Gmeindl David, BSc

GNSS Fingerprinting

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Geomatics Science

submitted to

Graz University of Technology

Supervisor:

Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Manfred Wieser

Institute of Geodesy

Graz, January 2015

Affidafit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis dissertation.

Eidesstaatliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Date/Datum

Signature/Unterschrift

Acknowledgement

I would like to thank my supervisor Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Manfred Wieser for his guidance in the course of this work. His insights and suggestions were very useful and were an important part to complete this thesis.

Special thanks goes to Dipl.-Ing. Petra Hafner, who was the guiding hand and had a friendly ear whenever problems arose. Furthermore, I would like to thank her for proofreading this work, her contributions and for giving very helpful hints.

The whole team of navigation of the Institute of Geodesy was supporting me, whenever possible. My appreciation for their friendliness, knowledge, and competence is great.

My family was always there fore me throughout the study. I would like to thank them, for their encouragement and for making this possible. Furthermore, I would like to thank my friends, who accompanied me in study, both directly and indirectly.

Abstract

This work deals with an alternative positioning approach to GNSS position solutions in urban areas. Due to high shadowing of satellite signals in cities, conventional GNSS Single Point Positioning is often subject to large errors. The presented method uses the shadowing of buildings of satellite signals. For this purpose, the visibilities to satellites are evaluated, which were measured by a GNSS receiver. A line of sight defines a visibility between the user and a satellite. The visibilities are then compared to a pre-computed model. The model - a fingerprint - contains information about the line of sight to satellites for pre-defined reference points at any given time. The visibilities are calculated from a 3D city model and satellite positions through a 3D intersection. After comparing the measurements with the model, a probability of a user position is assigned to each reference point.

Three approaches for the computation of GNSS fingerprints will be presented and analysed in this work. To handle the calculation of the model as efficiently as possible, methods are presented, which reduce the number of necessary calculations. Furthermore, error analyses are performed, which investigate the effects on the accuracy of GNSS fingerprints. The fingerprints were evaluated in an experiment and compared to GPS single point positioning solutions.

The results show, that GNSS fingerprinting can improve the positioning accuracy and can help to detect the side of a street. The method can be combined with a variety of absolute and relative navigation systems, and serve as input for filters. Due to the increasing number of available 3D city models, this technology can be used in urban areas in the future and can be applied to guidance systems of vehicles, navigation systems for the visually impaired and location-based services.

Kurzfassung

Diese Arbeit handelt von einer alternativen Positionsmöglichkeit zu GNSS Positionslösungen in urbanen Räumen. Durch hohe Abschattung der Satellitensignale in Städten, ist herkömmliches GNSS Single Point Positioning oft mit großen Abweichungen behaftet. Die vorgestellte Methode nutzt die Gebäudeabschattungen der Satellitensignale. Dazu werden die Sichtbarkeiten zu Satelliten ausgewertet, welche mittels eines GNSS Receivers gemessen wurden. Eine Line of Sight definiert daher eine Sichtbarkeit zwischen dem User und einem Satelliten. Die Sichtbarkeiten werden dann mit einem vorberechneten Modell verglichen. Das Modell - ein Fingerprint - enthält Informationen über die Line of Sight zu Satelliten zu einem bestimmten Zeitpunkt für vordefinierte Referenzpunkte. Die Sichtbarkeiten werden aus einem 3D Stadtmodell und Satellitenpositionen durch eine 3D Verschneidung berechnet. Nach dem Vergleich der Messungen mit dem Modell, ergibt sich für jeden Referenzpunkte eine Wahrscheinlichkeit der Userposition.

Es werden in dieser Arbeit drei Ansätze für die Berechnung von GNSS Fingerprints vorgestellt und untersucht. Um die Vorausberechnung des Modells möglichst effizient zu handhaben, werden Methoden vorgestellt, die die Anzahl der nötigen Berechnungen vermindern. Des Weiteren werden Fehleranalysen durchgeführt, die Einflüsse auf die Genauigkeit des GNSS Fingerprints untersuchen. Die Fingerprints wurden in einem Experiment evaluiert und mit GPS Single Point Positionslösungen verglichen.

Die Ergebnisse zeigen, dass GNSS Fingerprinting zur Verbesserung der Positionsgenauigkeit und vor allem zur Detektion der Straßenseite beitragen kann. Die Methode kann mit einer Vielzahl an absoluten und relativen Navigationssystemen kombiniert werden und für Filter als Input dienen. Durch die wachsende Anzahl an verfügbaren 3D Stadtmodellen kann diese Technologie in Zukunft in urbanen Räumen eingesetzt werden und vor allem für Leitsysteme von Fahrzeugen, Navigationssysteme für sehschwache Personen und Location-Based Services verwendet werden.

Contents

Abbrevations

CG	computer graphics
CSV	comma separated list
DB	database
DEM	digital elevation model
FTP	file transfer protocol
GNSS	global navigation satellite system
GPST	GPS time
IIR	infinite impulse response (filter)
JD	Julian Date
LLF	local level frame
LoS	line of sight
LSA	least squares adjustment
NLoS	non-line of sight
OSM	Open Street Map
PRN	pseudo random noise; used for distance determination as well as to explicitly identify GPS satellites
RSS	Received Signal Strength
SNR	signal to noise ratio
SPP	single point positioning
UTC	universal time, coordinated
WLAN	Wireless Local Area Network

1. Introduction

1.1. Motivation

GNSS Fingerprinting is a positioning method using Global Navigation Satellite System (GNSS) signals. Conventional positioning with GNSS is performed by measuring pseudoranges to satellites. For this, code or phase measurements are used. As GNSS signals are shadowed by buildings especially in urban areas, where tall buildings and narrow streets are common, conventional GNSS positioning performance is poor, because of non-optimal satellite constellations, availability and coverage. For a GNSS positioning solution, at least four satellites must be visible. To enhance positioning in urban areas, several methods are used today. Examples include the launching of additional satellites, WiFi fingerprints [1], the usage of cell phone signals, dead reckoning using inertial sensors, map-matching algorithms, magnetic compass, and barometric altimeter. However, these techniques improve the continuity and robustness, whereas GNSS fingerprinting can also improve the cross-street accuracy of the position solution [10].

GNSS fingerprinting uses a different approach for positioning compared to GNSS positioning, as it makes use of blocked as well as direct signals to its advantage. Therefore, no pseudorange information is required. To determine the visibility to satellites, a 3D model of buildings is introduced. Since the visibility to a satellite changes depending on the location of the user and the building's shape, determination of the user's position based on the knowledge of visible and invisible satellites is possible. The pre-computation of satellite visibilities for a predefined user location (reference point) is referred to as fingerprint. It uses the 3D city model and satellite positions for a fingerprint computation. As satellite positions are time-dependent, the fingerprint is only valid for a specific point in time. By comparison of visibility measurements taken by a GNSS receiver and the pre-computed model - the fingerprint - an estimation of the possible user location can be obtained. The proposed method is applicable on a smartphone, as satellite visibilities can be deviated from the signal to noise ratio (SNR) measured by GNSS receivers integrated in these devices. The fingerprint computation itself is carried out on a server, as it is the most demanding part of GNSS fingerprinting concerning the number of mathematical operations. Thus, the computational effort of matching a fingerprint, obtained from the server, to the measured satellite visibilities is low, and can therefore be executed on a smartphone. An additional advantage is, that there is no necessity of unobstructed view to four satellites as in conventional GNSS positioning. The method works for areas with no visibility to satellites as well.

It should be stated, that the outcome of GNSS fingerprinting is no absolute position solution, but a probability of user locations for a predefined grid of given positions. However, by introducing a location from another positioning technique with a provided standard deviation, this position can be enhanced in accuracy by the proposed method. Investigations showed, that especially the cross-street accuracy was significantly enhanced [10]. Therefore, the method is applicable for street-level applications, which require a sufficient positioning accuracy; examples are the lane detection of lane guidance systems, intelligent transportation systems, location-based services and advertisement, augmented-reality applications as well as step-by-step guidance for the visually impaired [11].

1.2. State of Research

GNSS fingerprinting is a relatively new approach for improving the positioning accuracy in urban areas. Until today, only a few investigations have been conducted in that area. Especially two papers can be emphasised. Their results are summarised, to give an overview on the current state of the art.

In [1], the proposed technique was implemented on a smartphone, where SNR information was obtained from NMEA data. Evaluations of SNR test measurements yielded visibility models, to distinguish visible from invisible satellite signals. To mitigate the effect of incorrect estimation of satellite visibility, an allowable Hamming distance was used. An experiment around Osaka station was performed, where GPS measurements were made at the same time. It resulted in an average distance of 51.2 m between a GPS fixed position and its real position. Furthermore, the method could narrow down GPS estimated regions to 17% with the correct ratio of 81% (number of correct estimations over the total number of estimations).

The work of [11] showed, that by improving the computational efficiency, GNSS fingerprinting could be brought from a personal computer solution proposed in earlier works, to a smartphone. Furthermore, [11] introduced a scoring scheme, to weight the candidate user locations to account for diffracted and reflected satellite signals. The weighting is based on diffraction modelling and SNR. An algorithm, similar to k-nearest neighbour, has been used, to retrieve the position solution from a grid. Statistical tests on the position solution have been made, which resulted in reduction of the average cross-street error by 77.5% on average, compared to conventional GNSS positioning solutions. Also, the success rate for a correct determination of a side of the street was averaged 54.4 percent, compared to 20.9% for the conventional positioning solution. The across-street accuracy was enhanced significantly, but the conventional solution outperformed GNSS fingerprinting in the accuracy of the along street direction.

The thesis at hand is based on the ideas of the above mentioned papers. Where methods have been implemented, reference to the respective work is given.

1.3. Problem statement

The objective of this thesis is to investigate an approach for GNSS fingerprinting, which employs the visibilities to satellites as observations. The basis of this approach is established by introducing a 3D model of buildings in the area of study. Therefore, the introduction to the theory of 3D models, and the creation of a 3D model in the area of the university campus is required. The resulting surface model is used to calculate the line of sight to satellites from reference points at any given time. Additionally, a database is to be created, which contains information about time and visibility to satellites for each reference point by means of the line of sight calculations. A dataset of coorinates of the reference points has to be known. Fingerprints from the generated database should then be compared to measured visibilities, which are obtained from the signal to noise ratio of signals measured by a GNSS receiver. The comparison yields the user's position. Finally, the GNSS fingerprinting should be evaluated.

1.4. Outline of the thesis

The thesis is sub-divided into five chapters, of which the first is the introduction to the topic. Basic theories to develop a GNSS fingerprinting are discussed in chapter 2. The first topic covers the fundamentals of fingerprints and GNSS fingerprints. The proof of visibility in 3D provides the mathematical background to calculate, whether a line of sight between a reference point to a satellite exists. Additionally, the methodology to obtain satellite positions from a set of Kepler parameters is given.

Chapter 3 introduces the methods for developing a GNSS fingerprinting. Initially, the data preparation is covered, which is about the generation of a 3D model and reference points from a set of input data. Three different approaches for a line of sight computation are presented in the following, including deliberations about accuracy and computational effort, as well as comparison of the methods. In the following, the development of satellite positions, GNSS fingerprints and the therefore required projections and transformations are discussed.

Investigations on measured signal to noise ratios commence chapter 4. Both an insight on finding an appropriate SNR threshold for satellite visibility detection and comparison between two antennas is given. Additionally, the reduction of measurement errors concerning SNR measurements is presented. The calculated fingerprints are evaluated in chapter 4.2. A trajectory has been measured and the results have been compared to GPS positioning solutions. A final overview of effects on shadow matching accuracy is given in the last section of chapter 4.

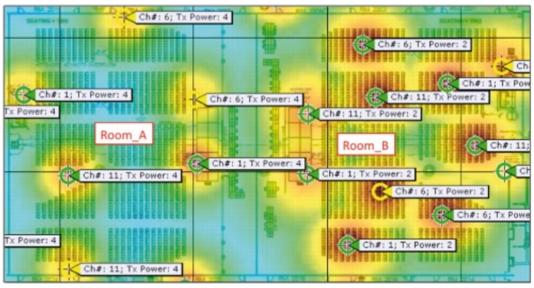
All conclusions drawn in the course of this thesis are summarised in the final chapter 5. Furthermore, a short outlook, on what is possible for future work and which aspect in daily life can benefit from GNSS fingerprinting, concludes the thesis.

2. Fundamentals

The fundamentals cover basic concepts, on which GNSS fingerprinting is based on. It gives an overview of fingerprints in general and GNSS fingerprints in particular (section 2.1). The concepts described in chapter 2.2 are the mathematical representation of Intersections in 3D. They are used for the computation of line of sights, in particular for the Intersection in 3D and the Sky Plot approach which are described in more detail in chapter 3.3. For the line of sight computation, in addition to the mathematical background, the positions of satellites at a given time have to be know which is addressed in chapter 2.3.

2.1. Fingerprints

Definition of fingerprints In the context of positioning, a fingerprint refers to a data set of properties, that link a physical value to a location. Properties can be the Received Signal Strength (RSS) in the case of Wireless Local Area Network (WLAN) for the purpose of indoor positioning or the Signal to Noise Ratio (SNR) in the field of GNSS-based fingerprinting, but other measurable physical variables are conceivable. A fingerprint is recorded at a certain time and is dependent on the location of observation. Fingerprints are correlated with a certain location, and therefore consist of a feature (it can be time-based), which is bound to coordinates. With a set of fingerprints, a database can be established, before the actual positioning takes place. A user, who requires a positioning solution, measures the RSS or SNR values and withdraws fingerprints from the database. Positioning algorithms compare the measured observations and the fingerprints to provide a possible location within the boundaries of the recorded fingerprints. See figure 2.1, which shows an example of a WLAN fingerprinting.



Source: http://www.cisco.com/web/JP/product/hs/wireless/airo3500/prodlit/images/design_guide_c07-693245-36.jpg

Figure 2.1.: WLAN Fingerprinting

The creation of fingerprints and the corresponding database is conducted in the *Offline Training Phase*. Using the example of WLAN fingerprinting, the respective RSS are measured at reference locations within and/or outside a building. The user then launches the *Online Determination Phase* by measuring a RSS sample. The most appropriate fingerprint - that means the fingerprint which best correlates to the measurements - results in the estimated user position.

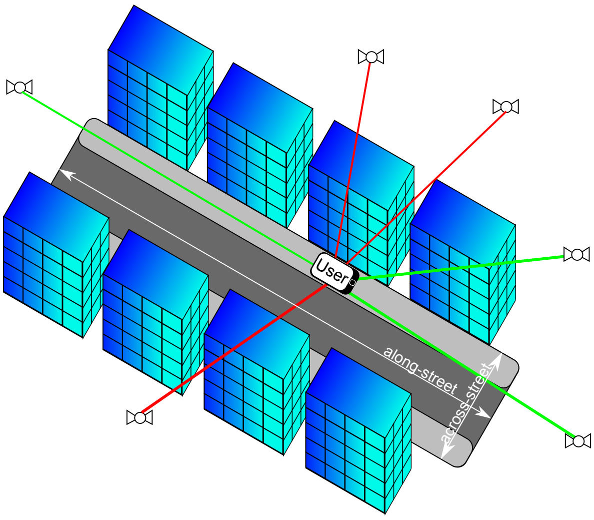
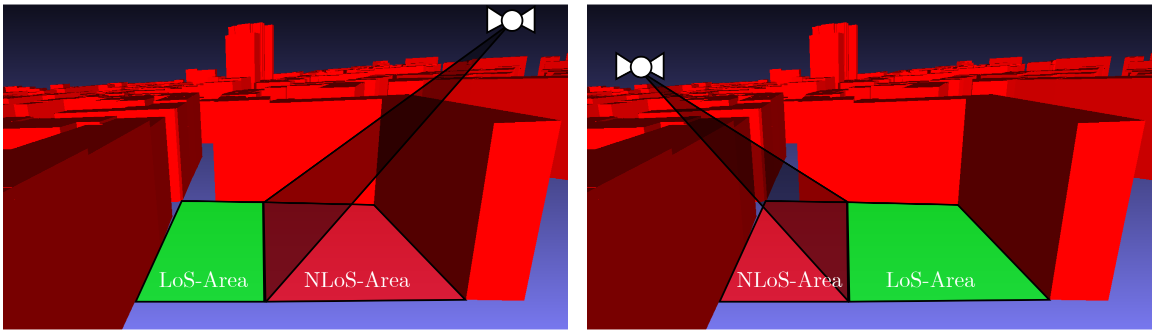


Figure 2.2.: Satellite visibilities in urban canyons

GNSS Fingerprinting GNSS fingerprinting is basically based on the same principle as WLAN fingerprinting. The similarities of model and measurements yield a position. Due to high GNSS signal shadowing caused by urban canyons in metropolis, the availability and accuracy of conventional GNSS Single Point Positioning (SPP) solutions is greatly impaired. If less then four satellites are visible, GNSS SPP is not feasible. Beside the information derived from visible line of sight (LoS) signals, GNSS fingerprinting utilizes also shadowed non-line of sight (NLoS) GNSS signals for positioning. For this purpose, 3D city models and satellite positions are evaluated in a ray tracing algorithm, which determines the visibility from a satellite to a point within a metropolitan area. This step is executed in the *Offline-Phase* which is discussed in more detail below.

Figure 2.2 demonstrates the GNSS fingerprinting in the *Online Determination Phase*. Non-line of sight signals are coloured red, which are either blocked by surrounding buildings, or reflected off the environment producing multipath effects. Visible satellites (green signals) can be expected mainly along the road axis or high overhead. Thereby, in GNSS positioning, the accuracy of the along-street component - that means along the street canyons - is generally better than the across-street component [11]. In figure 2.2, the user is located on a sidewalk and measures signal to noise ratios (SNR) of GNSS signals. SNRs indicate the signal strength in Dezibel (dB) and are an indicator of whether a visibility can be achieved, or the signal is too weak to be considered as LoS. Related works [11] emphasize the ability of the proposed fingerprinting method to detect road sections and sides. Figure 2.3 illustrates the visibility depending on the angle of incidence of a GNSS signal and the receiver's position.



(a) LoS-Area on street's left-hand side, NLoS to the right

(b) NLoS-Area on street's left-hand side, LoS to the right

Figure 2.3.: LoS and NLoS in urban canyons

In the following, the work flow is described. It starts with the creation of a 3D city model, continues with the computation of lines of sight and finishes with the matching of a fingerprint model from the database (DB) to the receiver's measurements. Some steps are computationally intensive, but must be carried out only once. Others have to be executed for every new positioning request. An overview of the entire process is shown in figure 2.4.

I. **Data Preparation** The aim of the Data Preparation Phase is to provide a 3D city model as accurately as possible. Nowadays, such models are available in urban areas with high degrees of detail. In this thesis however, a 3D model of buildings in a limited area in Graz is created for lack of an appropriate model. In order to define reference locations for the line of sight computations, a three-dimensional grid of points is required.

   The 3D model of buildings as well as the 3D grid points are created from a digital elevation model (DEM), building groundplans and 2D coordinates of grid points respectively. The building groundplans and the grid point coordinates are provided as shapefiles. Once generated, the models are stored in a spatial DB as 3D geometries due to the simplified data handling of geometries (geodata) in this type of DB. Since buildings of cities and their groundplan do not change considerably over a period of a few months, the creation of a 3D model has to be done only once. This model can then be considered valid for a longer period of time. The computational effort for the creation of a 2.5D model are very low to low, depending on the extent. The more detailed a model is desired, the more computationally intensive the generation becomes. However, a detailed modelling of roofs and buildings in general improves the accuracy of shadow matching significantly. Section 3.2 describes the creation of a 3D city model in more detail.

II. **Offline Training Phase** In the Offline Training Phase, satellite positions on the one

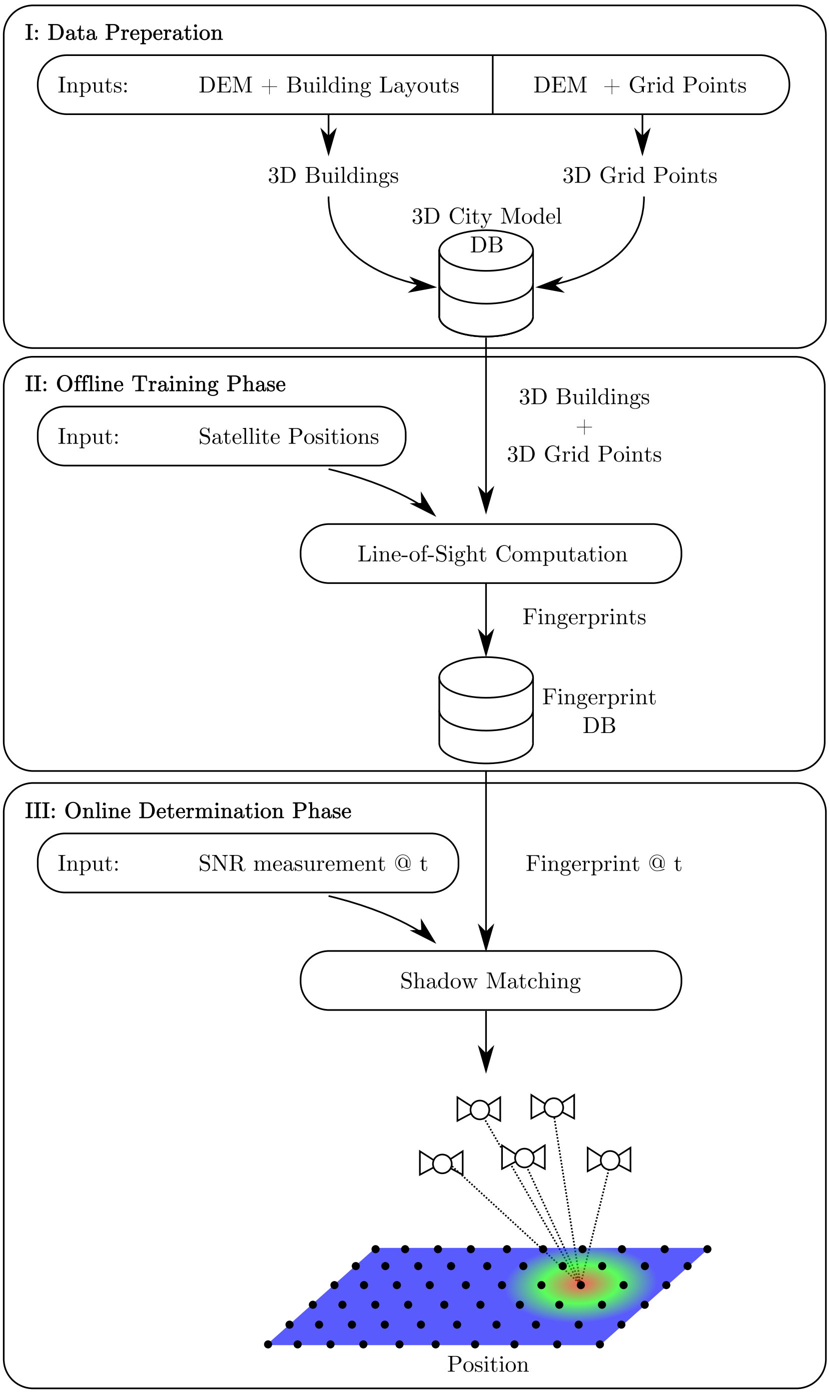


Figure 2.4.: Processes of GNSS Fingerprinting

hand and 3D models on the other hand serve as input. The satellite positions are given at a particular point in time, whereas the 3D models are retrieved from the database. The line of sight computation, which is part of the Offline Training Phase, calculates the visibilities of every grid point to all given satellite positions. In this thesis, three different ray tracing approaches were implemented and compared. The result of the LoS computations corresponds to the visibility of the defined reference points for each satellite, which are stored in the fingerprint DB. The structure of the DB will be outlined in section 3.5.2.

The fingerprints are subject to satellite positions and therefore time-dependent. A computation in advance is possible, nevertheless a processing only a short time before the Online Determination Phase is recommended, since satellite positions are calculated from broadcast ephemeris and are not predictable for longer periods. Satellite orbits are perturbed by gravitational and non-gravitational forces, therefore the calculated positions are valid for +/- two hours only. However, a projection further into the future is still possible, with the disadvantage of a decrease of precision in the determination of satellite positions. Chapter 2.3 deals with the computation of satellite positions from broadcast ephemeris and chapter 3.4.2 investigates the effect of various sources of errors on the accuracy of satellite position calculation.

III. **Online Determination Phase** The final step in GNSS fingerprinting is the positioning of the user within the grid. The measured SNR values - either measured by a GNSS receiver or a build-in GNSS smartphone receiver - and a time-stamp serve as inputs. The user fetches the fingerprint for the time t specified in the time stamp. One fingerprint corresponds to a matrix with the dimension *number of satellites* times *the number of grid points*. Comparing the actual visible and invisible satellites to the visible/invisible satellites of every grid point (DB) yields a grid with information on the level of accordance. The points with the highest probability, that is the highest level of accordance, are regarded as positioning results. Because of the possibility of obtaining more than one grid point with the highest probability, the positioning results can be ambiguous.

The evaluation of matching takes place in real time and may well be carried out on a smartphone [10]. In this thesis, the matching is computed in post-processing as the representation and evaluation of results in MATLAB is more advanced than on a smartphone. The mathematical formulation of shadow matching, which is needed for comparison of the measurements to the pre-computed fingerprint, is given in section 3.5.1.

Characteristics The characteristics of GNSS fingerprinting were partially mentioned in the previous sections. It can be expected, that the across street accuracy is better than the along-street component, opposing to GNSS positioning. Additionally, conventional GNSS positioning requires at least four satellites for providing a result. Therefore, an important aspect is the possible deployment of shadow matching in areas where less than four satellites are visible.

2.2. Proof of visibility in 3D

One of the approaches to calculate the visibility to satellites (see chapter 3.3) uses the mathematical formulations of planes and lines, where planes of building polygons are intersected in space by a line of sight. The following chapter presents the definition of lines and planes, as well as geometric operations for the proof of visibility.

2.2.1. Line

A three-dimensional line in the euclidean space can be expressed in the parameter or normal form. A line of sight between a grid point and a satellite is a line segment. Lines are used when calculating the intersection of building polygons and lines in space (used for ray tracing in general) and when determining whether a line intersects another line (used for LoS determination in 2D, see chapter 3.3.6).

Parameter form Equation (2.1) shows the parameter form of a line l, consisting of a position vector $\mathbf{r}_0$ and a direction vector $\mathbf{r}$ scaled by a scalar s. A ray is defined by an initial point formed by $\mathbf{r}_0$ which divides the line into two parts, each facing in another direction with a positive or negative scalar s.

$$l: \quad \mathbf{x} = \mathbf{r}_0 + s\mathbf{r} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + s \begin{bmatrix} x_d \\ y_d \\ z_d \end{bmatrix} \tag{2.1}$$

Normal form The normal form of a line is shown in eq. (2.2), where the intersection of two planes (p_1, p_2) yields a line in space. The vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ indicate possible normal vectors perpendicular to the vector $\overline{\mathbf{xr}}_0$, so that the scalar product is zero.

$$\begin{aligned} p_1: &\quad \mathbf{n}_1 \cdot (\mathbf{x} - \mathbf{r}_0) = 0 \\ p_2: &\quad \mathbf{n}_2 \cdot (\mathbf{x} - \mathbf{r}_0) = 0 \end{aligned} \tag{2.2}$$

2.2.2. Plane

The plane in the three-dimensional Euclidean space, on which a polygon is defined, is mathematically described by formulas (2.3) to (2.8). The intersection computation of polygons and line segments are based on the intersection of planes and lines.

Point-normal form A plane normal vector $\mathbf{n}$ and a vector $\overline{PP_0}$ are perpendicular, if P is on a plane p. P is an arbitrary point within the plane with a position vector $\mathbf{r}$ and P_0 represents the plane's origin P_0 with its origin vector r_0.

$$p: \quad \mathbf{n} \cdot (\mathbf{r} - \mathbf{r}_0) = 0 \tag{2.3}$$

Coordinate form Reformulating the point-normal form of a plane by expressing the vectors by their components yields

$$p: \quad \mathbf{n} \cdot (\mathbf{r} - \mathbf{r}_0) = a(x - x_0) + b(y - y_0) + c(z - z_0) = 0, \tag{2.4}$$

which is a linear equation and can be shortened to:

$$p: \quad ax + by + cz + d = 0, \tag{2.5}$$

$$\text{where } d = -(ax_0 + by_0 + cz_0) \tag{2.6}$$

Parameter form A plane may be described in the parameter form, so that the vectors $\mathbf{v}$ and $\mathbf{w}$ are pointing along the plane, starting at $\mathbf{r}_0$. The vectors $\mathbf{v}$ and $\mathbf{w}$ are linearly independent vectors which can be perpendicular but not parallel.

$$p: \quad \mathbf{r} = \mathbf{r}_0 + s\mathbf{v} + t\mathbf{w} \tag{2.7}$$

Calculation of a plane from three points The cross-product of two vectors in space is perpendicular to the plane p formed by the vectors.

$$p: \quad \mathbf{n} = (\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1) \tag{2.8}$$

2.2.3. Proof of the visibility based on geometric operations

The geometric operations listed in this section are used for the ray tracing algorithm implemented in this thesis. The operations include:

- Intersection of lines
- Intersection of plane and line
- Point in polygon validation

Further information, of how the operations are used, can be found in chapter 3.3.2.

Intersection of lines The intersection of lines in the context of this thesis is used for the LoS computation in 2D (see chapter 3.3.6). An intersection of line l_a and line l_b is where the coordinates are equal:

$$\begin{aligned} l_a: \quad \mathbf{x} \quad &= \mathbf{r_0} + s\mathbf{r} \\ l_b: \quad \mathbf{x}' \quad &= \mathbf{r_0}' + s'\mathbf{r}' \end{aligned} \tag{2.9}$$

$$\begin{aligned} \mathbf{x} &= \mathbf{x}' \\ \mathbf{r_0} + s\mathbf{r} &= \mathbf{r_0}' + s'\mathbf{r}' \\ (\mathbf{r_0}' - \mathbf{r_0}) &= s\mathbf{r} - s'\mathbf{r}' \end{aligned} \tag{2.10}$$

$$\mathbf{y} = (\mathbf{r_0}' - \mathbf{r_0}) = s\mathbf{r} - s'\mathbf{r}' \tag{2.11}$$

Formula 2.11 represents a linear equation system and can thus be solved in matrix notation for the unknown scalar parameters s and s'. Substitution of the parameter vector ξ (s and s') into eq. (2.1) yields the coordinates of the intersection point $\mathbf{x}_{\text{int}}$.

$$\begin{aligned} \mathbf{y} &= [\mathbf{r}, -\mathbf{r}'] \begin{bmatrix} s \\ s' \end{bmatrix} \\ \mathbf{y} &= \mathbf{A}\xi \\ \mathbf{A}^{-1}\mathbf{y} &= \xi \end{aligned} \tag{2.12}$$

$$\mathbf{x}_{\text{int}} = \mathbf{r}_0 + s\mathbf{r} = \mathbf{r}_0' + s'\mathbf{r}' \tag{2.13}$$

If $\mathbf{r}$ and $\mathbf{r}'$ are parallel, then no intersection is possible. Thus, when solving the linear equation, a linearly dependent design matrix is formed. Therefore, this cases have to be omitted in the calculation of intersections. The parallelism of lines can be determined by comparing both slopes:

$$\begin{aligned} \text{slope}_{l_a} &\neq \text{slope}_{l_b} \\ \left|\frac{r_y}{r_x}\right| &\neq \left|\frac{r_y'}{r_x'}\right| \end{aligned} \tag{2.14}$$

Intersection of plane and line The 3D intersection of a line representing a vector from a grid point to a satellite and a plane of a building's polygon is used in the first approach of LoS computation: Intersections in 3D. The algorithm of the computation of intersections corresponds to [7] and can be divided into five steps. The vectors and scalars used in the algorithm are described in table 2.1 below.

Table 2.1.: Vectors and scalars of the intersection of plane and line

Vector/Scalar		Description
$\mathbf{r_{d0}}$	$[x_{d0}, y_{d0}, z_{d0}]^T$	satellite position
$\mathbf{r_0}$	$[x_0, y_0, z_0]^T$	position vector to plane, i.e. one point of a building-polygon
$\mathbf{r_d}$	$[x_d, y_d, z_d]^T$	direction vector from satellite to grid point
$\mathbf{n}$	$[a, b, c]^T$	normal vector of plane
s		distance from satellite to intersection point

1. Insert the components of the parameter form of lines (eq. 2.1) into the coordinate form of a plane (eq. 2.5):

   $$a(x_{d0} + sx_d) + b(y_{d0} + sy_d) + c(z_{d0} + sz_d) + d = 0 \tag{2.15}$$

   with

   $$d = -(ax_0 + by_0 + cz_0) = - < \mathbf{n}, \mathbf{r_0} > \tag{2.16}$$

2. The above equation has to be solved for the distance s to determine the intersection point along the line.

   $$s = \frac{-(ax_{d0} + by_{d0} + cz_{d0} + d)}{ax_d + by_d + cz_d} \tag{2.17}$$

   $$s = \frac{-(< \mathbf{n}, \mathbf{r_{d0}} > + d)}{< \mathbf{n}, \mathbf{r_d} >} = \frac{v_0}{v_d} \tag{2.18}$$

3. If the scalar product of $< \mathbf{n}, \mathbf{r_d} > = v_d > 0$, then the ray points away from the plane normal. This is the case, when the angle between the normal vector $\mathbf{n}$ and the direction vector $\mathbf{r_d}$ from a satellite to a grid point is greater than -90°, but is less than 90°. However, it is desired that the surface normal faces the ray. This is accomplished by changing the direction of the plane normal $\mathbf{n}$ and therefore the plane itself:

   $$< \mathbf{n}, \mathbf{r_d} > = |\mathbf{n}||\mathbf{r_d}| \cos(\mathbf{n}, \mathbf{r_d}) \qquad \cos(\mathbf{n}, \mathbf{r_d}) > 0 \text{ if } -90° < \alpha < 90° \tag{2.19}$$

Listing 2.1: Inverting of the direction of planes

```
        if vd<0:
            n = n;
        else:
            n = -n;
```

If the direction of $\mathbf{n}$ is inverted, the scalars v_0 and v_d have to be recalculated, as they depend on the normal vector $\mathbf{n}$.

4. If $s < 0$, then the line intersects the plane behind the origin of the line, thus no actual intersection occurs. Otherwise the intersection point can be calculated:

$$\mathbf{r_i} = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} x_{d0} + sx_d \\ y_{d0} + sy_d \\ z_{d0} + sz_d \end{bmatrix} \tag{2.20}$$

5. The last step is to proof, if the intersection point r_i lies within the polygon.

Point in polygon validation Two methods have been obtained from [4] and tested for their applicability in 3D space. For matters of completeness, all methods are listed together with notes, why they work or do not work for testing an intersection point for being within a polygon in $\mathbb{R}^3$.

I For 2D polygons, the angles θ_i between the vectors from grid point to vertices of a polygon can be calculated:

- $\sum \theta_i = 2\pi$: point is in the interior of the polygon
- $\sum \theta_i = 0$: point is outside the polygon

Note: This algorithm does not work, as it is only possible in 2D; re-projection of a polygon and point from 3D to 2D would have to be done prior, which is time consuming. Furthermore, the formula $\cos\theta = \frac{<\mathbf{a},\mathbf{b}>}{|\mathbf{a}||\mathbf{b}|}$ is not appropriate, as it always returns a positive angle. The sum of angles will never end up to be $\sum \theta = 0$ accordingly.

II Calculate the area A^p between vectors from an arbitrary point p to vertices of a polygon v and compare to the polygon's area A:

$$insidePolygon(p) = \begin{cases} \sum A_i(\mathbf{v}_i, \mathbf{v}_{i+1}) = \sum(A_i^p(\mathbf{pv}_i, \mathbf{pv}_{i+1})), & \text{true} \\ \sum A_i(\mathbf{v}_i, \mathbf{v}_{i+1}) \neq \sum(A_i^p(\mathbf{pv}_i, \mathbf{pv}_{i+1})), & \text{false} \end{cases} \tag{2.21}$$

where $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ indicate vectors from an arbitrary point within the polygon (e.g. centroid) to the polygons vertices i and $i+1$. The area of the polygon A is computed by adding the area of all sub-areas A_i. The vectors $\mathbf{pv}_i$ and $\mathbf{pv}_{i+1}$ are defined by an arbitrary point p in $\mathbb{R}^3$ to the vertices of the polygon i and $i+1$. The area between the vectors $\mathbf{pv}_i$ and $\mathbf{pv}_{i+1}$ is A_i^p. If a point p is in the interior of the polygon, the sum of A_i and A_i^p have to be equal, otherwise the area of $\sum A_i^p$ is always greater than $\sum A_i$.

A_i and A_i^p are calculated by:

$$A_i = \frac{1}{2}|\mathbf{v}_i \times \mathbf{v}_{i+1}| \qquad A_i^p = \frac{1}{2}|\mathbf{p} \times \mathbf{v}_i| \tag{2.22}$$

Note: This will not work for polygons that are convex. The solution is provided by MATLAB's `inpolygon` function for 2D, which was used for the top and bottom of each building, as these are the only convex polygons in the 3D model.

2.3. Satellite Positions

Satellite positions given at specific times are required for LoS computation. Employing the computed positions, shadowing of buildings in urban areas can be determined in advance. From this, the fingerprints are computed. Unlike single point positioning (SPP), the position of each GPS satellite for a given point in time t has to be predetermined in order to get real time positioning. This is done by using broadcast ephemeris. Broadcast ephemeris are transmitted by satellites, but can also be provided later on, for example in the form of RINEX files.

In SPP the satellite position is calculated iteratively together with the user's position and the receiver clock error through a least squares adjustment (for an estimation on the effect of neglect of the receiver clock error, see section 3.4.2 on page 70). This can be accomplished in real time. Since the pseudoranges and thus a least squares approach can not be used in GNSS fingerprinting, an iterative approach is not possible. Another drawback is the computation time of the fingerprint, whereby a calculation in real time is not possible. This is not to be confused with the matching, which can be performed in real time indeed (see chapter 3.5.1). The following chapter provides an introduction to the topic of orbit determination based on Kepler parameters transmitted by broadcast ephemeris, and the derivation of parameters from the Newton mechanics. The implementation is presented in chapter 3.4.

2.3.1. Kepler Orbit

From Newton mechanics follows, that two bodies in space attract each other with a gravitational force. Assume, that body one has a mass of m_1 and body two a mass of m_2 respectively. The relative movement between the masses can be expressed by following the law of universal gravitation. It is a homogeneous differential equation of second order:

Table 2.2.: Quantities of Newtons law of universal gravitation

$\mathbf{r}$	relative position vector
$\ddot{\mathbf{r}}$	relative acceleration vector
G	universal gravitational constant

$$\ddot{\mathbf{r}} + \frac{G(m_1 + m_2)}{r^3}\mathbf{r} = \mathbf{0} \tag{2.23}$$

For a satellite that orbits the earth, the ratio $\frac{m_{\text{sat}}}{m_{\text{earth}}}$ is near to zero, thus one can neglect the satellites' mass. Solving the homogeneous differential equation yields six integration constants which are represented by the six Kepler orbital parameters for two integrations of a three-dimensional vector equation. Kepler used his first two laws of planetary motion established in 1609 to determine the six parameters empirically. The six parameters are listed in table

Table 2.3.: Keplerian orbital parameters

Ω	Right ascension of ascending node
i	Inclination of orbital plane
ω	Argument of perigee
a	Semimajor axis of orbital ellipse
e	Numerical eccentricity of ellipse
T_0	Epoch of perigee passage

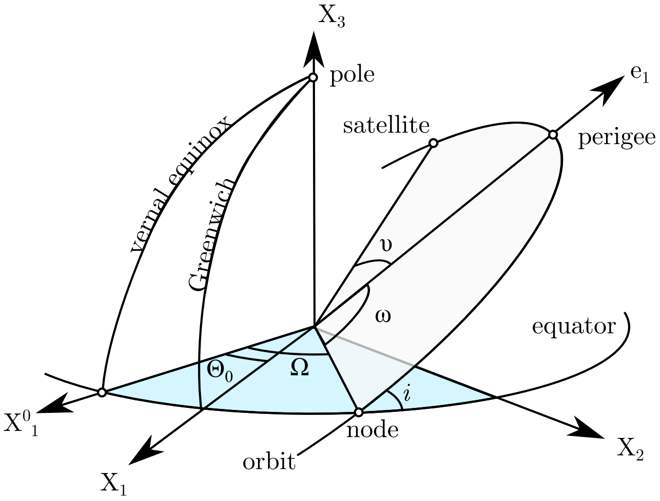


Figure 2.5.: Kepler Orbit and parameters

2.3. Using these parameters, a satellite's position in an elliptic orbit around the earth's centre of mass can be determined. Figure 2.5 illustrates a satellite orbiting the earth. The right ascension of the ascending node Ω, the inclination i and the argument of perigee ω describe the alignment of the satellite's orbit referred to the vernal equinox and the equator.

The perigee is the satellite's closest approach to the earth's centre of mass, whereas the apogee is the furthest. The position within the orbital plane is determined by an angular quantity called the true anomaly ν. It can be computed by knowing the revolution period P and thus the mean satellite's angular velocity n (eq. 2.24), as well as the mean anomaly M and the eccentric anomaly E. The mean anomaly is a mathematical abstraction, while the eccentric and the true anomaly can be visualized graphically (see figure 2.6). Equation 2.25 gives the formulas for the according anomalies.

$$n = \frac{2\pi}{P} = \sqrt{\frac{GM}{a^3}} \tag{2.24}$$

$$
\begin{aligned}
M(t) =& n(t - T_0) \\
E(t) =& M(t) + e \sin E(t) \\
\nu(t) =& 2 \arctan \left[\sqrt{\frac{1+e}{1-e}} \tan \frac{E(t)}{2} \right]
\end{aligned} \tag{2.25}
$$

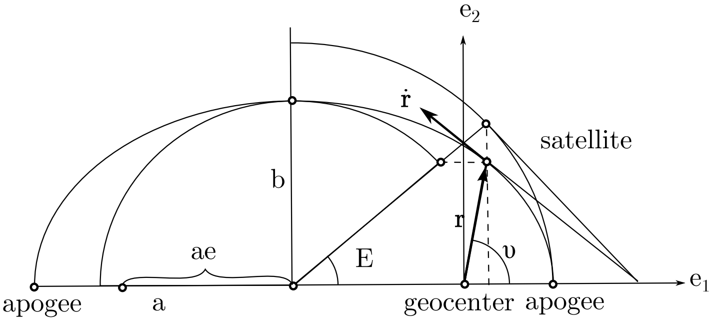


Figure 2.6.: Anomalies and orbital ellipse

Satellite position in the orbital plane Figure 2.6 shows the satellite's position in the orbital plane. With the true and the eccentric anomaly given, the satellite position $\mathbf{r}$ and speed $\dot{\mathbf{r}}$ can be obtained by:

$$
\mathbf{r} = a \begin{bmatrix} \cos E - e \\ \sqrt{1-e^2} \sin E \end{bmatrix} = r \begin{bmatrix} cos\nu \\ sin\nu \end{bmatrix} \tag{2.26}
$$

$$
\dot{\mathbf{r}} = \frac{d\mathbf{r}}{dt} = \frac{na^2}{r} \begin{bmatrix} -\sin E \\ \sqrt{1-e^2} \cos E \end{bmatrix} \tag{2.27}
$$

with

$$
r = a(1 - e \cos E) = \frac{a(1-e^2)}{1 + e \cos \nu} \tag{2.28}
$$

Figure 2.5 introduces two coordinate systems, the Earth Centred Earth fixed (ECEF) and the Earth Centred Space Fixed (ECSF). Both systems are centred in the earth's centre of mass (in the case of WGS84) and share a common X_3 axis through the earth's pole. The ECFS can be distinguished from ECEF system by the x-axes X_1^0 and X_1 respectively. Their X_2 axes are orthogonal to their X_1 and X_3 axes. The transformation from the orbital plane to ECSF is accomplished by means of three rotation matrices using the angles argument of perigee ω, inclination i and the right ascension of the ascending node Ω, in this order. The satellite position in the ECSF, donated as $\boldsymbol{\rho}$, is therefore defined by

$$
\boldsymbol{\rho}_{ECSF} = \mathbf{Rr} = \mathbf{R}_3(-\Omega)\mathbf{R}_1(-i)\mathbf{R}_3(-\omega)\mathbf{r} \tag{2.29}
$$

To determine the satellite position $\boldsymbol{\rho}_{ECEF}$ (eq. 2.30) in the ECEF system, an additional rotation about the z-axis by the Greenwich sidereal time Θ_0 has to be performed. This is the angle between the vernal equinox and the Greenwich meridian.

$$\boldsymbol{\rho}_{ECEF} = \mathbf{R}_3(\Theta_0)\mathbf{R}_3(-\Omega)\mathbf{R}_1(-i)\mathbf{R}_3(-\omega)\mathbf{r} \tag{2.30}$$

Perturbed orbit A disturbance of the orbit occurs as a result of gravitational and non-gravitational forces. Hence gravitational forces from the sun and the moon exert influence on the orbit, but also frictional forces, solar pressure and other forces affect the orbital parameters. A distinction is made between long-periodic and short-periodic disturbances. Due to the influences, the parameters and thus the orbit as well as the orbit's position and alignment in space are time dependent. This is modelled by the disturbing accelerations $d\ddot{\mathbf{r}}$. The corrections of the time varying interference terms is applied to a reference orbit at a reference time (see paragraph Corrections of section 2.3.2 at page 21).

$$\ddot{\mathbf{r}} + \frac{G(m_1 + m_2)}{r^3}\mathbf{r} = d\ddot{\mathbf{r}} \tag{2.31}$$

2.3.2. Orbit Determination

The determination of satellite orbits from broadcast ephemeris is discussed in this chapter, which is based on the principles of chapter 2.3.1. The formulas from the Navstar GPS Space Segment Document [12, p. 103-105] are used. Chapter 3.4 describes the implementation of calculations.

Ephemeris and input parameters For the calculation of the satellite's position, the time and the approximate user position are required in addition to the ephemeris. The broadcast ephemeris and their description are listed in table 2.6. Table 2.4 gives additional input parameters. The computation time t must be in GPS time (GPST) of the respective GPS week and the approximate user position is used for the estimation of the signal's transit time. Since the position information of the GPS satellites are based on the WGS84 ellipsoid, the corresponding parameters are provided in table 2.5. These include the geometric parameters a and b, describing the shape of the ellipsoid, as well as the physical parameters GM and ω_E.

Table 2.4.: Input parameters

	Unit	Description
t	[sec]	GPS system time within current GPS week
$\mathbf{x}_{R_{approx}}$	[m]	Approximate User position

Table 2.5.: WGS84 Constants

		Unit	Description
GM	3.9860044418e14	$m^3\ s^{-2}$	Geocentric Gravitational Constant
ω_E	7.2921151467e-5	rad/s	Earth's angle of rotation
a	6378137.0	m	Earth's semi major axis
b	6356752.3142	m	Earth's semi minor axis

Table 2.6.: Ephemeris Input parameters

Orbital Parameter	Description
t_c	Satellite clock reference epoch
t_e	Ephemeris reference epoch
$\sqrt{a}$	Square root of the semi major axis of the satellite orbit
Δn	Mean motion difference
M_0	Mean anomaly at reference epoch
e	Eccentricity
ω_0	Argument of perigee
C_{uc}, C_{us}	Correction coefficients (argument of perigee)
C_{rc}, C_{rs}	Correction coefficients (geocentric distance)
C_{ic}, C_{is}	Correction coefficients (inclination)
i_0	Inclination
$\dot{i}$	Rate of Inclination angle
Ω_0	Right ascension of ascending node
$\dot{\Omega}$	Rate of node's right ascension

UTC time to GPST If the input time t is given in UTC, it has to be converted to GPST (GPS Time) first. This conversion is covered in chapter 2.3.3 and results in the GPST for the input time, donated as t_R^{GPST} in the following.

Adjustment of the satellite time by transit time The transit time t_T has to be considered and applied to the time t_R^{GPST}, which then represents the satellite time t. This presupposes the user position to be known to calculate the difference vector between a satellite position and the user position. As the satellite position is not known at this time of computation, an iteration has to be initiated. The transit time can be considered after the first iteration. By calculating the distance between an approximate user position $\mathbf{x}_{R_{approx}}$ in the ECEF system and the calculated satellite position $\mathbf{x}_{Satellite}^{i-1}$ of the previous iteration $i-1$, divided by the speed of light c, an approximation for the transit time can be conducted. A final satellite position is obtained by defining a termination condition for the iteration over $\mathbf{x}_{Satellite}^{i}$. A suitable termination condition would be the comparison of the distance of two consecutive satellite position solutions $\mathbf{x}_{Satellite}^{i-1}$ and $\mathbf{x}_{Satellite}^{i}$ to a threshold.

$$t = t_R^{GPST} - t_T, \tag{2.32}$$

with

$$t_T = \frac{|\mathbf{x}_{Satellite}^{i-1} - \mathbf{x}_{R_{approx}}|}{c} \tag{2.33}$$

Time t_k The time of ephemeris t_e (TOE), which is the reference time for which the ephemeris have been calculated, and the input time t differ from each other by the time t_k. This time is used for calculating the positions in the following. The input time t is the GPST time corrected for the transit time.

$$t_k = t - t_e \tag{2.34}$$

To account for the issue of week crossovers [12, p.97] suggests to check t by:

$$t_k = \begin{cases} t - t_e > 302400[sec], & t - 604800[sec] - t_e \\ t - t_e < 302400[sec], & t + 604800[sec] - t_e \end{cases} \tag{2.35}$$

where 604800 is the number of seconds of one week.

Mean Anomaly The mean anomaly is computed analogous to section 2.3.1. Additionally, the disturbances have to be considered by adding the mean motion difference.

Computed mean motion

$$n_0 = \sqrt{\frac{GM}{a^3}} \tag{2.36}$$

Corrected mean motion

$$n = n_0 + \Delta n \tag{2.37}$$

Mean anomaly at time t_k

$$M_k = M_0 + nt_k \tag{2.38}$$

Eccentric Anomaly As the eccentric anomaly is part of the solution itself, an iteration as proposed below has to be initialized, starting with the mean anomaly as first approximation of the eccentric anomaly.

$$E_0 = M_k \tag{2.39}$$

$$E_i = M_k + e \sin E_{i-1} \qquad i = 1, 2, 3...n \tag{2.40}$$

$$E_k = E_n \qquad \text{with n beeing the last index of iteration} \tag{2.41}$$

True Anomaly Below, two different computations for the true anomaly ν are given. The second definition has been used in this thesis, as it is numerically more stable.

$$\tan \nu_k = \frac{\sqrt{1-e^2} \sin E_k}{\cos E_k - e} \tag{2.42}$$

$$\nu_k = 2 \arctan \left(\sqrt{\frac{1+e}{1-e}} \tan \frac{E_k}{2} \right) \tag{2.43}$$

Argument of Latitude The true anomaly and the argument of perigee yield the argument of latitude.

$$\Phi_k = \nu_k + \omega_0 \tag{2.44}$$

Corrections The corrections are given for short-periodic disturbances of the argument of latitude, the radius and the inclination.

$$\delta u_k = C_{uc} \cos 2\Phi_k + C_{us} \sin 2\Phi_k \qquad \text{argument of latitude correction} \tag{2.45}$$

$$\delta r_k = C_{rc} \cos 2\Phi_k + C_{rs} \sin 2\Phi_k \qquad \text{radius correction} \tag{2.46}$$

$$\delta i_k = C_{ic} \cos 2\Phi_k + C_{is} \sin 2\Phi_k \qquad \text{inclination correction} \tag{2.47}$$

The corrections are applied by:

$$u_k = \Phi_k + \delta u_k \qquad \text{corrected argument of latitude} \tag{2.48}$$

$$r_k = a(1 - e \cos E_k) + \delta r_k \qquad \text{corrected radius} \tag{2.49}$$

$$i_k = i_0 + \dot{i} t_k + \delta i_k \qquad \text{corrected inclination} \tag{2.50}$$

$$\Omega_k = \Omega_0 + (\dot{\omega} - \omega_e) t_k - \omega_e t_e \qquad \text{corrected longitude of ascending node} \tag{2.51}$$

Position in the orbital plane The position in the orbital plane in Cartesian coordinates can be calculated with the corrected radius and the corrected argument of latitude.

$$\overline{X}_k = r_k \cos u_k \tag{2.52}$$

$$\overline{Y}_k = r_k \sin u_k \tag{2.53}$$

ECEF Satellite Coordinate The transformation of the satellite's position from the orbital plane to the earth centred earth fixed (ECEF) coordinate system is accomplished by:

$$X_k = \overline{X}_k \cos \Omega_k - \overline{Y}_k \sin \Omega_k \cos i_k \tag{2.54}$$

$$Y_k = \overline{X}_k \sin \Omega_k + \overline{Y}_k \cos \Omega_k \cos i_k \tag{2.55}$$

$$Z_k = \overline{Y}_k \sin i_k \tag{2.56}$$

Rotate ECEF Satellite Coordinate for transit time After the last iteration, the satellite position vector has to be rotated by α which corresponds to the angle of the earth's revolution during transit time t_T.

$$\mathbf{X}_k^{rotated} = \mathbf{R}_3(\alpha)\mathbf{X}_k^{ECEF} = \mathbf{R}_3(\omega_e t_T)\mathbf{X}_k^{ECEF} \tag{2.57}$$

2.3.3. Time Systems and Conversions

This chapter introduces time systems which have been used in this work in order to transform the common CEST or UTC to a time system suitable for the calculation of the satellites' positions. The time scale mentioned in this thesis are based on atomic clocks and are therefore highly uniform.

UTC The Universal Time Coordinated is an atomic time scale and an adoption to UT1, which is obtained by astronomical observations and is bound to the Earth's rotation. A positive or negative leap seconds is applied to UTC whenever the difference $\Delta UT1 = UT1 - UTC$ becomes larger than 0.9 seconds. This difference occurs due to the non-uniform Earth rotation.

CET and CEST The Central European Time is a time scale which has an offset of +1 hour to UTC and is used in most middle European countries. The CEST is the Central European Summer Time which has an offset of +2 hours to UTC and is used from 1:00 UTC on the last Sunday of March to 1:00 UTC on the last Sunday of October.

GPS Time The GPS time scale was implemented with the intial launch of GPS and started on the 6^{th} of January 1980 at 0:00 UTC. It is an atomic time scale and is not incremented by leap seconds which leads to an offset to UTC. The leap seconds are part of the RINEX Navigation Message and can therefore be applied to the time conversion:

$$GPS\ time - UTC = leap\ seconds \tag{2.58}$$

The GPS time is expressed in GPS weeks and seconds within this week starting with week one at the reference epoch January 6th, 1980. A GPS week starts at midnight from Saturday to Sunday and comprises 604800 seconds. To calculate the GPS weeks and seconds for a given UTC time, a conversion into the Julian Date is applicable.

Julian Date and Conversions The Julian Date (JD) is the number of solar days elapsed since January 1^{st} 12:00, 4713 before Christ. There exist standard epochs in JD which define for example the modified Julian date or the GPS standard epoch and reduce the number of days for the sake of simplicity and digit saving. Table 2.7 and the following computations are taken from [9].

Table 2.7.: Julian Date Standard Epochs

Civil Date	Julian Date	Description
January 1.5^d, 2000	2 451 545.0	Current standard epoch (J2000.0)
January 6.0^d, 1980	2 444 244.5	GPS standard epoch

The civil date in the following computations is expressed by year y, month m, day d and the hours t in the UTC time scale.

The conversion from civil date to JD is given by:

$$y = y - 1;\ m = m + 12 \qquad \text{when } m \leq 2 \tag{2.59}$$

$$JD = INT[365.25y] + INT[30.6001(m + 1)] + d + t/24 + 1720981.5 \tag{2.60}$$

The inverse conversion is:

$$a = INT[JD + 0.5] \tag{2.61}$$

$$b = a + 1537 \tag{2.62}$$

$$c = INT[(b - 122.1)/365.25] \tag{2.63}$$

$$d = INT[365.25c] \tag{2.64}$$

$$e = INT[(b - d)/30.6001] \tag{2.65}$$

Table 2.8.: Mapping of RINEX and UTC Parameters

RINEX parameter	UTC Parameter	Description	Unit
LEAP SECONDS	Δt_{LS}	leap seconds	[sec]
GPS week #	WN	current week number	[week]

and

$$d = b - d - INT[30.6001e] + (JD + 0.5) - INT[JD + 0.5] \tag{2.66}$$

$$m = e - 1 - 12\, INT[e/14] \tag{2.67}$$

$$y = c - 4715 - INT[(7 + M)/10] \tag{2.68}$$

$$t = 24(JD - INT[JD]) \tag{2.69}$$

The number of weeks elapsed from a reference epoch (for example the GPS standard epoch given in table 2.7) is:

$$WEEK = INT[(JD_{observation\ epoch} - JD_{reference\ epoch})/7] \tag{2.70}$$

Conversion from GPST in civil representation to GPST weeks and seconds Algorithm 1 gives an idea of how to convert GPST given in the form of yyymmdd and hhmmss to GPST week and GPST seconds within this week with reference to the GPS Standard Epoch.

Algorithm 1 GPST yyyymmdd hhmmss to GPST week and seconds

```
Input: t in year, month, day, hour, minute, second
$t$ to $t_{JD}$ (see Julian Date)
$t_{GPS}[week] = INT[t_{JD}$ – GPS STANDARD EPOCH]/7 with GPS Standard Epoch = 2444244.5 JD
$t_{GPS}[sec]$ = round $[(t_{GPS}[week] - t_{reference\ epoch}[week]) \cdot 7 \cdot 24 \cdot 3600]$ with $t_{reference\ epoch}$ being the ephemeris GPS week provided in the data record of the measurement of the RINEX navigation file
```

UTC to GPST The UTC parameters given in table 2.8 are used to calculate the GPST time from a given UTC time. This is used when calculating the satellites' positions for a given UTC time (see [12, p. 118,123]).

$$t_{GPST} = t_{UTC} + \Delta t_{LS} \tag{2.71}$$

3. Development

3.1. Overview

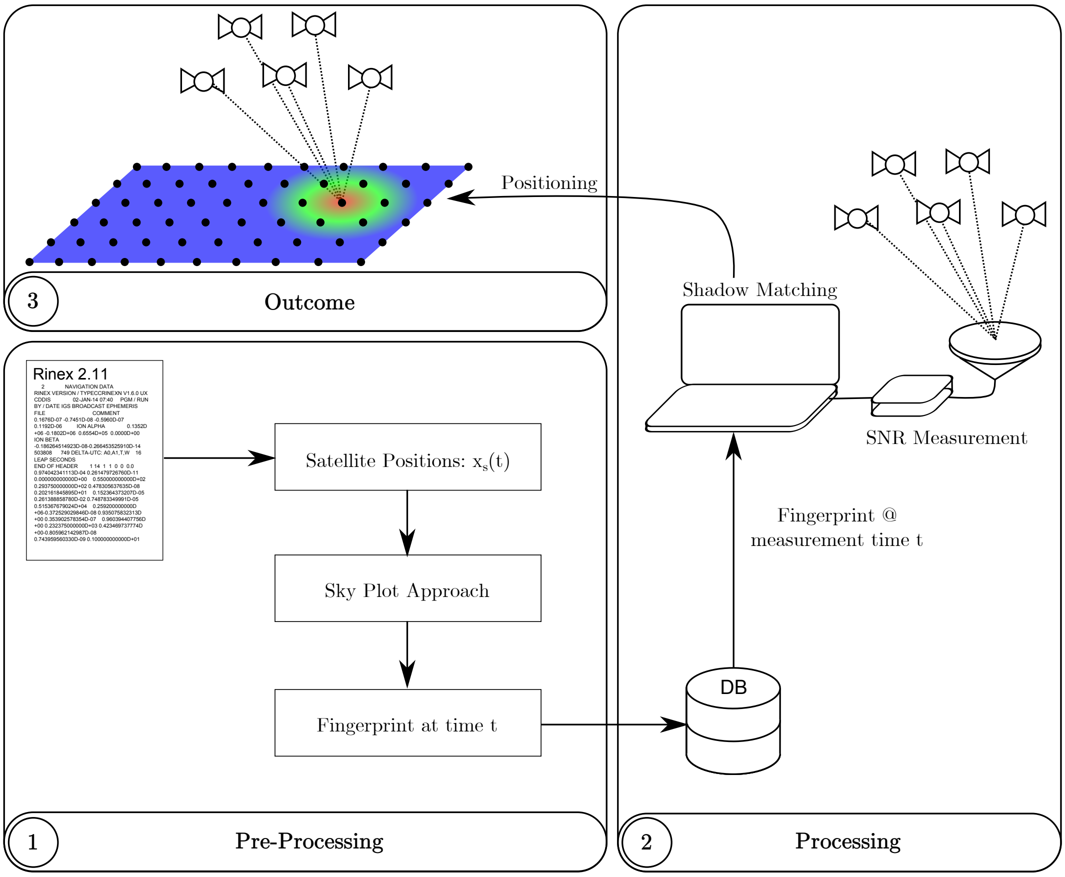
This chapter covers the processes, implementations and thoughts on GNSS fingerprinting. Data Preparation (chapter 3.2) is about the generation of 3D data, which is required for the calculations of fingerprints. To determine the line of sights, three approaches are introduced in chapter 3.3. Comparison between these methods and possible optimizations are discussed. In Satellite Positions (chapter 3.4), the implementation of satellite orbit determination and error analysis is discussed. The positioning with fingerprints is covered in the chapter 3.5.

3.1.1. General Principle of GNSS Fingerprinting

The creation of a fingerprint is subdivided into an off-line training phase and, at the very least, into a determination phase as outlined in section 2.1. In this thesis the *off-line training phase* is designated as *pre-processing step* and the *determination phase* is further divided into a *processing step* and an *outcome.* The goal when creating a fingerprint is to calculate the direct visibility to satellites for each reference location within a predefined grid. Based on the shadowing of buildings, the visibilities to satellites are strongly limited, thus resulting in different visibilities for each epoch and position within the grid. The visibilities of a grid point to all satellites are stored as a visibility vector in the database (DB), having 1 for LoS and 0 for NLoS.

Employing SNR measurements, the actual visibility of satellites is determined, which is compared to the fingerprints of the DB. The similarities of visibilities (based on the measurements) to fingerprints, give an indicator of the users position probability within the grid. Due to similarities within the user's environment, the same probabilities may occur at different locations of the grid. Especially along a street, visibilities change insignificantly. Figure 3.1 gives an overview of the different processing steps.

Data Preparation step Section 3.2 covers the creation of a 3D building model as well as a grid of reference locations. Additionally, a dataset of building boundaries seen from each grid point (reference location) is computed (see chapter 3.3.3 for its deployment). The data preparation step has to be done only once, but is very time consuming, dependent on various factors which are covered in the corresponding chapter.



```
Rinex 2.11
     2            NAVIGATION DATA
RINEX VERSION / TYPECCRINEXN V1.6.0 UX
CDDIS              02-JAN-14 07:40     PGM / RUN
BY / DATE IGS BROADCAST EPHEMERIS
FILE                              COMMENT
0.1676D-07 -0.7451D-08 -0.5960D-07
0.1192D-06          ION ALPHA              0.1352D
+06 -0.1802D+06  0.6554D+05  0.0000D+00
ION BETA
-0.186264514923D-08-0.266453525910D-14
503808       749 DELTA-UTC: A0,A1,T,W    16
LEAP SECONDS
END OF HEADER        1 14  1  1  0  0  0.0
0.974042341113D-04 0.261479726760D-11
0.000000000000D+00    0.550000000000D+02
0.293750000000D+02 0.478305637635D-08
0.202161845895D+01    0.152364373207D-05
0.261388858780D-02 0.748783349991D-05
0.515367679024D+04    0.259200000000D
+06-0.372529029846D-08 0.935075832313D
+00 0.353902578354D-07    0.960394407756D
+00 0.232375000000D+03 0.423469737774D
+00-0.805962142987D-08
0.743959560330D-09 0.100000000000D+01
```

Figure 3.1.: Fingerprint creation and shadow matching

Pre-Processing step With the computed building boundaries and the satellite positions $\mathbf{x}_S(t)$ as inputs, the Sky Plot approach (see chapter 3.3.3) calculates a visibility matrix, containing visibilities from each grid point to all satellites. The satellites' positions are derived from ephemeris given in Rinex 2.11 Navigation files provided daily by NASA's Archive of Space Geodesy Data [3]. Considering the GPS system, the ephemeris are given for 32 GPS satellites of the previous day. The algorithm for computation of the satellite position uses the latest ephemeris data of each satellite. A detailed implementation is given in chapter 3.4.

Finally, the fingerprint created by the Sky Plot approach is stored in a PostgreSQL DB, along with the time which the fingerprint has been computed for. A time reference is later needed to fetch the appropriate fingerprint for a measurement conducted at time t. The coordinates of the reference points for a fingerprint are not stored in the DB, as they do not change over time. Therefore, a coordinate dataset, which joins a grid point by ID with it's location, is sufficient.

Processing step and outcome The processing phase can also be designated as online-phase and may be computed in real-time or in post-processing, dependent on the complexity of shadow matching. The complexity correlates with the shadow matching method, as well as the detail of the 3D model and the design of the grid (size, number of grid points). The SNR measurements are performed with a conventional GPS antenna and a u-blox receiver. No C/A code solution is determined. Along with the SNR measurements, a UTC time-stamp t for each epoch is recorded. A comparison of the measurements and the fingerprint at time t fetched from the DB yields the similarities of the pre-computed model and the measurement and thus the location probability for each grid point.

3.1.2. Hardware and Software

The hardware and software used in the course of this work are listed in the tables 3.1 and 3.2. A short explanation is given for each entry.

Table 3.1.: Used hardware

Hardware	Description
u-blox 6 receiver	This receiver was provided by the Institute of Navigation of the Graz University of Technology and was used for all GPS/SNR measurements.
u-blox 6 patch antenna	The antenna was provided along with the receiver and was used for comparison to the Aero antenna.
Aero GNSS antenna	The antenna was provided by the Institute of Navigation at the Graz University of Technology and was the prime antenna used in the course of this thesis.

Table 3.2.: Used software

Software & Programming Languages	Description
QGIS	QGIS is an open-source geographic information system (GIS). Data handling and preparation with shape-files and the DEM was performed.
starspan [5]	A shell program to extract pixel values at the locations of overlaying shape files. These are available as Linux binaries only.
PostgreSQL with PostGIS extension	The database is a free and open-source software. With the PostGIS extension, geographic objects are supported and projections as well as functions for geographic data handling are provided.
MATLAB	MATrix LABoratory is a programming language and a multi-paradigm numerical computing environment. The majority of the work was implemented in Matlab.
Java	The programming language Java was used for .x3d file creation of data provided by the database.
Python	The Python programming language was used for Blender.
MeshLab	A lightweight tool, to visualize 3D models.
Blender	A 3D computer graphics software which is free and open-source.
u-center	A desktop tool, to measure and evaluate GPS measurements taken with a u-blox receiver.

3.2. Data Preparation

This chapter describes the procedures to create a 3D city model as well as reference points, referred to as grid points (chapter 3.2.2). The created city model and reference points serve as basis for different methods of line of sight computation discussed in chapter 3.3. Additionally, the data, which serves as input is discussed. Methods for providing the 3D models in an appropriate form for the third LoS computation approach Rendering Shadows is the final topic of this chapter (the methods for Tree Detection and the creation of a Continuous Grid).

3.2.1. Input Data

A digital elevation model (DEM) and data of buildings provided by Open Street Map (OSM) are the input data that was used for this thesis. All models are based on this data, therefore errors in this initial steps have grave impact on later results.

DEM The digital elevation model (DEM) is a result of a photogrammetric evaluation which was conducted on behalf of the city of Graz. The provided area covers parts of the Graz University of Technology grounds and parts of the inner city. The grayscaled image can be seen in figure 3.2, where white areas indicate high, and dark areas low elevation. The DEM's horizontal resolution is 0.2 m and the vertical precision is 0.2 m. DEMs, where only vegetation and the surface of the earth is content of the image, can serve as input for the extraction of building heights. A building model can then be derived by subtraction of a DEM of the earth's surface and vegetation and a DEM containing buildings. Due to the non-availability of both DEMs, other methods were used (see chapter 3.2.2).

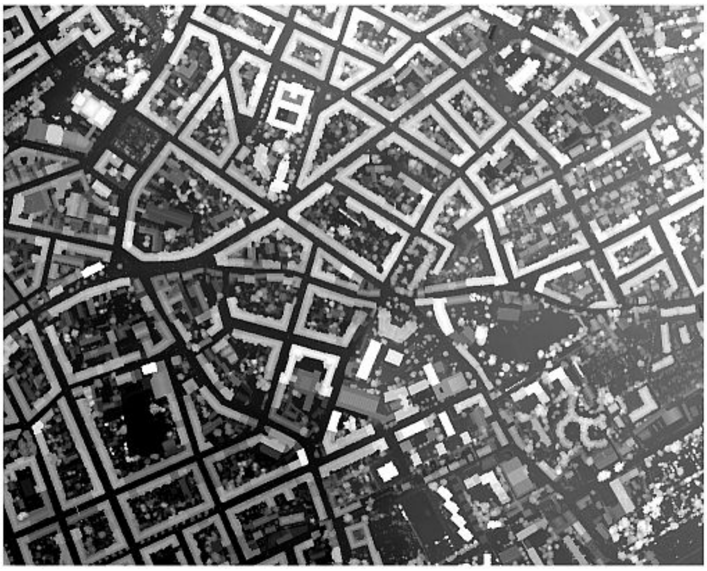


Figure 3.2.: Digital Elevation Model of an area around the Graz University of Technology

Open Street Map Data The data from Geofabrik [6] is an extract of the Open Street Map data set and is provided in the OSM data format and the shape format. With QGIS and a QGIS extension named OSMEditorRemoteControl, an automatic OSM data download can be performed. The extent of the raster layer (DEM) was chosen, and the OSM data was downloaded as XML file. Only building layers were selected from the large amount of OSM data objects and saved as shape file for the area of interest. Additional buildings where created for areas where the data was missing. Figure 3.3 shows the building groundplans clipped by the extent of the DEM.

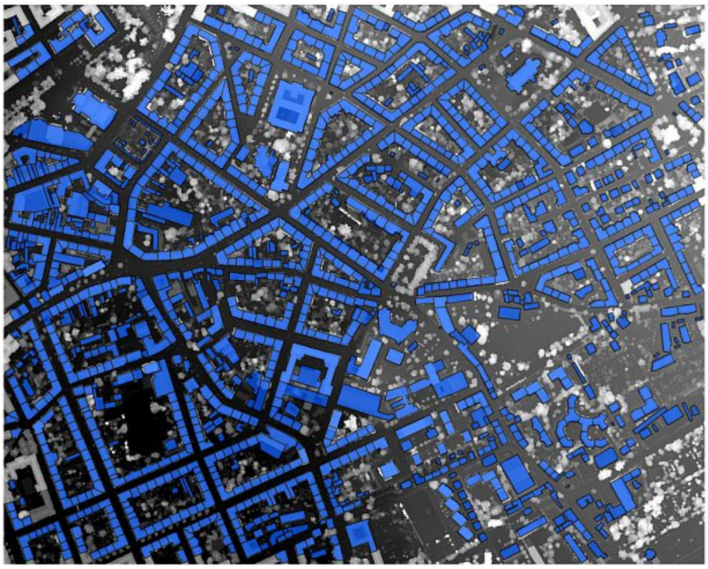


Figure 3.3.: DEM and OSM Geofabrik building groundplans

3.2.2. 3D Model Creation

The creation of a 3D model is subject to several steps. Figure 3.4 on page 32 gives an overview of the required processes. In the following, these will be explained in more detail. The final result should be a suitable 3D city model for a LoS computation, consisting of 3D building models.

I **OSM Data:** The shape file creation from Geofabrik OSM data has been discussed in chapter 3.2.1.

II **Building Groundplans:** The building groundplans serve for the extraction of building heights and the creation of the 3D models.

III **Digital Elevation Model:** The DEM contains the height information of all objects in the area of interest. This includes buildings and streets, but vegetation as well.

IV **Starspan:** A shell program was used to extract height values of buildings, where the vector files overlap the raster pixels. Starspan [5] provides the desired functionality and saves statistical values (for example the maximum and minimum heights of each building) to a comma separated value (CSV) file together with an ID of the corresponding shape. An example of a shell command of starspan is shown in listing 3.1. It extracts the minimum and maximum values of `DOM_20CM_AUS_BILDFLUG_2011_actualizedSRID.tif` where the vector file `gfBuildings.shp` overlays the raster file. Additionally, all fields - that are the attributes of the shapes of the vector file - are stored along with the extracted values to a CSV file called `minMax.csv`.

VI **PostgreSQL/PostGIS DB:** The PostGIS DB serves as back-end, which creates the 3D model and provides the computed fingerprints (see chapters 3.5). The 3D building models are created via a pgsql function, which extrudes the building groundplans based on the heights, which were stored in the CSV file.

VII **3D City Model:** The model is stored in a `polyhedral surface` data type provided by the PostGIS extension. It allows for storing a 3D model together with a EPSG projection. A pgsql function `ST_AsX3D` is provided by the extension, which was used for exporting the model to X3D, a xml-based 3D modelling data format. X3D can be imported to a 3D rendering program or can be converted to other 3D modelling data formats, if not supported by default. Figure 3.5 illustrates the created 3D model.

Listing 3.1: Starspan Bash Command

```
starspan --vector ../osmGeofabrikBuildings/gfBuildings.shp --raster ../
    DOM_20CM_AUS_BILDFLUG_2011_actualizedSRID.tif --stats minMax.csv min max
     nulls --fields
```

Detail of Groundplan

The accuracy of 3D models depends both on the accuracy of the groundplan and the methodology of the extraction of building heights. An example will show the outcome, if the initial data is not properly verified and revised. Figure 3.6 on page 33 shows the 3D model and the DEM with the building groundplans. Since the groundplan of the physics building was not further subdivided into sections, and the height was determined by the maximum grayscaled value (height) of the DEM within the region of the shape file, the 3D building model in some regions is about 17 m too high. Because of this, LoS computations, which are based on the 3D models, are error prone. This problem could be resolved by further dividing the provided shape files, and creating a ground plan for each region of a building with different height. Starspan will then evaluate the height for every shape file. The drawback of this method is, that more 3D models are created (parts of buildings instead of buildings), which increases the computational effort of LoS computation.

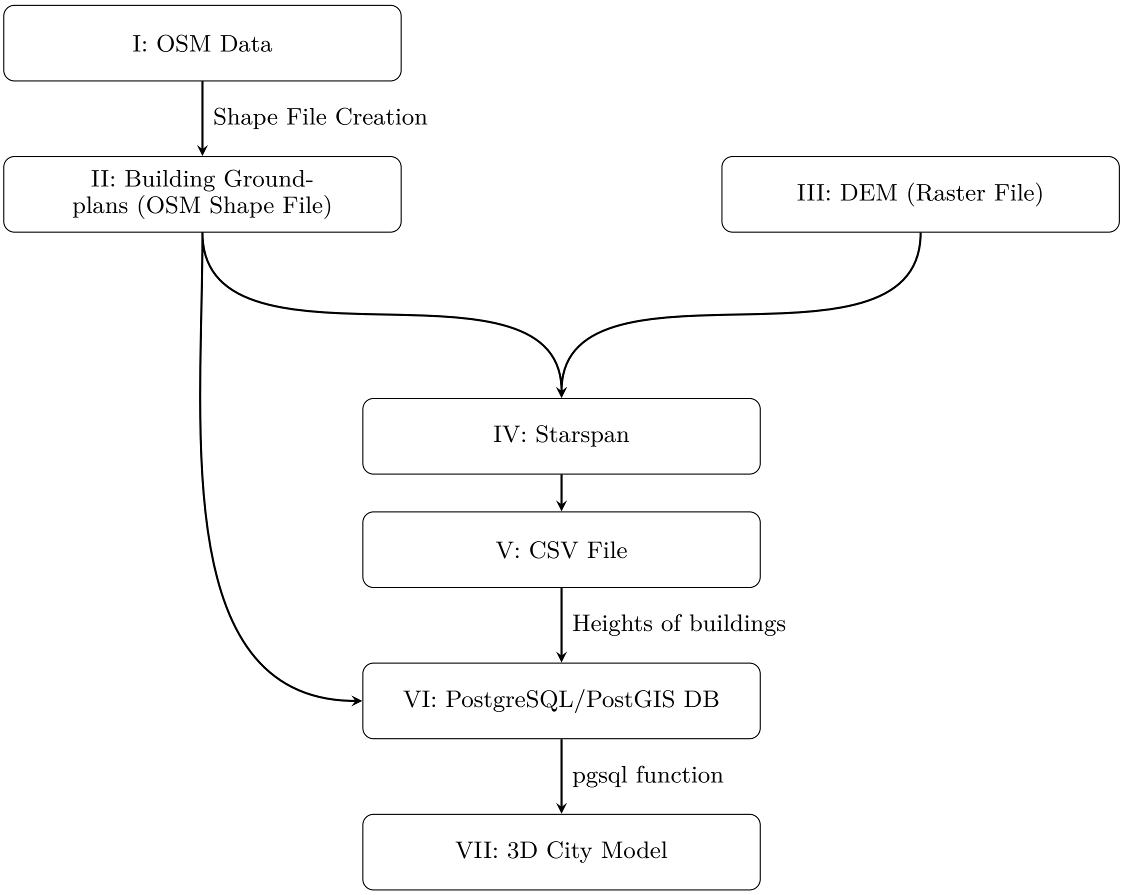


Figure 3.4.: Work-flow of the 3D Model Creation

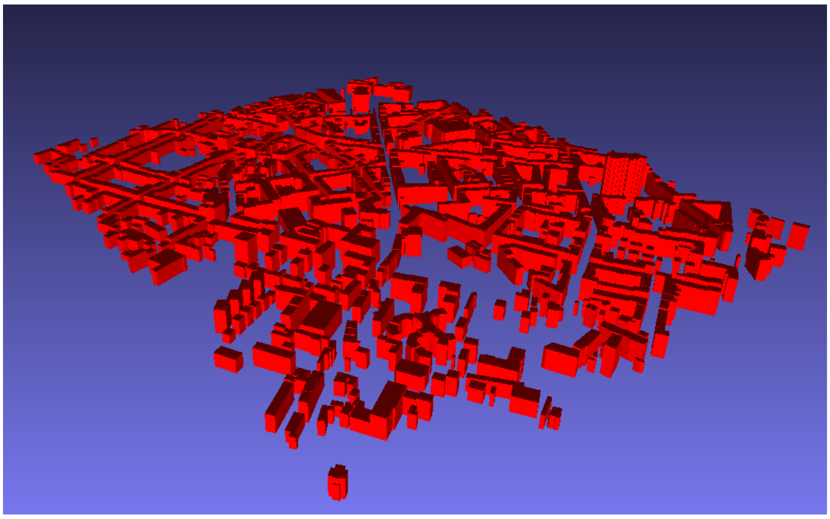
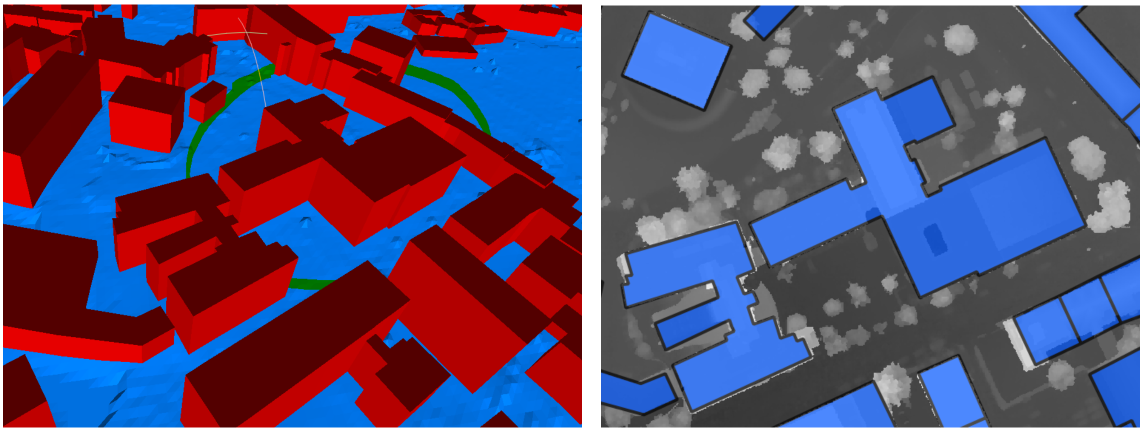


Figure 3.5.: 3D City Model



(a) Erroneous physics building model

(b) DEM and overlayed shape files

Figure 3.6.: 3D model and DEM

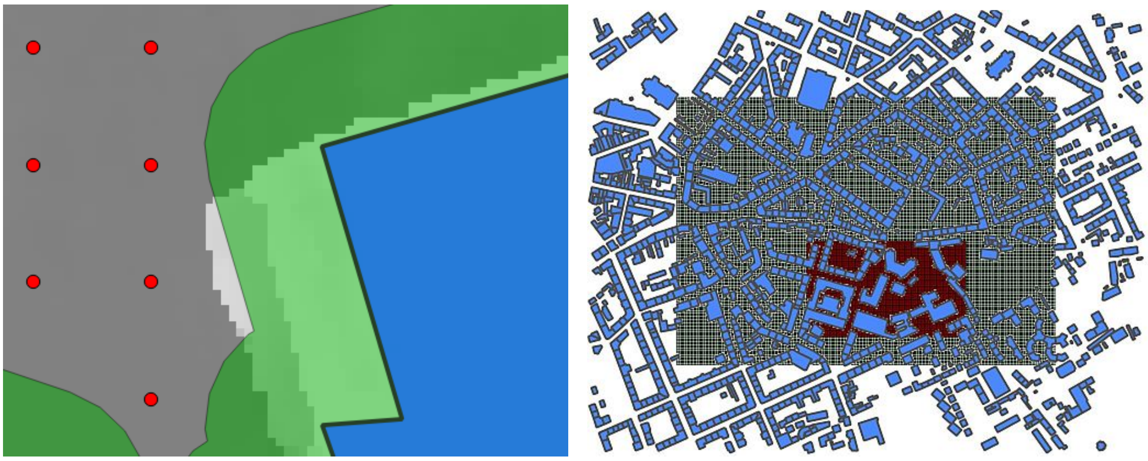
3.2.3. Reference Points

The reference points serve as possible locations of a user within the computed area. They are used to calculate the LoS by determining if a LoS between a satellite and a reference point is obstructed by a 3D model. The creation of a grid of reference points works in the same manner as the generation of the 3D city model does. The DEM and a shape-file, which contains the 2D points, are evaluated with starspan to extract the heights of the points. Thereafter, the points are stored as 3D objects in the database.

A grid size of 3 m was chosen for this project; on the one hand, to be able to detect the side of a road in narrow urban canyons, and on the other hand, in order to minimize the computational effort. Larger grid distances mean lower accuracy, while less points demand less computational cost.

The grid was intersected with the shape-file of the building groundplans, so only points outside buildings remained. As a result of the inaccuracy of the OSM building groundplans, some points of the grid are wrongly set to building height when using starspan. To prevent this, a buffer zone of 3 m was placed around each building. All points within the buffer were deleted. The disadvantage of this method is the loss of reference points in narrow streets. Figure 3.7a illustrates the problem. In green, the buffer is shown, whereas the grid points are red. The underlying raster image suggests a different building boundary (in white) as the OSM model (blue).

To further increase the processing speed, the area of reference points has been reduced. In figure 3.7b, the original grid is shown in grey, while the grid of the test area of this work is coloured red.



(a) 3 m buffer around buildings **(b)** Grid and smaller grid

Figure 3.7.: Grid buffer and size of grid

Tree Detection As initially mentioned, the DEM includes vegetation, which reduces the performance of GNSS fingerprinting regarding accuracy. The height of the reference locations would be set incorrectly to heights of bushes and trees - simulating users standing atop of them - and thus would influence the outcome of the LoS computation. The tree detection attempts to detect and remove trees from the grid by evaluating statistical values of several grid points in a small area around the point of interest. This procedure has to be repeated for each grid point. The result is a vegetation-free surface.

The algorithm proposed is an iterative procedure and queries a small area around a point from which the median and the standard deviation are computed. In the following, the implemented algorithm is described.

1. Search of the nearest neighbours of the reference point, and determination of mean and standard deviation:
   - Perform a Nearest Neighbour search with a maximum of nine points, including the provided one.
   - Just take the nearest neighbours if the distance is not greater than 4.5 m (in a 3 x 3 points = 9 x 9 m grid).
   - If a point is extremely isolated, then the calculation of the standard deviation fails, because a division by null $(1/(n-1)$ for $n = 1)$ occurs. Therefore, only points with at least one neighbour within a 9 x 9 m grid are considered. Extremely isolated points may still contain vegetation afterwards.
   - The median height and the standard deviation based on the median of the sur-

rounding points are calculated by:

$$\tilde{h} = \begin{cases} x_{\frac{x+1}{2}}, & \text{n is odd} \\ \frac{1}{2}\left(x_{n/2} + x_{x/2+1}\right), & \text{n is even} \end{cases} \tag{3.1}$$

$$s_h = \sqrt{\frac{1}{(n-1)}\sum_{i=0}^{n}(h_i - \tilde{h})^2} \tag{3.2}$$

2. Detection of trees: A suitable threshold for the standard deviation must be defined, in order to detect only trees and not unwanted objects, as for example stairs. If a computed standard deviation is greater than the threshold, then the minimum height of that area for the corresponding point is set. The threshold is calculated by assuming the following: a satellite cut-off angle of 15°, a grid spacing of 3 m and a GPS receiver held at a height of 1 m are chosen; consequently, 1.77 m is the minimal height of trees, computed by the following formula:

   $$3\,m\sin(15°) + 1\,m = 1.77\,m \tag{3.3}$$

   The maximum threshold is determined, if 5 out of 9 grid points are trees with a height $h = 1.77\,m$, which results in $s = \sqrt{(1.77^2 * 4/8)} = 1.25\,m$.

3. The termination condition for the iteration has to be defined. If the difference of standard deviations of s_h for the whole grid of two consecutive iterations differ no more than 0.1 m, then the iteration is stopped.

Figure 3.10 displays the grid before and after applying the tree detection algorithm. The same colour scales have been used. A three-dimensional visualization of the detected trees can be seen in the figures 3.8 and 3.9, where the green dots are trees, and the underlying grid points in magenta are the tree cleared grid.

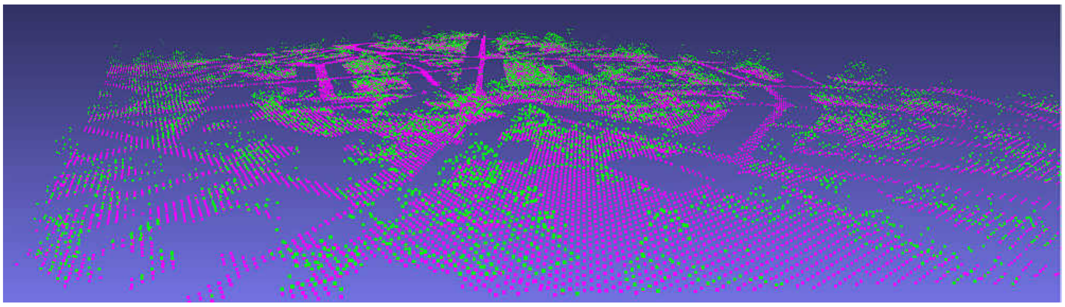


Figure 3.8.: Detected trees within the 3D grid

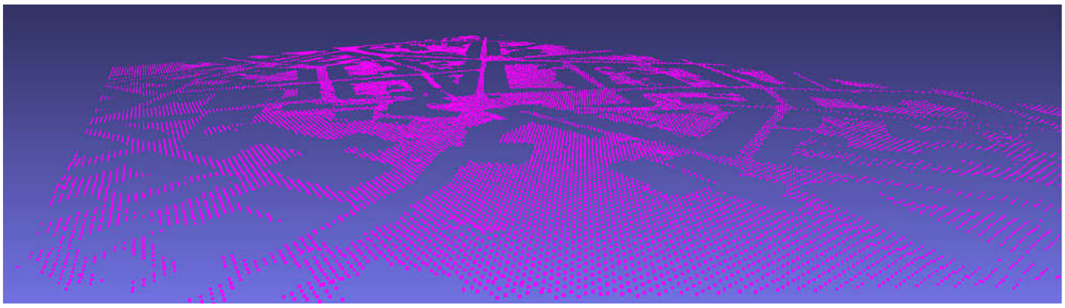
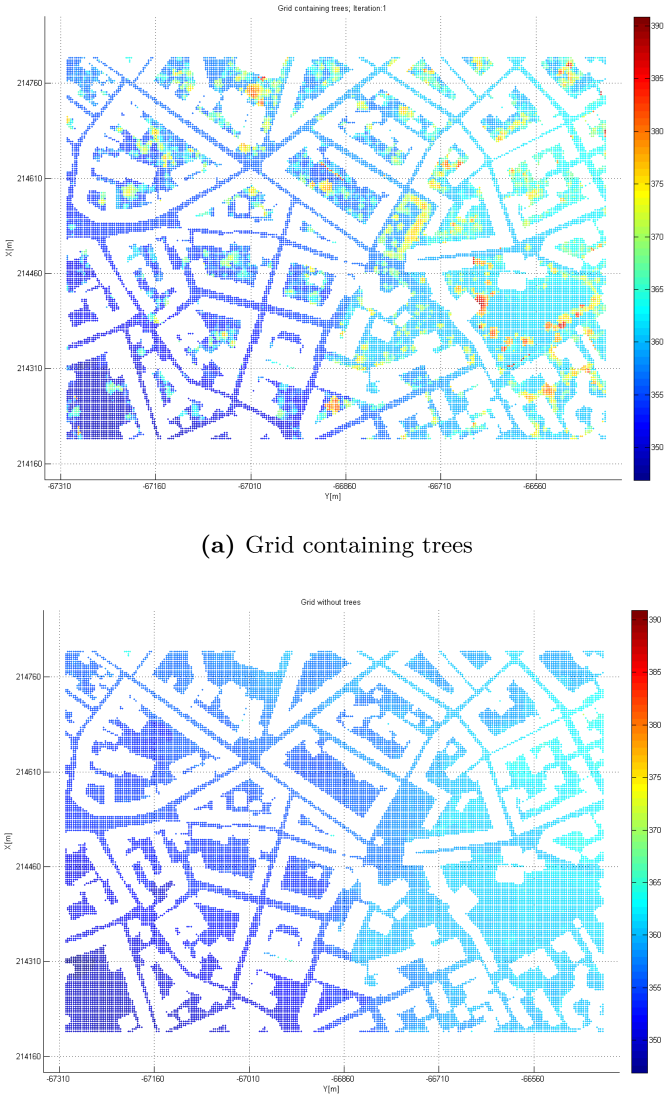


Figure 3.9.: The 3D grid cleared of detected trees

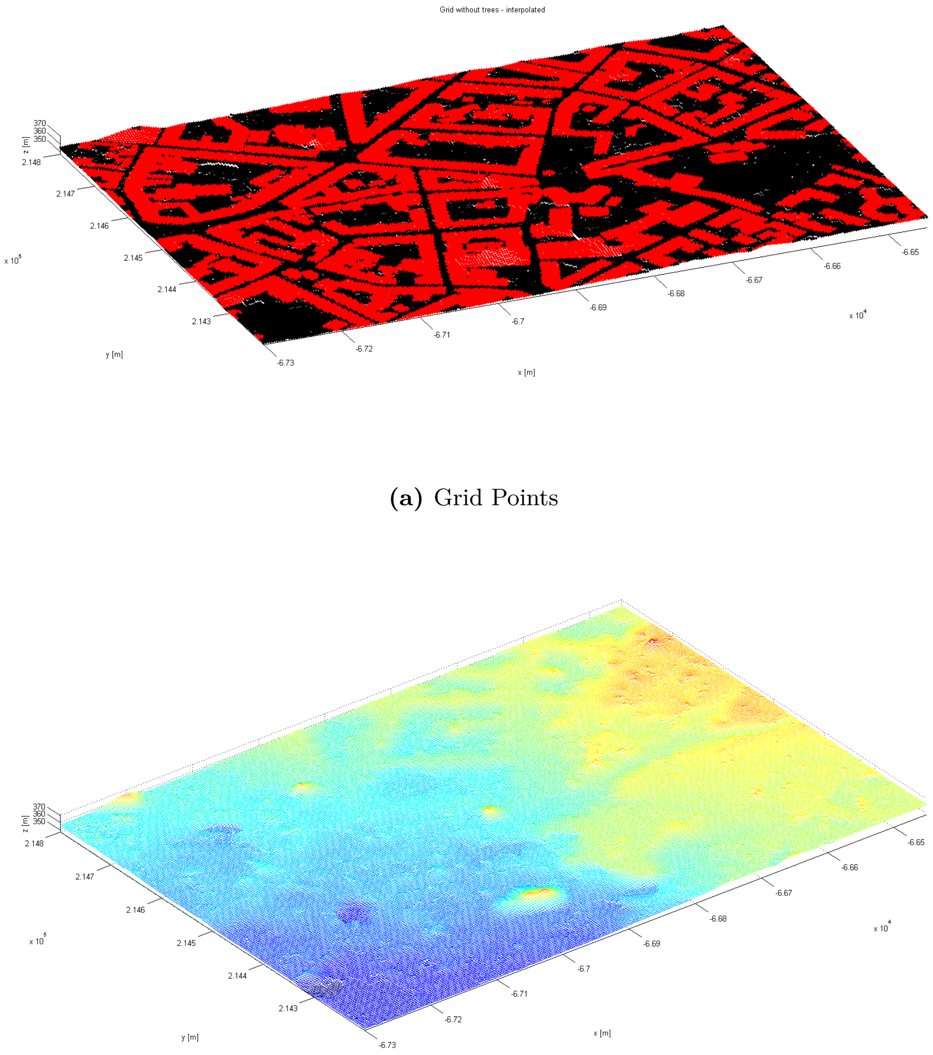


(a) Grid containing trees

(b) Grid without trees

Figure 3.10.: Grid before and after tree detection

Continuous Grid The continuous grid is required for the Rendering Shadows approach in chapter 3.3.4. Due to the intersection of building groundplans and grid points, gaps in the grid where created. For the calculation of shadows, the surface around buildings needs to be continuous, otherwise the light emitted cannot be reflected. Figure 3.11a shows the gaps in red and the grid points in black. A two-dimensional interpolation over the entire surface leads to the representation in figure 3.11b.



(a) Grid Points

(b) Grid Points as continuous surface

Figure 3.11.: Rendering Shadows - Continuous Grid

3.3. Approaches for Line of Sight computation

3.3.1. Overview

The line of sight computation calculates the satellite visibility from the viewpoint of users, which is used for the determination of fingerprints. The underlying principle is the computation of the intersection between a vector, which is oriented from the user to a satellite, and all polygons of a building. If none of the polygons of this building are intersected, then the polygons of the next building will be validated. This continues until all buildings are verified, or an intersection occurs. If this is the case, the next LoS to a satellite is validated. A line of sight to a satellite exists, if no intersections were found. The procedure is repeated for all buildings and satellites, as well as for all reference points within the grid. The final result of LoS computation is a list of satellites, which are visible from a reference point (grid point). The input data are satellite positions and reference points, as well as the three dimensional model of the buildings.

This chapter deals with three concepts of line of sight computation and their implementation (section 3.3.2 to section 3.3.4). Afterwards, in section 3.3.5, the advantages and disadvantages of the concepts as well as their accordances are compared.

1. **Intersections in 3D:** It computes the intersection points between the building polygons and the LoS in space as covered by chapter 2.2.
2. **Sky Plot:** It calculates the visible building boundaries seen from each reference point; the elevation of boundaries is then compared to the elevation of each satellite.
3. **Rendering Shadows:** The rendering software Blender is used to calculate the shading of buildings, based on the satellites' positions. Reference points located in shades have no line of sight to the satellite. The evaluation of rendered images is done in Matlab.

As some sub-areas of the Intersections in 3D and Sky Plot approaches are very time consuming, chapter 3.3.6 deals with concepts, that are applied to the algorithms to decrease the computational effort drastically. Additionally, an offset has to be applied to the height of grid points, to simulate a person carrying a GPS receiver hand-held. The offset value of 1 m was chosen. This has to be done for all three approaches.

3.3.2. Intersections in 3D

This approach determines whether a building polygon is intersected by a vector between the user location (each grid point is possible) and a satellite. From a reference point (grid point) a LoS vector must be analysed for intersections with all building polygons provided by the 3D city model. If an intersection is detected, then the satellite cannot be seen from the user's position and therefore the calculation for the remaining polygons of the building can be terminated. This is repeated for all satellites. Thereafter, the LoS for the next grid point is calculated.

The mathematical formulation of chapter 2.2 is the basis of this approach. In contrast to the other approaches introduced in this work, it is a very precise method, because no abstraction of the determination of a LoS takes place. In comparison, the resolutions of lines of view in the Sky Plot approach induce errors in the calculated building boundaries and are therefore no exact representation of reality (see Comparison of sky plot resolutions on page 45). However, this also increases the computational effort.

In order to accelerate the calculations, only a subset of buildings is considered. Based on a known azimuth α, which is the horizontal angle between a reference direction (north) and the vector from the reference point to the satellite, only a subset of buildings has to be considered. These are buildings, that interfere the LoS in the groundplan of the 3D city model (see section 3.3.6 on page 61 about getting a subset of buildings). The amended parts of the algorithm are highlighted in green.

Algorithm The notations for the variables in the algorithm below are listed in table 3.3. The algorithm requires 3D buildings, the coordinates of a grid point g and a satellite s.

Table 3.3.: Notations for the algorithm: Intersections in 3D

k	index of building
k_α	index of subset of buildings
g	index of grid point
s	index of satellite
$\mathbf{r_d}^{gs}$	direction vector between a grid point g and a satellite s
$\mathbf{n}_{jk}$	normal vector of the plane of polygon j of building k
$\mathbf{r}_i$	intersection point

1. Apply height offset of 1 m to grid point height

2. Calculate direction vector from grid point g to satellite s: LoS vector $\mathbf{r_d}^{gs}$

   Amendment: Get a subset of all buildings which are in the direction of the line of sight (see section 3.3.6, Get subset of buildings):

$$\text{buildings}_\alpha \subseteq \text{buildings} \tag{3.4}$$

3. Get normal vector $\mathbf{n}_{jk}$ from a polygon j of a building k
4. Calculate intersection of LoS vector $\mathbf{r_d}^{gs}$ and the plane $\mathbf{n}_{jk}$ of the polygon (see section 2.2)
5. Proof if the intersection point $\mathbf{r}_i$ lies within the polygon j (see paragraph Point in polygon validation in section 2.2):

   true: Intersection occurs, thus satellite s is not visible from current grid point g

   Exit computation with LoS = *false* for current grid point g and satellite s

   false: Intersection of LoS and plane occurs outside the polygon and thus no intersection occurs between LoS and polygon for current grid point g

   Continue computation with LoS = *true*
6. Repeat for all polygons j of a building
7. Repeat for all buildings k

   Amendment: Instead of looping over k, repeat the calculation for the subset of buildings, which is donated as buildings$_\alpha$ with the indices $k = k_{\alpha,1}..k_{\alpha,n}$
8. Repeat for all satellites s
9. Repeat for all grid points g

Results and Conclusion This approach proved to be the slowest of all approaches, however, the LoS determination is the most accurate. For a high number of buildings and grid points, this approach is not applicable any more, if no essential improvements on the algorithm's implementation are made. On the other hand, no pre-processing is required and it is therefore highly adaptive to possible changes of the 3D model and reference points.

Figure 3.12 shows the 3D city model, a reference point (green dot) at the ground of the grid and a simulated satellite position (blue dot) in the upper left corner, in the vicinity of the 3D model for illustration purposes. Additionally, the subset of buildings, which are located along the azimuth angle of the LoS, are displayed (green buildings). Only these buildings are further considered in the intersection calculations. The black buildings indicate, that they are intersected by the vector. Therefore, no line of sight is accomplished and the satellite is invisible to the user. Figure 3.13 illustrates a similar scenario, visualizing only the concerned buildings. The intersection points (yellow) are visible. Here, all possible intersection points are shown, although only one intersection would suffice to determine, that no LoS is accomplished.

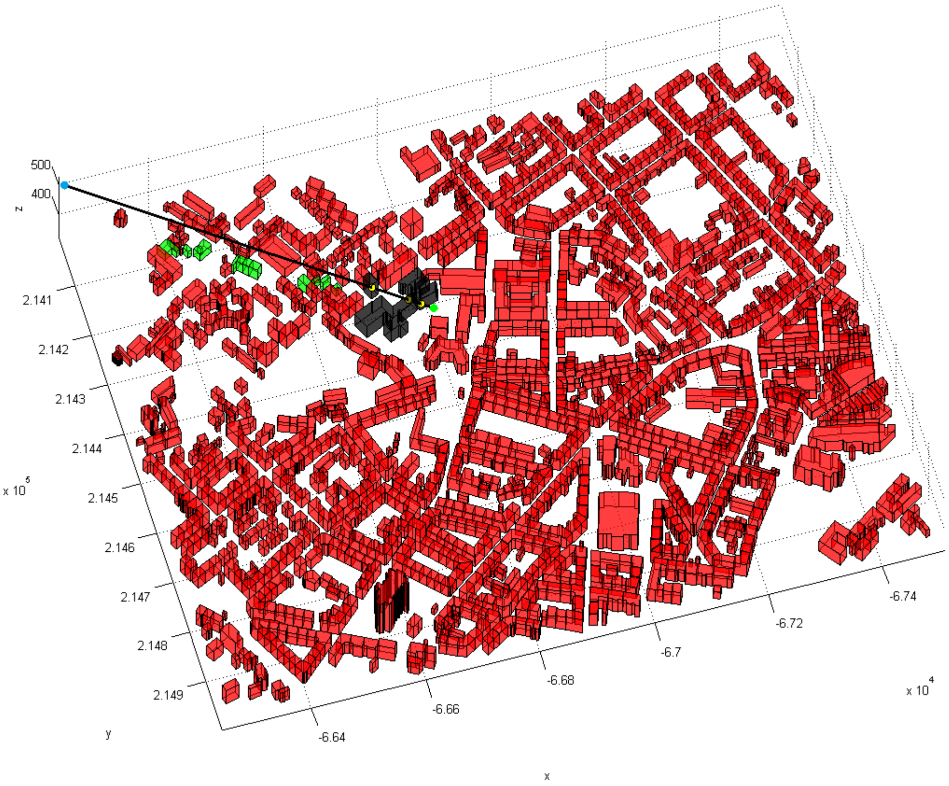


Figure 3.12.: Intersection between a building and a LoS

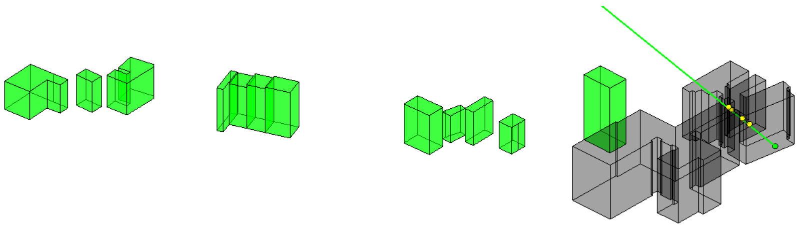
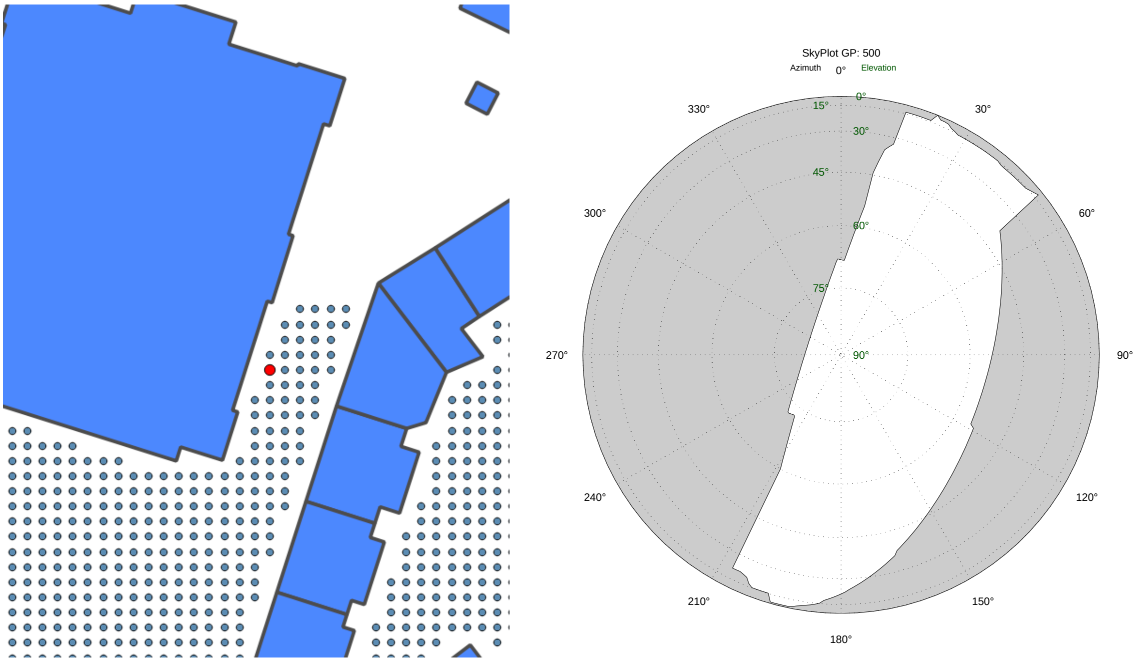


Figure 3.13.: Intersection in 3D

3.3.3. Sky Plot

The Sky Plot approach proposed by [11] generates a sky plot for each grid point that represents the building boundaries. Figure 3.14 illustrates the sky plot and the buildings surrounding the grid point with the ID 500. Gray areas of the sky plot define buildings, that limit the field of vision for a user. Satellites with lower elevation than given by the boundaries are consequently not visible. In figure 3.15 on page 44, an exemplary sky plot with computed satellite positions is displayed. From a user's perspective, the satellites *1, 11, 14, 19* and *32* are visible.



(a) Vicinity and buildings **(b)** Sky plot

Figure 3.14.: Sky Plot and groundplan for grid point 500

The LoS determination is divided into two phases: the pre-processing and the computation step. The pre-processing step must be performed only once, thereafter all sky plots are available. This step must be repeated only, if the 3D city model changes. The LoS computation step must be performed for each positioning anew.

1. **Pre-processing step:** The building boundaries are calculated for each grid point. This phase generally requires several hours to days, depending on the requirements regarding the accuracy of the results of the LoS computation step. The calculation time depends on the number of buildings, the number of grid points and the resolution of the sky plot. The pre-processing step is further divided into two phases:

    a) *Intersection in 2D:* In the two-dimensional space (the groundplan of the 3D model), lines of view with discrete angular intervals are generated in all directions (360°)

from the grid point. Along the lines of view, all buildings are intersected. To accelerate this procedure, only a subset of buildings is used. The method is described in more detail in chapter 3.3.6.

b) *Determination of building boundaries:* The z-coordinate of the points of intersection along a line of view are then set to the heights of buildings. Thus, 3D points are created, which can be defined in terms of the grid point by elevation, azimuth and distance. For the determination of building boundaries, azimuth and elevation are used. A building's boundary is defined by:

$$\varepsilon_b(\alpha_i) = max(E_i), \text{ with} \tag{3.5}$$

$$E_i = E(\alpha_i) = \{\varepsilon_1, \varepsilon_2, ..., \varepsilon_n\}, \text{ and} \tag{3.6}$$

$$\alpha = \{0°, \Delta\alpha, 2\Delta\alpha, ..., 360° - \Delta\alpha\} \tag{3.7}$$

A set E contains the elevations of intersection points along a line of view with the azimuth α_i, where α is a set of discrete azimuths, ranging from 0° to $360° - \Delta\alpha$. The resolution of the lines of views is specified by $0° < \Delta\alpha \leq 180°$. Therefore, the building boundary $\varepsilon_b(\alpha_i)$ at a particular azimuth α_i is obtained by the maximum of the set of elevations E_i.

2. **LoS computation step:** Azimuth and elevation of a grid point to a satellite can be compared to the elevation information contained in the sky plot and thus the visibility v from the grid point g to a satellite s is determined by:

$$v(\mathbf{r_d}^{gs}) = \begin{cases} \varepsilon_{r_d} > \varepsilon_{sp}, & \text{LoS: visibility to satellie exists} \\ \varepsilon_{r_d} \leq \varepsilon_{sp}, & \text{NLoS: visibility to satellite does not exist} \end{cases} \tag{3.8}$$

with:

$$\mathbf{r_d}^{gs} = \Delta\mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{grid\ point} \tag{3.9}$$

$$\alpha_{r_d} = \alpha = \arctan\frac{\Delta x_y}{\Delta x_x}, \qquad \varepsilon_{r_d} = \varepsilon = \arcsin\frac{\Delta x_z}{|\Delta\mathbf{x}|} \tag{3.10}$$

where ε_{r_d} denotes the elevation from the grid point to a satellite and ε_{sp} is the elevation of a building boundary gathered from the sky plot of the grid point in question. The azimuth angle from a grid point to a satellite is referred to as α_{r_d}. Instead of comparing all elevations of a sky plot, only elevations for the azimuth α_{r_d} are evaluated. Because the building boundaries were determined by discrete lines of view, the direct neighbours of α_{r_d} have to be evaluated, denoted as $\alpha_{sp,\ min}$ and $\alpha_{sp,\ max}$. The azimuth α_{r_d} is used for gathering elevations of the sky plot at azimuths $\alpha_{sp,min}$, $\alpha_{sp,max}$, that enclose α_{r_d}:

$$\alpha_{sp,\ min} \leq \alpha_{r_d} \leq \alpha_{sp,\ max} \text{ and } \alpha_{sp,\ max} - \alpha_{sp,\ min} = \Delta\alpha \tag{3.11}$$

where $\Delta\alpha$ is the resolution of lines of view of the sky plot. Figure 3.33 on page 67 displays lines of view with a reslolution of 5°, and figure 3.30 on page 62 shows a subset of buildings selected along a line, originating from a grid point with α_{r_d}.

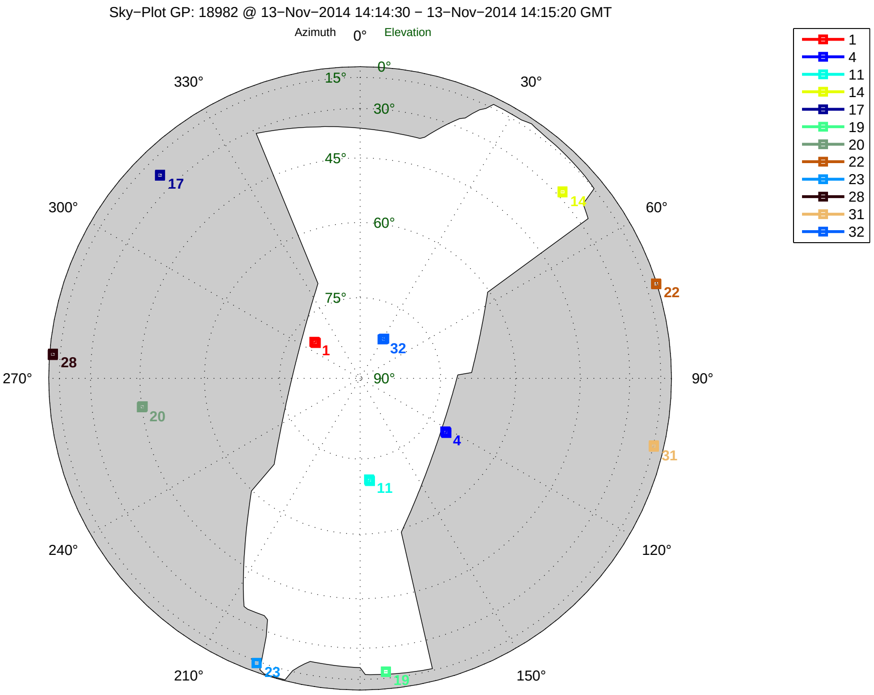


Figure 3.15.: Sky Plot containing satellites for visibility determination

Algorithm The input data for the computation of lines of sight is depending on which of the two steps of LoS computation is executed. The pre-processing step (sky plot generation) requires the 3D city model and the coordinates of grid points, whereas the computation step uses the computed sky plots and satellite positions.

1. **Pre-processing step:** Generate a sky plot for each grid point

   1.1. Generate a set of lines of views.

   1.2. Get a subset of buildings which are in direction of a line of view (see section 3.3.6)

   1.3. Intersect the boundaries of each building from the building subset in 2D with a line of view. Return x- and y-coordinates of the 2D intersection point.

   1.4. Retrieve the height of the building intersected and assign it to the z-coordinate of the intersection point. A 3D intersection point is created.

   1.5. Apply a height offset of 1 m to the grid point height.

   1.6. Calculate the elevation angle of the vector from grid point to the 3D intersection point.

   1.7. Repeat this for every building along the line of view and store the highest elevation angle to the grid point's sky plot.

1.8. Repeat for every direction (for every line of view).

1.9. Repeat for every grid point.

2. **LoS computation step:** Compare the elevations of the sky plot model to the computed satellite elevations to determine the visibility to satellites.

   2.1. Calculate the direction vector from grid point g to satellite s: LoS vector $\mathbf{r_d}^{gs}$

   2.2. Calculate the azimuth α_{r_d} and elevation angle ε_{r_d} of $\mathbf{r_d}^{gs}$

   2.3. Compare the elevation angle for the given azimuth to the elevation angle of the sky plot ε_{SP} of grid point g

   - *if* $\varepsilon_{r_d} > \varepsilon_{SP}$*:* Satellite is visible
   - *if* $\varepsilon_{r_d} \leq \varepsilon_{SP}$*:* Satellite is not visible

   2.4. Repeat for all satellites s

   2.5. Repeat for all grid points g

Comparison of sky plot resolutions The investigation of the impact of the azimuth resolutions $\Delta\alpha$ of line of views on the detection of building boundaries is an important factor for the accuracy estimation of GNSS fingerprinting. The higher the precision of detected building boundaries of a sky plot, the more precise the computation of LoS can be accomplished. The error of boundary detection can be estimated by calculating the arc length for a building boundary furthest away from a grid point:

$$l_{error} = \Delta\alpha \, d_{max} \tag{3.12}$$

with l_{error} being the perpendicular error in direction, $\Delta\alpha$ the azimuth resolution in radian and d_{max} the distance from the grid point to the farthest building boundary. For an arbitrary grid point with the ID 25031 this results in an error l_{error} of $[43.46 \quad 21.73 \quad 10.87]$ m for $\Delta\alpha = [2 \quad 1 \quad 0.5]°$ respectively. For the maximal occurring distance from a grid point to a building boundary within the limits of the grid ($d_{max} = 1245.12$ m) this estimation leads to an error of $[45.05 \quad 22.53 \quad 11.26]$ m. Even under the assumption of standing in an inner courtyard with building boundaries being approximately 100 m away from the user position, the error proofs to be still significant: $[3.49 \;\; 1.75 \;\; 0.87]$ m.

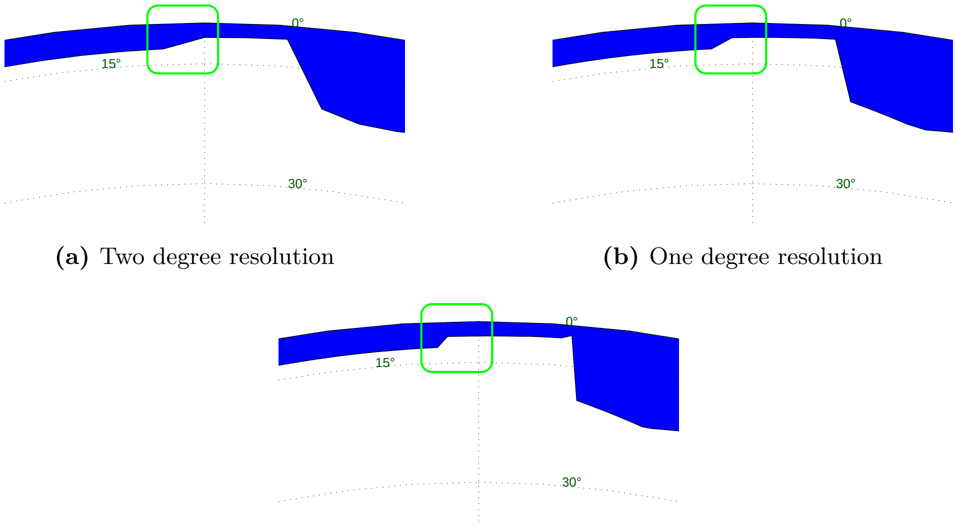
For an evaluation of different sky plot azimuth resolutions, sky plots for three $\Delta\alpha$ have been computed: 2°, 1°and 0.5°. In figure 3.16 the sky plots were zoomed to an azimuth and elevation of 0°, leading to a better interpretable comparison. A considerable difference in the detection precision of building boundaries can be seen when comparing figure 3.16a to 3.16c. Borders in sky plots with higher resolution differ significantly from those of lower resolution. Figure 3.17 indicates even worse errors of lower resolutions.

This deliberations lead to the conclusion, that even a high resolution will not result in satisfying results, if the buildings are far off (for example 11.26 m error in perpendicular direction

- see examples above). However, the sky plot is determined by the highest elevation angle in every direction. The possibility of a building near to a user position having the highest elevation angle is significantly higher, than a building further away. Near buildings are subject to little errors. When increasing the resolution of azimuth angles, the outcome is a better detection of building boundaries, but higher resolutions will increase the computation time as well. Investigations show, that the computational cost increases linearly with the resolution (see table 3.4). On the other hand, other factors, like the influence of the detail of the 3D model, proofed to have even more impact on the accuracy of LoS computations (see chapter 4.3).

Table 3.4.: Computational cost of Sky Plot computation for one grid point

Resolution $\Delta\alpha$ [°]	2	1	0.5
Computation time [s]	117	241	475



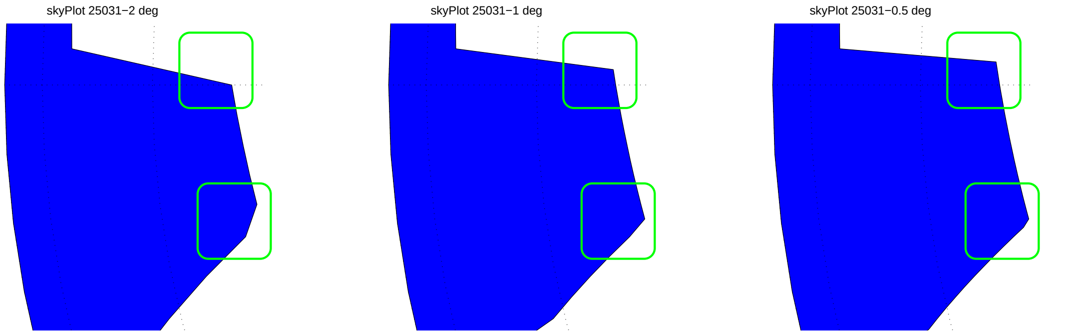
(a) Two degree resolution

(b) One degree resolution

(c) A half degree resolution

Figure 3.16.: Sky Plot for grid point with ID 25031, zoomed differences - 1

Conclusion Due to the generation of sky plots in the pre-processing step, a fingerprint determination (that means the LoS computation alone) can be performed many times faster than the Intersections in 3D approach. However, this entails drawbacks in the accuracy of the results. As a consequence of the high computational effort of the building boundary detection, a repeated application (e.g. daily), which would have to be carried out in consequence of changes in the 3D city model, is not possible. The investigations have shown, that a significant error in the calculation of building boundaries may occur by using a discrete azimuth resolution. This also means, that the fingerprint calculations based thereon are error-prone.



(a) Two degree resolution

(b) One degree resolution

(c) A half degree resolution

Figure 3.17.: Sky Plot for grid point with ID 25031, zoomed differences - 2

For example, if an edge 20 m distant has an offset of 0.5 m in the sky plot, this would lead to an angular error of 1.43°.

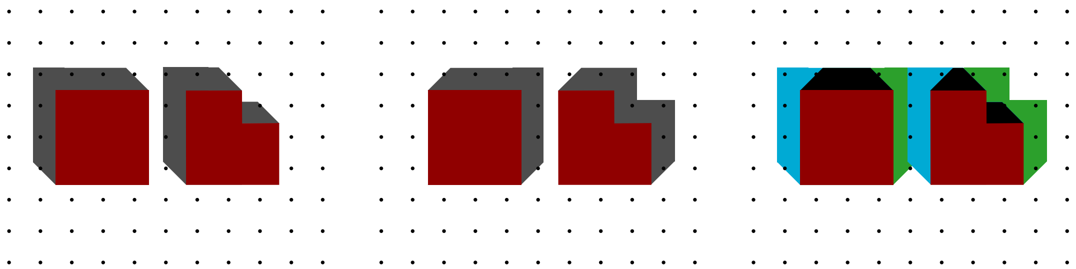
In this thesis, the Sky Plot approach was chosen with an azimuth resolution of $\Delta\alpha = 1°$ for GNSS fingerprinting. The errors induced by the resolution of 1° are small enough to not significantly influence the results at short distances. As in cities only small to moderate distances between a grid point and an obstructing building arise, the application of GNSS fingerprinting in urban areas is reasonable. The advantage is a shorter computation time for a large number of grid points when pre-processing, compared to computation times at higher resolutions.

Contrary to the high computational effort of pre-processing, the effort of elevation comparison between the sky plot and the satellite elevations for a larger number of grid points is small. Thus, given a number of 3727 points, an average computational time of 5.5 seconds for 32 satellites was achieved. Hence, a fingerprint is available for every six seconds.

3.3.4. Rendering Shadows

The method of rendering 3D models for the generation of GNSS fingerprints was proposed by [1]. By rendering the 3D city model in a 3D scene, visibility information for each grid point can be derived. Therefore, a satellite in a rendering program is represented by a light source. The output of a rendering process is an image of cast shadows over the 3D city model. To get the results for a fingerprint, images rendered for 32 satellites (or at least for all satellites in sight) are overlapped. Each grid point's location in the image is evaluated and a list of visible satellites is assigned to it. A satellite is designated as visible, if light is cast to the grid point location.

The combination (overlapping) of shadows is shown in figure 3.18. The shadows cast by satellite 1 and satellite 2 differ from each other, depending on the position of the satellite related to the model. The combination of rendered images is shown in figure 3.18c. Each grid point is evaluated thereafter. For instance, at points in the blue area all satellites except satellite 1 are visible, as they are located in the shadow of buildings cast by that satellite. The black area refers to locations, where the shadows of satellite 1 and 2 overlap, thus neither satellite 1 nor 2 can be seen.



(a) Satellite 1 **(b)** Satellite 2 **(c)** Combined shadows

Figure 3.18.: Combination of shadow rendering results

The procedure of rendering is shown in figure 3.19. The rendering was performed using the computer graphics (CG) software Blender (see figure 3.20). Based on the 3D building model and the 3D grid points, a scene was created. The satellites serve as light source, which are aligned by the azimuth α and the elevation angles ε relative to the model.

Blender includes a Python console environment, which allows for modification of created scenes by Python scripts (figure 3.20b). This makes it possible to carry out Python code via shell commands, which changes the alignment of the light source in Blender, based on the calculated satellite positions, and then renders the scene, producing an image. The shell commands executing the Python code can be started through Matlab, hence operating the process by a single software is possible. Necessary parameters (α, ε) are passed to the rendering process. The rendered image is loaded by Matlab and converted from a grey-scaled image to a black and white image. At each grid point, a shadow detection is performed and repeated for each rendered image. The final result is a list of visible satellites for every grid point.

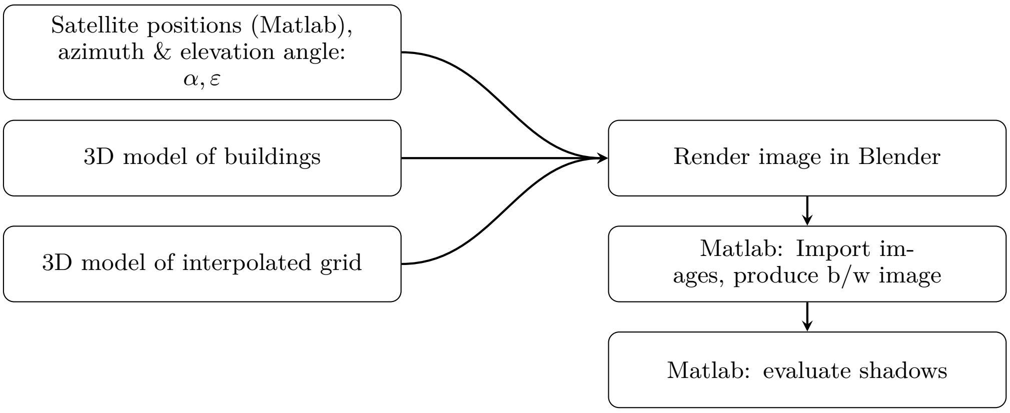

Satellite positions (Matlab), azimuth & elevation angle: α, ε
3D model of buildings
3D model of interpolated grid
Render image in Blender
Matlab: Import images, produce b/w image
Matlab: evaluate shadows

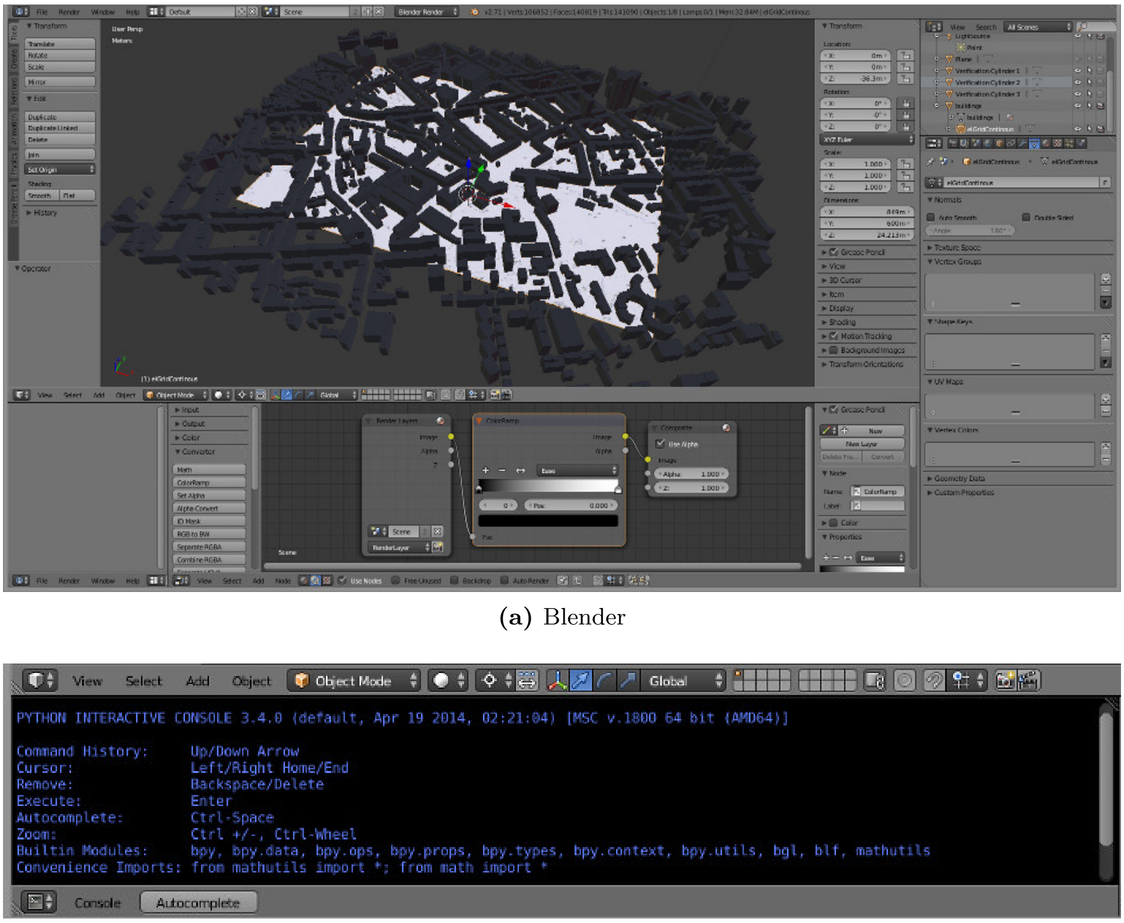
Figure 3.19.: Workflow of Rendering Shadows - Shadow Detection

To render the 3D model in Blender and conduct shadow detection, three consecutive steps are necessary:

1. *Setup:* For rendering, an interpolated grid is required, since only a continuous surface reflects the emitted light. The creation of a continuous grid is discussed in chapter 3.2.3. To import the 3D city model to Blender, the model must be exported from the PostgreSQL/PostGIS database in a 3D format (.x3d). The model is given in Gauss-Krueger coordinates. To allow for easier handling in the GC program, the model has to be transformed to the local level frame (LLF) of Blender. A 3D datum transformation sets the centre of the model to the origin of the Blender coordinate system (see chapter 3.6 for details). Since Blender does not provide native support for the .x3d format, the model was converted to .obj format. The following settings were made:

Blender project Setup

a) Create a Blender project, where the building's 3D model is centred to the coordinate origin of the project system
b) The project's units must be in meters
c) Add the continuous grid surface to the project
d) Apply a height offset of 1 m to the surface (height of hand-held when walking)
e) Add a camera and place it in the zenith of the project plane
f) Change the camera type to `orthographic` so that no perspective is visible and set `orthographic scale` to 2000. This scales the camera to an extent of 2000 x 2000 m.
g) Change the resolution of the rendered image to 1920 x 1080 pixels



(a) Blender

```
PYTHON INTERACTIVE CONSOLE 3.4.0 (default, Apr 19 2014, 02:21:04) [MSC v.1800 64 bit (AMD64)]

Command History:     Up/Down Arrow
Cursor:              Left/Right Home/End
Remove:              Backspace/Delete
Execute:             Enter
Autocomplete:        Ctrl-Space
Zoom:                Ctrl +/-, Ctrl-Wheel
Builtin Modules:     bpy, bpy.data, bpy.ops, bpy.props, bpy.types, bpy.context, bpy.utils, bgl, blf, mathutils
Convenience Imports: from mathutils import *; from math import *
```

(b) Blender's Python console

Figure 3.20.: 3D computer graphics (CG) software Blender

2. *Rendering:* In order to set the light source at the satellite's position, azimuth and elevation of the satellite must be known. For this, the difference vector between satellite and grid point is needed. Every grid point may be calculated (rendered) to get the vector's elevation and azimuth separately, but this would increase the computational effort heavily, as it would produce n images for each epoch (n = number of grid points × number of satellites). Unlike previous line of sight computation approaches proposed in this work, it is not recommended to compute (render) every grid point separately. Therefore - as approximation, the grid's centroid is used for azimuth and elevation calculations:

$$\mathbf{r_d}^{gs} = \Delta\mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{grid's\ centroid} \tag{3.13}$$

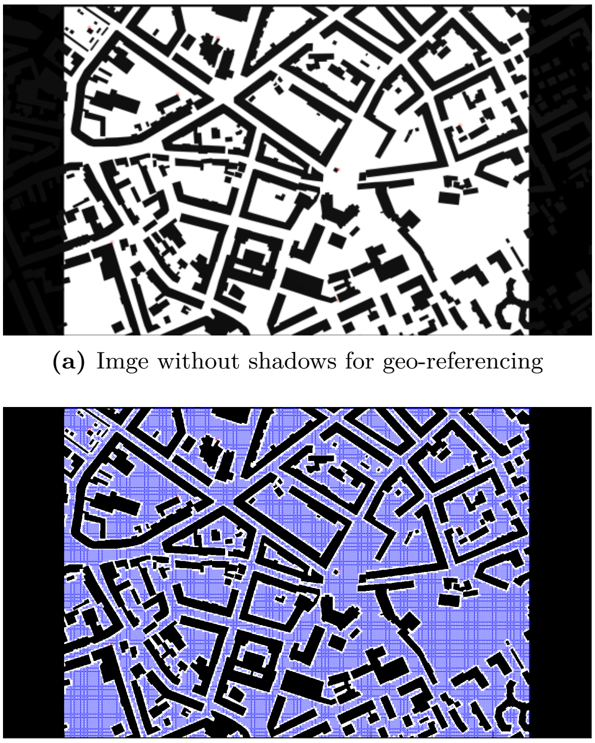
$$\alpha = \arctan\frac{\Delta x_y}{\Delta x_x}, \qquad \varepsilon = \arcsin\frac{\Delta x_z}{|\Delta\mathbf{x}|} \tag{3.14}$$

The light source is positioned for each satellite through a Python script executed by Matlab.

3. *Shadow Detection:* The rendered image is evaluated in Matlab. This is done by georeferencing a reference image, which was rendered without shadows (see figure 3.21). As control points, building vertices were chosen. The parameters for the four-parameter transformation (2D/Helmert transformation), needed for the transformation between MGI Austria Gauss Krueger (GK) coordinates and the image body frame, were determined by a least squares adjustment (LSA). As observables, pixel and GK coordinates of the control points are given. The rotation parameter of the LSA was fixed to $\alpha = -90°$. The following values were determined for the Helmert parameters:

$$\mu = 1/0.56, \quad \mathbf{x} = \begin{bmatrix} -214802.86 \\ 67413.91 \end{bmatrix}$$

The scale factor is therefore 1.88, by which one meter corresponds to approximately two pixels. Since the chosen grid spacing is 3 m, a resolution of 1920x1080 is chosen to be sufficient. Higher resolutions can be rendered, but this increases the computational complexity. If the grid spacing would have been selected in sub-meter range, then the resolution would have to be increased in order to achieve accurate results. The rendered grey scaled image is processed by a black/white filter. At the position of grid points, the grey-scale value is determined; white means, that the satellite is visible, areas of shadow are coloured in black (see figure 3.22).

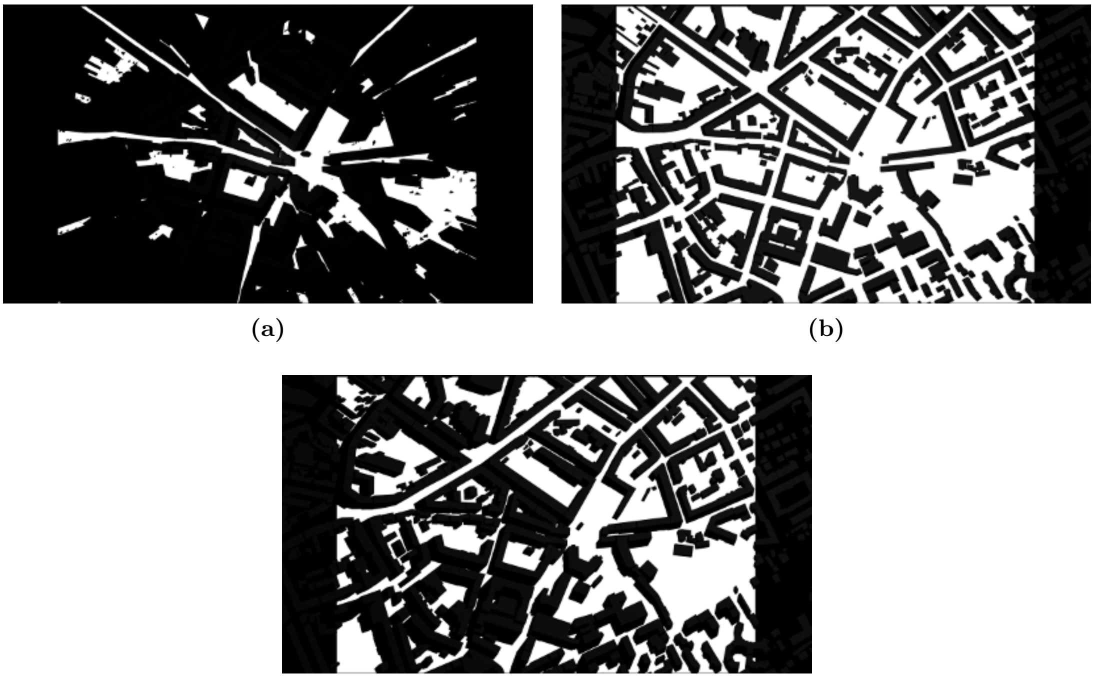


(a) Imge without shadows for geo-referencing

(b) Grid points in the reference image transformed to pixel coordinates

Figure 3.21.: Rendered Image for Georeferencing without Shadows

The rendered images from Blender for different satellite positions are visible in figure 3.22, where shadows cast by the light source can be seen. Dependent on the position of the light source, the shadow changes orientation and size. In contrast to the approaches presented before, the entire grid is calculated, rather than a subset. Nevertheless, the method is comparable to the Sky Plot approach concerning performance (see Comparison of methods on page 57).



(a) **(b)**

(c)

Figure 3.22.: Rendered images with shadows

Since the purpose, why rendering programs are designed, is a photo-realistic generation of images and films of 3D scenes, and not the simulation of shadows cast by satellites, several approximations have to be considered. These are described in the following.

Light Sources in Rendering Software Solutions In a 3D rendering program a point light source has finite energy, so it can not be set at large distance from the 3D model. Therefore, the light source has to be positioned in the vicinity of the model. The idea is to downscale the satellite position to the vicinity of the grid (for example to 1 km from the central grid point when the grids extents are approximately 850 m x 650 m). The grid point coordinates are in MGI Austria GK East projection, whereas the satellite coordinates are in the WGS84 reference frame. To determine the vectors between a user position and a satellite, the grid points and the satellite positions have to be in the same system. A Transversal Mercator Projection is used for the reverse projection of grid point coordinates to WGS84 coordinates (see chapter 3.6).

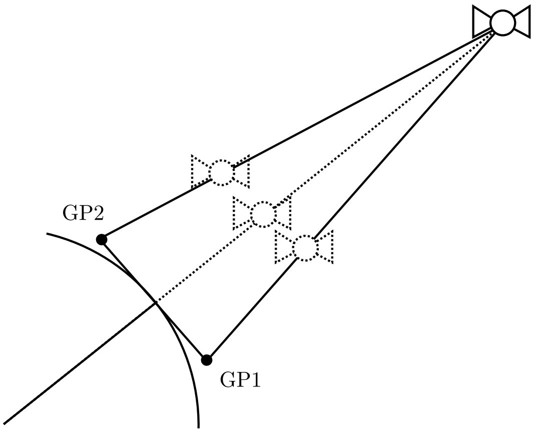
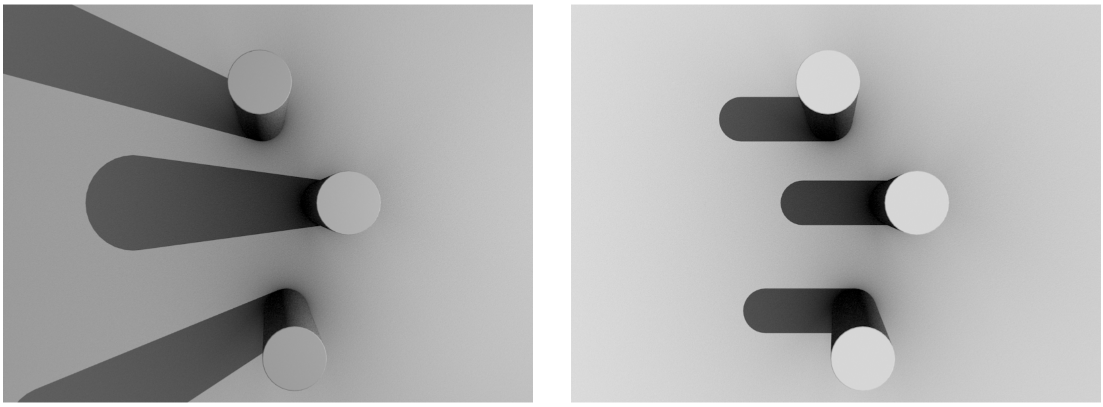


Figure 3.23.: GK Projection: downscaling of satellite position



(a) Point **(b)** Sun

Source: http://wiki.blender.org/index.php/Doc:2.4/Books/Essential_Blender/11.2.Lighting:_Discussion#Sun

Figure 3.24.: Light Sources in Blender

However, a simple downscaling of the satellite's position to the grid's vicinity is not possible, as the downscaled satellite position is dependent on the grid point used to calculate the downscaling (see figure 3.23). The solution is to set the type of light source to the type 'sun'. A sun in Blender is independent of it's position and the cast light is only dependent on the sun's alignment in space. Its emitted light has constant intensity and is therefore suitable for simulating a satellite in space. However, the sun light source emits parallel rays, as if it would be infinitely large and far away, which is not the case with satellites. This induces an error of cast shadows (see figure 3.24). Therefore an analysis of this error has been made.

Analysis of error induced by a sun as light source The error depends on the distance of the light source to the objects that cast shadows, as well as the angle of light rays that hit the objects. Another factor is the extent of the object under investigation - in this case, the extent of the grid is used.

In figure 3.25 the difference of cast shadows between the sun as light source and a point light source is displayed. The light source (satellite) is positioned in the zenith of the grid and therefore the zenith angle is zero degrees. As the sun casts parallel rays, no shadow is cast. The shadow caused by the point light source is $a = 2\,cm$ with a satellite distance of $d = 20000\,km$, a grid width of $b = 850\,m$ and a maximum building height of $h = 95\,m$ (church). The formulas for the estimation of a cast shadow a are given below.

$$\alpha = \arctan\left(\frac{b/2}{d}\right) \tag{3.15}$$

$$a = h \tan\alpha \tag{3.16}$$

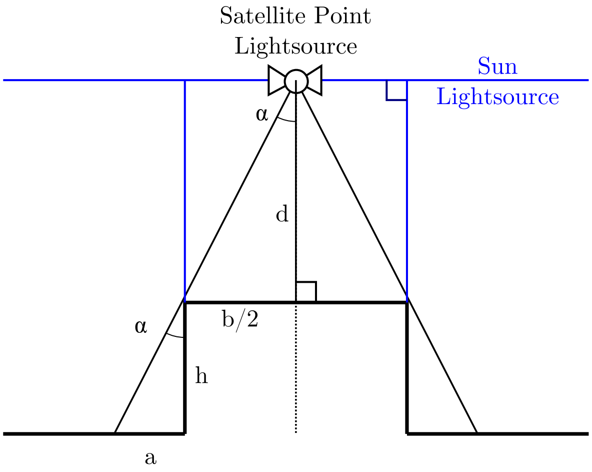


Figure 3.25.: Sun as light source - error analysis

As the satellite moves over the horizon, the cast shadow changes for a point light source and a sun light source differently. Figure 3.26 illustrates the geometrical constellation for the error analysis. The lowest shadowing error cast by a sun would be when the satellite has a high elevation angle. The sun light rays are parallel and the cast shadow c is smaller than the cast shadow $a' + c$ by a point light source. With an zenith angle of $\gamma = 89.9°$ the error in shadow casting a' would be 1.16 m. However, such a case is not likely to occur in an urban area, as the cast shadow $a' + c$ would be approximately 54 km long. A more realistic estimation is to set the maximal cast shadow by a point light source c before it hits another building and then estimate the error a'. Assuming an open space of 250 m between buildings, the resulting error a' is only 5.7 mm. The results for different zenith angles can be taken from table 3.5. Following formulas give the error between cast shadows of distinct light sources a'.

The results of table 3.5 indicate, that a sun light source can be used, as it induces only small errors compared to errors induced by the 3D models (see section 4.3).

$$\delta = \frac{\pi}{2} + \gamma \tag{3.17}$$

$$d' = \sqrt{\left(\frac{b}{2}\right)^2 + d^2 - 2\frac{b}{2}d\cos\delta} \tag{3.18}$$

$$\alpha = \arcsin\left(\frac{b\sin\delta}{2d'}\right) \tag{3.19}$$

$$c = h\tan\gamma \tag{3.20}$$

$$a' = h\tan(\gamma + \alpha) - c \tag{3.21}$$

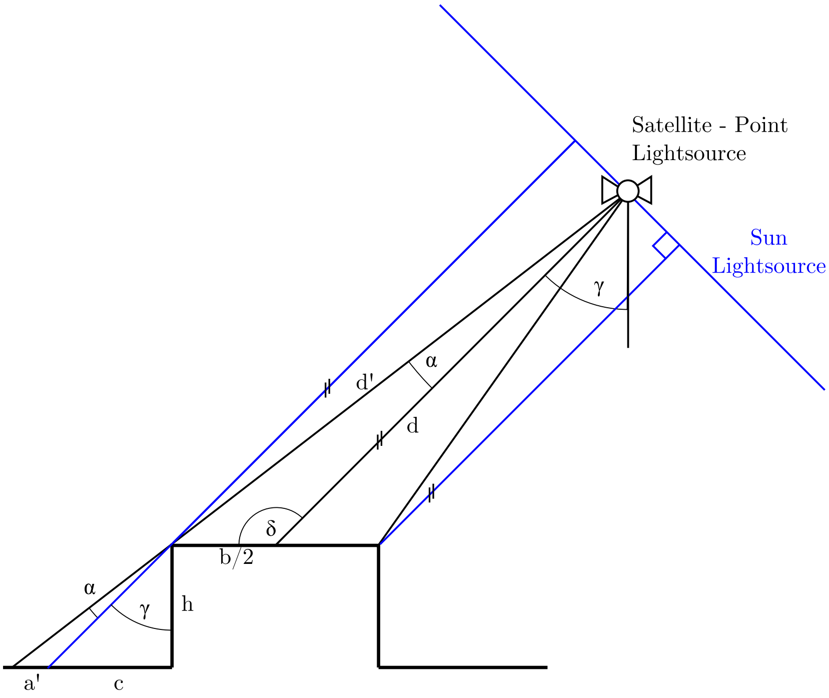


Figure 3.26.: Sun as light source - error analysis with $\gamma \neq 0°$

Table 3.5.: Analysis of distinct light source induced errors

γ [°]	c [m]	a'
0.0	0	2 mm
57.7	150	3.8 mm
69.2	250	5.7 mm
80.0	539	11.6 mm
85.0	1086	23.2 mm
89.9	54431	1.16 m

Conclusion It has been shown, that LoS computation can be performed by rendering the 3D city model. What is needed, is a continuous surface model and 3D building information, as well as the ability to position the light source by source code. The evaluation of the rendered images is performed by geo-referencing and retrieval of grey-scale values at the grid points' coordinates. A greater effort must be considered, if the 3D building model changes, because the model would have to be exported from the DB and imported to Blender anew. However, a computational expensive recalculation as in the Sky Plot approach is not necessary. Due to the limitation of the applicable type of light source (only a sun light source can be used), an error in the range of 2 mm ($\gamma = 0°$) to 1.16 m ($\gamma = 89.9°$) is generated in the casting of shadows.

3.3.5. Comparison of methods

The comparison of the three LoS computation approaches and possible optimizations are the topics of this chapter.

Comparison In table 3.6 the characteristics of the proposed approaches are listed. They are a summary of the conclusions drawn at the end of each method. Especially due to it's advantage in computational speed, the Sky Plot approach has been chosen for fingerprint generation in this thesis.

Table 3.6.: Characteristics of the LoS computation approaches

Approach	Advantages	Disadvantages
Intersections in 3D	• accurate • no pre-processing required • highly adaptive to 3D model changes	• high computational cost
Sky Plot	• fast	• inaccurate, depending on the resolution of the lines of view • pre-processing required
Rendering Shadows	• moderate to fast speed • little pre-processing required (continuous grid) • adaptive to 3D model changes	• accuracy is appropriate

Figure 3.27 shows the accordance of the Intersections in 3D and the Sky Plot approach, whereas figure 3.28 compares the Intersections in 3D to the Rendering Shadows approach. For comparison, a fingerprint for a satellite at one epoch was calculated. The corresponding numerical values are given in table 3.7. The results show a high correlation of the Intersections in 3D and the Sky Plot approach. The comparison to the rendering indicates, that the calculated continuous grid, which is used in the Rendering Shadows approach exerts influence on the result. If the differences of the highlighted area of figure 3.27 and figure 3.28 are compared, one can see the influence of the rendering approach on the percentage of compliances (compliance of Intersection in 3D - Sky Plot: 99.14% versus Intersection in 3D - Rendering Shadows: 98.31%).

Possible Optimizations As mentioned before, the approaches are very different concerning the calculation time. In this chapter, only the time for the LoS computation itself, and not the time for possible pre-processing is evaluated (for example the pre-processing step of the sky plot). Figure 3.29 on page 60 demonstrates the computational effort for a fingerprint of the whole grid determined for one satellite. The calculation time of the Intersections in 3D and

Table 3.7.: Accordance of Approaches

	Sky Plot	Rendering Shadows
Intersection in 3D	99.14 %	98.31 %
Sky Plot	-	97.83 %

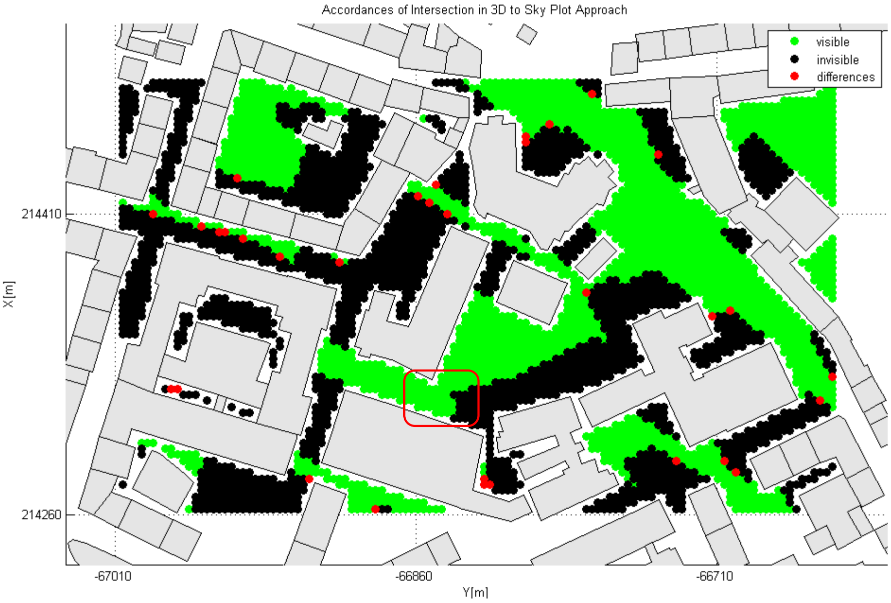


Figure 3.27.: Accordance of Intersection in 3D/Sky Plot approaches

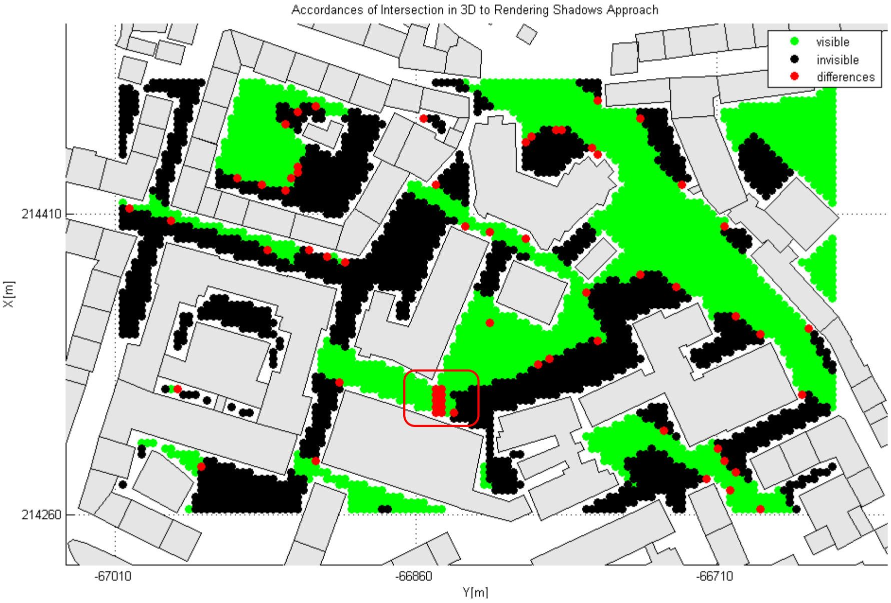


Figure 3.28.: Accordance of Intersection in 3D/Rendering Shadows approaches

the Sky Plot approach increases linearly with the number of points. The Rendering Shadows approach's computational time is independent of the number of points to be calculated. It is constant, since the entire grid (26040 points), and not a smaller subset (3727 points) was calculated. Since a high image resolution of 1920 x 1080 pixels was selected, every grid point is represented by at least 5.5 pixel (1 m corresponds to 1.88 pixel). Thus, an agglomeration of the grid to a spacing of 1 m instead of 3 m is without effect on the calculation time - provided, the grid's extent does not change.

The Sky Plot approach, which is chosen in this thesis for further fingerprint calculations, achieved an average computation time of 5.5 seconds for 32 satellites and 3727 grid points. It is independent of the resolution of lines of view, since the comparison of the elevations is evaluated only for the satellite's azimuth.

By using the acceleration concepts introduced in chapter 3.3.6, the computation complexity has been significantly reduced. Another concept proposed by [10] selects only the closest buildings for the LoS computation. For example, a selection within a radius of 100 m around the reference point would be possible. However, since distant buildings with large heights - which would be excluded from the calculation - could block the signal in reality, this method is not recommended. Another possibility would be to index the buildings in relation to the building's distance to a grid point. Thereafter, the LoS computation algorithm could intersect the nearest buildings to a grid point first. It would therefore exit the computation earlier, if an intersection occurs near to a grid point, which is more likely, than an intersection far off the grid point's vicinity. Hence, the computational effort would be decreased further. However, the effort of indexing in the pre-processing phase must be considered as well.

In general, the computation time depends on:

- Details of the 3D model of buildings
- Number of buildings
- Number of grid points
- The resolutions of images/lines of view for the Sky Plot and Rendering Shadows approach only

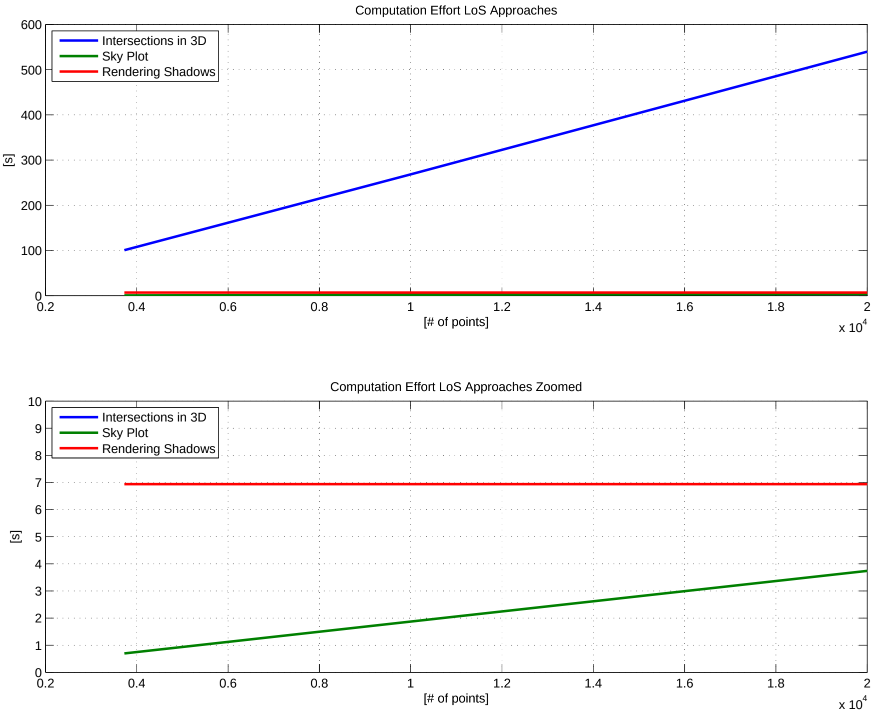


Figure 3.29.: Computation times for one satellite for each LoS computation approach

3.3.6. Acceleration Concepts

These concepts were used for improving the performance and computational efficiency of the approaches introduced in sections 3.3.2 and 3.3.3. These concepts reduce the amount of computation by up to 1800 times and therefore their implementation is crucial. The list gives an overview of implemented improvements:

1. Get subset of buildings: This process is used by the Intersection in 3D approach for LoS computation and in the pre-processing step of the Sky Plot approach.
2. Calculate azimuth angles to buildings: It is required for the retrieval of subsets of buildings (see above).
3. Calculate LoS for every azimuth in degree resolution: The Sky Plot approach uses this method in the pre-processing step.

Get subset of buildings

The LoS computations can be accelerated many times, by using a small selection of buildings for intersection. Based on a given azimuth α of a grid point to a satellite, a line with the grid point as origin is created. Then, a list of buildings which intersect this line is calculated. Figure 3.30 illustrates the procedure. The selected buildings are coloured blue. Two steps are needed for the calculation of subsets:

- Calculate azimuth angles to buildings: Generates a matrix of a minimal and a maximal azimuth angle from each grid point to each building (see section below).
- *Selection of Subset*: Get a subset denoted as buildings$_\alpha$ by comparing the azimuth of LoS α_{LoS} to the minimum α_b^{min} and maximum α_b^{max} azimuth of building boundaries observed from the current grid point:

$$min(\alpha_{building}) \leq \alpha_{LoS} \leq max(\alpha_{building}) \tag{3.22}$$

Calculate azimuth angles to buildings

The aim is to create a matrix, which contains the minimum and maximum azimuth angle from each grid point to each building. The algorithm for the determination of the minimum and maximum azimuth has to distinguish several cases, depending on the involved quadrants the line segments of buildings are located in, considered from the grid point's perspective. The quadrants are numbered based on the geodesic direction angle. There exist four possibilities (see figure 3.31):

1. The building is located in one of the four quadrants (black area in the figure).

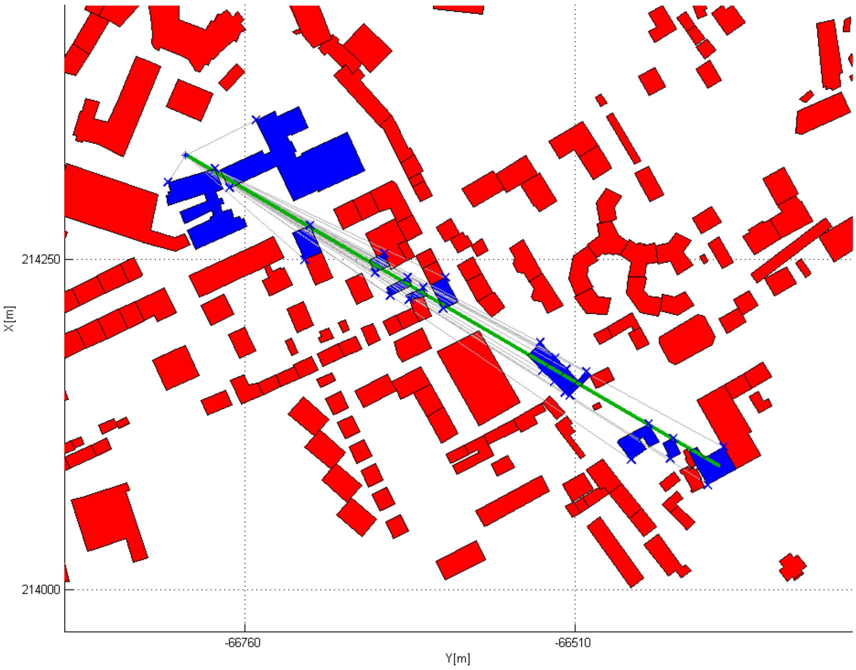


Figure 3.30.: Subset of buildings for a given azimuth and grid point

2. The building spans over two quadrants. To find the minimum and maximum angle, especially the transitions from quadrant 1 to 4 or 4 to 1 have to be taken care of (blue area).
3. Three quadrants are spanned by the building. Again, the transitions from quadrant 4 to 1 or 1 to 4 are crucial (green area).
4. Vertices and line segments of the buildings are in every quadrant (yellow area). The determination of the correct minimum and maximum azimuth is solved differently from the other approaches.

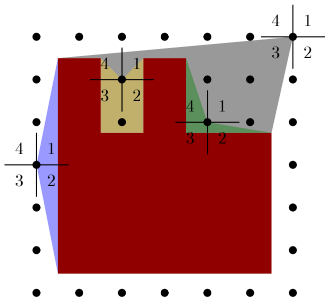
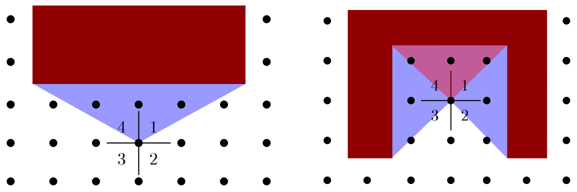


Figure 3.31.: Quadrants relative to grid points

Advantage: The selection of a subset of building is significantly faster.

Disadvantage: The minimal and maximal azimuth for a building is hard to obtain as there are many exceptions. For example:

1. If the vertex points of a building span from quadrant 4 to quadrant 1, then the building is to the north of the grid point. Thus, the maximum angle $270° < \alpha < 360°$ is the minimum and vice versa (see figure 3.32a).
2. If the building surrounds the grid point on three sides (see figure 3.32b), its edges cover all four quadrants (for example a u-formed building). The minimal and maximal azimuth angle are difficult to obtain, as it is unclear which angle is the minimal and which is the maximal angle. From determining the minimal and maximal azimuth to each building vertex, the wrong angles would be determined (see red triangle in figure 3.32b), although the blue area displays the actual range between the minimum and maximum azimuth. By determination of visible points from the grid point's perspective, the minimum and maximum angle can be obtained. For this, a visibility test has to be performed for each vertex of a building.



(a) 1 **(b)** 2

Figure 3.32.: Special cases of quadrant detection of building's line segments

Algorithm: The pseudo code given in algorithm 2 presents one possibility to handle the determination of azimuths to buildings from a given grid point. The invoked functions are given in algorithm 3. An explanation of the used notations is given in table 3.8.

Table 3.8.: Notations for the algorithm to detect the building's azimuth angle

b	Buildings: Set of vertices
e	Edge of a building
α_{all}	Set of azimuths of a grid point to all vertices of a building
e_s	Set of edge sectors
n_{sec}	Number of sectors the building's vertices are located in
v_p	Set of visible vertices
α_b	The result of the algorithm: Set of one minimal and one maximal azimuth of a building: $\alpha_b = \{min(\alpha_{all}), max(\alpha_{all})\}$

Algorithm 2 Get the maximum and minimum Azimuth from a grid point to a building

```
$\alpha_{all} \leftarrow$ calculate azimuths $\alpha$ from grid poit to each vertex in $b$
$e_s \leftarrow$ call *getEdgeSectors(b)*
$n_{sec} \leftarrow$ determine number of sectors of $s$
$e_s, n_{sec} \leftarrow$ call *handleSpecialSectorCases($e_s, n_{sec}$)*
**if** $n_{sec} \leq 2$ **then**                                  ▷ building is within one or two sectors
    $\alpha_b \leftarrow$ call *getMinMaxAlpha($\alpha_{all}$)* for all azimuths $\alpha$ of building $b$
**else if** $n_{sec} == 3$ **then**
    **if** $e_s == s\{4,1\}$ OR $e_s == s\{1,4\}$ **then**          ▷ when building spans from 4. to 1. sector
        $\alpha_b \leftarrow$ call *getMinMaxAlpha($\alpha_{all}$)*
    **else**
        $\alpha_b \leftarrow$ call *getMinMaxAlpha*($\alpha_{all}$)*
    **end if**
**else**                                                        ▷ building has vertices in all 4 sectors
    $v_p \leftarrow$ call *getVisiblePoints(b)*
    $\alpha_{all} \leftarrow$ calculate all azimuths $\alpha$ from a grid point to all vertices in $v_p$
    $\alpha_b \leftarrow$ call *getMinMaxAlpha($\alpha_{all}$)*
**end if**
```

Algorithm 3 Function invoked in algorithm 2

```
function getEdgeSectors($b$)
    for all $e$ in $b$ do
        $sectors$: add sectors of vertices of edge $e$
    end for
    return $sectors$
end function

function handleSpecialSectorCases($e_s, n_{sec}$)
    $\Delta e_s \leftarrow$ determine the difference of sectors of every edge
    if $\Delta e_s == 2$ then    ▷ at least one building edge spans over 3 sectors; increase number of sectors $n_{sec}$
        if $n_{sec} == 3$ then
            proof which additional sector the edge would cross
            if if there is already a point $p$ of $b$ in this sector then
                exit
            else
                continue
            end if
        end if
        $n_{sec} \leftarrow n_{sec} + 1$
        if $n_{sec} == 3$ then
            if building edges $e_s$ span from 4->2 OR 2->4 OR 1->3 OR 3->1: building north of grid point then
                $e_s \leftarrow$ add edge sectors 4 to 1
            end if
        end if
    end if
    return $e_s, n_{sec}$
end function

function getMinMaxAlpha($\alpha$)
    $\alpha_{min}, \alpha_{max} \leftarrow$ call method getMinMaxAlpha*($\alpha$)
    if $\alpha_{max} - \alpha_{min} > \pi$ then
        for all $a$ in $\alpha$ do
            if $a > \pi$ then
                $a \leftarrow a - 2\pi$
            end if
        end for
        $\alpha_{min}, \alpha_{max} \leftarrow$ call method getMinMaxAlpha*($\alpha$)
    end if
    return $\alpha_{min}, \alpha_{max}$
end function

function getMinMaxAlpha*($\alpha$)
    $\alpha_{min} \leftarrow \min(\alpha)$
    $\alpha_{max} \leftarrow \max(\alpha)$
    return $\alpha_{min}, \alpha_{max}$
end function

function getVisiblePoints($b$)
    $d_{max} \leftarrow$ maximum distance from grid point to building vertices
    for all $p$ in $b$ do
        $x_p \leftarrow$ vector from grid point to vertex $p$ with length $d_{max}$
        $v_p \leftarrow$ if vector $x_p$ from grid point to $p$ intersects no edges $e$, add vertex $p$ to set of visible points
    end for
    return $v_p$
end function
```

Calculate LoS for every azimuth in degree resolution

To increase the calculation speed of the Sky Plot approach, the intersection of generated lines of view with line segments of buildings is calculated first. These are 2D intersections, which means, that only the groundplan polygon of a building is considered. The lines of view are referred to as the 2D representation of the lines of sight (LoS) in the groundplan of the model.

The lines of view are generated for each grid point in all directions with a discrete angular spacing (see figure 3.33). The blue circles indicate the intersections along the lines of view. The computation results in a list for every grid point, that indicates, which buildings intersect the lines of view at a certain azimuth. It contains the coordinates of the corresponding intersections. Without optimization, the calculation would be very time consuming. The time complexity of the non-optimized algorithm can be given by:

$$O(l \cdot n \cdot m \cdot o \cdot p), \tag{3.23}$$

with

l	number of lines of view
n	number of grid points
m	number of buildings
o	average number of line segments/building
p	number of arithmetic operation/intersections

The process would take up to approximately 33 days using a Intel i7-4500U 1.8-3GHz dual core processor. To speed up the computation process significantly, a subset of buildings in the azimuth's direction is used (see Get subset of buildings on page 61). With this, an acceleration of 1800 times is achieved - which is 0.5‰ of the previous computation time. This corresponds to 26 minutes for a grid of 26040 points. The disadvantage is, that the lines of view are generated for discrete azimuth angles. Therefore, some buildings, especially those far off, are not intersected. This may lead to inaccuracies (see figure 3.34).

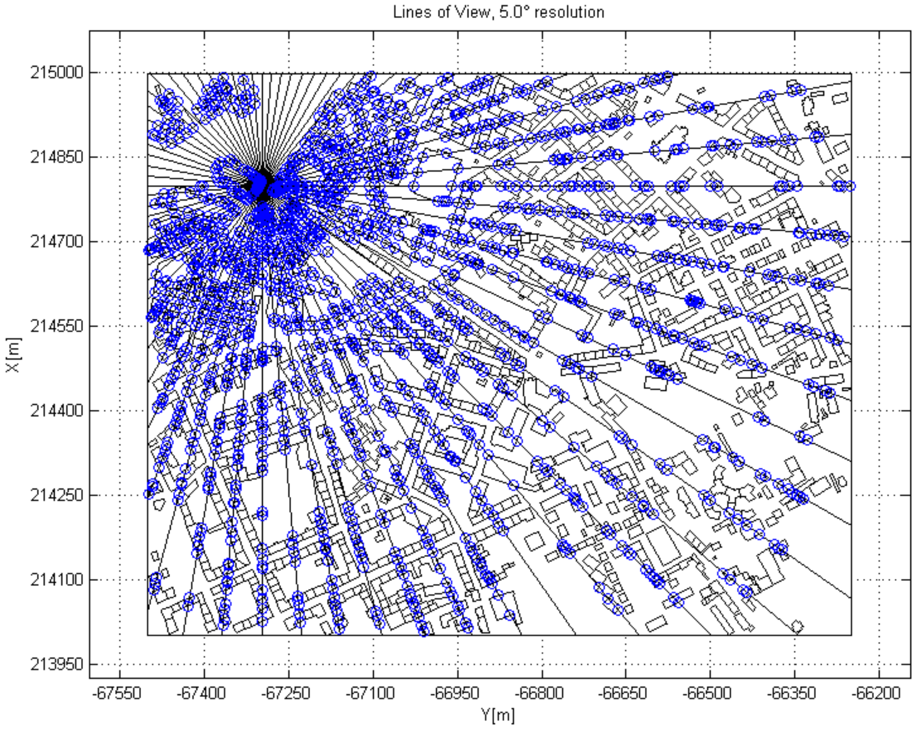


Figure 3.33.: Line of sights in 2D

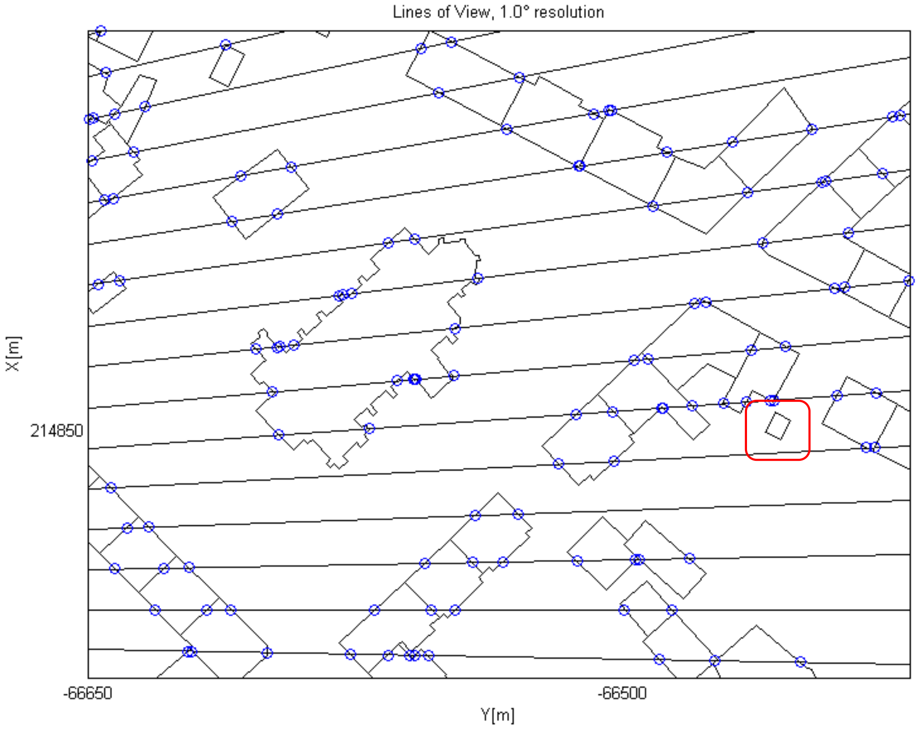


Figure 3.34.: Lines of view from first grid point

3.4. Satellite Positions

The implemented algorithm for the calculation of satellite positions is the topic of this chapter. Furthermore, several error analyses have been investigated in terms of the accuracy of satellite position determination.

3.4.1. Orbit Determination with Broadcast Ephemeris

The navigation files of the RINEX 2.11 format serve as data source for the orbit determination. A description of the format is given in [15]. An FTP server provided by [3], provides Broadcast Ephemerides for previous days for both GPS, as well as other GNSS. In this thesis only GPS satellites are used. Since fingerprints have to be calculated in advance (see section 2.1), the ephemeris send by the GNSS satellites cannot be utilized. Therefore, not current, but outdated ephemeris are used for calculation, if positioning in real time is required. Chapter 3.4.2 introduces an error analysis based on dated ephemeris and their repercussion on the accuracy of orbit determination.

The provided RINEX files contain ephemeris of the day prior. They comprise data for all 32 GPS satellites throughout the day at intervals of two hours. Broadcast ephemeris are valid for +/- two hours, so the last ephemeris specification available are used for calculations to minimize the effect of dated ephemeris. The last entry of ephemeris in the RINEX files from [3] is usually provided for 12 p.m. of the previous day and thus 0 a.m. on the current day. If no ephemeris are found for a satellite for 12 p.m., the previous valid set of latest ephemeris is used. The parameters, that are specified in the RINEX file, are mapped to orbit parameters listed in table 3.9. The mapped orbit parameters can be inserted in the formulas given in section 2.3.2 for satellite orbit calculation. For a description of the RINEX file and containing parameters, see [8].

3.4.2. Error Analysis concerning Orbit Determination

Effect of dated Ephemeris

Since dated ephemeris are used, the effect of the age on the accuracy of computed satellite orbits was investigated. Satellite positions with the latest, and in contrast, with older ephemeris were calculated. From this, position differences based on the latest ephemeris were derived. The position differences are expressed by the relative distance to each other. The evaluation was carried out for ephemeris, which were published a week ago, up to currently published ones. Each consecutive day's positions were compared to the positions yielded from the actual and therefore most accurate ephemeris. This simulates the effect of predicted positions a week ahead, with the advantage of having accurate ephemeris for comparison. If an approximate

Table 3.9.: Mapping of RINEX to Orbital Parameters

RINEX parameter	Description/Unit	Orbital Parameter	Description
Toc	Time of Clock (GPST) [y, m, d, H, M, S]	t_c	Satellite clock reference epoch
GPS Week #	Time of Ephemeris (GPST) [week]		
Toe	Time of Ephemeris (GPST) [sec]	t_e	Ephemeris reference epoch
sqrt(A)	[sqrt(m)]	$\sqrt{a}$	Square root of the semi major axis of the satellite orbit
Delta n	[radians/sec]	Δn	Mean motion difference
M0	[radians]	M_0	Mean anomaly at reference epoch
e		e	Eccentricity
omega	[radians]	ω_0	Argument of perigee
Cuc, Cus	[radians]	C_{uc}, C_{us}	Correction coefficients (argument of perigee)
Crc, Crs	[m]	C_{rc}, C_{rs}	Correction coefficients (geocentric distance)
Cic, CIS	[radians]	C_{ic}, C_{is}	Correction coefficients (inclination)
i0	[radians]	i_0	Inclination
IDOT	[radians/sec]	$\dot{i}$	Rate of Inclination angle
OMEGA	[radians]	Ω_0	Right ascension of ascending node
OMEGA DOT	[radians/sec]	$\dot{\Omega}$	Rate of node's right ascension

distance of 20 000 km to the satellite is assumed, the error on the elevation angle would be calculated by

$$\Delta \mathbf{x}_{\mathrm{Sat}}(t, e, e_{pre}) = \mathbf{x}_{\mathrm{Sat}}(t, e) - \mathbf{x}_{\mathrm{Sat}}(t, e_{prev}) \tag{3.24}$$

with

$$e_{prev} = e(t - i), \quad i = 7, 6, ...1 \tag{3.25}$$

$$b = r\Delta\varepsilon = |\Delta \mathbf{x}_{\mathrm{Sat}}| \tag{3.26}$$

$$\Delta\varepsilon = \frac{|\Delta \mathbf{x}_{\mathrm{Sat}}|}{20 \cdot 10^3 \text{ km}} \tag{3.27}$$

with $|\Delta \mathbf{x}_{\mathrm{Sat}}|$ being the error in distance between a satellite position calculated with actual ephemeris e at time t, and positions calculated with dated ephemeris e_{prev}, also at time t.

The i donates the days subtracted from the actual date t up to one week. The error in the elevation angle is donated as $\Delta\varepsilon$ and r is the distance from a user to a satellite.

The investigations have been conducted for three points in time, separated by five hours from each other. Figure 3.35a shows the error in distance and the impact of this on the elevation angle perceived by the user. The errors were computed by averaging the values for all 32 satellites day by day. The corresponding standard deviations can be seen in figure 3.35b. It can be seen, that the error both in distance and elevation decreases with more up to date ephemeris. The maximum error, occurring with ephemeris dating a week back, is of about 6 km in distance which corresponds to an error of 65" in elevation. The standard deviation from a maximum of 3.5 km (36") decreases sequentially, reaching a minimum of 0.3 km (3") at $i = 1$. Thus, the impact of wrong building boundary detection due to an inaccurate 3D city model (see section 4.1.5) induces far graver errors than dated ephemeris. Investigations with different days as evaluation points in time, and therefore other ephemeris data, show the same results.

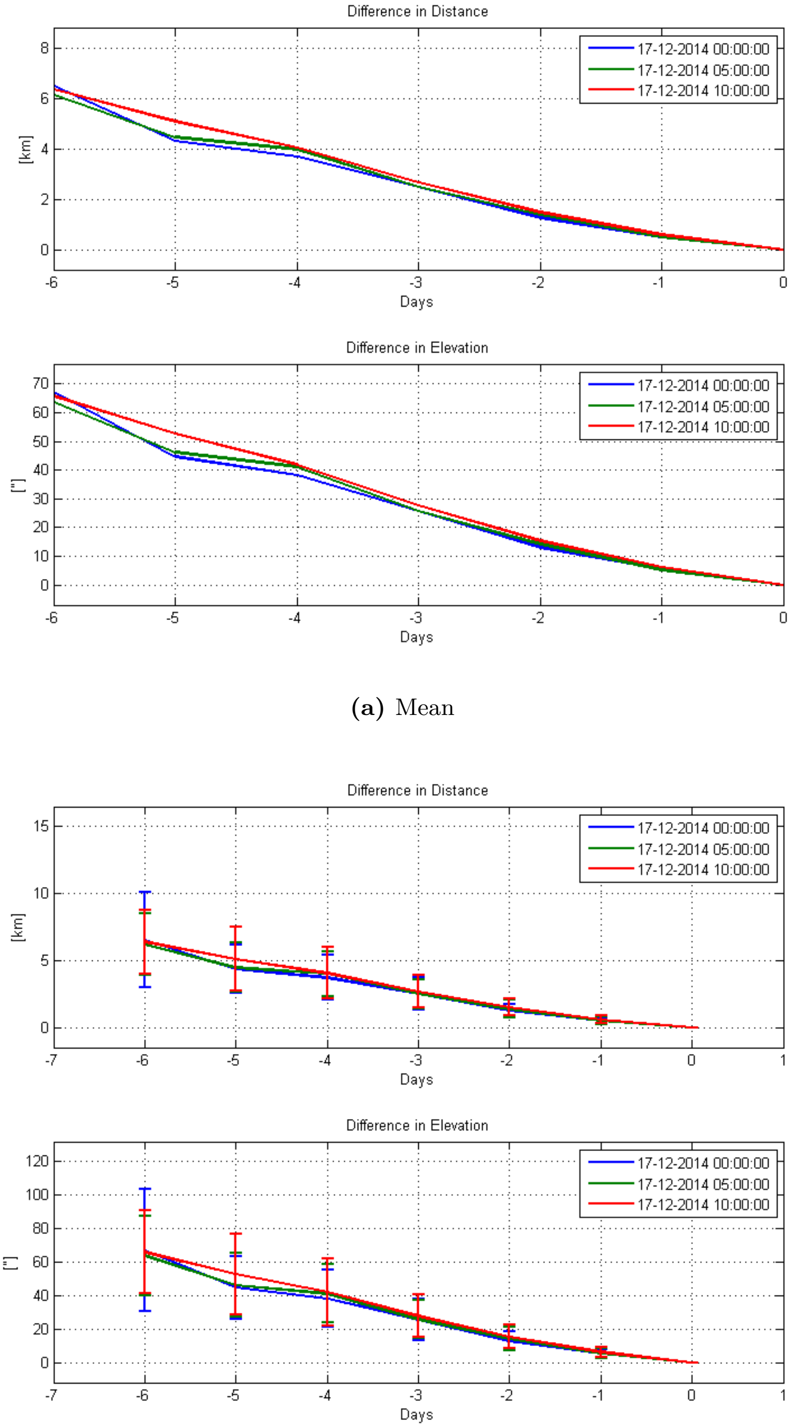
Neglect of Receiver Clock Error

In a single point solution, the unknown satellite and receiver clock error as well as errors induced by the earth's ionosphere and troposphere are modelled as part of the least squares adjustment which yields the user's position. It is an iterative computation, as the receiver clock error and the satellite clock error are used for the determination of the geometrical distance $\rho(x_{satellite}, x_{receiver})$ and therefore also induce errors in the determination of the satellite position. In this thesis, the receiver clock error can not be estimated due to the satellite position computation, which is performed in advance. Hence, an estimation of the impact of neglecting the receiver clock error on the satellite position is important. In formula 3.28 the error with respect to the satellite position is given:

$$\Delta\mathbf{x}_{\text{Sat}}(\delta t_R(t)) = \begin{bmatrix} v_x^S(t) \\ v_y^S(t) \\ v_z^S(t) \end{bmatrix} \delta t_R(t), \tag{3.28}$$

where $\Delta\mathbf{x}_{\text{Sat}}$ is the difference in position, given in meters, $\mathbf{v}^S$ is the vectorial satellite speed and $\delta t_R(t)$ is the receiver clock error. With an calculated speed $\mathbf{v}^S$ of approximately 3000 m/s and a maximal receiver clock error for a u-blox 6 GPS receiver of 1 ms, this gives an error of 3 m. Table 3.10 lists some computed errors for δt_R of the u-blox 6 GPS receiver. The satellite speeds $|\mathbf{v}^S|$ given, represent the minimum and maximum values computed over several epochs. The variations of speeds can be explained by the eccentricity of the satellite's orbits as well as disturbing forces caused by the earth, moon, and sun.

The impact on the position is relatively small, even smaller than the error from dated ephemeris. The neglect of the receiver clock error has no negative consequences on GNSS fingerprinting at all.



(a) Mean

(b) Standard Deviation

Figure 3.35.: Estimation of error induced by dated ephemeris

Table 3.10.: Receiver Clock Error impact on the satellite position

$\|\mathbf{v}^S\|$ [m/s]	δt_R [s]	$\|\Delta \mathbf{x}_{\mathrm{Sat}}\|$ [m]
3811	1e-3	3.811
3959	1e-3	3.959
3811	2e-7	7.622e-04
3959	2e-7	7.918e-04

Usage of an approximate user position for transit time calculation

For the transit time calculation an approximate user position is used (see chapter 2.3.2). As an approximate position, the center of the grid was chosen. This causes an error in the transit time and subsequently an error in satellite position. Figure 3.36 illustrates the issue, if a user is located at site A or B, but location C at the grid's centre would be used as approximate position for the user's location. Consequently, the distances x_1, x_2 and x would differ from each other. The error can be estimated by knowing the extent of the used grid. In this thesis, a grid dimension of 849 x 600 m has been used. The error is higher for low satellite elevations and lower for very high elevations.

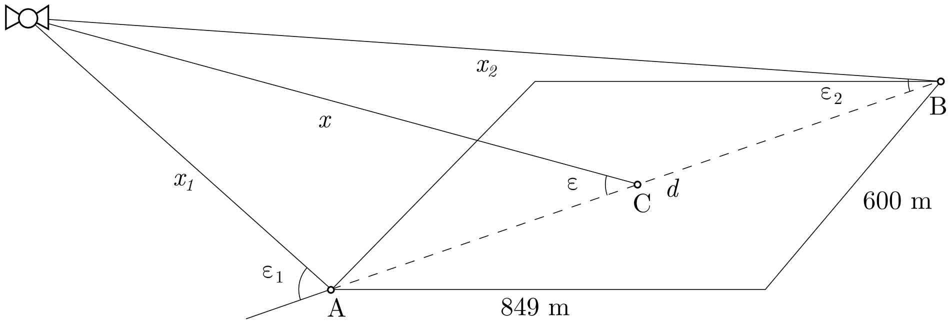


Figure 3.36.: Estimation of error of transit time

By applying the law of cosines, the error in distance $dx_1 = x - x_1$ and $dx_2 = x_2 - x$ can be calculated. The error in satellite position $\Delta \mathbf{x}_{\mathrm{Sat}}$ is the transit time error in seconds, multiplied with the satellites speed:

$$dx_1 = x - \sqrt{\left(\frac{d}{2}\right)^2 + x^2 - 2\frac{d}{2}x\cos\varepsilon} \tag{3.29}$$

$$dx_2 = -x + \sqrt{\left(\frac{d}{2}\right)^2 + x^2 - 2\frac{d}{2}x\cos(\pi - \varepsilon)} \tag{3.30}$$

$$\Delta \mathbf{x}_{\mathrm{Sat}_i} = \frac{dx_i}{c}\mathbf{v}^S \tag{3.31}$$

$$\text{with c= 299792458 [m/s], and } i = 1, 2 \tag{3.32}$$

The values for dx_1 and dx_2 have been calculated by using $x = 20\ 000\,\text{km}$ as approximate distance to a satellite and $d = \sqrt{849^2 + 600^2}\,\text{m}$ as maximal distance d between two points on the grid. The resulting satellite position error can be seen in table 3.11 below. Again, the errors are negligible compared to errors introduced by inaccurate 3D city modelling. An approximate position - the grid's centre - has thus been used in the thesis.

Table 3.11.: Error of transit time, expressed in meters

ε [°]	dx_1 [m]	dx_2 [m]	$\lvert\mathbf{v}^S\rvert$ [m/s]	$\lvert\Delta\mathbf{x}_{\text{Sat}_1}\rvert$ [mm]	$\lvert\Delta\mathbf{x}_{\text{Sat}_2}\rvert$ [mm]
1	519.729	519.729	3000	5.2	5.2
30	450.165	450.169	.	4.5	4.5
70	177.779	177.791	.	1.8	1.8
90	6.75e-3	6.75e-3	.	6.8e-5	6.8e-5

3.5. Fingerprints

3.5.1. Shadow Matching

The measured SNR values are assigned to visible or invisible LoS groups based on a defined SNR cut-off (see chapter 4.1). A vector $\mathbf{v}_m$, the measured visibility vector, is formed with these groups, where visible satellites are donated as logical ones and invisible as logical zeros. The dimension of $\mathbf{v}_m$ is equal to the number of GPS Satellites accordingly: $dim(\mathbf{v}_m) = (1 \times 32)$. A pre-computed fingerprint $\mathbf{FP}_{sat_i}(t)$ for each GPS satellite (PRN 1-32) at time t is provided as binary array by the DB and contains ones and zeros, corresponding to LoS and NLoS seen from the users perspective. Therefore, one fingerprint contains information of one satellite's visibility to all grid points.

The binary arrays $\mathbf{FP}_{sat_i}(t)$ for each satellite are stored in a fingerprint matrix $\mathbf{M}_{FP}(t)$ (eq. 3.33) with the dimensions $dim(\mathbf{M}_{FP}) = (\#\text{ satellites} \times \#\text{ of grid points}) = (n \times m)$. Note that the dimension of $\mathbf{M}_{FP}(t)$ does not change over time, as the number of satellites and grid points is constant. A fingerprint matrix consists of m visibility vectors $\mathbf{v}_{gp_i}$.

$$\mathbf{M}_{FP}(t) = \begin{bmatrix} FP_{sat_1}(t) \\ FP_{sat_2}(t) \\ \vdots \\ FP_{sat_n}(t) \end{bmatrix} = \begin{bmatrix} v_{sat_1,gp_1} & v_{sat_1,gp_2} & \cdots \\ v_{sat_2,gp_1} & v_{sat_2,gp_2} & \cdots \\ \vdots & & \\ v_{sat_n,gp_1} & \cdots & v_{sat_n,gp_m} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_{gp_1} & \mathbf{v}_{gp_2} & \cdots & \mathbf{v}_{gp_m} \end{bmatrix} \quad (3.33)$$

$$\text{e.g. } \mathbf{M}_{FP}(t) = \begin{bmatrix} 01011100\ldots \\ 10000010\ldots \\ \vdots \\ 01001011\ldots \end{bmatrix} \quad (3.34)$$

A bit-wise comparison of the measured visibility vector $\mathbf{v}_m$ and the database visibility vectors $\mathbf{v}_{gp_i}$ yields the compliance of the measurement and the model. If the model corresponds to the measurement, λ_{sat_k,gp_i} in (3.35) is set to true. The comparison is repeated for each grid point gp_i and satellite sat_k at each time epoch t. Hereby an assertion about the user location probability λ_{gp_i} - the mean of the compliance vector λ_{sat_k,gp_i} - can be taken (eq. 3.37). In general, multiple positions with the highest probability $\max \lambda_{gp_i}$ exist. The vector of location probabilities for the whole grid at one epoch is given in equation 3.37.

$$\lambda_{sat_k,gp_i} = \begin{cases} 1, & \text{if } v_{sat_k,gp_i} = v_m^{sat_k} \\ 0, & \text{otherwise} \end{cases} \quad (3.35)$$

$$\boldsymbol{\lambda}_{gp_i} = \begin{bmatrix} \lambda_{sat_1,gp_i} \\ \lambda_{sat_2,gp_i} \\ \vdots \\ \lambda_{sat_n,gp_i} \end{bmatrix} \tag{3.36}$$

$$\mathbf{L_p}(t) = \begin{bmatrix} \lambda_{gp_1} & \lambda_{gp_2} & \dots & \lambda_{gp_m} \end{bmatrix} \qquad \text{with } \lambda_{gp_i} = \tilde{\boldsymbol{\lambda}}_{gp_i} = \frac{1}{n}\sum_{k=1}^{n} \lambda_{sat_k,gp_i} \tag{3.37}$$

An alternative representation is given in equation (3.38) of [1] by calculating the Hamming distance H between $\mathbf{v}_m$ and $\mathbf{v}_{gp_i}$. H represents the number of different bits between both vectors and can be calculated by logical XNOR, also called logically equivalence. A location gp_i can be estimated by the shortest Hamming distance (3.39), which is the model vector which best applies to the measurement vector.

$$H_i(\mathbf{v}_m, \mathbf{v}_{gp_i}) = \sum_{k=1}^{n} v_{sat_k,gp_i} \oplus v_m^{sat_k} \tag{3.38}$$

$$\underset{gp_i \in FP}{\arg\min}\, H_i(\mathbf{v}_m, \mathbf{v}_{gp_i}) \tag{3.39}$$

3.5.2. The Fingerprint Database

This section treats the method of storing fingerprints to the database. It is different from the retrieval of geometric data (as for example the 3D building model or the grid of reference location) discussed in the previous section 3.2.2 on page 30, as it holds no information about location whatsoever. The link to a grid point's location must be made through the ID and a table which holds the geometric data of the grid points. Another difference is the amount of data that needs to be stored. An array of the size: *number of satellites* times *the number of points* (32 × 3727) is created for each epoch in advance. To reduce the amount of data processing the database has to perform, this information is stored within a single entry, represented by a single line of a table in the database.

To reduce the data rate, the computed fingerprint from the Sky Plot approach is converted from a binary array to a hexadecimal beforehand. The binary array contains zeros where satellites cannot be seen and ones where a visibility is predicted. With this approach, the data rate and storage can be reduced by four times. The hexadecimal arrays of all satellites are then transmitted as one string array to the database. A UTC time-stamp, which indicates at which time the model is applicable, is stored along with the fingerprint array, consuming one data entry per epoch. Fingerprints of invisible satellites are stored as empty strings (see listing 3.2). Timestamps are of the PostgreSQL data type `timestamp with time zone`, whereas the fingerprints are of the type `character varying array` (see listing 3.3). Listing 3.2 shows an example of one data entry in the fingerprint database. The first entry is an ID, increasing sequentially, the second is the fingerprint string array followed by the time stamp displayed in the local time zone (CET). Internally, the time-stamp is stored in UTC time. Access to a fingerprint is accomplished by providing a time-stamp. At the client side (the user measuring the SNR values), the received fingerprint is decoded to a binary array and compared to the measurements.

Listing 3.2: Example of a fingerprint array

```
1; "{"","",1C00007FFF18FC0001FFFCFF00007...,(...)}"; "2014-09-25
    11:00:00+02"
```

Listing 3.3: SQL datatypes for the fingerprint database

```
CREATE TABLE fingerprints
(
  id integer NOT NULL DEFAULT nextval('footprints_id_seq'::regclass),
  fingerprint character varying[],
  utctime timestamp with time zone,
  CONSTRAINT footprints_pkey PRIMARY KEY (id)
);
```

3.6. Projections and Transformations

The reference systems and frames of the data involved in the processing of the thesis are listed in this chapter. Additionally, an overview of the required transformations/projections is given.

Reference Frames and Projections The following tables list the required reference frames and projections as well as the transformations and mappings.

Table 3.12.: Reference frames and projections

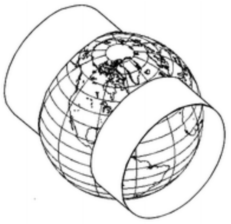
Object	Ref. Sys./Frm.	Geo. Date	Projection	Heights
DEM		MGI	MGI Aut GK East	Orthometric
3D Buildings		MGI	MGI Aut GK East	Orthometric
3D Grid		MGI	MGI Aut GK East	Orthometric
3D Model in Blender	LLF			Orthometric
Satellites (GPS)	WGS-84	WGS-84		Ellipsoid
Rendered Image (2D)	Image Body Frm.			-

As can be seen in table 3.13, 3D datum transformations, 2D datum transformations (both Helmert transformations) and a Reverse Transverse Mercator Projection were used to provide the data in the appropriate form. To calculate the difference vector between a satellite position (WGS84) and a reference position (grid point in MGI Austria GK East projection), both coordinates have to be in the same reference system.

A projection of WGS84 satellite coordinates to the Gauss Krueger projection is not possible, as an error would be induced to coordinates, due to the distance of GPS satellites to the earth's surface ($h_{GPS} \sim 20\,200\,km$ above ground). This is, because a Gauss Krueger projection is defined for points on the earth's surface, and the reference ellipsoid for the projection is not defined globally, but adjusted to a confined region. Therefore, a conversion of grid

Table 3.13.: Transformation and mapping

			Transformation / Projection
3D Buildings/Grid	⇒	3D Model in Blender	3D Datum Transformation (Helmert)
3D Grid (MGI Aut GK East + Orthometric Heights)	⇒	3D Grid (WGS84)	Reverse Transverse Mercator Projection & 3D Datum Transformation (Helmert)
2D Grid (MGI Aut GK East)	⇒	Rendered Image (2D)	2D Datum Transformation (Helmert)



Source: http://kobi.nat.uni-magdeburg.de/patrick/uploads/BEng/trans_mercator.png

Figure 3.37.: Transversal Mercator Projection

point coordinates to the WGS84 system is used. A simple Transversal Mercator Projection illustration can be seen in figure 3.37.

The context of projections and transformations is shown in figure 3.38. A grid point, given in the Gauss Krueger projection is first reverse projected to ellipsoidal coordinates on the Bessel ellipsoid. Here, the geoid undulation is applied to the orthometric height provided by the DEM. The undulation for the area of Graz is $N = 0.08\,m$. Then, the ellipsoidal coordinates are expressed as Cartesian coordinates and with transformation parameters given in the appendix A.1, the coordinates are transformed to WGS84 Cartesian coordinates. Thus, the grid points are in the same system as the satellite positions computed from the broadcast ephemeris. From this, the difference vector can be determined, which is transformed to the local level frame (LLF) to get azimuth and elevation from a user position to a satellite. They serve as input quantities for the Sky Plot and Rendering Shadows approach. For the Intersections in 3D approach, only the Cartesian positions are needed. The formulas for the transformations and projections are given in the appendix A.1.

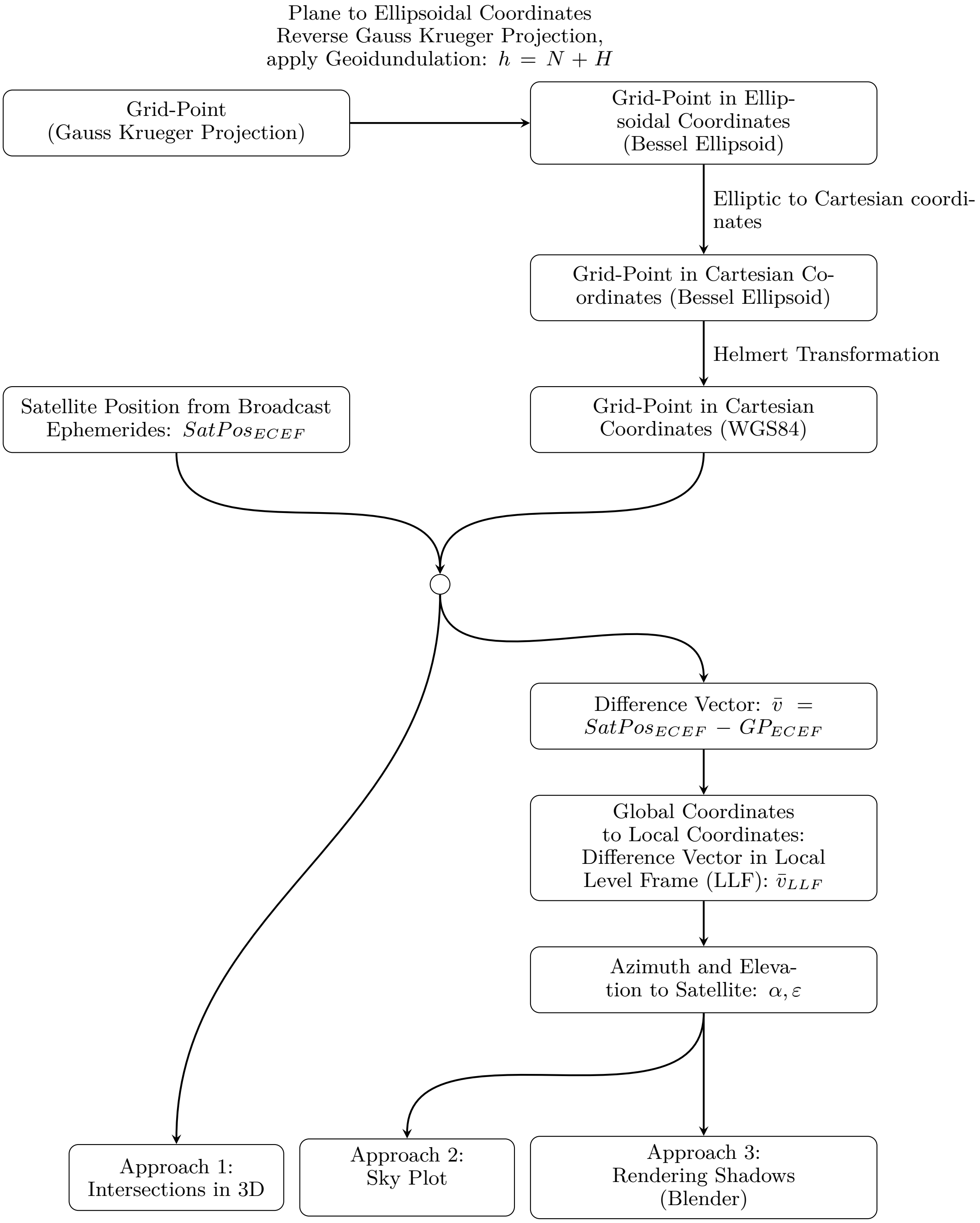
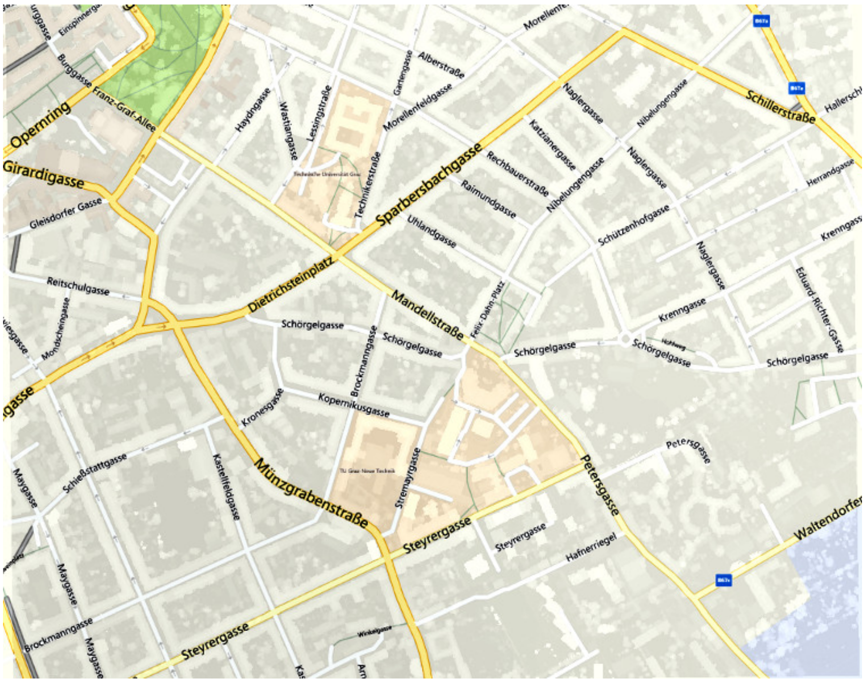

Plane to Ellipsoidal Coordinates
Reverse Gauss Krueger Projection,
apply Geoidundulation: $h = N + H$
Grid-Point (Gauss Krueger Projection)
Grid-Point in Ellipsoidal Coordinates (Bessel Ellipsoid)
Elliptic to Cartesian coordinates
Grid-Point in Cartesian Coordinates (Bessel Ellipsoid)
Helmert Transformation
Satellite Position from Broadcast Ephemerides: $SatPos_{ECEF}$
Grid-Point in Cartesian Coordinates (WGS84)
Difference Vector: $\bar{v} = SatPos_{ECEF} - GP_{ECEF}$
Global Coordinates to Local Coordinates: Difference Vector in Local Level Frame (LLF): $\bar{v}_{LLF}$
Azimuth and Elevation to Satellite: α, ε
Approach 1: Intersections in 3D
Approach 2: Sky Plot
Approach 3: Rendering Shadows (Blender)

Figure 3.38.: Overview projections & transformations

4. Investigations

The chapter investigations discusses the methods to detect visible satellites from SNR measurements (section 4.1), to compare the measurements with the predicted fingerprint model thereafter. Two GPS antennas have been compared on their performance regarding SNR measurement sensitivity and multipath mitigation. Therefore, several test locations were evaluated. The positioning with fingerprints is the topic of chapter 4.2. A trajectory has been measured, where the fingerprint positioning results at way points were evaluated. Concepts to determine the position within the grid of location probabilities are pointed out, as well as the performance compared to GPS measurements, which were taken at the same time and at the same locations. The effects on the accuracy of GPS fingerprinting (shadow matching) is discussed in the final chapter 4.3.

The test area (see figure 4.1) is part of the campus of the Graz Technical University and its surrounding areas, where a diverse arrangement of buildings, building heights and orientation of street canyons is present, to investigate different locations for their influence on GNSS fingerprinting.



Source: https://www.google.at/maps

Figure 4.1.: Test area for SNR and GNSS fingeprint investigations

4.1. SNR Investigations

4.1.1. Overview

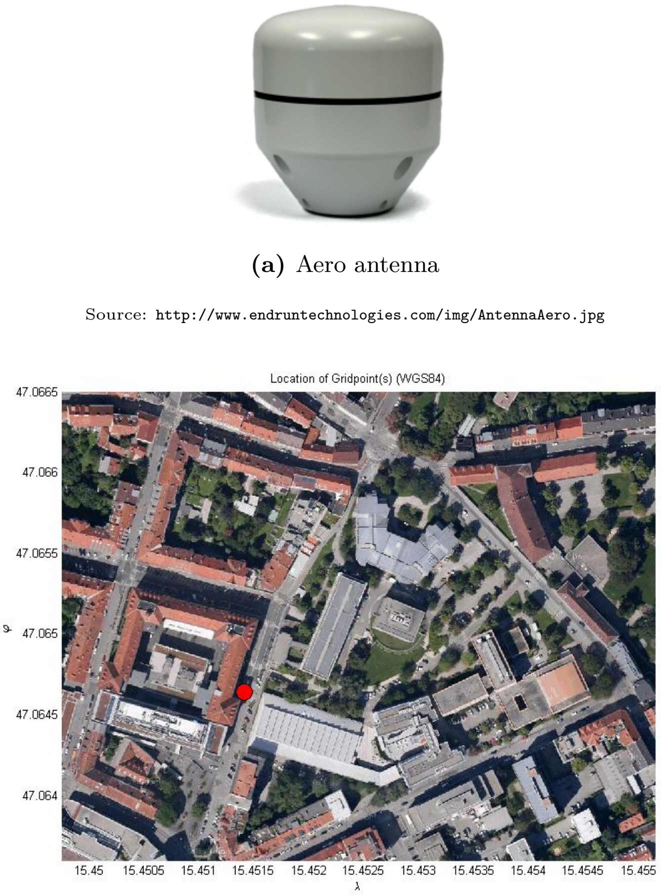
The signal to noise ratio (SNR) indicates the signal's strength, and therefore the visibility to a satellite. To distinguish between visible (LoS) and invisible (NLoS) satellites, a threshold for the SNR measurements is introduced, called SNR cut-off in the following. The investigation of different SNR cut-off values, to determine whether a satellite is visible from a users perspective, has turned out to be a crucial factor for more accurate positioning results with GNSS fingerprinting. Another focus was the difference of a high sensitive (Aero) compared to a low sensitive (u-blox ANN-MS) GPS antenna, regarding the sensitivity of SNR measurements. Measurements with the high sensitive GPS antenna showed high SNR values from satellites, which were blocked by buildings. By using a less sensitive antenna, there might be less measurements of high SNR of satellites, that should be invisible corresponding to the computed model. Therefore the comparison between a high and low sensitive antenna, as well as the investigation of their SNR cut-off value's impact on the probability of the correct user location, are conducted in this chapter. SNR cut-off values are referred to as cut-off values in the following.

4.1.2. SNR cut-off investigation

In this subsection, different cut-off values for both antennas are discussed. To determine the most promising cut-off, SNR time series, as well as figures of user location probabilities and sky plots at the actual user position have been compared and analysed. Measurements were conducted at five different locations with both antennas measuring simultaneously. The results below show representative results, results where GNSS fingerprinting may enhance the quality of other positioning methods and in contrast, locations where GNSS fingerprinting is very difficult to use. Both antennas were positioned similar to holding a smartphone at hip height (approximately 1 m above the ground) and therefore also the alignment of the user conducting the measurements is important, as the user blocks LoS as well.

Aero GPS Antenna

The Aero GPS antenna shown in figure 4.2a is a high sensitive antenna and was used in combination with a u-blox 6 GPS receiver. Figure 4.2b displays the location of the first measurement in the Stremayergasse. As can be seen in the aerial image, the position measured is located in an approximate north-south directed street. Thus, signals emitted from east and west directions, seen from a user perspective, are blocked.



(a) Aero antenna

Source: http://www.endruntechnologies.com/img/AntennaAero.jpg

(b) Location 1

Source: https://www.google.at/maps

Figure 4.2.: Antenna and location 1 of Aero antenna measurements

Location 1 The pre-computed satellite positions and the boundaries of the buildings from a users perspective at the grid point 20744 can be seen in figure 4.3. The sky plot was computed from the elevation and azimuth angles of building boundaries as well as satellite positions, computed from ephemeris for the measured timespan. These angles are used, when computing the visibility for a grid point in advance, by comparing the elevation angle of satellites to those of buildings at the satellites azimuth. From the sky plot, the visibility of satellites 32 and 4 is evident, thus satellites in the zenith or north-south direction of the user position are visible. On the other hand, satellites (PRN 11, 14) are predicted to be invisible, but the measurements yielded high SNR values. When comparing the SNR values of location 1 over a timespan of approximately 30 seconds, an investigation concerning the accordance of the measurements to the model can be performed.

Figure 4.4 shows the SNR values for the Aero antenna. The chosen SNR cut-off value is indicated by a dashed line at a SNR of 40 dB-Hz, which led to the best results. The determination of the cut-off value, and why 40 dB-Hz has been used, is discussed later in this section. However, two satellites were measured (PRN 11 and PRN 14) with a high SNR value, which should not be observable, corresponding to the computed sky plot model. This influences the results of possible user locations.

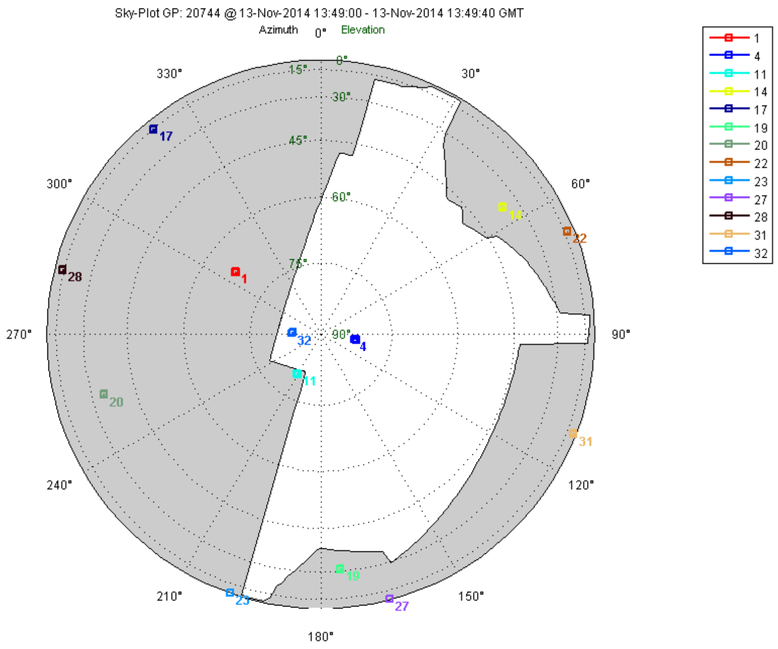


Figure 4.3.: Sky Plot at location 1

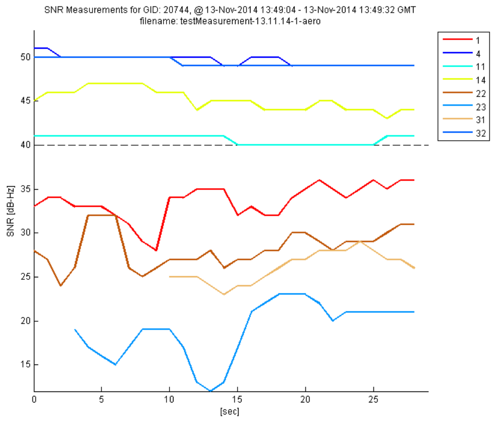


Figure 4.4.: SNR measurement of Aero antenna at location 1

In table 4.1, the numerical values of the SNR measurements over two epochs are displayed. Measurements of PRNs, that are visible with respect to the sky plot, are coloured in green (computed model). For each epoch, two cut-off values have been tested. The optimal case would be a cut-off indicating only satellite 4 and 32 as measured. The decision for a appropriate cut-off is difficult, as it depends on the location of the user. If a higher cut-off -

for example 47 dB-Hz - would have been used, this would work for location 1 (compare the table below), but not for other locations in the grid. Satellite's SNRs would be cut-off, which should be measured.

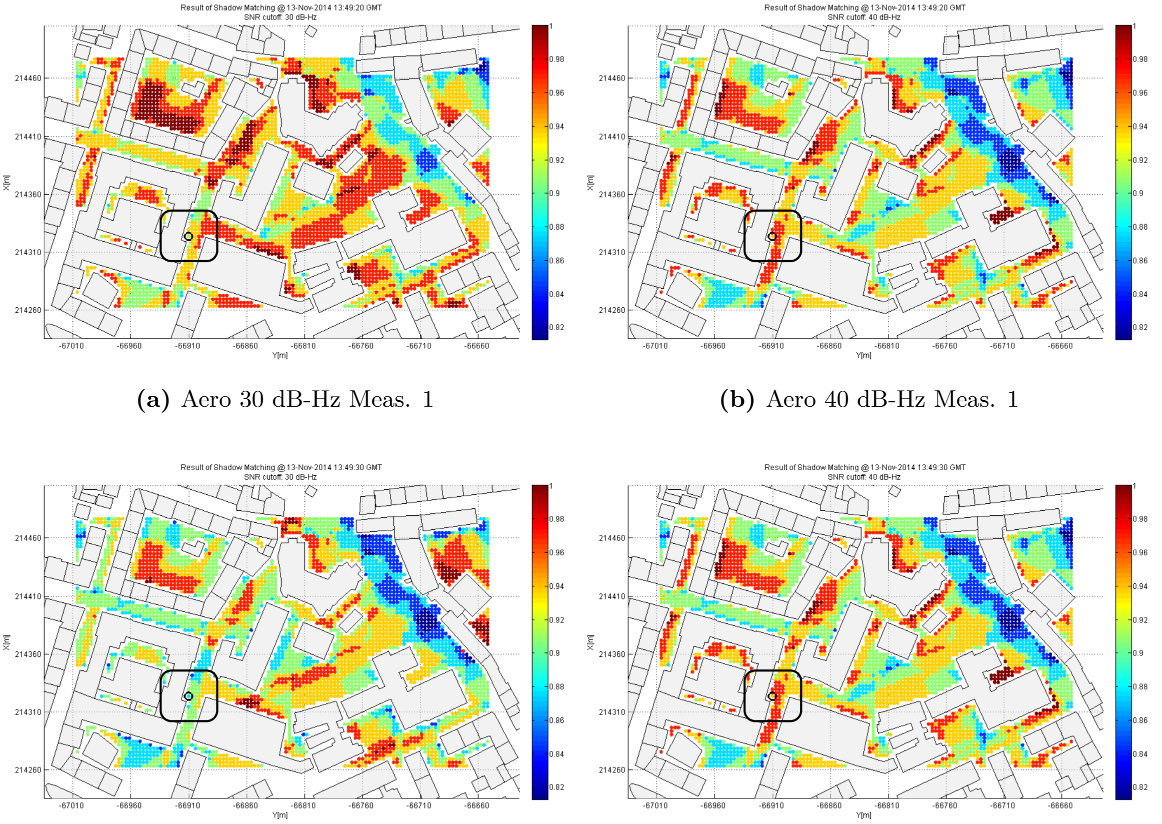
Table 4.1.: SNR measurements of an Aero antenna at location 1

				SNR Measuremets [dB-Hz]							
Meas.	GMT	Cut-off [dB-Hz]	PRN:	1	4	11	14	22	23	31	32
1	13:49:20	0		34	50	41	46	27	19	25	50
		30		34	50	41	46				50
		40			50	41	46				50
2	13:49:30	0		35	49	40	44	30	23	27	49
		30		35	49	40	44	30			49
		40			49	40	44				49

In figure 4.5 on page 86, the impact of the two cut-off values of 30 and 40 dB-Hz on the location probability can be seen. Locations of high user position probability are coloured red, and locations of low probability are coloured blue. If comparing the first measurement epoch (first line of figures), then a higher probability is reached at the users true location indicated by a black circle with a 40 dB-Hz cut-off (right figure). The direct neighbours of the user's true location have higher probabilities in figures 4.5b and 4.5d (40 dB-Hz), which produces a better result in general.

Conclusion The comparison of a cut-off of 30 and 40 dB-Hz yields following results:

- A 40 dB-Hz barrier works better for the Aero antenna, because the probability of correct location determination increases.
- Satellites near building borders, which are indicated to be invisible by the Sky Plot model, were measured with high SNR values. Thus, a false detection of possible user locations is possible. Reasons for this phenomena are discussed later in this chapter (multipath effects or other influences).
- Higher cut-off values are problematic, since LoS satellites can have lower SNR than NLoS satellites.


Result of Shadow Matching @ 13-Nov-2014 13:49:20 GMT
SNR cutoff: 30 dB-Hz
Result of Shadow Matching @ 13-Nov-2014 13:49:20 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 13-Nov-2014 13:49:30 GMT
SNR cutoff: 30 dB-Hz
Result of Shadow Matching @ 13-Nov-2014 13:49:30 GMT
SNR cutoff: 40 dB-Hz
X[m]
214460
214410
214360
214310
214260
Y[m]
-67010
-66960
-66910
-66860
-66810
-66760
-66710
-66660
1
0.98
0.96
0.94
0.92
0.9
0.88
0.86
0.84
0.82

(a) Aero 30 dB-Hz Meas. 1

(b) Aero 40 dB-Hz Meas. 1

(c) Aero 30 dB-Hz Meas. 2

(d) Aero 40 dB-Hz Meas. 2

Figure 4.5.: Fingerprints of an Aero antenna at location 1

Patch GPS Antenna

The patch antenna (see figure 4.6) used in this section is a u-blox ANN-MS antenna which is delivered with the u-blox 6 receiver. The measurement location is the same as the one for the Aero antenna in figure 4.2b.

Figure 4.6.: Patch antenna

Source: `http://www.u-blox.com/images/stories/Products/ANN-MS/antenna_web.jpg`

Location 1 The SNR measurements taken by the patch antenna are visible in figure 4.7. The cut-off value has been set to 40 dB-Hz, indicated by the dashed line. The reason for choosing this value, is discussed later in this section. In table 4.2 different cut-off values and their results are displayed. PRNs which are predicted to be visible by the model are coloured green. With a cut-off of 40 dB-Hz, the detection of visible satellites for location 1 is better, than with the Aero antenna. This can be explained by less sensitive SNR measurements in general, due to better mitigation of multipath signals (at this location). In this example, the measurement of satellite 11 is below the cut-off, because the GPS antenna is less sensitive and a false detection of a LoS to satellite 11 can be prevented. Table 4.2 indicates that only one of three satellites (PRN 14) is wrongly detected to be visible by the measurement, in contrast to two of four measurements by the Aero antenna.

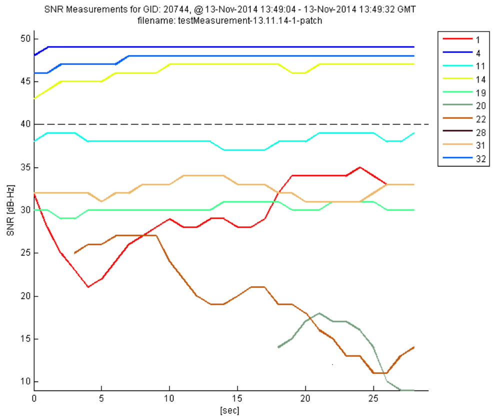
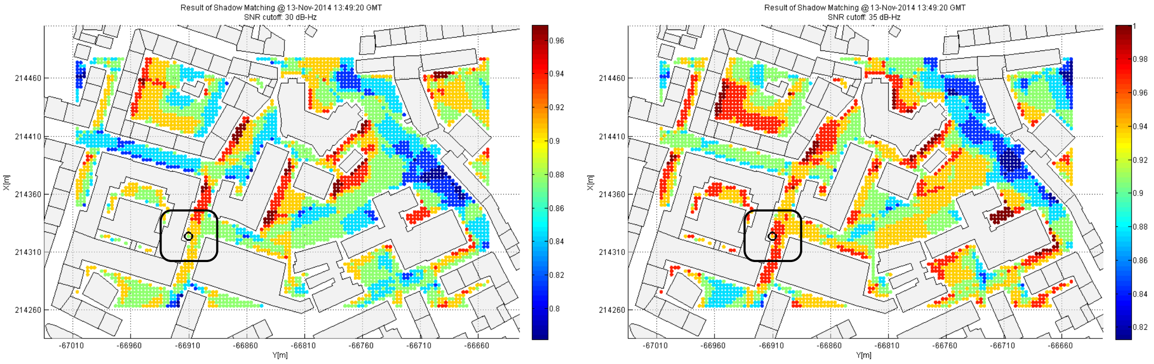


Figure 4.7.: SNR measurement of patch antenna at location 1

Figures 4.8 and 4.9 show the comparison of location probabilities with a cut-off of 30, 35 and 40 dB-Hz. In figure 4.8a the location probability at the user position (black circle) and the surrounding neighbours, is significantly lower than in figure 4.8b. The best results can be reached with a 40 dB-Hz cut-off in figure 4.9b. They indicate a significantly high probability at the user's position. Therefore, a cut-off of 40 dB-Hz is chosen for the patch antenna.

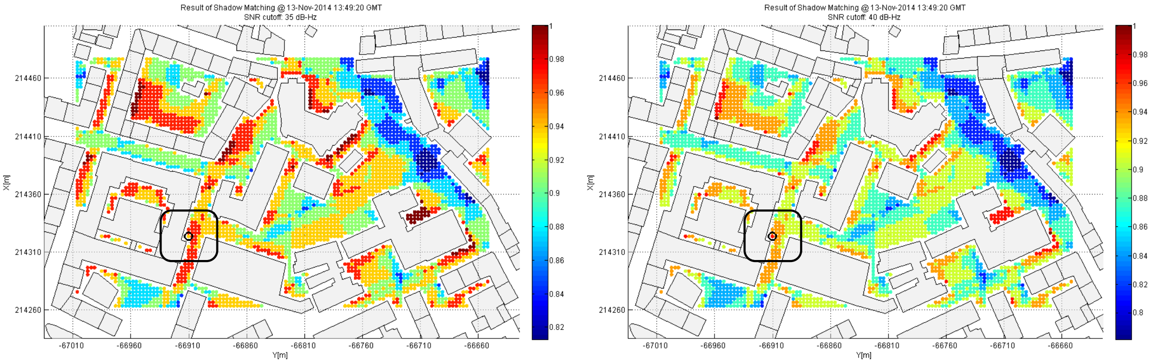
Table 4.2.: SNR Measurements of the patch antenna at location 1

		Cut-off		SNR Measuremets [dB-Hz]							
Meas.	GMT	[dB-Hz]	PRN:	1	4	11	14	19	22	31	32
2	13:49:20	0		29	49	38	47	30	24	33	48
		30			49	38	47	30		33	48
		35			49	38	47				48
		40			49		47				48



(a) Patch 30 dB-Hz Meas. 2 **(b)** Patch 35 dB-Hz Meas. 2

Figure 4.8.: Fingerprints of a patch antenna at location 1 - 1



(a) Patch 35 dB-Hz Meas. 2 **(b)** Patch 40 dB-Hz Meas. 2

Figure 4.9.: Fingerprints of a patch antenna at location 1 - 2

Conclusion Similar to the Aero antenna, several influences aggravate the detection of LoS and NLoS signals (multipath, building borders, etc.). The patch antenna behaves slightly different to the high sensitivity antenna, which leads to following conclusions:

- The concept of setting a lower cut-off value when using a less sensitive antenna, proofs to be false.
- A cut-off value of 40 dB-Hz is satisfying for the patch antenna.

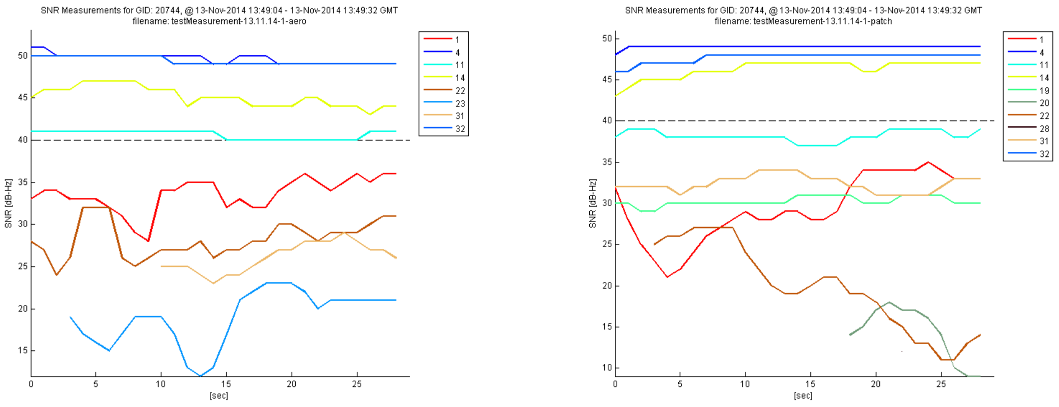
4.1.3. Comparison of antennas

This section shows the differences between the two antennas regarding SNR measurements. As can be shown, the differences are significant, and influence the results at a large scale. Several parameters affect the quality of fingerprints, beginning from selecting an appropriate SNR cut-off (threshold) for the antennas, to the correlation of antenna performance to a location. Synchronous measurements with both antennas were performed at several locations. Some of these measurements were chosen, to exemplarily show the performance of the antennas.

Problematic cut-off selection

Within the previous investigations in chapter 4.1.2, the SNR cut-off for each antenna has been determined, but without comparison of the antennas. Figures 4.10a and 4.10b display the measurements of both antennas at location 1. When comparing the sky plot (see figure 4.3 on page 84) to the SNR time series, only one false PRN is in the SNR range of visible satellites (PRN 14) with the patch antenna, compared to two false PRN measurements with the Aero antenna (PRN 11 and 14). Additional satellites in the measurements are multipath signals, and signals, which bend around building corners. The cut-off for the patch antenna dismisses the occurring multipath/bending signal of PRN 11 correctly. The patch antenna with a cut-off of 40 dB-Hz yields the best result (see the dark red dot indicating the highest probability at the users position in figure 4.11b).

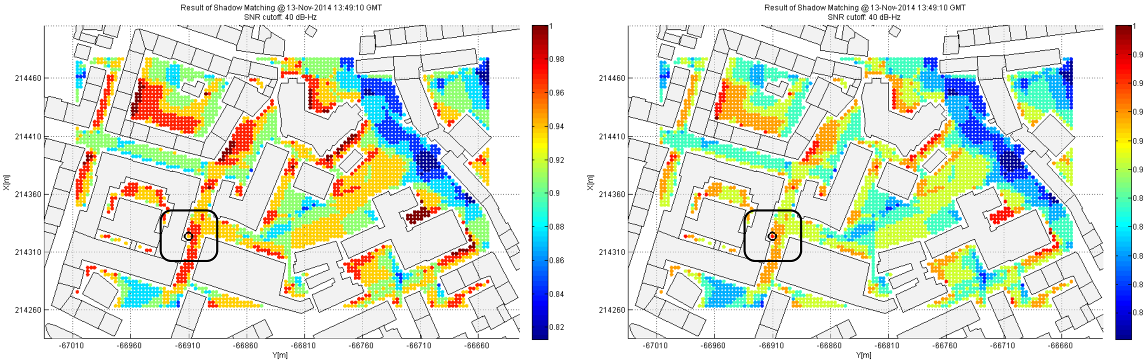
The selection of a correct cut-off proves to be difficult, considering no difference between multipath/bending and direct LoS signals can be made from measured SNR values.



(a) Aero SNR measurement

(b) Patch SNR measurement

Figure 4.10.: SNR measurements of Aero/Patch antenna at location 1



(a) Aero 40 dB-Hz Meas. 1

(b) Patch 40 dB-Hz Meas. 1

Figure 4.11.: Fingerprints Aero-Patch at location 1, 40 dB-Hz cut-off

Significant time variation of SNR measurements

Investigations in several distinct locations revealed significant time variations of measurements over a time span of a few seconds. A certain fall off can be expected from satellites that descent and are blocked behind buildings. However, this is a slow process and cannot be observed within several seconds, as the SNR measurements are not sensitive enough. Most likely those higher frequency fluctuations originate in passers-by, that block the signal for a short amount of time (when the signal decreases), or are caused by small changes of the alignment of the antenna (and the user holding it, blocking signals), or by changes of the position of objects reflecting the satellite signal. The user's heading and reflecting objects can be regarded as measurement set-up, which depends on the location.

Figure 4.13 shows time variations, particularly in the patch antenna measurements. This time variation of measurements introduce significant fluctuation of fingerprint results over only a few seconds (see figure 4.15 on page 93). Table 4.3 displays the satellite PRNs, derived from the measurements (Aero and patch antennas). Additionally, it highlights PRNs, that are visible by the model and the probability for a user standing at location 2. Further study of figure 4.12 and table 4.3 reveals, that the overall probability of correct PRN detection for the patch antenna is lower, than for the Aero antenna. The Aero antenna holds more separable measurements of invisible and visible satellites (compare figures 4.13a and 4.13b).

Table 4.3.: Visible satellites of measurements and model at location 2

Meas.	GMT	Aero	Prob.	Patch	Prob.	Model
1	14:40:40	1, 4, 11, 14, 20, 32	96.88 %	4, 11, 14, 20, 32	93.75 %	1, 4, 11, 14, 20, 23, 32
2	14:40:50	1, 4, 11, 14, 20, 32	96.88 %	1, 4, 11, 20, 23, 32	96.88 %	1, 4, 11, 14, 20, 23, 32
3	14:50:00	1, 4, 11, 14, 20, 23, 32	100.00 %	1, 4, 11, 14, 20, 32	96.88 %	1, 4, 11, 14, 20, 23, 32

Using only one epoch of SNR measurements for location determination is problematic, as it is unaware of sudden jumps in the measurements and therefore errors cannot be detected.

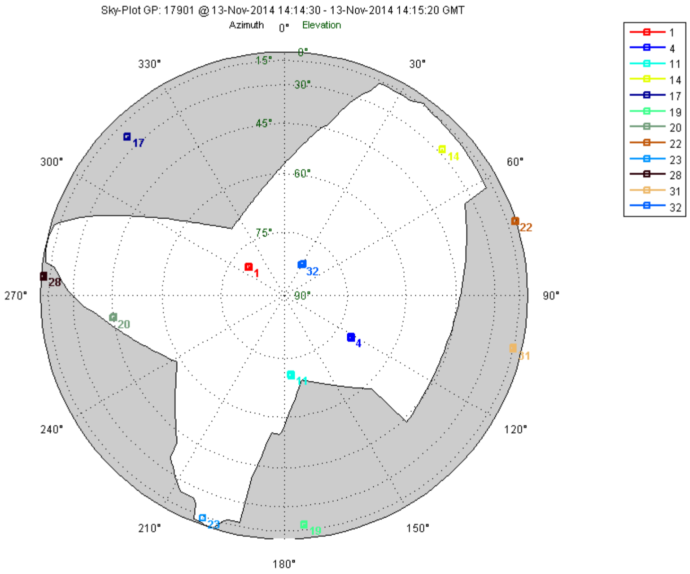
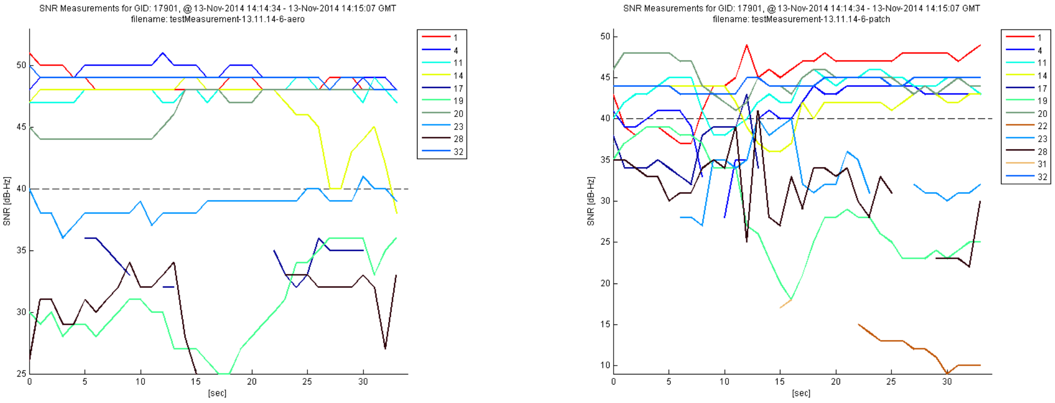


Figure 4.12.: Sky Plot at location 2



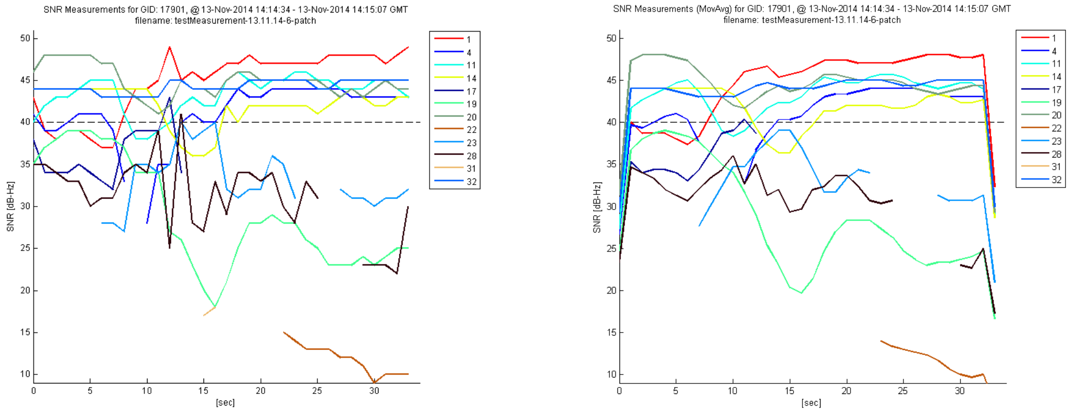
(a) Aero SNR measurement

(b) Patch SNR measurement

Figure 4.13.: SNR measurements of patch/aero antenna at location 2

The solution could be a filtering of measurements.

Two low order lowpass IIR (infinite impulse response) filters were tested (Butterworth and elliptic), with a passband corner frequency of 0.4 Hz and a stopband corner frequency of 0.6 Hz. However, these filters need a certain time to work correctly and are not suitable for a small amount of measurements. A moving average filter yields better results, but the measurements comprise data gaps. The moving average filter's outcome is truncated at the beginning and at the end of the time series by n data samples, where n is the order of the filter divided by 2. The filter truncates the outcome, as it needs this information to compute the average values for direct neighbours. Figure 4.14 shows the patch antenna's measurements unfiltered and filtered. Short jumps can be filtered successfully (PRN 28), but on the other hand, drops and rises of several seconds cannot be mitigated (e.g. PRN 14). Further research in this scope must be made to determine the improvement of filtered over unfiltered data. Another approach would be a filtering with a Kalman filter when measuring (in real-time), where parameters for past measurements are estimated to obtain a smoothed representation of the time series. Only past measurements would then be used for filtering.



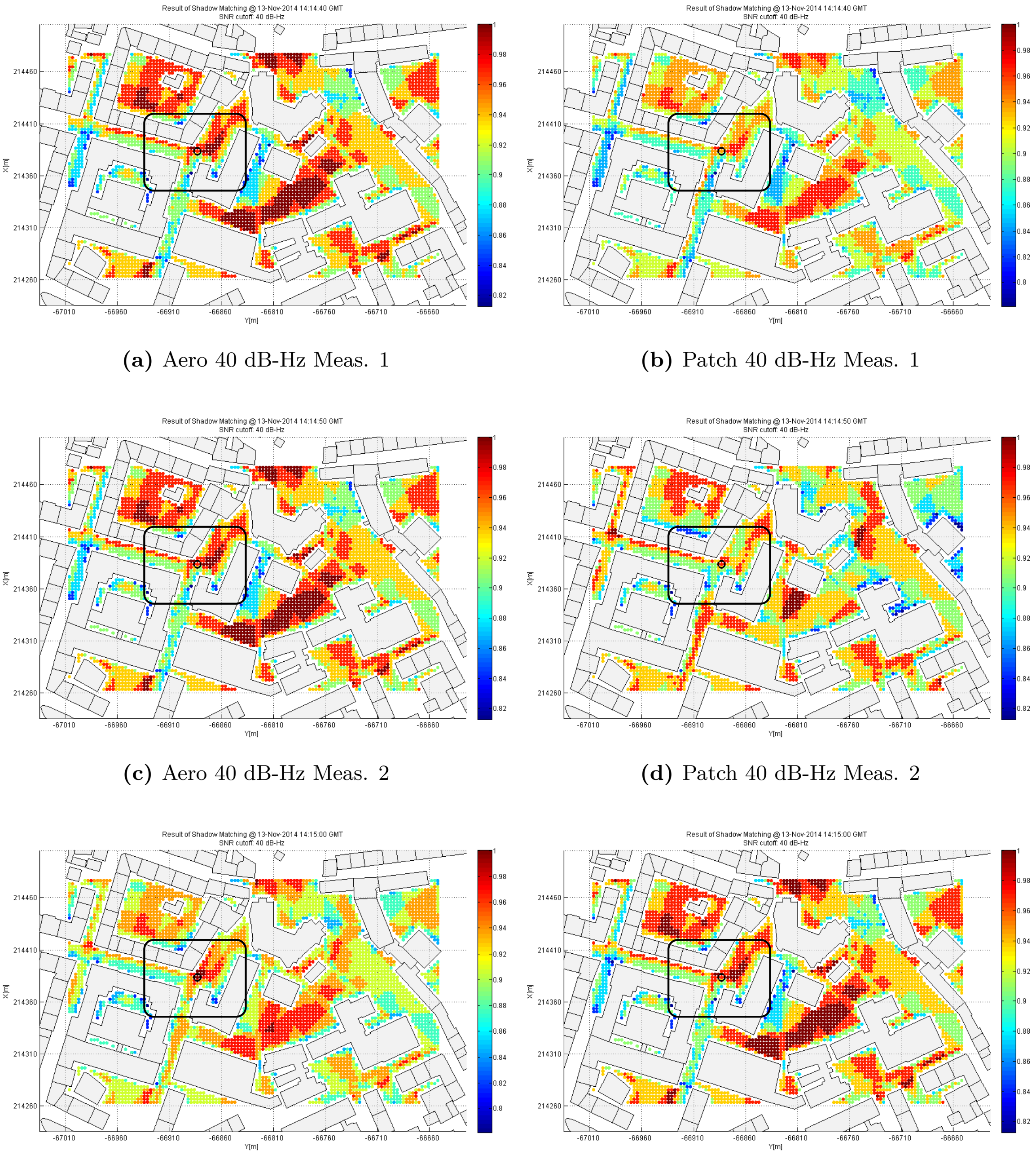
(a) SNR measurements

(b) SNR measurement with moving average filter

Figure 4.14.: SNR measurements of patch antenna and moving average

Correlation of the antenna performance to locations

The antenna performance indicates a large correlation of the results with the location, at which the measurements have been taken. In some locations, the sensitive Aero antenna performs better, in others the low-sensitive patch antenna outperforms the Aero antenna. Also the time variance with the patch antenna is higher, as pointed out in the previous paragraph. Figure 4.16a on page 95 shows the sky plot at a east-west directed street. Thus, a better location performance in the north-south (across street direction) is expected. The fingerprints in figure 4.16 on page 95 support this assumption, as the correct side of the street is detected. A noticeable exception is measurement 3 of the patch antenna (figure 4.16e) in



(a) Aero 40 dB-Hz Meas. 1

(b) Patch 40 dB-Hz Meas. 1

(c) Aero 40 dB-Hz Meas. 2

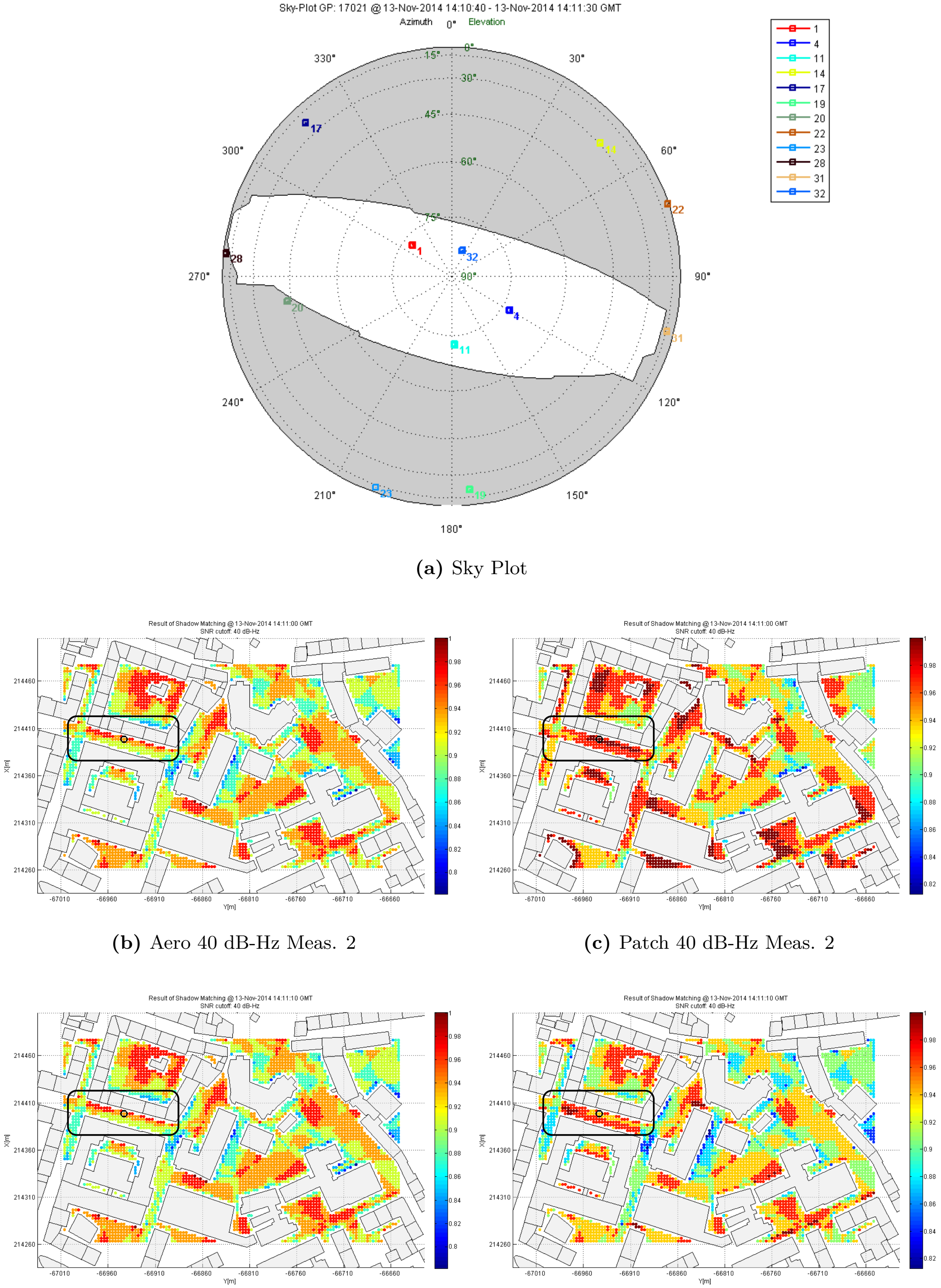
(d) Patch 40 dB-Hz Meas. 2

(e) Aero 40 dB-Hz Meas. 3

(f) Patch 40 dB-Hz Meas. 3

Figure 4.15.: Time variation of fingerprints of Aero-Patch at location 2

contrast to a good prediction of the correct location in measurement 2 (figure 4.16c). On the other hand, the patch antenna outperformed the Aero antenna at location 1 (see figure 4.11 on page 90). Thus, assessing a reliable statement on which antenna works better is hard at this stage, especially when regarding the measurement set-up mentioned on page 90.



(a) Sky Plot

(b) Aero 40 dB-Hz Meas. 2

(c) Patch 40 dB-Hz Meas. 2

(d) Aero 40 dB-Hz Meas. 3

(e) Patch 40 dB-Hz Meas. 3

Figure 4.16.: Sky Plot and Fingerprints Aero-Patch with 40 dB-Hz cut-off at location 3

Erroneous Location Detection

Strong multipath, especially in inner courtyards, makes fingerprint location detection erroneous. Figure 4.19 displays the original fingerprint in an inner courtyard. The SNR of multipath signals from PRN 1 and 4 were higher than that of the direct LoS signals from satellites 11 and 32 (see the sky plot and SNR measurements below).

To investigate the effect of multipaths, the measurements were manipulated measuring only LoS singals from satellites 11 and 32, as indicated by the sky plot in figure 4.17. The corresponding fingerprint is visible in figure 4.20. The location probability at the real user location improved significantly, showing that multipath is evident. Comparing the performance of the two antennas regarding multipath mitigation, the Aero antenna should behave better, because a build in ground plate rejects multipath signals reflected from the ground. From figure 4.18 however, it can clearly be seen that the SNR of multipath signals of satellite 1 is significantly higher than that of LoS signals of satellite 11. As signals are not only reflected by the ground, this signals cannot be mitigated by a ground plate and thus lead to strong signals. A solution would be an implementation of a multipath model, but this is beyond the scope of this thesis.

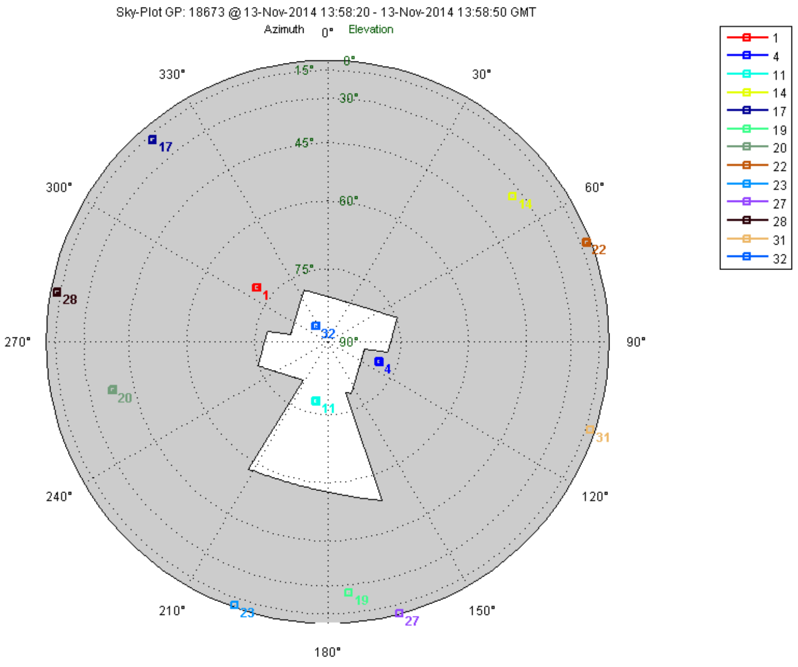
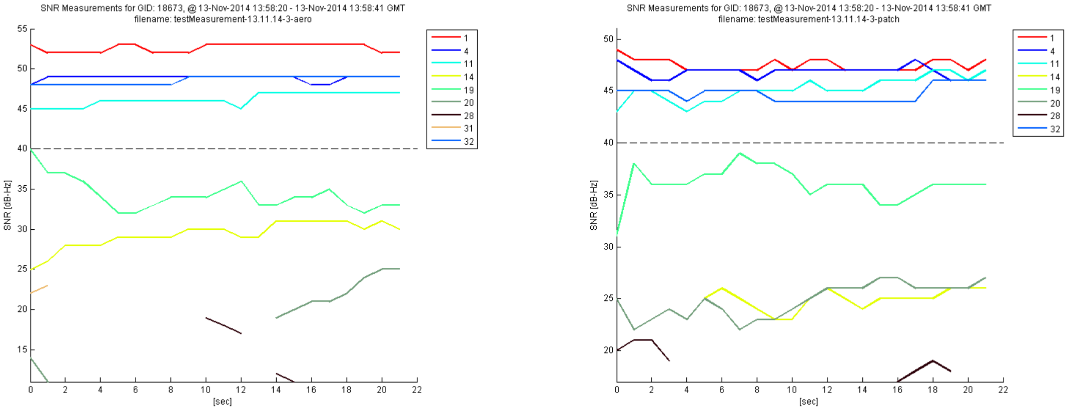
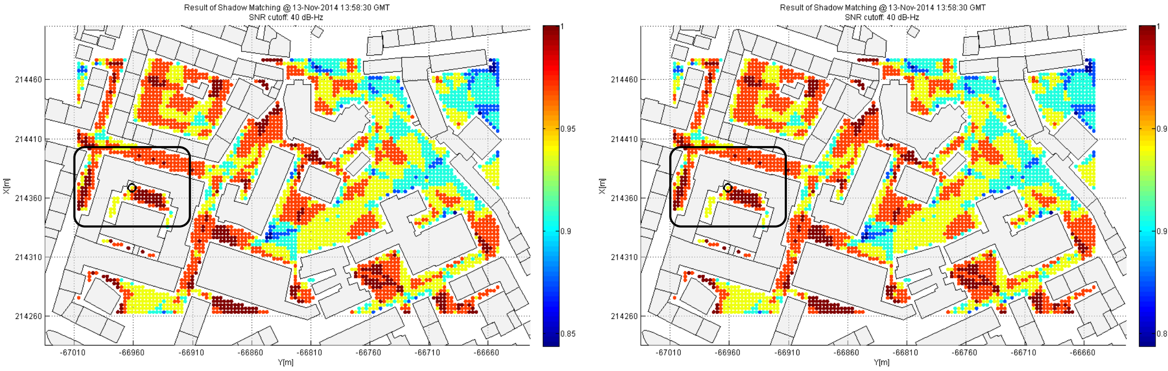


Figure 4.17.: Sky Plot at location 4



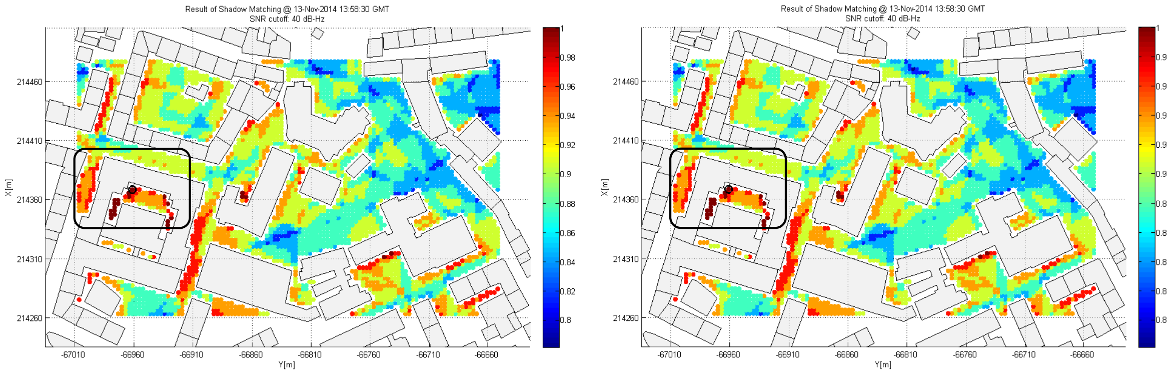
(a) Aero SNR measurement **(b)** Patch SNR measurement

Figure 4.18.: SNR measurements of Aero/Patch antenna at location 3


Result of Shadow Matching @ 13-Nov-2014 13:58:30 GMT
SNR cutoff: 40 dB-Hz
X[m]
Y[m]
Result of Shadow Matching @ 13-Nov-2014 13:58:30 GMT
SNR cutoff: 40 dB-Hz
X[m]
Y[m]

(a) Aero 40 dB-Hz Meas. 2 **(b)** Patch 40 dB-Hz Meas. 2

Figure 4.19.: Fingerprints Aero-Patch at location 4 with 40 dB-Hz cut-off


Result of Shadow Matching @ 13-Nov-2014 13:58:30 GMT
SNR cutoff: 40 dB-Hz
X[m]
Y[m]
Result of Shadow Matching @ 13-Nov-2014 13:58:30 GMT
SNR cutoff: 40 dB-Hz
X[m]
Y[m]

(a) Aero 40 dB-Hz Meas. 2 **(b)** Patch 40 dB-Hz Meas. 2

Figure 4.20.: Fingerprints Aero-Patch at location 4 with 40 dB-Hz cut-off manipulated

4.1.4. General SNR Conclusion

By comparing the investigation results of both antennas, following observations were made:

- The cut-off detection is difficult, due to unknown multipath and bending of signals.
- Time variations in SNR measurements may distort fingerprint results if not filtered or averaged over a time span; further investigations must be made to determine the improvement of filtered over unfiltered data.
- The antenna performance is heavily correlated to its location and measurement set-up (alignment of the user, objects which reflect signals that lead to multipath, etc.). Therefore, an overall conclusion on which antenna works best is difficult to obtain.
- Multipath is evident, especially in inner courtyards, and leads to grave positioning errors. A multipath model would address this issue.

The contradiction between model and measurements can further be explained by the following:

- The 3D model, on which the Sky Plot approach is based on, is not detailed enough. For example, the abstraction of detailed rooftops by cubes is error prone and therefore rooftops block satellite signals in the model, which are visible in reality.
- The precision of the initial data (DEM and building groundplans), on which the 3D model is based on.
- Bending of satellite signals around buildings is not modelled.

4.1.5. Reducing Measurement Errors

Since the negative effects on SNR measurements were discussed in the section before, the reduction of measurement errors is the topic of this chapter. Especially two possible solutions for better fingerprint results are discussed.

Multipath modelling

When measured signals which are clearly caused by multipath, can be mitigated, the fingerprint results improve drastically (see section 4.1.3). Studies in this area [14] show, that it is possible to detect multipath signals by using a 3D city model and a ray-tracing algorithm. Using an approximate user position, LoS signals can be distinguished from multipath and thus NLoS signals. Other studies [2] use ray-tracing algorithms to determine the signal scattering and delay introduced in urban environments.

Introduction of a mask angle for 3D models

To address the bending of signals around building borders, as well as the inaccuracy of the 3D models, [10] suggests to implement a 3° diffraction zone for each building. Signals passing this zone have a lower weight, than signals with a direct LoS. The weighting is implemented by a scoring theme, displayed in table 4.4. The diffraction zone is calculated by reducing the elevation and azimuth of each building seen from a users perspective by 3°. Only buildings in direct neighbourhood of a grid point were included for building border computation (for example a circumcircle of 300 m around a grid point).

Table 4.4.: Scoring scheme for diffraction zones of buildings

		Prediction		
		Invisible	Diffracted	Visible
Observation	Not Tracked	1	1	-1
	Weak Signal	0	2	0
	Strong Signal	-1	1	1

To determine the error induced by the decrease of building dimensions, an error analysis was conducted. Two different approaches were tested for their applicability:

- Reducing the height of each building by $h_b = h_b^{\text{original}} - 1$ m to compromise the errors of signal bending and non-detailed roof modelling
- Introduction of an elevation mask angle of 2° for each building

The two methods are explained in more detail below. In figure 4.21, a layout of a grid point and the distance to the surrounding buildings b_1 and b_2 is given. The point is located in an inner court-yard, hence most satellites are invisible to the user. The distances and heights of the grid point and buildings for error estimation are listed in table 4.5.

Table 4.5.: Height of grid point and distance to buildings

	Absolute Height	Relative Height	Distance [m]
	b_{hi} [m]	$b_{hi} - gp_h$[m]	d_i
gp	353.40	-	-
b_1	388.56	35.16	4.60
b_2	376.80	23.40	13.60

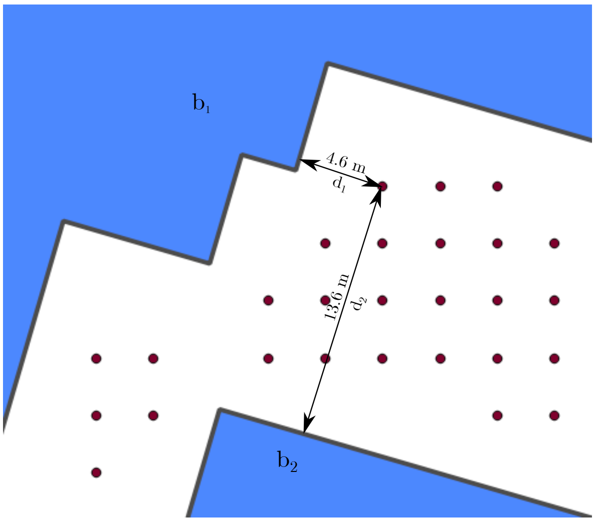


Figure 4.21.: Overview of a grid point and surrounding buildings

Decrease of building heights The effect of a reduction of building heights on the elevation angle seen from a grid point, can be conducted by trigonometric functions. The arctangent from the relative building height and the distance to the building returns the elevation angle (see figure 4.22). The tangent approaches the pole for high elevations (eq. 4.3), thus the elevation angle of a high building near a grid point is effected only slightly by a building height reduction of 1 m (eq. 4.2). Table 4.6 gives the results of error estimation for building 1 and 2. The effect on the elevation is too small for buildings in the vicinity of the grid point, thus would not change the outcome of fingerprints (compare figure 4.23 with a mask angle of 2°). For buildings further away, the impact on the elevation angle would decrease further.

$$\varepsilon_i = \arctan \frac{b_{hi} - gp_h}{d_i} \tag{4.1}$$

$$\varepsilon_{i,\text{small}} = \arctan \frac{b_{hi} - gp_h - 1\text{ m}}{d_i} \tag{4.2}$$

$$\lim_{\varepsilon \to 90^\circ} \tan \varepsilon = \pm\infty, \qquad \lim_{\varepsilon \to 0^\circ} \tan \varepsilon = 0 \tag{4.3}$$

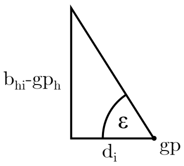


Figure 4.22.: Gridpoint and building front view

Table 4.6.: Estimation of errors introduced by a 1 m height reduction

	Original Value	Unit	$\varepsilon_{i,\text{small}}$ (b_h - 1 m)	Difference
ε_1	82.55	°	82.33	0.22
ε_2	59.84	°	58.74	1.10

Decrease of building boundaries elevation The second approach is a decrease of elevation angles to building boundaries, as proposed in [10]. When decreasing the elevation angle, the corresponding building height will be decreased, depending on the distance to the grid point (eq. 4.5). Therefore, high buildings near the observation point and buildings remote to the grid point of interest will be effected drastically by such an approach. Table 4.7 gives an overview of the estimated errors induced. For building 1 in direct neighbourhood of the grid point and a height of ~35 m, the building would be decreased by 7.5 m, an effect, which makes an appropriate 3D model unnecessary. The same error would apply for an azimuth mask angle to decrease the width of buildings.

$$b_{hi} = d_i \tan \varepsilon \tag{4.4}$$

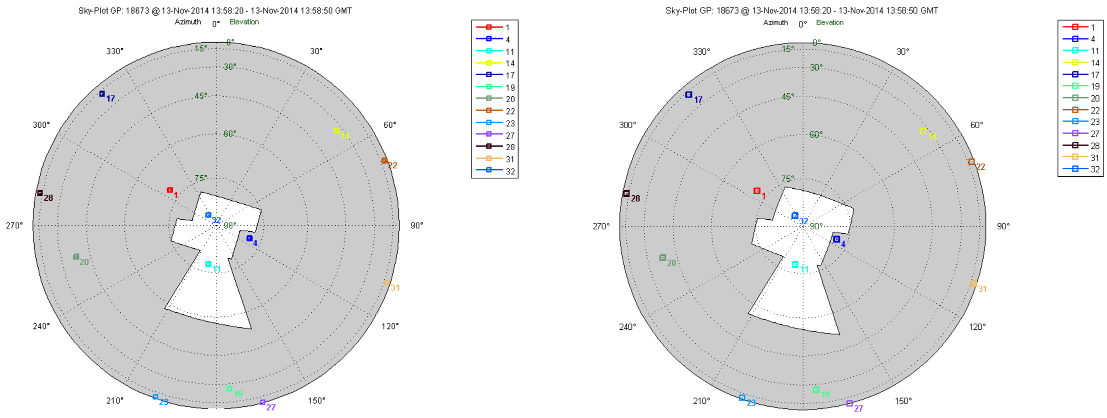
$$b_{hi,\text{small}} = d_i \tan(\varepsilon - 2^\circ) \tag{4.5}$$

Table 4.7.: Estimation of errors introduced by a 2° mask

	Original Value	Unit	$b_{hi,\text{small}}$ (ε-2°)	Difference
b_{h1}	35.16	m	27.63	7.53
b_{h2}	23.40	m	21.63	1.77

Conclusion Above investigation show, that a decrease of building heights would have no impact on the elevation angle and result of fingerprinting. Therefore, this discards the approach, although the errors induced to the heights of the model are small ($h_b = h_b^{\text{original}} - 1$ m). Figure 4.23 shows an original sky plot on the left hand side, and a sky plot with a 2° mask angle. Satellite 4 may be inappropriately tagged as invisible because of the inaccuracy of the 3D model. However, a 2° larger elevation angle would not change the visibility to the satellite from this grid point. Hence, a larger decrease of heights would have to be applied to the model, inducing grave errors. On the other hand, a mask angle of 2° would lead to grave errors in building heights.

Both above investigated methods are therefore not applicable to increase the detection rate of fingerprinting and negating signal bending around building borders and inaccurate 3D models.



(a) Sky Plot

(b) Sky Plot with 2° elevation mask angle

Figure 4.23.: Comparison of sky plots with/without mask angle at location 4

4.2. Fingerprints

This chapter deals with the results of GPS fingerprinting. Test measurements were carried out on the campus of the Institute of Navigation at the Graz University of Technology. To simulate a trajectory, way points were sequentially measured along a predefined route (see chapter 4.2.1). Therefore, SNR and GPS code measurements were taken at the same time, in order to verify and compare the positions. A comparison between the positioning based on GPS fingerprinting and GPS code-based single point positioning was conducted in chapter 4.2.2.

4.2.1. Trajectory

As a trajectory, a collection of way points was specified before taking measurements, to illustrate the detection of sides of the street, and the alteration of location probabilities depending on the visibility to satellites in certain directions.

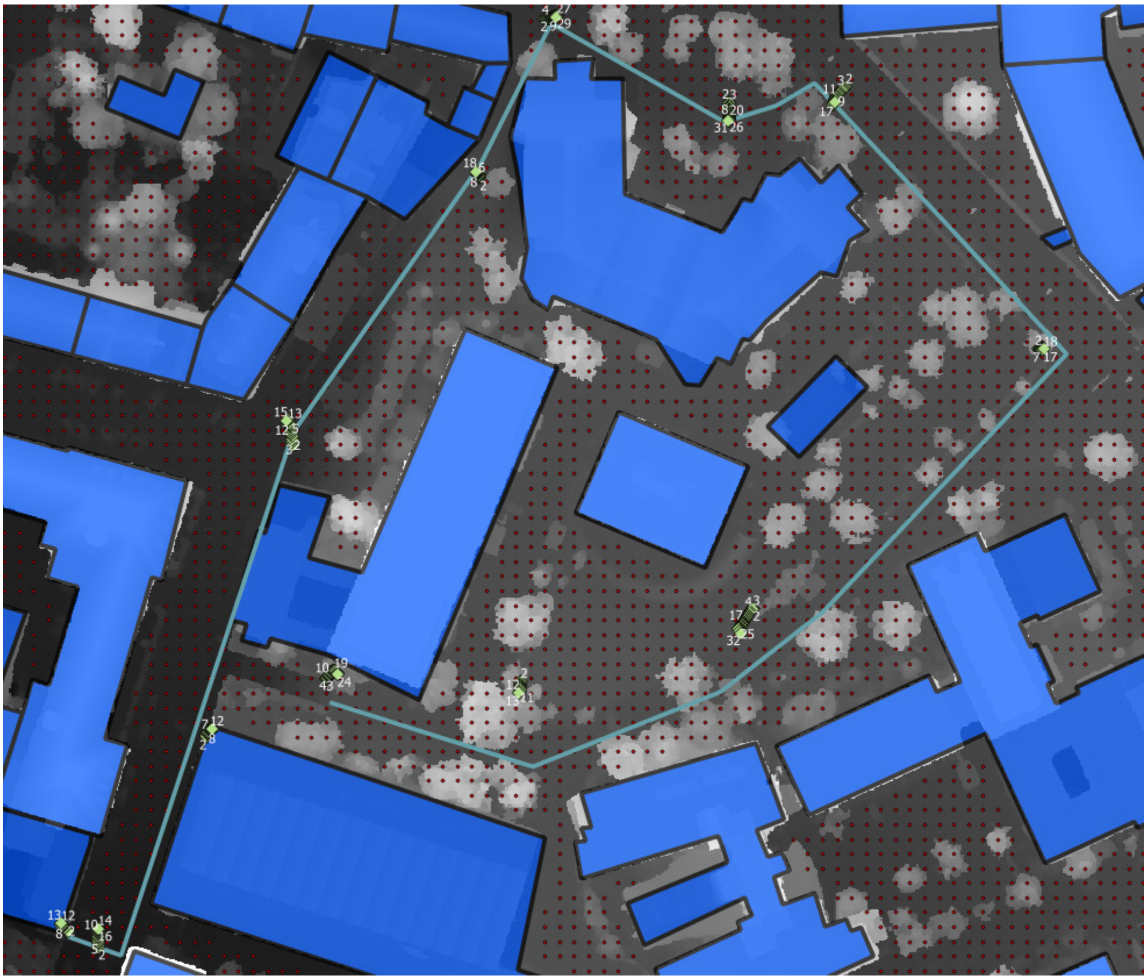


Figure 4.24.: Overview of the measured trajectory

Figure 4.24 displays the predefined trajectory and the corresponding GPS code solutions at several way-points. The trajectory leads through a street canyon (south-south-west to north-north-east oriented) and then back in a loop on a tree-lined street and parking area. Deviations of the GPS-based position measurements to the trajectory are visible. In the provided figure, which displays the DEM and shape-file (buildings) atop, you can see trees, which were not modelled in the 3D city model. This leads to additional loss of visibility in reality, which makes the prediction of satellite visibility inaccurate, since no vegetation is considered. Since the measurements took place in December, the shading caused by vegetation is reduced. In order to develop an accurate positioning using GPS fingerprinting, it is necessary to take the vegetation into account.

Figures 4.25 and 4.26 on the pages 106 to 107 show the results of GPS fingerprinting. The grid points are coloured corresponding to their location probability from blue (low probability) to red (high probability). All twelve way points are shown sequentially. The travelled trajectory is illustrated as black line. Furthermore, previous way points are shown as white dots and black circles indicate the current true location of the user.

In the following, the way of the trajectory and the results of fingerprinting for the twelve way points are discussed. Each letter corresponds to a fingerprinting result of a way point along the trajectory shown in the figures 4.25 and 4.26:

a) The beginning of the trajectory was chosen at the vertex of the chemistry building. The probability of the true grid point (black circle) is 93%, but to the south-west, points with higher probabilities are located. However, the correct side of street is detected.

b) The probabilities in north-south direction of the true position are low. Therefore, an east-west variation exists and the correct side of the street is not clearly identified.

c) - d) At the corner of the staircase, the grid points hold high probabilities. Along the street, the correct side is detected (see the previous way point of the trajectory). Figure 4.25d displays a larger area with identical probabilities. A more accurate position allocation is therefore difficult.

e) The possibility of detection of this point is low, but the adjacent points are arranged in a funnel, which opens to the south (satellites are most likely visible to the south). By implementing an algorithm, that involves not only the highest probability, but also the number of points of equal probability for user position detection, this grid point might be detected correctly. However, this might not be the case for all locations. Further investigation concerning algorithms for the determination of an absolute position within a search area have to be conducted.

f) The change of probabilities is clearly visible from figure 4.25e to figure 4.25f. After the narrow passage from point e to f, the visibility to the south is severely limited due to the tall building. The model and the measurements match at the location, where the measurements took place, and a high probability is evaluated. Since a large contiguous area with high probability exists at this location, an implementation of higher frequency measurements and a time-based filter would be appropriate (for instance a Kalman filter). By considering previous closely-spaced measurements (higher measurement frequency) and a dynamic model introduced by a Kalman filter, some of the points of high probability could be excluded and the final result would be more precise.

g) - h) on page 107: the cloud of points with high probabilities coincide with the trajectory.

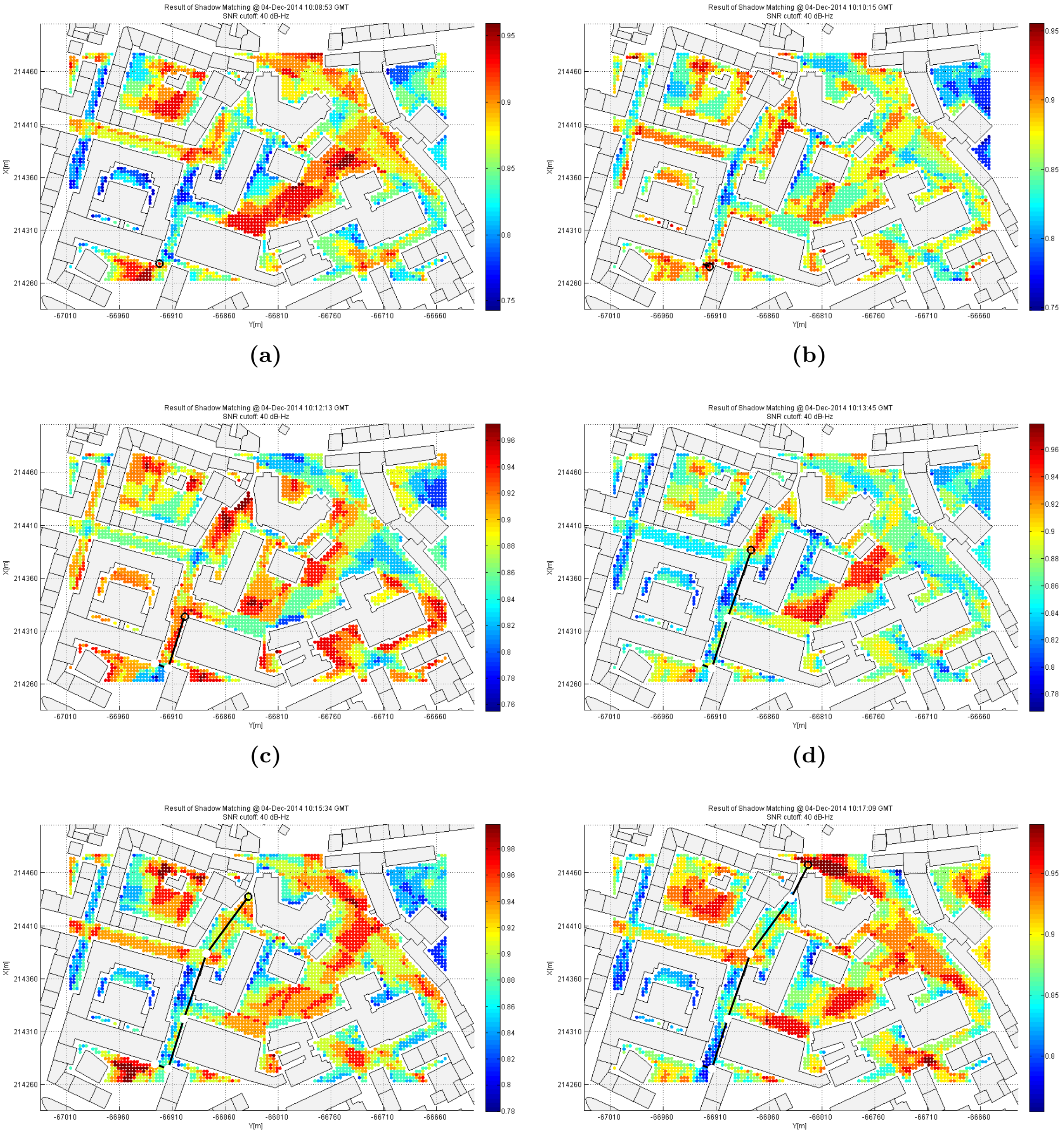
i) Again, only low probabilities at the true grid point can be achieved. This is mainly a result of the building to the south of the grid point, which was modelled wrongly (see section 3.2.2 on page 31). In the model, the building is up to 17 m taller than in reality. Thus, the matching between SNR measurements (visible satellites) and the model yields erroneous location probabilities. The large number of trees in this area restrict the field of vision additionally.

j) Despite the influence of the incorrect building model (see point above), the detection of the correct point in the grid is appropriate. Compared to the GPS measurements (see section 4.2.2 and figure 4.27a on page 109), which have a large offset to the true location, the correct position is detected with fingerprinting.

k) - l) The area of the cloud of high probabilities decreases from point k to l corresponding to the true position.

Conclusion From the investigations of the trajectory, following insights were made:

- The modelling of vegetation is crucial as it has a great impact on the visibility to satellites. However, this is subject to seasons.
- In street canyons, and areas with high shading, the correct side of the road is detected.
- Adjacent areas with a large number of high probabilities lead to ambiguity in positioning with GNSS fingerprinting.
- After the change of direction at edges of tall buildings, detections are particularly good. Along roads similar probabilities occur, due to the similarity of building models and therefore similar visibilities.
- The involvement of a time-based filter would lead to better results. The filter could conclude the true position of the user from the history of results and a dynamic model.
- Errors in the building model have a very negative impact on fingerprinting.
- The cloud of high probabilities moves in sequential order with the true user position of the trajectory.

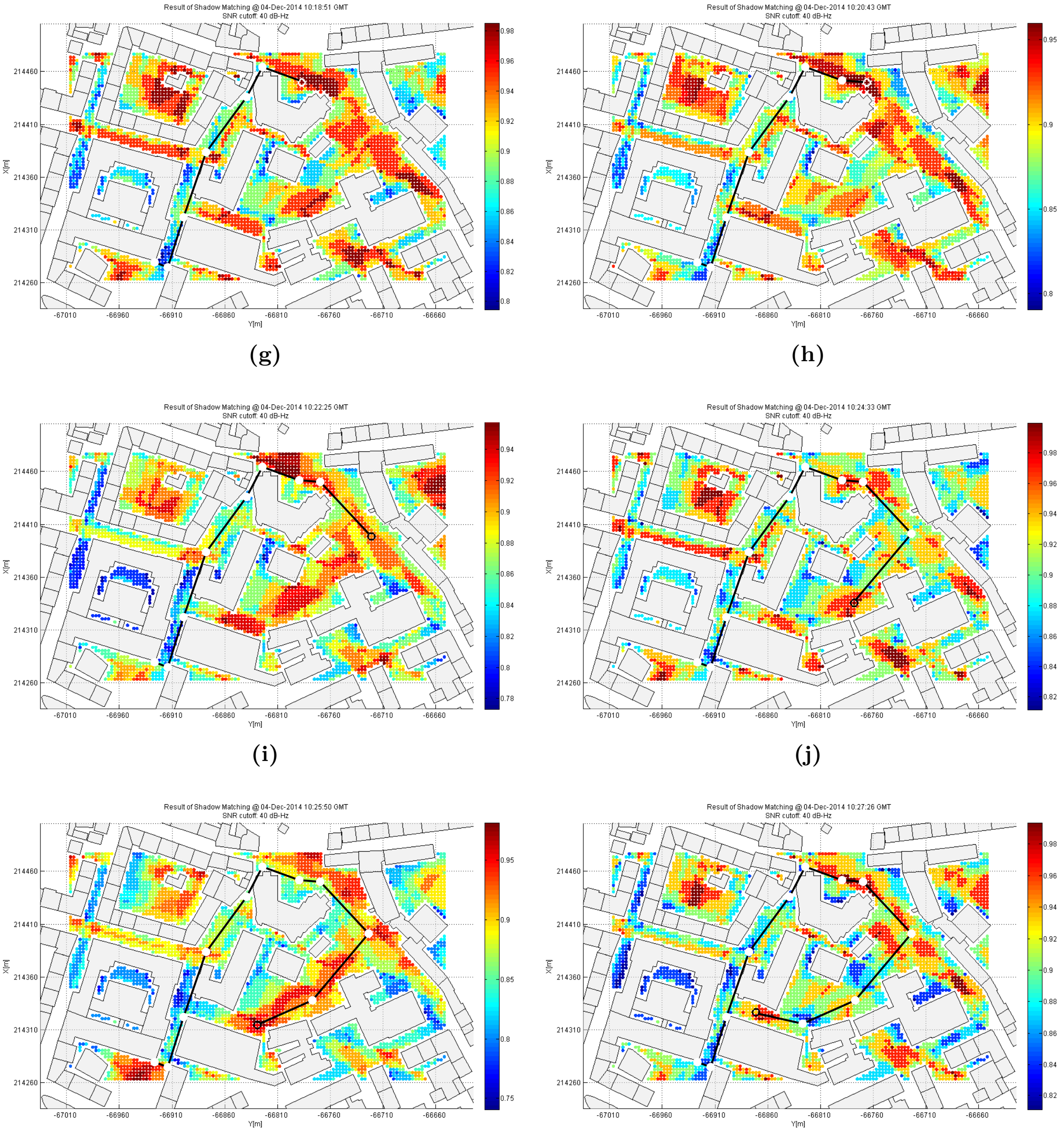

Result of Shadow Matching @ 04-Dec-2014 10:08:53 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:10:15 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:12:13 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:13:45 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:15:34 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:17:09 GMT
SNR cutoff: 40 dB-Hz
X[m]
214460
214410
214360
214310
214260
Y[m]
-67010
-66960
-66910
-66860
-66810
-66760
-66710
-66660

(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure 4.25.: Trajectory part I


Result of Shadow Matching @ 04-Dec-2014 10:18:51 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:20:43 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:22:25 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:24:33 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:25:50 GMT
SNR cutoff: 40 dB-Hz
Result of Shadow Matching @ 04-Dec-2014 10:27:26 GMT
SNR cutoff: 40 dB-Hz
X[m]
Y[m]
214460
214410
214360
214310
214260
-67010
-66960
-66910
-66860
-66810
-66760
-66710
-66660

(g) **(h)**

(i) **(j)**

(k) **(l)**

Figure 4.26.: Trajectory part II

4.2.2. Comparison to GPS Measurements

The comparison between the positions of GNSS (GPS) fingerprinting and positions from GPS measurements is the subject of this chapter. The fingerprints yield a location probability for each grid point at each epoch, but no absolute position. Therefore, an initial position is required, on the basis of which is sought for the position within a certain area. The initial position may be provided by a variety of complementary positioning methods - for example dead reckoning, relative positioning via an Inertial Measurement Unit (IMU), or GNSS measurements. As initial position, GPS measurements were used in the course of this thesis. A circle with the radius r was created, with the GPS positioning solution of an epoch as its centre. Grid points within this limited area were evaluated. There are several ways to determine the evaluation area around an initial position. For example, [10] suggests a variable ellipse based on the precision of GPS measurements. In this work, a circle with fixed radius was applied to each GPS measurement at a location where measurements took place. The points were selected by a nearest neighbour search pattern around the initial position with a maximum distance r.

The evaluation of the selected grid points is based on the highest probability of user locations inside the circumcircle. The average of n grid point positions with highest probabilities, yields the GPS fingerprint position.

$$\hat{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.6}$$

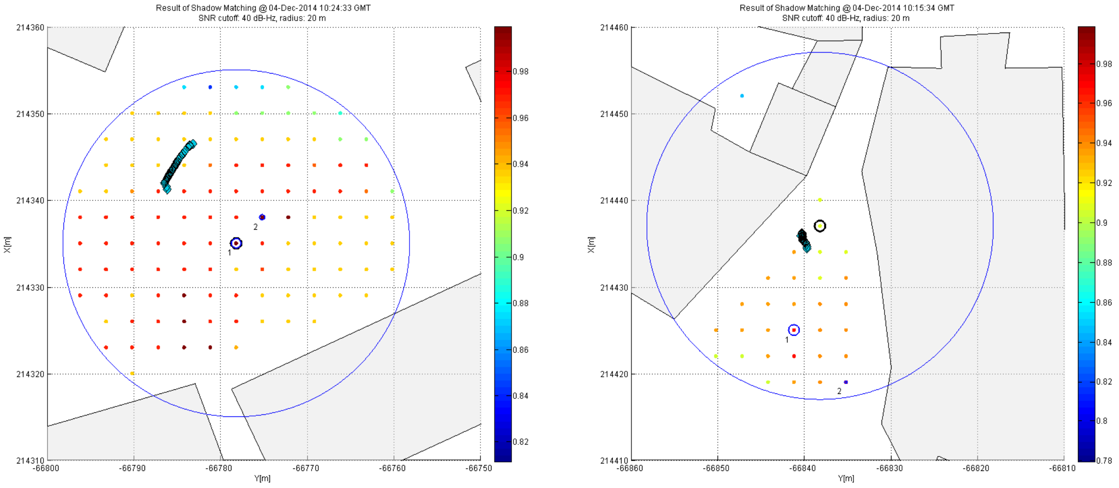
$$\hat{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{4.7}$$

Other methods, such as Bayesian filters are conceivable, without the need for a search area. Since the position solutions of each epoch can vary significantly, the accordance of the GPS fingerprint positioning solutions over all epochs is shown. If for example measurements for two epochs have been taken, and the same GPS fingerprint position is obtained from both measurements, then the accordance would be 100%. The GPS fingerprint positioning solutions can vary over several epochs, due to:

- SNR measurements (high time variations, multipath effect, etc.)
- The calculated fingerprint, because a satellite is visible in one epoch, but invisible in the next.
- The location of the initial GPS position.

The accordance of respective GPS fingerprint positions for the following examples is given in tabular form. This indicates the scattering of position solutions over all epochs at one location.

Figure 4.27 shows the comparison between GPS (green diamonds) and fingerprint position solutions at two locations. The true position, which was defined to be the nearest grid point to a true user location, is shown as a black circle and the fingerprint positions as blue circles, whose size depend on the number of same detected positions over all epochs. The area of search is indicated as a blue circle around the grid points. Note, that it is centred at the true location. When computing location probabilities for an epoch, the search area is centred at the epoch's GPS position solution, since the true location is not known. Therefore, each epoch comprises a different set of grid points to evaluate the most probable position. Since the location probabilities change for each epoch, the probabilities displayed are the mean values of all epochs.



GP-ID	occurrence [%]
1	75
2	25

(a) Trajectory way point j

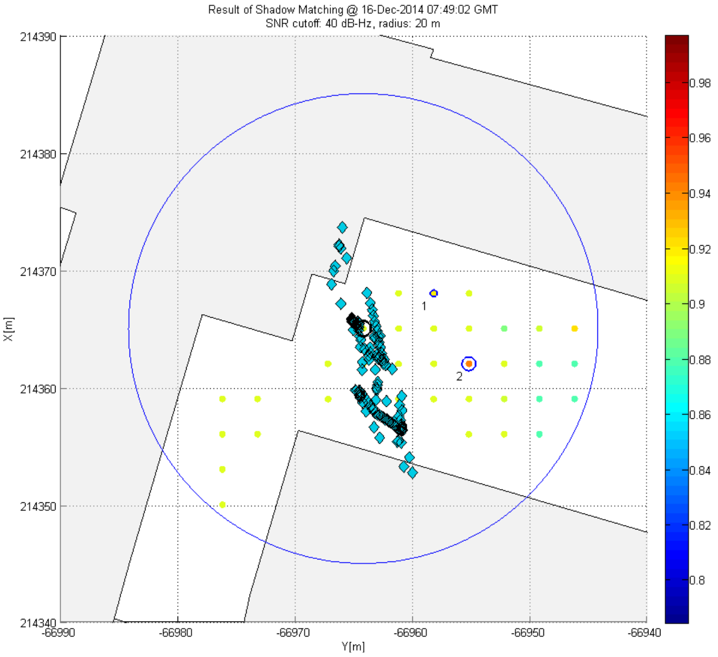
GP-ID	occurrence [%]
1	95
2	5

(b) Trajectory way point e

Figure 4.27.: Comparison of fingerprint results and GPS measurements - way point j and e

An example of a good position solution is figure 4.27a. The GPS solution suggests, that the measurement has taken place on the opposite side of the street. This is about 8.10 to 14.3 m deviation in horizontal position from the true location. The fingerprint solution detects the true location in 75% of the epochs. A quarter of the measured epochs gets associated with a location 4.25 m distant to the true location. In contrast, a large error in position is caused by fingerprinting in figure 4.27b. Although only one position is suggested in 95% of the cases, the horizontal error with 12.3 m is significant (GPS: ∼1.5-3 m horizontal error).

Another example in an inner courtyard (see figure 4.28) shows the variance of GPS positions in north-north-west and south-south-east directions. The fingerprint positions are far off the true user position in east-west direction (6-9 m), but are more resistant to variations to the north and south direction, than the GPS measurements.



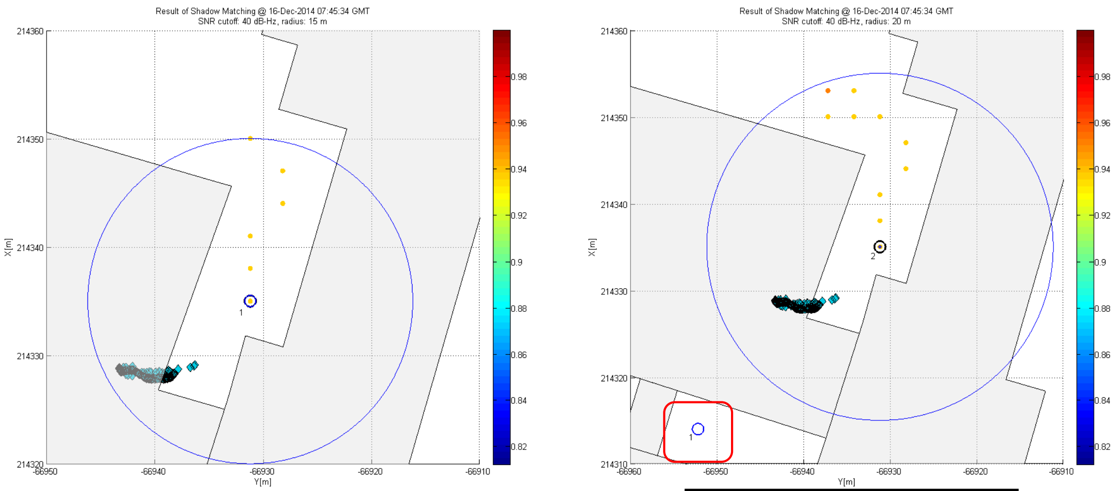
GP-ID	occurrence [%]
1	20.86
2	79.14

Figure 4.28.: Comparison of fingerprint results and GPS measurements

Investigation of radii As the selection of grid points to evaluate depends on the circle centred at a GPS position, the initial position and the radius of the circle affect the outcome of fingerprint positions intensively. Several tests of radii variation for the same measurements indicated, that a fixed radius is in some situations problematic. If the true position of a user is in a small inner courtyard of for example 15 m distance in every direction, then the initial GPS position results may not be within the courtyard at all, since most of the satellite signals will get blocked, or multipath takes place. A circumcircle with a small radius of a few meters may not include points inside the courtyard at all, which results in a fingerprint position outside the courtyard. On the other hand, if the initial GPS positions are inside the courtyard, a larger radius may include grid points outside the inner courtyard with high probabilities. The fingerprint positions would be outside the courtyard as well.

Thus, the selection of a radius appropriate to the particular location is difficult. To indicate the effect of different radii on positioning results, figure 4.29 shows a location within a narrow passage between two buildings. As can be seen, the GPS positions (green diamonds) have offsets of 7.8 to 13.6 m from the true position. Some of the GPS positions are located within the building, therefore a circumcicle with a large radius centred at these positions may include grid points outside the courtyard for the evaluation of location probabilities.

Figure 4.29a displays the outcome of fingerprint positioning at the true location with an occurrence of 100%, thus, positions for all epochs are detected correctly. The chosen radius is 15 m from an initial position. On the other hand, a slightly increased radius ($r = 20$ m) effects the outcome significantly (see figure 4.29b). With an occurrence of 94%, the grid point with the ID 1 (see red rectangle) is detected outside the courtyard. The detected grid point



GP-ID	occurrence [%]
1	100.0

(a) Radius of 15 m

GP-ID	occurrence [%]
1	94.07
2	5.93

(b) Radius of 20 m

Figure 4.29.: Comparison of fingerprint results and GPS measurements - radii impact

is not within the visualised search area, as the initial position of the search area depends on the epoch's GPS positioning solution, as mentioned before. The detected grid point with the ID 1 results in a greater error in horizontal distance to the true location, than the initial GPS positions.

4.2.3. Conclusion

Large areas of the same location probability lead to ambiguities. Therefore, an implementation of a time-based filter or a filter including knowledge about past results or measurements is recommended. Additionally, a higher frequency of measurements (for example 1 Hz) would provide additional information for a filter providing a dynamic model (for example a Kalman filter).

It has been shown, that based on wrong modelling of buildings, the results are heavily effected. Furthermore, the results are better, the more shading is cast by buildings (narrow areas like alleyways). As was also shown in chapter 4.1.5 on page 98, multipath heavily effects the SNR measurements, and thereby the predicted fingerprint model and the SNR measurements do not fit.

The methods of defining a search area around an initial position, and consequently the evaluation of selected location probabilities within the search area, play a major role in fingerprint positioning. An initial position could be provided by other positioning methods (GPS, IMU), and a search area can be implemented in several ways (fixed radius circle, ellipse based on the accuracy of GPS measurements). Further it has been shown, that the size of the search area is heavily correlated to the fingerprint positioning results.

4.3. Effects on Shadow Matching Accuracy

The effects on shadow matching accuracy are summarised in this chapter, and references to chapters where error analysis and deliberations were made, are given. Furthermore, it is a short conclusion on what to focus and take care of, when preparing data for GNSS fingerprinting, computing LoS, taking SNR measurements and match the predicted model to the measurements.

Satellite Positions The precision of the calculated satellite positions effect the shadow matching accuracy directly, as the vector from a reference point to a satellite is error prone. The different influences on the satellite position precision are listed in the following.

- The effect of dated ephemeris (section 3.4.2 on page 68) on the satellite position accuracy is negligible. Investigations showed, that for ephemeris dating a week back, the error in position is 6 km, which corresponds to an error of 65” in elevation.
- The neglect of the receiver clock error is covered in section 3.4.2 on page 70. It was evaluated, that the impact is even smaller, than the error originating from dated ephemeris.
- If an approximate user position is used for transit time calculation (see section 3.4.2 on page 72), the resulting error in satellite position is based on the size of the grid, and the elevation of the corresponding satellite. For a grid size of 849 x 600 m, the maximum error, which occurs at an elevation of $\varepsilon = 1°$, would be $\Delta \mathbf{x}_{\text{Sat}} = 5.2\,m$.
- The precision of the calculated satellite positions depends additionally on the used ephemeris. Sources of ephemeris are precise ephemeris, given for example in the NGS-SP3 file format [13]. They are available as final (+/- 2 cm, available in 13 days), rapid (+/- 5 cm, available next day), or ultra rapid (+/- 15 cm, available every 2 hours) precise ephemeris. The broadcast ephemeris used in this thesis have an accuracy of +/- 1 to 2 m, valid for 2 hours. The third possibility for calculating satellite positions is to use almanac data, which offer a precision of a few kilometres.

Grid Points As the precision of satellite positions, the precision of grid points influences the shadow matching accuracy, because the LoS vector is directly affected.

- The precision of the 3D grid points depends on the precision of the DEM, which is used for extracting height information.
- The spacing of the grid points defines the possible accuracy of GNSS fingerprinting. The closer the points are defined to each other, the more accurate a final GNSS fingerprint positioning can be obtained.
- The 3D grid points are not modelled as obstacles. This is an unsatisfying abstraction of the real world, since possible user locations can block satellite signals (hills, staircases, bridges).

3D Model As was mentioned in the chapters 3 and 4, a major increase of the accuracy concerning GNSS fingerprinting can be achieved by an accurate 3D city model. The details are listed in the following.

- The detail of model, and abstraction of roofs (2.5 or 3D model) effects the accuracy of shadow matching greatly; for an analysis of the influence of the detail on the Sky Plot approach, see section 4.1.5 on pages 99 and 100. This is one of the major effects, which influence the accuracy of shadow matching.

- When the model is based on other data (for example building groundplans), its precision is effecting the 3D model as well (see section 3.2.2 on page 31).

- The vegetation is not modelled, which implies errors in the prediction of satellite visibilities. Therefore, a seasonal vegetation modelling is important for better GNSS fingerprinting results.

LoS Computation The accuracy of the LoS computation can be effected for the Sky Plot and the Rendering Shadows approach directly. The Intersections in 3D approach is effected indirectly by the detail of the 3D model, as well as the precision of the grid points and satellite positions.

- Sky Plot:
  - The resolution of sky plots (lines of view) effects the accuracy of the building boundary detection (section 3.3.3 on page 45). Because discrete elevation and azimuth angles for the building boundaries were calculated, the chosen angle resolution influences the detection of visible and invisible satellites as well.

- Rendering Shadows:
  - The resolution of rendered images controls the accuracy of shadow matching directly, as each grid point has to be assigned to a pixel. If the resolution is too low, then several grid points may be assigned to a pixel, which would result in erroneous shadow detection.
  - An error is induced by the usage of a sun light source, instead of a point light source (see section 3.3.4 on page 53).

SNR Measurments Investigations in SNR measurements showed, that the fingerprint results are heavily dependent on correct measurements. Some negative effects are listed below.

- Multipath is evident in some places and leads to grave errors, as invisible multipath satellite signals may wrongly be tagged as visible, because of high SNR values.

- Each GNSS antenna behaves differently, concerning the sensitivity of measuring SNR signals and tracking invisible satellites. Also the multipath mitigation differs from antenna to antenna. A pre-evaluation of an antenna and the adoption of the SNR cut-off value for the shadow matching process is required.

Shadow Matching Finally, the direct effects on shadow matching are discussed, which are covered in chapter 4.2.

- Large areas of same location probabilities lead to ambiguities. Without information on previous positions, no accurate fingerprint positioning can be performed.
- The detection of a position within the grid can be improved by:
  - The implementation of measurements with higher frequency (for example 1 Hz) and a time-based filter introducing a dynamic model (for example a Kalman filter) can improve GNSS positioning. By using past measurements and incorporating a dynamic model, a statement about possible locations can be made. The higher the measurement frequency, the more information can be passed to the filter.
  - An appropriate method of defining a search area around an initial position, as well as the evaluation of selected location probabilities, plays a major role in fingerprint positioning.
  - The search for a position within a search area can be implemented in different ways. In this thesis, the mean position of the grid points with highest location probability has been used. Non-parametric Bayesian filters (grid-based estimation or particle filter), considering the user's dynamics, could improve the position estimation process.
- The more shading is present (for example narrow areas like streets), the better are the results of fingerprinting.

5. Conclusions and Outlook

The conclusions made in the context of this thesis are summarised in the following. Furthermore, a short outlook of possible improvements and applicabilities of GNSS fingerprinting is given.

It has been shown, that the creation of an appropriate 3D city model for both buildings and grid points can be accomplished from a DEM and building groundplans. Statistical methods were used to extract vegetation contained in the created grid, to generate a grid representing the earth's surface of a city.

As has been pointed out in section 3.3, various methods are suitable for the calculation of GNSS fingerprints. Three methods of LoS computation were introduced and compared, each with its advantages and disadvantages, corresponding speed, adaptivity to 3D model changes and accuracy. Several deliberations were made, whether abstractions and simplifications of processes (to increase the computational speed) are suitable. Furthermore, an error analysis was performed on the abstraction and simplification processes.

1. The approach of intersecting LoS with the building polygons in 3D proved to be the most accurate, but slowest approach.

2. The Sky Plot approach chosen for further investigations in this thesis, outperformed the other approach in terms of speed, although the detection of building boundaries turned out to be very time consuming in the pre-processing step.

3. As an alternative to the above mentioned approaches, a 3D rendering of the city was conducted with a 3D rendering program. Evaluation of the produced shadow images yielded the fingerprints. Several pre-processing steps had to be implemented, but the outcome proved to be similar to the other methods, showing that several methods can be used for GNSS fingerprint calculations.

Possible optimizations for each approach are given and different acceleration concepts, which were implemented for the approaches in the course of this thesis, were discussed.

Satellite position calculations as input for the LoS computations and possible error sources have been evaluated. Error analyses have been made on the impact of dated ephemeris, neglect of the receiver clock error, as well as the usage of an approximate user position for the transit time calculation. Errors caused by the sources mentioned above are much smaller than errors caused by false modelling of the city.

Investigations on SNR measurements were performed to define a threshold, that declares, which SNR values belong to visible or invisible satellites. It was demonstrated, that the multipath of signals and time variations of SNR measurements are effecting the outcome of GNSS fingerprinting. Furthermore, the antenna performance is heavily correlated to its location. Methods to reduce the effects have been introduced (see chapter 4.1.5).

Positions calculated by GNSS fingerprinting have been evaluated by measuring a trajectory and comparing the results to GPS code solutions. Fingerprint positioning works best in narrow street canyons, where especially the side of street could be detected correctly. Therefore, the across-street component is more accurate than the along-street component, opposing to GPS positioning. Especially at crossroads, where a user changes direction, the detection of the correct position works well. At open spaces, similar location probabilities result in ambiguities regarding a possible user location.

Improvements of the presented methods include implementation of a more detailed 3D city model, modelling of vegetation, faster algorithms for fingerprint calculations (LoS computations) to calculate fingerprints in real time and further research in SNR threshold and multipath effects. To yield more precise positions, a sophisticated determination of a user position within a grid of points, carrying location probability informations, is needed. Several navigation and positioning techniques are suitable for using the fingerprints as inputs. For example, the grid of location probabilities can be used as input for a non-parametric Bayes filter (e.g., grid-based estimation), which estimates the state (position) of the user, based on observations and a dynamic model.

Further research in the field of multi-sensor integrated navigation systems - where GNSS fingerprinting could act as a supporting technology - is necessary in the future. Especially in urban areas with a large number of shading structures, the usage of GNSS fingerprinting would emphasize its strengths by not only improving the availability, but also enhancing the quality of the positioning results. This can be used for the detection of lanes for guidance systems of vehicles, for location-based services, as well as navigation systems for visually impaired people [10].

A. Appendix

A.1. Projections and Transformations

The following formulas are extracted from [9]. Parameters required for the transformations and projections are given in table A.1.

Table A.1.: WGS84 and Bessel Parameters

Parameter	Value	
a_{WGS84}	6378137.0	m
b_{WGS84}	6356078.9628	m
a_{Bessel}	6377397.1550	m
b_{Bessel}	6356078.9628	m

Plane to Ellipsoidal Coordinates - Reverse Gauss-Krüger Projection Auxiliary quantities for computation of the footpoint latitude:

$$n = \frac{a-b}{a+b} \tag{A.1}$$

$$\bar{\alpha} = \frac{a+b}{2}\left(1 + \frac{1}{4}n^2 + \frac{1}{64}n^4 + ...\right) \tag{A.2}$$

$$\bar{\beta} = \frac{3}{2}n - \frac{27}{32}n^3 + \frac{269}{512}n^5 + ... \tag{A.3}$$

$$\bar{\gamma} = \frac{21}{16}n^2 - \frac{55}{32}n^4 + ... \tag{A.4}$$

$$\bar{\delta} = \frac{151}{96}n^3 - \frac{417}{128}n^5 + ... \tag{A.5}$$

$$\bar{\varepsilon} = \frac{1097}{512}n^4 + ... \tag{A.6}$$

The reversal of the meridian arc length is the footpoint latitude φ_f which can be calculated by a series expansion and is based on the meridian arc length $\bar{B}$ of equation A.8:

$$\varphi_f = \bar{B} + \bar{\beta}\sin 2\bar{B} + \bar{\gamma}\sin 4\bar{B} + \bar{\delta}\sin 6\bar{B} + \bar{\varepsilon}\sin 8\bar{B} + ... \tag{A.7}$$

$$\bar{B} = \frac{x}{\bar{\alpha}} \tag{A.8}$$

Further quantities have to be introduced for the computation of the ellipsoidal latitude φ and longitude λ in equation A.12:

Radius curvature in prime vertical:

$$N = \frac{a^2}{\sqrt{a^2 \cos^2 \varphi_f + b^2 \sin^2 \varphi_f}} \tag{A.9}$$

Auxiliary quantities:

$$\eta = \frac{\cos \varphi_f}{b} \sqrt{a^2 - b^2} \tag{A.10}$$

$$t = \tan \varphi_f \tag{A.11}$$

Longitude of the central meridian:

$$\lambda_0$$

Ellipsoidal Coordinates:

$$\begin{aligned}
\varphi =& \varphi_f + \frac{t}{2N^2}(-1 - \eta^2)y^2 \\
&+ \frac{t}{24N^4}(5 + 3t^2 + 6\eta^2 - 6t^2\eta^2 - 3\eta^4 - 9t^2\eta^4)y^4 \\
&+ \frac{t}{720N^6}(-61 - 90t^2 - 45t^4 - 107\eta^2 + 162t^2\eta^2 + 45t^4\eta^2)y^6 \\
&+ \frac{t}{40320N^8}(1385 + 3633t^2 + 4095t^4 + 1575t^6)y^8 + ... \\
\lambda =& \lambda_0 + \frac{1}{N \cos \varphi_f} y \\
&+ \frac{1}{6N^3 \cos \varphi_f}(-1 - 2t^2 - \eta^2)y^3 \\
&+ \frac{1}{120N^5 \cos \varphi_f}(5 + 28t^2 + 24t^4 + 6\eta^2 + 8t^2\eta^2)y^5 \\
&+ \frac{1}{5040N^7 \cos \varphi_f}(-61 - 662t^2 - 1320t^4 - 720t^6)y^7 + ...
\end{aligned} \tag{A.12}$$

Cartesian and Ellipsoidal Coordinates The cartesian coordinates for a point P in space are

$$X = (N + h) \cos(\varphi) \cos(\lambda) \tag{A.13}$$

$$Y = (N + h) \cos(\varphi) \sin(\lambda) \tag{A.14}$$

$$Z = \left(\frac{b^2}{a^2} N + h \right) \sin \varphi \tag{A.15}$$

with the prime curvature in the primal vertical given in equation A.9 and replacing the footpoint latitude φ_f by the latitude φ.

Three Dimensional Helmert Transformation A 7-parametric transformation is given by the formula

$$\mathbf{X}_T = \mathbf{c} + \mu\mathbf{R}\mathbf{X} \tag{A.16}$$

where $\mathbf{c}$ is the shift vector, μ is a scale factor, $\mathbf{R}$ is the rotation matrix. It transforms the position vector $\mathbf{X}$ to the position vector $\mathbf{X}_T$.

The rotation matrix $\mathbf{R}$ consists of three elementary matrices rotating around the respective rotation axis:

$$\mathbf{R} = \mathbf{R}_3(\alpha_3)\mathbf{R}_2(\alpha_2)\mathbf{R}_1(\alpha_1) \tag{A.17}$$

$$\mathbf{R}_1(\alpha_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_1 & \sin\alpha_1 \\ 0 & -\sin\alpha_1 & \cos\alpha_1 \end{bmatrix}, \tag{A.18}$$

$$\mathbf{R}_2(\alpha_2) = \begin{bmatrix} \cos\alpha_2 & 0 & -\sin\alpha_2 \\ 0 & 1 & 0 \\ \sin\alpha_2 & 0 & \cos\alpha_2 \end{bmatrix}, \tag{A.19}$$

$$\mathbf{R}_3(\alpha_3) = \begin{bmatrix} \cos\alpha_3 & \sin\alpha_3 & 0 \\ -\sin\alpha_3 & \cos\alpha_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.20}$$

As the transformation parameters are given for WGS84 to Bessel Ellipsoid and are only valid for Austria (table A.2), the transversal three dimensional Helmert Transformation is:

$$\mathbf{X} = -\mathbf{c} + \frac{1}{\mu}\mathbf{R}^T\mathbf{X}_T \tag{A.21}$$

Global and Local-Level Coordinates A difference vector $\mathbf{X}_{ij}$ is given by:

$$\mathbf{X}_{ij} = \mathbf{X}_j - \mathbf{X}_i \tag{A.22}$$

This can be applied to a point in the grid $\mathbf{X}_{gp}$ and the position of a satellite $\mathbf{X}_{Sat}$:

$$\mathbf{X}_{GP-Sat} = \mathbf{X}_{Sat} - \mathbf{X}_{GP} \tag{A.23}$$

Table A.2.: Transformation Parameters from WGS84 to Bessel

		Script of Reference Systems
dx	[m]	-577.33
dy	[m]	-90.13
dz	[m]	-463.92
α_1	[cc]	15.8640
α_2	[cc]	4.5370
α_3	[cc]	16.3580
μ	[ppm]	-2.42

The difference vector $\mathbf{X}_{ij}$ in the local level frame is donated as $\mathbf{x}_{ij}$ and is rotated by $\mathbf{R}_i$ consisting of:

$$\mathbf{R}_i = \begin{bmatrix} \mathbf{n}_i, \mathbf{e}_i, \mathbf{u}_i \end{bmatrix} \tag{A.24}$$

with the components for north, east and up:

$$\mathbf{n}_i = \begin{bmatrix} -\sin\varphi_i \cos\lambda_i \\ -\sin\varphi_i \sin\lambda_i \\ \cos\varphi_i \end{bmatrix}, \qquad \mathbf{e}_i = \begin{bmatrix} -\sin\lambda_i \\ \cos\lambda_i \\ 0 \end{bmatrix}, \qquad \mathbf{u}_i = \begin{bmatrix} \cos\varphi_i \cos\lambda_i \\ \cos\varphi_i \sin\lambda_i \\ \sin\varphi_i \end{bmatrix} \tag{A.25}$$

The projection of the vector $\mathbf{X}_{ij}$ to the local level vector axes $\mathbf{n}_i$, $\mathbf{e}_i$ and $\mathbf{u}_i$ is accomplished by the transposed rotation matrix $\mathbf{R}_i$:

$$\mathbf{x}_{ij} = \mathbf{R}_i^T \mathbf{X}_{ij} \tag{A.26}$$

List of Figures

List of Tables

Bibliography

[1] Yuki Uejima Hirozumi Yamaguchi Akira Uchiyama Etsuko Katsuda and Teruo Higashino. "GPS Line-Of-Sight Fingerprinting for Enhancing Location Accuracy in Urban Areas". In: *2013 International Conference on Indoor Positioning and Indoor Navigation, 28-31st October 2013* (2013).

[2] Kasmiran Jumari Tiong Sieh Kiong Anwar H Ibrahim Mahamod Ismail. "Simple Method for Outdoor LOS Propagation Model Using Ray Tracing Tools". In: *Journal of Applied Sciences 6(12)* (2006).

[3] CDDIS GPS Archive. *RINEX 2.11 Navigation files.* Dec. 2014. URL: `ftp://cddis.gsfc.nasa.gov/gps/data/daily/2014/brdc/`.

[4] Paul Bourke. *Polygons and meshes.* June 2014. URL: `http://paulbourke.net/geometry/polygonmesh/`.

[5] J.A. Greenberg C.A. Rueda and S.L. Ustin. *StarSpan: A Tool for Fast Selective Pixel Extraction.* Dec. 2014. URL: `https://github.com/Ecotrust/starspan`.

[6] *Geofabrik - OpenStreetMap Data Extracts.* Dec. 2014. URL: `http://download.geofabrik.de/`.

[7] Andrew S. Glassner, ed. *An Introduction to Ray Tracing.* London, UK, UK: Academic Press Ltd., 1989. ISBN: 0-12-286160-4.

[8] Werner Gurtner. *The Receiver Independent Exchange Format Version 2.11.* Dec. 2014. URL: `http://igscb.jpl.nasa.gov/igscb/data/format/rinex211.txt`.

[9] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and Elmar Wasle. *GNSS - Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and more.* 2008th ed. Springer, Nov. 2007. ISBN: 9783211730126.

[10] Marek K Ziebart Lei Wang Paul D Groves. "GNSS Shadow Matching: Improving Urban Positioning Accuracy Using a 3D City Model with Optimized Visibility Prediction Scoring". In: *Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012)* (Sept. 2012).

[11] Marek K. Ziebart LeiWang Paul D. Groves. "Urban Positioning on a Smartphone". In: *Inside GNSS* (Nov. 2013).

[12] NAVSTAR. *Navstar GPS Space Segment / Navigation User Interfaces.* URL: `http://www.gps.gov/technical/icwg/IS-GPS-200H.pdf` (visited on 09/13/2014).

[13] *Precise Ephemeris Files.* Dec. 2014. URL: `http://www.nrcan.gc.ca/earth-sciences/geomatics/geodetic-reference-systems/tools-applications/9064`.

[14] Gerd Wanielik Sven Bauer Marcus Obst. "3D Environment Modeling for GPS Multipath Detection in Urban Areas". In: *9th International Multi-Conference on Systems, Signals and Devices* (2012).

[15] University of Berne Werner Gurtner Astronomical Institute. *RINEX: The Receiver Independent Exchange Format Version 2.11*. URL: `http://igscb.jpl.nasa.gov/igscb/data/format/rinex211.txt` (visited on 10/22/2014).