS) and cause miscalculations when determining the position of a receiver. The main reason for this is because GNSS signals have low energy and as a result they have a high risk of being affected by different types of noise and errors. Since the range that the GNSS receiver measures is altered by errors this range is called the pseudorange as it will not be the exact true range. The app GNSS Compare takes into consideration some of these and they will be described below.

Receiver noise is one of the errors that occur in GNSS applications and is a more broad term for the different errors that occur by the receiver when measuring the signals of the satellites. These errors can not be eliminated and as a result are white noise.
2.4.1 Clock-Related Errors

There are two different types of clock errors: clock bias and satellite clock error.

Normally, receivers have quartz crystal clocks while satellites are equipped with the more expensive and accurate atomic clock [5]. The former have lower power requirement and long life spans, although they are less stable. This is not a problem, however, as time is one of the unknown variables along with the three variables for dimensions x, y and z. Simply, four observations are needed to solve for these unknowns.
2.4.2 Ionosphere Error

The earth has an atmosphere which consists of different layers. The uppermost layer is called the ionosphere and is ionised by solar radiation, hence its name. When signals pass this layer, they will get refracted and their speed slows down. As a result, this causes a delay from the satellite to the receiver as well as a smaller alteration of direction. Since time is essential in determining where a satellite was positioned when it sent a signal in order to calculate the receiver’s location this causes communication problems. This type of error may fluctuate in size depending on whether a satellite used by the receiver is close to its horizon or by activity of sunspots. An important property of the ionosphere is the fact that signals of higher frequency that pass through this layer are not as affected as signals of lower frequency [6]. A GNSS receiver with dual frequency capability is able to receive two GNSS signals of different frequency from the same satellite. Since these signals will be affected differently by the ionospheric effect, the ionospheric delay can be estimated and as a consequence more precise positioning of the receiver can be calculated. An illustration of the effect of the ionosphere error is shown in figure 5.

![Figure 5: How the ionosphere error and the troposphere error affect the satellites’ signals shown.](image)

GNSS Compare only has implemented their Ionosphere correction algorithm on the GPS satellites as of now and the Galileo constellation returns 0.0. However, it can be applied to other constellations.
2.4.3 Troposphere Error

The troposphere is the lowest layer of Earth’s atmosphere. It is modelled as two components, a dry and a wet component. The dry component is connected to the pressure of the atmosphere and is responsible for the largest part of the delay. The wet component consists of water vapour. This component is harder to model and calculate the error since it depends on the weather conditions and as a consequence is more unpredictable. A major difference between the troposphere and the ionosphere is that the effects of the latter are not frequency dependent and thus, dual frequency measurements will not be helpful in the elimination of the delay. An illustration of the effect of the troposphere error is shown in figure 5.

2.4.4 Shapiro Delay

The Shapiro delay is also known as the relativistic path range correction and is the effect of when signals navigate through space and travel near an immense body. The mass causes the signal to reach their destination later than they would have if the object was not placed near the flight path of the signal.

2.5 Dual Frequency

Currently there are some telephone models that support dual frequency which is a technique where the receiver tracks more than one signal from each satellites on different frequencies. In the case of Galileo, these frequencies are called E1 and E5a. This results in a much more accurate and precise positioning, particularly in urban areas. If one of the frequency bands does not work or breaks down the other is used as backup. In addition, one of the errors, ionosphere delay, that occur when evaluating positions due to the atmosphere can be eliminated with two different signals since this particular error causes larger discrepancies for signals with lower frequencies compared to signals of higher frequency. This is discussed more in depth earlier.

2.6 GNSS Compare

GNSS Compare is an app that utilises raw data from mobile phones to calculate their positioning while using dual frequency on those devices that support this technique.

2.7 Coordinate System: SWEREF 99 18 00

SWEREF 99 18 00 is a coordinate system commonly used in Sweden on national level in areas such as land survey and cartography. The map projection uses a two dimensional coordinate system called Northing and Easting. These work similarly as an x and y coordinates on a usual graph, or in other words a
The geodetic control points use the SWEREF 99 18 00 coordinate system while the app GNSS Compare uses latitudes and longitudes. Since the app produces a .gpx file that should be compared with the control points one of these needs to be translated to the other. For this thesis, the positions using SWEREF 99 18 00 were chosen to be translated to latitudes and longitudes.

2.8 Phone Models

When testing the accuracy of the position calculator, two/three different phone models were tested. Since the essence of the project is to improve positioning by extracting a phone device’s raw GNSS data it makes sense to test on different models, some of newer release, and compare them to each other. One of these models has dual frequency capability and should therefore produce a better and more precise position. However, ideally they should all provide better calculations than before.

2.8.1 Xiaomi Mi 8

Xiaomi Mi 8 is the newest phone model amongst the phones in the testing pool. It has dual frequency capabilities, it is in fact the world’s first smartphone with dual frequency GNSS.

2.8.2 Samsung Galaxy S9

Samsung Galaxy S9 is a model launched in March 2018 and has single frequency.

2.9 Why Would BM Benefit from Better Positioning?

A big question that may arise is why BM would benefit from a more accurate positioning. Since it is a company that uses positioning for a lot of their different intelligent management systems this would improve their already existing systems as well as creating new ones with different approaches and areas of focus. Presently, like stated in section 2.2 previously, their positioning may be up to 5 to 10 meters incorrect and still be used. If this could be lowered to around a meter or perhaps even lower it would lead to higher accuracy and open up use for a broader spectrum of area of application and thus create new interesting ideas with a need of high precision. In the future, an error level of decimetres or centimetres may be possible and similarly open the doors for new possibilities in creating effective and powerful solutions.
One major product that BM provides is the Road Service System (or RSS), where road maintenance like snow removal plays a big part. An increased accuracy would lead to several valuable improvements. Salting roads, for example, is part of road maintenance during the months with snow. Better positioning may lead to knowing which lane or if the sidewalk is currently driven on by the user of the app and if that is known then the risk of accidentally pushing salt onto the ditch by the vehicle responsible could be lowered. This is an aspect which is both environmentally and profitably essential as salt is not wasted as well as the salting being more efficient.

Another area which falls more under general road maintenance all year round would be for the app to tell the operator where they should fix a part or perform some upkeep. An example of this may be a part of the game fence is broken and the app telling the operator where to go to fix it. Another example could be that a manhole may need some upkeep where the cover has to be replaced and the app could show where to find it.

3 Related Work

Several papers and theses were found during the research phase of the project. The three papers described below were the most relevant and interesting to this thesis. Their methods were slightly different to the one conducted in this paper and as a result it was interesting to add these for their contrasts.

A paper "Advances In Smartphone Positioning in Forests: Dual-frequency Receivers and Raw GNSS Data" by Julián Tomaštík, Juliána Chudák, Daniel Tuná was published in 2020 which describes how they proceeded in testing the possible improvement in accuracy by focusing on dual frequency GNSS receivers and by utilising raw GNSS data [10]. Their method consists of three types of experiments, two real time tests where individual points were measured as well as trajectories. The third test was of raw GNSS data from the device receiver. The first two tests were analysed by only using the last and final position that the receiver collected and not the average or similar estimations. The results that they obtained from the tests were interesting. As expected, it was found that the phone models performed the best during leaf off season with an average root square error of 4.10 m to 11.44 m than leaf on season with an average root square error of 5.41 m to 12.55 m. In addition, open area conditions produced the highest accuracy with the errors being the small 1.72 m to 4.51 m which is also to be expected since clear area above the receiver means no blockage of satellite connection. Another interesting find was made regarding the first test, which was the one concerning individual points. They found that Xiaomi Mi8 did not achieve noticeably different result from single frequency receivers. From the second test the authors discussed that smartphone receivers perform better in dynamic applications rather than static given the results of
the mean shift between the measure trajectories and the reference trajectories were between 1.23 m to 5.98 m in accuracy, a far better result than the static test. The third test which was on raw GNSS data, they observed even higher accuracy, centimetre accuracies for open areas. For areas in forests, these accuracies grew lower due to the lack of signal strength and values may be up to tens of metres. They summarised by saying that dual frequency receivers produced positions of higher accuracy and robust compared to its single frequency counterparts. However, further studies have to be made due to it can be affected by the different errors that occur related to finding positions with GNSS systems.

Another paper, "Enabling High Accuracy Dynamic Applications in Urban Environments Using PPP and RTK on Android Multi-Frequency and Multi-GNSS Smartphones", by Marco Fortunato, Joshua Critchley-Marrows, Małgorzata Siutkowska, Maria Loredana Ivanovici, Elisa Benedetti and William Roberts which was published in 2019 discusses the release of Android raw measurements and the positive effect the extra signals L5/E5 from dual frequency have on multi path in urban areas. Multi path is where the estimation of positions is affected due to signals being reflected by objects. The authors talk about performing tests in different environments, semi open sky and suburban environments in order to evaluate if the accuracy can be improved by using raw GNSS data. Although, phones are not the ideal receiver for GNSS due to its antenna being shared with others, for example Wi-Fi and Bluetooth, they found that the additional signals received from dual frequency resulted in a higher final accuracy.

The third paper, "Practical applicability of processing static, short-observation-time raw GNSS measurements provided by a smartphone under tree vegetation" published by Julián Tomaštík and Matej Varga and they discuss the accuracy of static GNSS measurements made by Xiaomi Mi 8 in areas of different tree canopy conditions. Their method is described as measuring the horizontal accuracy for ten minutes and was collected for 40 days while testing three points. They found that centimetre accuracy could be recorded for open sky conditions while vegetation decreased the accuracy. Apart from this, they also compared carrier phase based solutions to the code based measurements and found that the former performed better in some cases but had results of high variability while the latter, during optimal conditions, provided accuracy of less than a meter and being available to phones with Android 7 (or higher).

4 Methodology

The original plan was to investigate which files of the already existing app GNSS Compare that were needed for calculating positions with dual frequency and to translate those files from Java code to C#. As the project progressed it was discovered that this approach would be more time consuming than allowed in the existing time frame for the project and the implementation plan had to be
revised in order to be finished in time.

It seemed appropriate to modify GNSS Compare so that it would create .gpx files where the position was calculated to which could be compared to the positions found by Loco. These .gpx files were then inputted to google maps to create a map which enabled the positions to be compared visually.

The output of GNSS Compare was compared to Loco which uses Android’s Location API. The reasoning behind this was to evaluate if utilising raw GNSS data from phones and calculating positions directly results in more precise locations compared to Android’s Location API. If so, implementing and developing this would be an incentive to the company as better positions are highly beneficial for their intelligent systems.

Different routes were tried to compare the phones and apps to one another as well as testing them against geodetic control points. The reasoning behind this was that these points have a high accuracy of position and as a result if the phones and apps showed positions of close proximity the accuracy of them may be considered high as well.

Before the phones and apps were compared to the geodetic compare points, these points’ positions had to be translated from the coordinate system SWEREF 99 18 00 to latitudes and longitudes.

4.1 Code

The app GNSS Compare did not write to a .gpx file and therefore the code had to be altered in the appropriate place to add the modification, which was where the new position is found, i.e. the latitude and longitude. This could then simply be written to a file. It is important to note that only when there is a last location, i.e. lastLocation != null, this .gpx file will be opened or appended since there has to exist a location to be added. Therefore there is an if-statement that checks for this before trying to work with the position:

```java
if(lastLocation != null)
```

A .gpx file has certain sections that need to be included and those are the header, the name, track points for every position and lastly a footer before the file ends. The header, name and footer are saved as strings for clarity as shown below.

```xml
String header = "<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gpx xmlns="http://www.topografix.com/GPX/1/1"
	creator="MapSource 6.15.5"
	version="1.1"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com/GPX/1/1/gpx.xsd">
	<trk>
```

16
String name = "\t\t<name>TestGNSS</name>\n\t\t<trkseg>\n";
String footer ="\t\t</trkseg>\n\t\t</trk>\n</gpx>";

Any attempt in the code to create, write to or any other manipulations to any file are placed in try and catch blocks in order to catch any error that may occur due to these actions, instead of letting the app crash inexplicably. The following code block is needed to access the external storage of the phone that runs the app. The state of the storage has to be checked since both read and write access has to be exist. If there is both read and write access, the desired name of a directory is created and saved in the variable baseDir.

File baseDir;
String stateGPX = Environment.getExternalStorageState();
if (Environment.MEDIA_MOUNTED.equals(stateGPX)) {
    baseDir = new File(Environment.getExternalStorageDirectory(), "GNSS Compare/");
    baseDir.mkdirs();
} else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(stateGPX)) {
    Log.w(TAG, "Cannot write to external storage.");
    return;
} else {
    Log.w(TAG, "Cannot read external storage.");
    return;
}

In the next block the path to the desired file to write to is saved to the variable gpxFile. If this is the first time running the code, this file may not exist yet and therefore there is an if statement checking this. If that is the case, the header, name and footer has to be added to the file.

File gpxFile = new File(baseDir, "TestGNSS.gpx");

if(!gpxFile.exists()) {
    FileWriter writerGPX = new FileWriter(gpxFile, true);
    writerGPX.append(header);
    writerGPX.append(name);
    writerGPX.append(segments);
    writerGPX.append(footer);
    writerGPX.flush();
    writerGPX.close();
}

The following block is already existing code that logged positions.

Log.i(TAG, "locationFromGoogleServices: New location (phone): 
    + lastLocation.getLatitude() + ", 
    + lastLocation.getLongitude() + ", 
    + lastLocation.getAltitude());
A new try and catch block is introduced where the positions are appended to the already existing file "TestGNSS.gpx" (this file was created earlier). The idea here is that the sections header and name have already been added to the file and this is the case for the footer as well. However, it should always be the last thing in the .gpx file and as a result the footer is replaced by the newest addition of position as well as an identical footer.

```java
File gpxfile = new File(baseDir, "TestGNSS.gpx");
String footerString = "\t\t</trkseg>\n\t</trk>\n</gpx>
String locString = "\t\t\t<trkpt lat=""+lastLocation.getLatitude() + "" lon="" lastLocation.getLongitude() + ""/>\n";</
Scanner sc = new Scanner(gpxfile);
StringBuffer buffer = new StringBuffer();
while (sc.hasNextLine()) {
    buffer.append(sc.nextLine()+System.lineSeparator());
}
String fileContents = buffer.toString();
fileContents = fileContents.replace(footerString,locString + footerString);
FileWriter writerGPX = new FileWriter(gpxfile,false);
writerGPX.append(fileContents);
writerGPX.flush();
writerGPX.close();
```

5 Results

As seen in Table 1 below the different combinations of phones with the two apps are summarised, including the distance from the estimated points’ mean, median and best points to the actual location of the control points. The leftmost column is designated to the numbers of the controls and is labelled "Contr Point" which is short for control point. There is also a column for the type of area that the control point is placed in. U stands for Urban and Sub for Suburban. For control point 2721 the description of this is "Sub and F" which stands for the outer ranges of a forest in a suburban area. Only the median values, and not mean values, were calculated and presented in this project due to their slightly different properties. The mean is the preferred option when data values are close to one another and there are no outliers while the median is the better choice when there are outliers or otherwise skewed data points. The same reasoning was behind when choosing best values and not worst values as possible outliers will affect the result for worst values.
Table 1: A table showing the median and best (lowest) square error from each control point, for all combinations of phones and apps. The type of area or terrain each control point is placed in is included. The mean of each column is added as well.

<table>
<thead>
<tr>
<th>Control Point</th>
<th>Area</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>GNSS Median</td>
<td>Best Value</td>
<td>GNSS Median</td>
<td>Best Value</td>
<td>GNSS Median</td>
<td>Best Value</td>
<td>GNSS Median</td>
<td>Best Value</td>
</tr>
<tr>
<td>10748</td>
<td>U</td>
<td>2.04 m</td>
<td>0.37 m</td>
<td>12.58 m</td>
<td>0.52 m</td>
<td>26.41 m</td>
<td>8.73 m</td>
<td>12.58 m</td>
<td>0.52 m</td>
</tr>
<tr>
<td>10619</td>
<td>Sub</td>
<td>6.05 m</td>
<td>3.99 m</td>
<td>1.85 m</td>
<td>0.53 m</td>
<td>14.26 m</td>
<td>1.77 m</td>
<td>6.03 m</td>
<td>2.04 m</td>
</tr>
<tr>
<td>10622</td>
<td>Sub</td>
<td>5.12 m</td>
<td>0.61 m</td>
<td>12.26 m</td>
<td>0.52 m</td>
<td>3.74 m</td>
<td>0.07 m</td>
<td>3.87 m</td>
<td>0.13 m</td>
</tr>
<tr>
<td>10624</td>
<td>Sub</td>
<td>0.79 m</td>
<td>0.005 m</td>
<td>0.75 m</td>
<td>0.20 m</td>
<td>2.63 m</td>
<td>0.15 m</td>
<td>3.07 m</td>
<td>1.09 m</td>
</tr>
<tr>
<td>2721</td>
<td>Sub and F</td>
<td>10.57 m</td>
<td>4.53 m</td>
<td>4.39 m</td>
<td>2.19 m</td>
<td>6.36 m</td>
<td>3.07 m</td>
<td>11.24 m</td>
<td>4.02 m</td>
</tr>
<tr>
<td>10565</td>
<td>U</td>
<td>2.88 m</td>
<td>0.80 m</td>
<td>1.27 m</td>
<td>0.40 m</td>
<td>0.47 m</td>
<td>0.05 m</td>
<td>7.80 m</td>
<td>1.65 m</td>
</tr>
<tr>
<td>10747</td>
<td>U</td>
<td>4.60 m</td>
<td>0.14 m</td>
<td>4.74 m</td>
<td>1.03 m</td>
<td>4.49 m</td>
<td>2.22 m</td>
<td>5.79 m</td>
<td>1.18 m</td>
</tr>
<tr>
<td>8074</td>
<td>Sub</td>
<td>2.63 m</td>
<td>0.03 m</td>
<td>19.03 m</td>
<td>0.91 m</td>
<td>7.04 m</td>
<td>3.09 m</td>
<td>6.01 m</td>
<td>0.96 m</td>
</tr>
<tr>
<td>10213</td>
<td>Sub</td>
<td>21.10 m</td>
<td>0.95 m</td>
<td>12.67 m</td>
<td>2.57 m</td>
<td>2.79 m</td>
<td>0.08 m</td>
<td>1.76 m</td>
<td>0.10 m</td>
</tr>
<tr>
<td>Mean:</td>
<td></td>
<td>6.20 m</td>
<td>1.27 m</td>
<td>7.73 m</td>
<td>0.99 m</td>
<td>7.58 m</td>
<td>2.14 m</td>
<td>6.46 m</td>
<td>1.30 m</td>
</tr>
</tbody>
</table>
5.1 Control Point 10748

Figure 6: Control point 10748 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10748. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 6 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 7: Control point 10748. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10748. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 7 shows how the different combinations of the phones and the apps acted during the testing of control point 10748.
5.2 Control Point 10619

Figure 8: Control point 10619 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10619. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 8 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 9: Control point 10619. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10619. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 9 shows how the different combinations of the phones and the apps acted during the testing of control point 10619.
5.3 Control Point 10622

Figure 10: Control point 10622 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10622. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 10 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 11: Control point 10622. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10619. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 11 shows how the different combinations of the phones and the apps acted during the testing of control point 10622.
5.4 Control Point 10624

Figure 12: Control point 10624 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10624. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 12 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 13: Control point 10624. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10624. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 13 shows how the different combinations of the phones and the apps acted during the testing of control point 10624.
5.5 Control Point 2721

Figure 14: Control point 2721. Shows the position and coordinates in the SWEREF 99 18 00 coordinate system.

Figure 14 shows the location of the control point 2721. Its position is in SWEREF 99 18 00 and converted to longitude and latitude it stands at latitude 59.832069 and longitude 17.638933.
Figure 15: Control point 2721 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 2721. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 15 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 16: Control point 2721. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 2721. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 16 shows how the different combinations of the phones and the apps acted during the testing of control point 2721.
5.6 Control Point 10565

Figure 17 shows the location of the control point 10565. Its position is in SWEREF 99 18 00 and converted to longitude and latitude it stands at latitude 59.847559 and longitude 17.677893.
Figure 18: Control point 10565 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10565. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 18 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 19: Control point 10565. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10565. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 19 shows how the different combinations of the phones and the apps acted during the testing of control point 10565.
5.7 Control Point 10747

Figure 20: Control point 10747. Shows the position and coordinates in the SWEREF 99 18 00 coordinate system.

Figure 20 shows the location of the control point 10747. Its position is in SWEREF 99 18 00 and converted to longitude and latitude it stands at latitude 59.858670 and longitude 17.690983.
Figure 21: Control point 10747 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10747. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 21 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 22: Control point 10747. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10747. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 22 shows how the different combinations of the phones and the apps acted during the testing of control point 10747.
5.8 Control Point 8074

Figure 23: Control point 8074 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 8074. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 23 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 24: Control point 8074. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 8074. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 24 shows how the different combinations of the phones and the apps acted during the testing of control point 10747.
5.9 Control Point 10213

Figure 25: Control point 10213. Shows the position and coordinates in the SWEREF 99 18 00 coordinate system.

Figure 25 shows the location of the control point 10213. Its position is in SWEREF 99 18 00 and converted to longitude and latitude it stands at latitude 59.854902 and longitude 17.631978.
Figure 26: Control point 10213 is placed at the origin. Shows the different combination of phones and models when testing their accuracy to control point 10213. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 26 shows how scattered the data points are for the different combinations of the phones and the apps around the control point which is placed at the origin. The x-axis shows the error in east and west direction while the y-axis shows the error in the north and south direction.
Figure 27: Control point 10213. Shows the actual positions of the different combination of phones and models on a map when testing the accuracy to control point 10213. Xiaomi Mi 8 with GNSS Compare is blue, Xiaomi Mi 8 with Loco is green, Samsung Galaxy S9 with GNSS Compare is red and Samsung Galaxy S9 with Loco is yellow.

Figure 24 shows how the different combinations of the phones and the apps acted during the testing of control point 10747.

6 Discussion and Analysis

As seen in the table in section Result the different phones combined with both apps have been summarised with a value of the mean, median and closest points' distance to the corresponding control point. These different combinations will be compared and discussed below. The median values are the median distance of the data points to the actual position of the control points and similarly, the best values are the distance to the actual control point for the data point closest to it. From this point on in the thesis, these will be called the median values and best values as a shorthand.

Overall, it seems as though urban and suburban areas do not make much of a difference in the error for the closest points as well as the median error, although a larger number of control points would have to be tested to establish this. The one point tested in the edge of a forest, otherwise located in a suburban area,
produces the worst results for all combinations of phones and apps. This is probably due to trees blocking the receiver from satellites.

6.1 Xiaomi Mi 8 with GNSS Compare Versus Xiaomi Mi 8 with Loco

For the median values of these two combinations, Xiaomi Mi 8 with GNSS Compare was lower than Xiaomi Mi 8 with Loco four times out of nine of the control points with the values for control points 10624 and 10747 being very close to each other, 0.79 m for GNSS Compare compared to 0.75 for Loco and 4.60 m for GNSS Compare compared to 4.74 for Loco respectively. However, Xiaomi Mi 8 with GNSS Compare tended to have less outliers and thus slightly less large values of the median. The combination of Xiaomi Mi 8 and GNSS Compare had two noticeably larger values of median values for control points 2721 and 10213. The former is a point placed on the wall of a tall building which may have caused this larger value. Its best value is very small, 0.95 m which is of high accuracy. The latter has a very small best value of 0.95 m and thus the large median value was therefore probably due to uncertainty in computations before finding sufficient amount of satellites. In addition, point 2721 is the one point in the outskirts of a forest and the results are probably due to trees blocking the receiver. Xiaomi Mi 8 with Loco had four noticeably larger median values for control points 10748, 10622, 8074 and 10213. However, since all of the best values for these control points are very small these larger values may be due to outliers. As previously discussed, 10213 is placed on the wall of a tall building. Figure 6 visualises how the phones and models behaved during control point 10748 and for Xiaomi Mi 8 with Loco it is shown that there are several data points where this combination has larger distances. As seen in Figure 7 these are not the starting points which usually are misleading before finding the needed satellites but may be caused due to lesser satellites being available at that point in time or in case of loss of connection.

For the best values of these combinations, Xiaomi Mi 8 with GNSS Compare was lower than Xiaomi Mi 8 with Loco five times out of nine of the control points with the values for control points 10748, 10622, 10624 and 10565 being very close to each other. Overall in best values, Xiaomi Mi 8 with Loco tended to be more stable with values ranging from 0.20 m to 2.57 m while Xiaomi Mi 8 with GNSS Compare had lower values than the former at its best but being slightly less stable and its two largest values produced were 3.99 m and 4.53 m compared to the former with two values of 2.19 m and 2.57 m. However, this is a small difference. Its values ranges from 0.005 m to 4.53 m.

Overall, Xiaomi Mi 8 with GNSS Compare tended to have a higher precision than Xiaomi Mi 8 with Loco, meaning the data points being closer to each other. An exception to this can be shown in Figure 24 when testing control point 10213, where Xiaomi Mi 8 with GNSS Compare has a higher precision while also having the smaller error of its best value compared to the error of

42
the best value for Xiaomi Mi 8 with Loco. This might affect the results as two points at equally long distances from another on either side of the control point will cancel each other out. This can be seen in for example 22 where Xiaomi Mi 8 with GNSS Compare has a higher precision than Xiaomi Mi 8 with Loco and also a lower error for the best value, 0.14 m compared to 1.03 m. However, the median values are very similar to one another.

6.2 Samsung Galaxy S9 with GNSS Compare Versus Samsung Galaxy S9 with Loco

For the median values of these two combinations, Samsung Galaxy S9 with GNSS Compare was lower than Samsung Galaxy S9 with Loco five times out of nine of the control points with the values for control points 10622. 10624, 10747, 8074 and 10213 being very close to each other. In general, Samsung Galaxy S9 with GNSS Compare tended to reach lower values compared to Samsung Galaxy S9 with Loco while its two highest values being higher than the other. In other words, Samsung Galaxy S9 with GNSS Compare seems to reach lower median values while Samsung Galaxy S9 with Loco has values more similar to one another.

Overall, Samsung Galaxy S9 with GNSS Compare tended to have a higher precision than Samsung Galaxy S9 with Loco, meaning the data points being closer to each other. This might affect the results as two points at equally long distances from another on either side of the control point will cancel each other out. This can be seen in for example 10 where Xiaomi Mi 8 with GNSS Compare has a higher precision than Xiaomi Mi 8 with Loco.

6.3 Xiaomi Mi 8 with GNSS Compare Versus Samsung Galaxy S9 with GNSS Compare

For the median values of these two combinations, Xiaomi Mi 8 with GNSS Compare was lower than Samsung Galaxy S9 with GNSS Compare four times out of nine of the control points with the values for control points 10622, 10624, 10565 and 10747 being fairly similar to each other. Both Xiaomi Mi 8 with GNSS Compare and Samsung Galaxy S9 with GNSS Compare have two values noticeably larger than the rest. For the former these are 21.10 meters and 10.57 meters while the values of the latter are 26.41 meters and 14.26 meters. These do not occur for the same tests and may be due to differing reasons. For the highest value of Xiaomi Mi 8 with GNSS Compare, control point 10213, it is shown in Figure 26 that the precision is very low which is probably as a consequence of not finding sufficient satellites in the beginning of the test as well as there might be fewer of GPS satellites. Since the control point is placed on a tall building it is probably due to satellite blockage. Both of these combinations use the app GNSS Compare which can eliminate the ionosphere delay with the help of dual frequency. However, this is only implemented for GPS satellites presently and if there are less of those satellites found or available, the estimations may be
affected.

The second largest value of the median for Xiaomi Mi 8 with GNSS Compare which is 10.57 meters occurs for the test on control point 2721. As seen in figures 15 and 16 all combinations of phones and apps have a low precision, although Xiaomi Mi 8 with GNSS Compare has the highest amongst them. Since all of them have a low precision and that the control point is positioned on the edge of a forest, the probability of them not finding as many satellites as normally is high which is very likely the culprit of these results.

The highest value of Xiaomi Mi 8 with Loco, 26.41 meters, happens for the test on control point 10748. As seen in figure 6 there are several outliers shown. The figure reveals that these data points are not the start or end points. As a result, they may be due to loss of satellite connection. The second largest value, 14.26 meters, is found in the test for control point 10619. It is shown in figures 8 and 9 that the precision for Xiaomi Mi 8 with Loco is quite low which is probably because of either fewer satellites being available at the time or perhaps some trouble finding satellites.

6.4 Xiaomi Mi 8 with Loco Versus Samsung Galaxy S9 with Loco

For the median values of these two combinations, Xiaomi Mi 8 with Loco was lower than Samsung Galaxy S9 with Loco five times out of nine of the control points and for one of the control points they both shared the same value. Xiaomi Mi 8 with Loco had four values that were noticeably larger than the rest, three of these being around 12 meters and one being 19 meters. The other five values were very low, with an error ranging from a low 0.75 meters to 4.74 meters. Comparably, Samsung Galaxy S9 with Loco had two noticeably larger values of 12.58 meters and 11.24 meters while the others being low but not as much as the other app.

The values of 12.58 meters occur during the test of control point 10748 for both of the test subjects and their behaviour can be observed in figure 6 where Xiaomi Mi 8 with Loco is green and Samsung Galaxy S9 with Loco is yellow. Since the data points are very similar to each other and practically lay on each other only the green points are visible. Since both of them appears to have divergent points it may be due to less satellites available at the time which affects the estimations.

The rest of the high median values for Xiaomi Mi 8 with Loco occur when the other phone performs well during the tests for control points 10622, 8074 and 10213. Figure 10 and 11 show that the precision is slightly lower for Xiaomi Mi 8 with Loco compared to Samsung Galaxy S9 with Loco. However, since the latter has a larger percentage of its points closer to the control point, it performs better. These results can possibly be due to differences in the hardware of the phones. Loco does not utilise the dual frequency capacity that Xiaomi Mi 8
has which means that the ionosphere delay is not excluded and as a result that phone does not have an advantage in that particular area.

The second largest median value for Samsung Galaxy S9 with Loco is found for the test of control point 2721. It is seen in figures 15 and 16 that the precision of Samsung Galaxy S9 is lower than for its counterpart which is the reason for its much higher value of the median value. As discussed earlier, this can be due to some difference in hardware as the other phone performed better. However, as previously stated in earlier sections, control point 2721 is placed on the edge of a forest and is probably the reason behind its higher median error.

7 Conclusion

Overall, Xiaomi Mi 8 with GNSS Compare seems to achieve the best values of lowest error amongst the other combinations of phones and apps in the testing pool, with only two values being larger than 1 meter and its smallest error found to be just 0.005 meters from the position of the control point it is tested against, closely followed by Samsung Galaxy S9 with a best value of 0.05 meters. The other two, Xiaomi Mi 8 and Samsung Galaxy S9 with Loco also produce very low errors. As small numbers as 0.005 meters are very low numbers and it is important to keep in mind that there is a conversion of SWEREF 99 18 00 to latitudes and longitudes for the position of the control points and as such these small errors can be due to this particular conversion. For the same reason, errors may also be slightly larger than they appear to be. A reason why this is the result may be due to GNSS Compare calculating the positions by utilising the raw data of the phone using the app compared to Loco which rather using the Android Location API. In addition, Xiaomi Mi 8 is a phone that has dual frequency capabilities and since GNSS Compare uses this property when applying their ionosphere correction algorithm it may give this certain combination of phone and app an advantage compared to the others. Since the ionospheric delay differ in size, as previously mentioned, depending on how close the satellite is on the horizon seen from the receiver as well as the activity of sun spots.

When observing the median values of each combination of phones and apps, they all perform well and when testing GNSS Compare on the phones it produces two noticeably larger values than usual compared to the tests on the phones performed on Loco where Xiaomi Mi 8 has four large values but Samsung Galaxy S9 like the GNSS Compare app having two larger values.

When inspecting the mean values of the medians, Xiaomi Mi8 with GNSS Compare does perform the best followed by Samsung Galaxy S9 with Loco and the former is the second best when looking at the mean values of the best values. In addition, it achieves the lowest values of error with the smallest being 0.005 m. From this information it can be concluded that there is a slight advantage
in using Xiaomi Mi8 with GNSS Compare, i.e. calculating the positions by utilising the raw GNSS data from an android device with a phone with dual capabilities that are used to eliminate the ionosphere delay due to multiple frequencies from each satellite. However, this needs to be further studied such as testing the control points multiple times. An interesting angle would be to vary the time intervals. In addition, areas in forests could be examined and expand the tests as this project only expanded to urban and suburban areas in Uppsala.

Answering the research question formulated in the beginning of the project:

- Is it possible to improve the positioning for an Android device by utilising the GNSS data on devices with dual frequency receivers? If so, by how much?

- In which terrain does this work the best? How do the various areas differ? Suburban, urban areas, etc

Yes, it seems like utilising raw GNSS data on Android phones with dual frequency capabilities does perform better than with the other combinations of phones and apps. It is slightly more difficult to determine by how much better it is. However, its lowest square error is 5 mm which is a very small number. This can be compared to the smallest square errors of Xiaomi with Loco, Samsung with GNSS Compare and Samsung with Loco which are 20 cm, 7 cm and 10 cm respectively.

7.1 Future Work

For future work there are several things that can be examined and performed. Firstly, the initial plan for this project was to study and inspect the app GNSS Compare to find the vital parts of the code where the positions are evaluated as well as where the errors were modelled and calculated. Especially essential to find was where the ionosphere delay was eliminated due to the use of dual frequency implemented. This was then to be translated from the original programming language Java to C#. Since this plan had to be revised for the time limit of the project and as such the translation part had to be removed and instead the app GNSS Compare was edited to write to a .gpx file which then would be translated to Loco both in a visual manner on a map and in a graph, as well as extracting the positional values, for comparison. In a future project, a translation could be produced. The translation would not be a waste of time and would be highly interesting to view seeing the promising results.

Another interesting area to consider in the future is whether the app GNSS Compare performs better with GPS, with Galileo or with both. As of right now, the app has only implemented correction for the ionosphere delay and as a result a phone with dual capacity may perform better with only GPS. An additional task for subsequent work and projects would thus also be to implement this correction for the ionosphere delay on Galileo as well and perhaps even
other satellite systems. In any case, being able to receive an average accuracy to centimetres would be very beneficial for several areas of focus as discussed earlier in this thesis.

Furthermore, future works may also perform different testing methods as mentioned in the conclusion section. These being testing the control points multiple times as well as varying the time intervals of the tests. In this project, control points in urban and suburban areas and one in the outskirts of a forest in Uppsala were tested, however, areas in forests would be interesting to examine as it is known that the accuracy of positioning is lower in those areas.
References


Appendix

This section of the Appendix is provided to present the code responsible for writing to the .gpx file.

```java
if (lastLocation != null) {
    String header = "<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gpx xmlns="http://www.topografix.com/GPX/1/1"
    creator="MapSource 6.15.5"
    version="1.1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com/GPX/1/1/gpx.xsd">
    <trk>
        <name>TestGNSS</name>
        <trkseg>
            String segments = "";
            try {
                File baseDir;
                String stateGPX = Environment.getExternalStorageState();
                if (Environment.MEDIA_MOUNTED.equals(stateGPX)) {
                    baseDir = new File(Environment.getExternalStorageDirectory(), "GNSS Compare/");
                    baseDir.mkdirs();
                } else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(stateGPX)) {
                    Log.w(TAG, "Cannot write to external storage.");
                    return;
                } else {
                    Log.w(TAG, "Cannot read external storage.");
                    return;
                }
                File gpxfile = new File(baseDir, "TestGNSS.gpx");
                if (!gpxfile.exists()) {
                    FileWriter writerGPX = new FileWriter(gpxfile, true);
                    // true appends while false over writes
                    writerGPX.append(header);
                    writerGPX.append(name);
                    writerGPX.append(segments);
                    writerGPX.append(footer);
                    writerGPX.flush();
                    writerGPX.close();
                }
            } catch (IOException e) {
                // TODO Auto-generated catch block
                Log.e(TAG, "Error Writing Path", e);
            }
        }
    </gpx>
}
```

49
if (lastLocation != null) {
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            for (DataViewer dataViewer : mPagerAdapter.getViewers()) {
                dataViewer.onLocationFromGoogleServicesResult(lastLocation);
            }
        }
    });
}

Log.i(TAG, "locationFromGoogleServices: New location (phone): 
    + lastLocation.getLatitude() + ", 
    + lastLocation.getLongitude() + ", 
    + lastLocation.getAltitude()");

try {
    File baseDirectory;
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)) {
        baseDirectory = new File(Environment.getExternalStorageDirectory(), 
            "GNSS Compare/");
        baseDirectory.mkdirs();
    } else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
        Log.w(TAG, "Cannot write to external storage.");
        return;
    } else {
        Log.w(TAG, "Cannot read external storage.");
        return;
    }
    File testfile = new File(baseDirectory, "position.txt");
    FileWriter fileWriter = new FileWriter(testfile);
    fileWriter.write( lastLocation.getLatitude() + 
        + lastLocation.getLatitude() + 
        + lastLocation.getLongitude() + 
        + lastLocation.getLongitude() + 
        + lastLocation.getLatitude());
    fileWriter.close;
    /////////////////
    File baseDir;
    String stateGPX = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(stateGPX)) {
        baseDir = new File(Environment.getExternalStorageDirectory(), 
            "GNSS Compare/");
        baseDir.mkdirs();
    } else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(stateGPX)) {
        Log.w(TAG, "Cannot write to external storage.");
        return;
    } else {

Log.w(TAG, "Cannot read external storage.");
return;
}
File gpxfile = new File(baseDir, "TestGNSS.gpx");
String footerString = "\t\t</trkseg>\n\t</trk>\n</gpx>";
String locString = "\t\t\t<trkpt lat="" + lastLocation.getLatitude() + "\n" lon="" + lastLocation.getLongitude() + "">" + "</trkpt>" + "\n";
Scanner sc = new Scanner(gpxfile);
StringBuffer buffer = new StringBuffer();
while (sc.hasNextLine()) {
    buffer.append(sc.nextLine() + System.lineSeparator());
}
String fileContents = buffer.toString();
fileContents = fileContents.replace(footerString, locString + footerString);
FileWriter writerGPX = new FileWriter(gpxfile, false);
writerGPX.append(fileContents);
writerGPX.flush();
writerGPX.close();
// Add: Append trkpt for every position
// Example: "trkpt lat="59.9" lon="18.8"></trkpt>"
// Example: fileWriter.write("trkpt lat="" + lastLocation.getLatitude() + "lon="" + lastLocation.getLongitude() + "">" + "</trkpt>")
}
catch (IOException e) {
    Log.e("Exception", "File write failed: " + e.toString());
}
}