# Master Thesis

im Rahmen des
Universitätslehrganges “Geographical Information Science & Systems”
(UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

# “Modelling of Continuous Fields:

## Coverage Mapping Based on Dynamic In-situ WLAN Measurements”

vorgelegt von

Dipl.-Ing. (FH) Johannes Kapser

U1502, UNIGIS MSc Jahrgang 2010

Zur Erlangung des Grades
“Master of Science (Geographical Information Science & Systems) - MSc(GIS)”

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

München, 20. September 2012

# Acknowledgement

At first and foremost, I would like to express great gratitude to my family. My parents, my brothers and my grandparents were always extraordinary supportive and encouraging during the UNIGIS part-time studies and the creation of this Thesis.

I would like to express special thanks to my colleagues from Z_GIS University Salzburg, for discussions of the structure and content towards the development of this thesis. My thanks goes as well to the Research Group Geodesy Technical University of Vienna, for providing me the necessary dynamic WLAN measurement and GPS positioning data.

Particular thanks is dedicated to the UNIGIS Team for their support during the studies and Prof. Josef Strobl for the supervision of this thesis.

# Affidavit

Herewith I, Johannes Kapser, declare that I have written the thesis at hand fully on my own and that I have not used any other sources apart from those given.

Date

Signature

# Abstract

In literature (Honkavirta, Perala, Ali-Loytty & Piche 2009), (Parodi, Lenz, Szabo, Hui, Horn, Bamberger & Obradovic 2006), (Li, Wang, Lee, Dempster & Rizos 2005) and (Sen, Gümüsay, Kavas & Bulucu 2008) lots of different approaches have been applied for modelling of electro-magnetic field maps, for the purpose of WLAN (Wireless Local Area Network) positioning. However, none of them considers coverage mapping based on dynamic WLAN measurements, taking surrounding mobile objects' positions into account. This research introduces a method for reference measurement identification, out of dynamic field measurements, enabled by spatio-temporal filter strategies. Furthermore, it proposes a spatial database model for processing and storage of electro-magnetic field maps.

This is done by revision of theoretical concepts and methods on wireless network positioning systems and field modelling approaches. Available data sources of static and dynamic characteristic will be evaluated, modelled and analysed. On these groundwork of data, parametric and probabilistic field modelling approaches have been implemented and qualified against a measurement reference track. Thereby, it has turned out that Ordinary Kriging is best suitable for electro-magnetic field modelling. The findings of this research constitutes an excellent basis for spatio-temporal field modelling.

# Kurzfassung

In der Literatur (Honkavirta et al. 2009), (Parodi et al. 2006), (Li et al. 2005) und (Sen et al. 2008) werden viele verschiedene Ansätze der elektromagnetischen Feldmodellierung zum Zwecke der WLAN Positionierung aufgezeigt. Jedoch betrachtet keiner dieser Ansätze die in-situ Kartierung aus dynamisch bezogenen Messwerten und beachtet dabei die Positionen umgebender mobiler Objekte. Diese Forschungsarbeit führt eine Methode zur Extrahierung von Referenzmessungen aus dynamischen Feldmessungen ein. Dabei wird eine raum-zeitliche Filterstrategie angewendet. Hierfür werden zuerst theoretische Konzepte und Methoden zur Positionierung in drahtlosen Netzwerken und Feldmodellierungsansätze aufgeführt. Zur Verfügung stehende dynamische und statische Datensätze werden evaluiert, modelliert und analysiert. Auf dem Fundament dieser Daten werden parametrische und statistische Ansätze zur Feldmodellierung implementiert und gegen einen Spur von Referenzmessungen qualifiziert. Dabei stellt sich heraus, dass Ordinary Kriging am besten geeignet ist, für die Modellierung der räumlichen Verteilung des elektro-magnetischen Feldes. Die Ergebnisse dieser Arbeit bilden eine exzellente Basis für die raum-zeitliche Modellierung von Feldern.

# Contents

# List of Figures

# List of Tables

# Acronyms

**ADS-B**
Automatic Dependent Surveillance Broadcast.

**ANN**
Artificial Neural Network.

**AoA**
Angle of Arraival.

**AP**
Access Point.

**BIM**
Building Information Model.

**BMN**
Bundesmeldenetz.

**CAD**
Computer Aided Design.

**DBMS**
Database Management System.

**DOP**
Dilution of Precision.

**DPM**
Direct Path Model.

**DSSS**
Direct-Sequence Spread Spectrum.

**DWG**
Drawing.

**EDA**
Exploratory Data Analysis.

**EGNOS**
European Geostationary Navigation Overlay Service.

**EPSG**
European Petroleum Survey Group.

**FFG**
Research Promotion Agency.

**FMCW**
Frequency-Modulated Continuoues Wave.

**GeoDBMS**
Geographical Database Management System.

**GF**
Generating Function.

**GIS**
Geographic Information System.

**GNSS**
Global Navigation Satellite System.

**GPS**
Global Positioning System.

**HMI**
Human Machine Interface.

**ICAO**
International Civil Aviation Organization.

**IDW**
Inverse Distance Weight.

**ISM**
Industrial Scientific and Medical.

**LoS**
Line of Sight.

**MAC**
Media Access Control.

**MGI**
Military Geographic Institute.

**MLAT**
Multi-Lateration.

**MU**
Mobile Unit.

**MWM**
Multi Wall Model.

**NB**
Narrow Band.

**NMEA**
National Marine Electronics Association.

**PL**
Path Loss.

**RM**
Radio Map.

**RP**
Reference Points.

**RSS**
Received Signal Stregth.

**RToF**
Round-trip Time-of-Flight.

**SESAAM**
Geo-Spatially Enhanced Situational Awareness for Airport Management.

**SGeMS**
Standford Geostatistical Modeling Software.

**SHP**
Shape.

**SNR**
Signal-to-Noise Ratio.

**SS**
Smooth Seminorm.

**SSID**
Service Set Identification.

**SSR**
Secondary Surveillance Radar.

**TDoA**
Time Difference of Arrival.

**ToA**
Time of Arrival.

**UML**
Unified Modeling Language.

**UTC**
Coordinated Universal Time.

**UTM**
Universal Transverse Mercator.

**UWB**
Ultra Wide-Band.

**VM**
Virtual Machine.

**WAF**
Wall Attenuation Factor.

**WB**
Wide-Band.

**WLAN**
Wireless Local Area Network.

# 1. Introduction

Continuous fields are mostly based on physical phenomena and in nature their constitution is very often dependent on the variables of time and space. The modelling of such phenomena compound several challenges: (i) An appropriate discretisation approach has to be selected, allowing efficient numerical processing of the actual functional describable continuous phenomenon. (ii) Highly organised data structures are necessary, for storage and processing of modelled field coverages. (iii) In nature, ideally modelled field phenomena underlay the real-word of unforeseen physical influence with inherent statical occurrence.
This problem statement applies to many physical real-world phenomena, as for instance acoustic noise, optics and atomic or electro-magnetic radiation. On the latter will be focused in this thesis. The continuous field modelling thematic has been given through the context of the Austrian FFG (Research Promotion Agency) funded TAKE OFF project SESAAM (Geo-Spatially Enhanced Situational Awareness for Airport Management) from March 2010 to July 2012. One research target of SESAAM is the investigation of low-cost positioning methods for WLAN (Wireless Local Area Network) positioning of objects, moving on aviation ground environments. For this purpose the RSS (Received Signal Stregth) based fingerprinting WLAN positioning method has been selected by the Technical University of Vienna research group Geodesy. This technology requires a RM (Radio Map), modelling the electro-magnetic field coverage of available WLAN AP (Access Point)s. The research of this thesis, under the umbrella of Z_GIS (Centre for GeoInformatics) University Salzburg, should give major contribution to the electro-magnetic field modelling in this research project, on which the actual WLAN positioning algorithm is based. As the research on low-cost position technologies is one of the stated project targets (Bretz, Kapser & Roth 2011), the implementation of computational tools and software have been done in freely available Open Source environments.

The central research question of this thesis is already given by the title: “Modelling of Continuous Fields: Coverage Mapping Based on Dynamic In-situ WLAN Measurements”. Thereby, the colon divides the theoretical overall topic from the empirical part of the thesis. However, especially the latter part of the thesis title invites for more specific verbalisation of the research questions. The “Coverage Mapping Based on Dynamic WLAN Measurements” should not be confused with “Dynamic Coverage Mapping...”. The thesis does not focus on spatio-temporal field modelling, rather in modelling of continuous fields out of a bulk of dynamically gathered in-situ measurements. This leads to the following formulation of research questions:

- What modelling method is suitable for electro-magnetic fields in non-obstacle free environments?

- Is it possible to estimate a static field map out of dynamic in-situ measurements?
- What data structure is capable for multi-layer field mapping?

This questions should be answered by this thesis at the end of the day.

The thesis project provides a massive data volume of recorded in-situ WLAN measurement values, collected by MU (Mobile Unit)s, and aircraft positioning on the testbed area. Out of these dynamic data sets, measurement reference values will be extracted on a spatio-temporal rule set. These reference values should constitute the basis for relevant field modelling approaches. For handling of data and analysis, a tailor-made data processing frame-work has been developed and implemented accordingly.

This work is exclusively focused on modelling of continuous electro-magnetic fields, on the example of WLAN positioning infrastructures and does not consider any signalling, location sensing, position algorithmic or position display related topics. The thesis suggests theoretical methods, abstract processes and applied techniques towards field modelling, however it does not provide finalised electro-magnetic field maps for a kind of end solution.

The target audience of this thesis is everyone, who is interested in continuous field mapping with the aid of GIS (Geographic Information System) related tools and geo-statistic theory. Especially interested persons in electro-magnetic or physically similar distribution phenomena might be benefit from that work. However, this thesis should be accessible without any background in physical or electrical theories. As this thesis should constitute a sustainable groundwork for ongoing research of continuous field modelling, it might be relevant for each, who wants to expand that work in the direction of spatio-temporal field modelling and/or geo-statistical simulation.

The thesis structured into this introduction, three core chapters and has been finalised by a chapter about discussion and future directions. Additionally, a Research Methods Appendix is attached to this work, as it should not disturb the thread of arguments through the core of this thesis. A flow chart of the thesis's main chapter is shown in figure 1.1. Each of the three main chapters is depicted as functional block, extended by sections of first and second order.
After this **Introduction**, **Theoretical Methods and Concepts** basically focuses on theoretical fundamentals and literature review. Therein, *Wireless Network Positioning Techniques* introduces theoretical concepts and principles of communication systems and the wireless communication channel. Moreover, it regards positioning systems from a system point of view. *Methods of Radio Map Generation* does focus on the mathematically fundamental definition of the radio map model and illuminates electro-magnetic field modelling approaches by literature review.

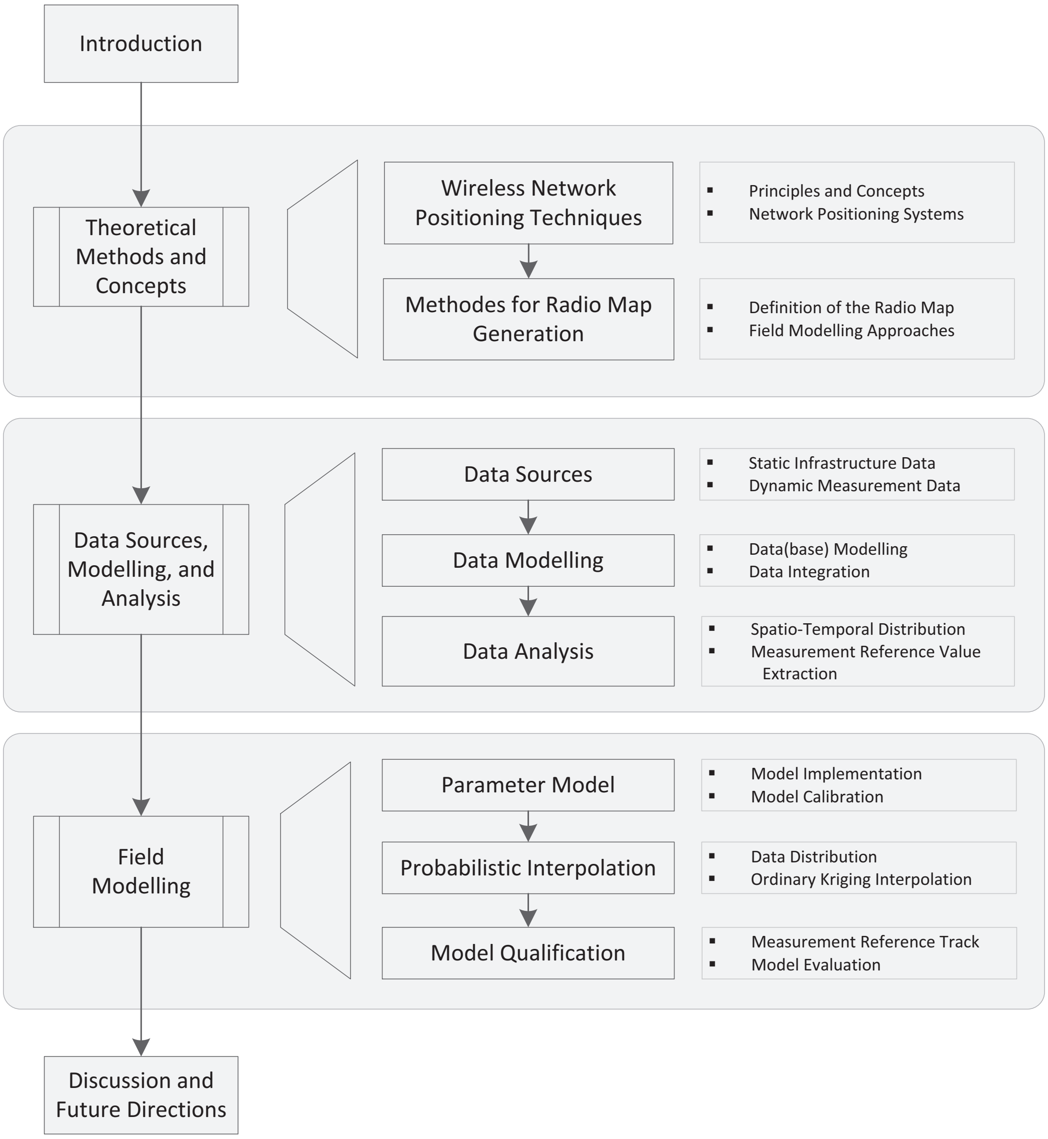

Introduction
Theoretical Methods and Concepts
Wireless Network Positioning Techniques
Principles and Concepts
Network Positioning Systems
Methodes for Radio Map Generation
Definition of the Radio Map
Field Modelling Approaches
Data Sources, Modelling, and Analysis
Data Sources
Static Infrastructure Data
Dynamic Measurement Data
Data Modelling
Data(base) Modelling
Data Integration
Data Analysis
Spatio-Temporal Distribution
Measurement Reference Value Extraction
Field Modelling
Parameter Model
Model Implementation
Model Calibration
Probabilistic Interpolation
Data Distribution
Ordinary Kriging Interpolation
Model Qualification
Measurement Reference Track
Model Evaluation
Discussion and Future Directions

Figure 1.1.: Sequence of chapters and sections

After this theoretically based block, **Data Sources, Modelling and Analysis** reviews the set-up material of this thesis project. This is done in three consecutive steps. First of all *Data Sources* reviews available static infrastructure and dynamic measurement data sets, for the purpose of static display of infrastructural data and measurement reference value extraction. Second, the *Data Modelling* section proposes a suitable model for data modelling and integration. This chapter ends by *Data Analysis* of dynamic data in terms of spatio-temporal distribution. On that basis, a suitable process has been elaborated for measurement reference value extraction.

The last core chapter cares about **Field Modelling**, on the basis of up to this point elaborated findings. It starts with implementation of a *Parameter Model* considering physical antenna

characteristic in connection with the radial model based field modelling approach. After implementation, this model is calibrated by previous extracted measurement reference values. The second approach illuminated in this section is the *Probabilistic Interpolation*. This realises Ordinary Kriging interpolation, based again on the measurement reference values. Additionally, a second interpolation point basis is considered, compounded of samples of the elaborated parameter model and measurement reference values. All these field modelling approaches will be analysed against a measurement reference track and compared against each other in the *Model Qualification* part of this thesis.
This work finishes with a **Discussion and Future Directions** chapter.

# 2. Theoretical Methods and Concepts

Theoretical methods and concepts are the key to each research and so to this thesis. They provide the fundamentals and current status of research, in the area of field modelling in reference to electro-magnetic field modelling. This will be introduced, by a mixture of reviewing research publications and standard literature in that field. Although this thesis is basically focused on modelling of continuous fields, this chapter goes beyond purely revising this regarding literature. It takes the concepts and physical boundaries of WLAN (Wireless Local Area Network) positioning into account, in order to get familiar with related terms and technology limitations. This is regarded as crucial basic knowledge, for following up later field modelling concepts of this thesis. However, the main focus is laid on field modelling concepts, or in other words: "Methods for Radio Map Generation".

Wireless network positioning techniques will be elaborated fist of all. Thereby, principles and concepts of communication systems in general will be introduced, focusing on the physical constitution and possibly interfering effects of the wireless communication channel. This will be followed by an introduction to network positioning systems, elaborating the core components of that system with their technological concepts behind. After this communication systems based overview, methods for radio map generation will be discussed. This starts by a mathematical definition of a possible radio map system structure and goes over to physical field modelling approaches. The latter part is subdivided into *deterministic* and *probabilistic* modelling approaches.

## 2.1. Wireless Network Positioning Techniques

Networked communication systems constitute the foundation of wireless network positioning. Basically, they provide a infrastructure for the purpose wireless positioning or geolocation determination. In the case of wireless network positioning, the infrastructure is primarily used for communication of data. This is due to the fact that wireless positioning is mostly used in indoor and campus environments, where given WLAN infrastructure is utilised for a variety of IT-Services (Honkavirta et al. 2009),(Parodi et al. 2006).

The concepts of positioning are on the forefront of this section. However, in order to get access to the idea later on in this thesis, first of all electro-physical grass roots will be revised. This will be done by guidance through a basic model of communication systems, whereas the focus is laid on the communication channel. It will not get into much detail, though a basic understanding of radio propagation effects is crucial to obtain the later on introduced methods of electro-magnetic field modelling. Besides that, different network positioning topologies, its functional architecture and inherent components from a system point of view will be elaborated. On the basis of a generic functional positioning system architecture, a basic illustration of the components' function will be given. It starts at different available positioning technology, goes further about positioning metrics and algorithms to positioning storage and display strategies.

### 2.1.1. Principles and Concepts

The science of communication systems is a complex field and fills scores of books, university courses' curriculum and scientific journals. The very basic understanding of a communication system and its underlying fundamentals should be shown on its basic elements, depicted in figure 2.1 from Haykin (2001).

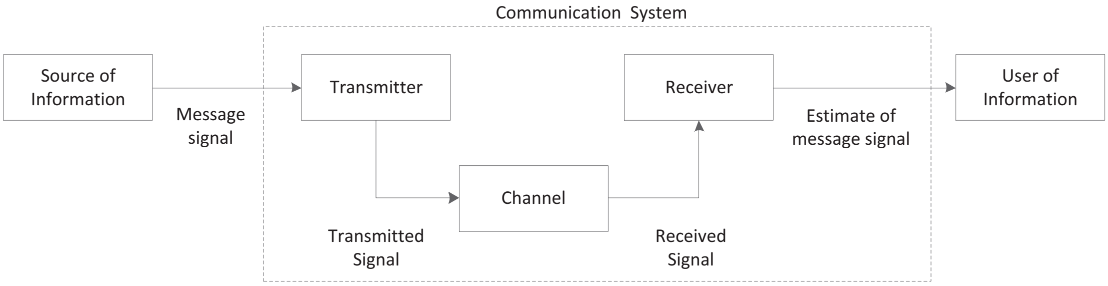


Figure 2.1.: Basic elements of a communication system

The basic principle of a communication system is the transmission of information, or messages, from a *Source of Information* to the *User of Information*. In telecommunication for instance four major source of information are employed: *speech*, *music*, *pictures* and *computer data*, whereas the latter one is relevant for WLAN communication. For the purpose of information transmission, the source of information is now inserted as a message signal to the communication system. The communication system itself includes three major components:

- The *Transmitter encodes* the source of information, independent of its corresponding information, appropriate for the consecutive communication channel. In the example of WLAN communication, a AP (Access Point) constitutes the transmitter of the communication system. The transmitted power $P_0$ is a crucial parameter of the transmitter.
- The *Channel* of a communication system is simplified spoken the connection line or space between transmitter and receiver. Two basic groups of channels can be distinguished: channels based on *free propagation* and those on *guided propagation*. Guided propagation channels are in most cases *telephone cable*, *coaxial cable* and *optical fibre* wave guides. Free propagation channels are *wireless broadcast*, *mobile radio* and *satellite* channels. Each of these channels in both groups can be parametrised in terms of *bandwidth* and *transmission*

*loss*. Additionally, the group of free propagation channels induces interfering *multipath* and *diffraction* propagation effects.

- The communication system's *Receiver* detect the incoming signal and *decodes* the original bits of information. The decoding process of the signal is *statistical* in nature, as the incoming signal is often interfered by a variety of physical effects. The major performance parameter of receivers is the sensitivity, counted as the SNR (Signal-to-Noise Ratio) at the receiver input as the ratio of the average received signal power to the average noise power, both measured at the same point. The SNR is often expressed in *decibels* (dB), defined as 10 times the logarithm (to base 10) of the power ratio. Finally, the receiver forwards an estimate of the message signal to the user of information.

The two primary resources in communication systems are *channel bandwidth* and *transmitted power*. Channel bandwidth in general is defined as the band of frequencies allocated for the transmission of the message signal and is measured in *Herz* (Hz). This means in turns, the broader the channel bandwidth, the more information can be transmitted at the same time. Transmitted power is the average power of the signal and determines the coverage of wireless communication systems (Haykin 2001).

The transmitted power $P_0$ of a communication system is a crucial parameter in wireless network positioning, as it defines implicitly the spatial availability of positioning in such systems. For the coverage in wireless systems a required minimum SNR$_r$ on the receiver side is necessary to establish a reliable communication link.
The SNR$_r$ in non-logarithmic form can be expressed as $P_r/N_0$, where $P_r$ denotes the average received signal power and $N_0$ the average noise power both counted on the receiver side. That points out, the lower the $N_0$, the internal noise flour of the receiver, the higher the SNR$_r$. Amongst others $N_0$ is a function of the receiver quality, which has to be considered at the consumer technology domain of WLAN devices (Carr 2000). Though, the most influencing value of the SNR$_r$ is the received power $P_r$. In wireless free propagation channels, the received signal power $P_r$ at a certain point of time is mainly a function of distance to the signal transmitter. This is expressed by the *Friis free-space equation*:

$$P_r = P_t G_t G_r \left(\frac{\lambda}{4\pi d}\right)^2 \quad [\text{W}] \tag{2.1}$$

$P_t$ denotes the transmitted power, $G_t$, $G_r$ the power gain of transmitting and receiving antenna and $\lambda$ the wave length, $\lambda = c_0/f$, of the transmitted carrier frequency $f$ and the speed of light $c_0$. For link budget calculation and ease of use, based on equation 2.1, the *path loss* $PL$, representing the signal *attenuation* between transmitter and receiver signal power in decibels, can be expressed as,

$$PL = 10\ log_{10}\left(\frac{P_t}{P_r}\right) \quad [\text{db}] \tag{2.2}$$

From the substitution of equation 2.1 into 2.2 follows,

$$PL = -10\ log_{10}(G_t G_r) + 10\ log_{10}\left(\frac{4\pi d}{\lambda}\right)^2 \quad [\text{db}] \tag{2.3}$$

The latter part of equation 2.3 is called the *free-space* loss, excluding the parametric influences of transmitting and receiving antenna. For the case of omni-directional antenna characteristic and constant carrier frequency $f$, these terms can be summarised as constant $C$. This leads to the most basic form of a propagation model, purely dependent on the distance $d$,

$$\begin{gathered} P_r(d) = 10\log_{10}(P_t) + 10\log_{10}(d)^2 + C \quad [\text{dbW}] \\ \text{where } C = 10\log_{10}\left(\frac{4\pi}{\lambda}\right)^2 - 10\ log_{10}(G_t G_r) \quad [\text{db}] \end{gathered} \tag{2.4}$$

Finally, the spatial coverage of a wireless communication system is mainly a concept of required minimum $\text{SNR}_r$ at the receiver. That parameter is directly connected with its inherent spatially dependent value of $P_r(d)$. The spatial distribution of the received power $P_r$ might be modelled on the basis of equation 2.4, between the communication system's transmitter and receiver station.

Since the spatial modelling of $P_r$ is a major concept of this thesis, lets have a closer look on interfering effects of the wireless communication channel.
In ideal multipath-free environments, a electro-magnetic wave propagates on the direct path from the transmitting to the receiving antenna. This is an ideal consideration which might be found under lab conditions. In real world however, each natural or artificial object, as buildings or vehicles, located between source and sink of wireless communication, reflect the electro-magnetic wave (Skolnik 1990). Figure 2.2 shows exemplarily the multipath effect in WLAN outdoor environments between an AP and MU (Mobile Unit).

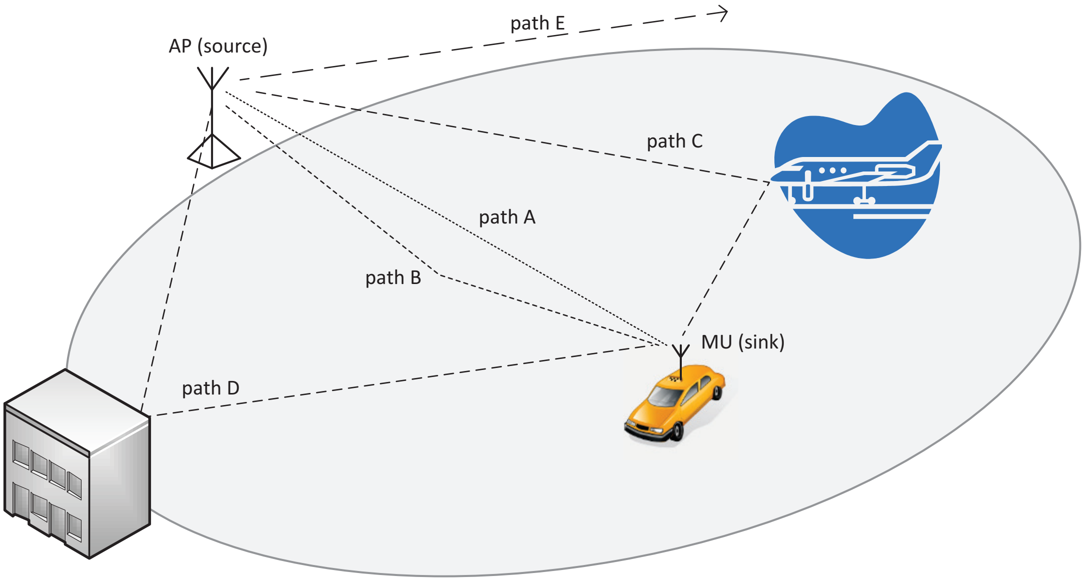


Figure 2.2.: Multipath effect in outdoor environments

Basically, figure 2.2 shows besides the direct *path A* only an exemplary subset of reflection options. Each line represents the direction of the *Poynting vector* of the electro-magnetic field, corresponding to the radiated sinusoidal plane wave. That vector points the direction of the propagating wave, while oscillating in magnitude and represents the rate of energy flow per unit area (Haykin 2001). In figure 2.2 the originally radiated signal is reflected by ground (*path B*), a surrounding building (*path D*) and an airplane (*path C*). A further part of the energy is even radiated in free space, without reflecting back to the signal sink. Thereby, each single signal path is attenuated differently. The signal attenuation is caused on the following physical considerations: (i) the travel distance of the electromagnetic wave, as shown in 2.4, and (ii) different material properties of the reflecting plane due to their corresponding dielectric constant (Skolnik 1990). Moreover, different surface materials induce different scattering effects at the reflecting plane's surface, as shown by (Il-Suek, Sewoong, Jae-Woo & Young Joong 2007). This in turns, attenuates the deflected signal further, as according to the law of energy conversation only a certain part of the scattered signal arrives at the receiver side.

The effect of multipath on the receiver side can be descriptively shown in the time domain. For that each sinusoidal signal path of the above introduced example is modelled in phase shift and amplitude. This can be done by the well known equation of a sinusoidal waveform, $y(t) = A\sin(2\pi f + \Phi)$, with the signal frequency $f$ and phase $\Phi$. For simulation of the introduced example[1] in figure 2.2 above, $f$ is selected to 2.4 GHz, according to the WLAN ISM (Industrial Scientific and Medical) band. The signal amplitude $A$ is normalised on the direct propagation amplitude of path A. Hence, $A$ of path B is due to reflection and extended travel distance weaker than the signal of path A. So does the amplitude of signal path C and D. The phase of signal path A, B, C and D depends on the signal's travel distance and wave length $\lambda$, counting 12.5cm at 2.4 GHz[2]. In this mind experiment, the phase is randomly selected to $\Phi_A$=0, $\Phi_B$=1/3$\pi$, $\Phi_C$=$\pi$ and $\Phi_D$=1/10$\pi$. At the receiver side all incoming signals will be accumulated to one resulting component in phase and amplitude.

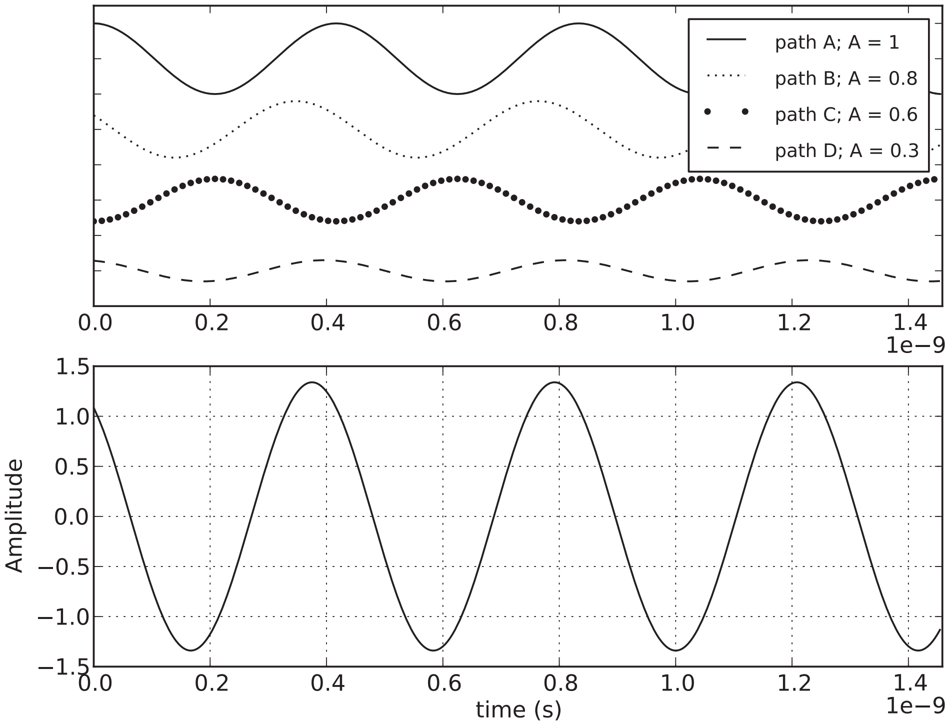


Figure 2.3.: Simulated multipath effect in an outdoor environment for sinusoidal signals

Figure 2.3 shows on the one hand the simulated signal paths A to D and on the other the resulting sum of all signal paths at the receiver side. At this parameter selection multipath has a constructive effect on the signal, as the resulting amplitude is higher than the direct path A's amplitude. However, in worst case the overall signal is eliminated completely due to adverse phase shifts in different signal paths.

The multipath propagation effect is influenced by different physical effects, like wave propagation mechanisms, reflection, diffraction and scattering. This makes it highly complex to model multipath effects of the communication channel.

[1] Please note that all parameter values of the simulation are selected corresponding to physical conditions, however do not base on empirical parameter research

[2] At laboratory conditions in vacuum, where the velocity corresponds to the speed of light $c_0$

The illumination of the basic model of communication systems has shown, the field strength of radiated electro-magnetic waves, carrying the information signal in WLAN communication, is a continuously distributed magnitude in space. The signal strength must reach a certain value above the receiver sensitivity threshold, in order to establish a reliable communication link. The area where this link is available, known as the concept of coverage, might be modelled straight forward under multipath free lab conditions. The effect of multipath, however constraints the way of deterministic electro-magnetic field modelling due to its inherent complexity.

### 2.1.2. Network Positioning Systems

The term *Positioning system* is introduced at this stage of the thesis as the angle of view on wireless communication systems is shifted. It turns from the pure electrical and physical point of view to a more positioning affine perspective on wireless networks.
Positioning is basically a rather general term and summarises the identification of an object's real-world geographic location. The position identification is either done actively by positioning of an technology device or passively by e.g. primary radar or ultrasonic technology (Skolnik 1990). Per se, positioning is not limited to certain technologies and might be even a solution of compounded positioning technologies (Groves 2008). A positioning system is an active position technology and consists of at least two spatially separated hardware components. The first location sensing unit measures received signals, radiated from the second signal transmitting component. Vossiek, Wiebking, Gulden, Weighardt & Hoffmann (2003) classifies local positioning systems in terms of their topology. Thereby, the main distinction is made between *self-* and *remote-positioning* systems. Whereas in self-positioning the location sensing unit is mobile, in remote-positioning systems the location sensing part is done by locally fixed receiver stations. Vossiek et al. (2003) introduces further *indirect* remote- and self-positioning approaches, where measured data will be transferred back to the opposite unit for the purpose of information, display or position computation.
This more general considerations of positioning systems will be regarded in the following section, in terms of their functional architecture, while focusing on networked systems.

From an architectural point of view, network positioning systems can be segmented into *signalling infrastructure*, *location sensing*, *position algorithm* and *position display & storage* components. Though, each component is not necessarily a closed hardware component. This is a more functional view on the system, where each component is neither fixed on a certain hardware or software component nor no its spatial distribution. Figure 2.4 depicts the functional architecture of a network positioning system in general. All these components are chained and deliver a certain signalling or information input to consecutive blocks. It covers the most general case, that multiple positioning technologies provide their measured variable to one position algorithm.

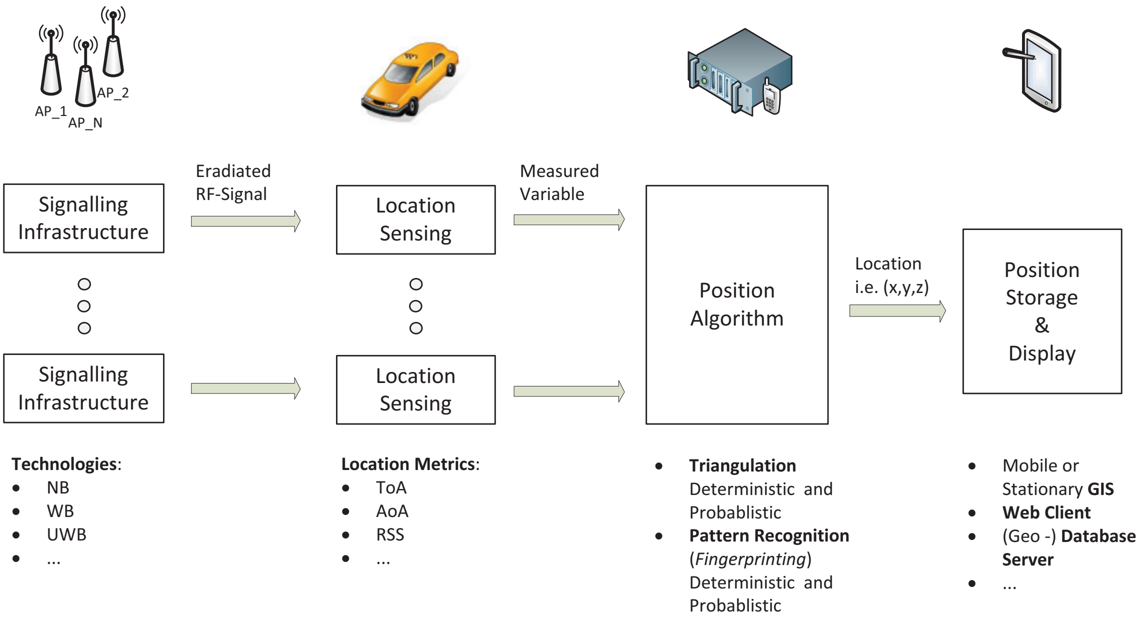


Figure 2.4.: Functional architecture of a network positioning system

Let's have a closer look on the function of each single block of that architecture.

### Signalling Infrastructure

The signalling infrastructure constitutes the skeleton of a positioning system. Thus, it defines the positioning technology and its inherent potential in system coverage, availability and precision. A well known example for such an infrastructure is the GPS (Global Positioning System) (Groves 2008), where the earth surrounding satellites establish the signalling infrastructure for positioning on earth, water and sky.
In wireless network positioning system however, the signalling infrastructure is established by WLAN AP or different technology dependent transceivers, surrounding the area of positioning. Basically, the positioning technologies in such systems can be divided into NB (Narrow Band), WB (Wide-Band) and UWB (Ultra Wide-Band) signalling systems (Pahlavan, Xinrong & Makela 2002). This classification is based on the occupied signal bandwidth, used for positioning related measurements. NB systems, as all GNSS (Global Navigation Satellite System) systems, measure mainly the phase difference between transmitted and received signal for ToA (Time of Arrival) computation. They occupy around 20MHz in bandwidth.WB systems apply the same principle of ToA measurements, however under utilisation of more bandwidth, around 150MHz, and complex modulation techniques as for instance DSSS (Direct-Sequence Spread Spectrum) (Pahlavan et al. 2002) or FMCW (Frequency-Modulated Continuoues Wave) modulation (Roehr, Gulden & Vossiek 2007) more robust signalling is achieved for use in multipath environments. UWB techniques exploit up to 1GHz in bandwidth and allow the most precise measurement in ToA positioning. Due to its robustness against multipath its the ideal positioning technique

for indoor environments. In brief, the higher the signal's bandwidth the more precise the signal timing, as shown by (Waldmann, Weigel & Gulden 2008), and therefore at the end of the day the positioning.

Although on market are many network based positioning technologies available, the decision for a certain technology is often a trade-off between position accuracy, deployment costs and already available infrastructure.

### Location Sensing

The location sensing part of the system receives first of all the radiated RF-signal from the signalling infrastructure. Depending on the inherent position system's metric it extracts the input variables for the positioning algorithm. In the case of self-positioning systems the location sensing is part of the MU, as depicted in figure 2.4. In remote-positioning systems it is vice-versa.

The location metrics can be classified into three different measurement principles: propagation based, AoA (Angle of Arraival) and RSS (Received Signal Stregth). Propagation based systems can be further sub-classified into ToA, RToF (Round-trip Time-of-Flight) and TDoA (Time Difference of Arrival) systems. While in ToA and RToF systems the travel time of the signal is used for range measurements, in so called TDoA systems the metric is based on the difference of signal travel time between MU and APs (Vossiek et al. 2003). The measured variable includes accordingly either travel times or differential times. In AoA systems the relative angle of bearing between two stations is measured. This is usually achieved by using directional antennas or antenna arrays (Niculescu & Badri 2003). In that method the bearing angle between MU and the surrounding infrastructure stations constitutes the measured variable. Positioning in RSS systems is based on the propagation loss of signals radiated by each AP. The MU simply measures the signal strength to each detectable AP and forwards that information to the positioning algorithm.

For AoA or propagation based sending metrics, highly specified technology is necessary to enable accurate angle and time measurements in wireless networking infrastructures. For these positioning method existing standardised off-the-shelf technology is not yet available. Amongst all metrics, RSS is the easiest adaptable to already existing WLAN communication infrastructures, as it does neither claim additional hardware implementation nor acquisition effort. This is because most off-the-shelf WLAN communication interfaces support power measurement per se.

### Position Algorithm

The position algorithm constitutes the core of a positioning system. It receives all measured variables of the location sensing units and processes the position solution, which could be forwarded to consecutive system components. In networked infrastructures the position algorithm can be either located together with the location sensing component on the same hardware device or on a separate hardware component within the network.

*Triangulation* is used for position computation of AoA and propagation based metrics. The *deterministic* triangulation approach is based on geometric triangulation methods, where the signal geometry and the spatial constellation of navigation transmitters around the MU, plays a primary role in position accuracy (Mansfeld 2004). Due to estimation of errors in inaccurate ToA measurements, the deterministic triangulation techniques provide a spatial uncertainty of the position solution, known as DOP (Dilution of Precision) concept, rather than a fixed position. To overcome that issue, a variety of *probabilistic* position algorithms have been developed, providing an estimate of location coordinates as fix position (Pahlavan et al. 2002). RSS metrics use pattern recognition techniques, also known under the concept of *fingerprinting*, for position determination. For that technique a priori measurements of the electromagnetic field distribution for each AP has to be applied in the area of interest. On that "field-map" spatial correlation to the measured field-strength values is achieved by the deterministic nearest neighbour approach (Honkavirta et al. 2009), yielding into the position solution. For enhancing the position solution, more complex probabilistic methods using Bayesian filtering implemented as Particle filter (Widyawan, Klepal & Pesch 2007) for fixing the position.

In RSS metrics the choice of the position algorithm does considerably influence the accuracy of the later position solution. In challenging areas regarding interference effects on the communication channel, e.g. by multipath influences, the position algorithm has to compensate these. While in outdoor environments deterministic approaches often fulfils that purpose, indoor environments claim for complex probabilistic algorithm models to meet strong requirements in position accuracy.

**Position Display and Storage**

The position display unit constitutes the HMI (Human Machine Interface) of a positioning system. It directly receives the finally processed geographical position as coordinate tuple from the position algorithm or an intermediate storage component. Display and storage component can be physically divided onto different network entities or merged on the same machine or software system. In some cases no display unit exists, if positions are exclusively used for process based analysis, as for instance in the case of package tracking systems in logistics.

Position storage systems can be different in their characteristics. Often, in order to achieve networking flexibility, the storage system is a web-server database combination. For security purpose, standardised security services of the web-server can be utilised. As database a conventional DBMS (Database Management System) can be utilised, with or without geo-spatial extension, as for instance *PostGIS* or *Oracle Spatial*. This might be utilised as well as basis for a distributed Web GIS (Geographic Information System) solution on top of the DBMS server. The display component in networked environments could be a light weight *thin-client* browser based web-application, retrieving positions regularly from a DBMS or application server, as shown by Fu & Sun (2010). Either it can be designed as a *thick-client*, where positions are directly stored underneath the display client application. In both cases, thin- and thick-client applications, the physical display device can be shaped as mobile or stationary platform. Another possibility for displaying positions is the direct use of a GIS, where tracked positions can be displayed and analysed in the time and space domain (Press & Environmental Systems Research 2004). The GIS can be either shaped as client/server or standalone and desktop or mobile application.

Although, GIS is moving towards the field of real-time data analysis and display, the impression of the author is, it has not yet achieved the break through in this domain. There is high potential in the application of spatio-temporal analysis capability, on conventional positioning and tracking systems (Miller & Bridwell 2008).

The above highlighted considerations on network positioning systems have given an overview on a bunch of positioning technologies and their generic functional architecture. Nonetheless, many of them are not yet available as standardised off-the-shelf products. Closed environments envisaged for positioning of devices, vehicles or even people, often already apply available communication networks of the IEEE 802.11x standard family, generally known as WLAN; this is for example the case in airport environments (Bretz et al. 2011). In such environments, positioning technologies based on WLAN is preferred, as it sustainably saves installation and investment effort on the positioning infrastructure as well as the location sensing side.

## 2.2. Methods for Radio Map Generation

A centralised object in WLAN positioning by fingerprinting constitutes the RM (Radio Map). The RM holds the information of electro-magnetic field strength distribution over the entire area, where positioning should be made available. On the basis of RM information, the positioning algorithm computes the actual position of active MUs by correlation of measured field strength of the sensor and modelled field strength in the RM. Thus, the precision of the radio map model is essential for the accuracy in positioning.

This subsection gives first a few basic definitions about the constitution and mathematical description of the RM model. Then, it goes over to field modelling approaches. These are initially based on deterministic field modelling and empirical field measurements and later on deterministic and probabilistic interpolation methods. Deterministic modelling starts with the fairly simple approach of radial field modelling and goes over to the ray tracing, multi wall model and dominant path model. On the interpolation side, the deterministic inverse distance weight and spline methods will be covered. For probabilistic interpolation Kriging methods will be shortly revised.

### 2.2.1. Definition of the Radio Map

The RM model constitutes the groundwork of RSS positioning and builds the foundation of the entire RM generation process. Though, first of all it has to be defined in its mathematical dimension. That RM model is independent of field modelling approaches, discussed later on in this section.

Basically the RM describes the representation of a continuous field phenomenon, the electromagnetic field strength. Surrounding APs of the signalling infrastructure radiate that signals. Each AP generate its own electrical field, thus multiple radio maps exists for a single positioning

infrastructure. The continuous field model can be either represented by a set of mathematical functions or a discrete representation, sampled from the continuous field phenomenon. The latter one can be stored in a matrix of spatially regularly distributed values. Thereby, each cell of the matrix refers to a geographical location. For the ease of use in terms of the expected field complexity, map manipulation and fast value accessibility the discrete model is selected for further considerations.
Mathematically, each sampling point $x_{ij}$ and thus matrix element is defined as a tuple:

$$x_{ij} = (rss_{ij}, \theta_{ij}) \tag{2.5}$$

where $rss_{ij}$ is the radio signal strength and $\theta_{ij}$ represents a universal parameter for additional use during the location estimation phase. This could be for instances used for the vehicles orientation in north, east, south or west direction, influencing the radio signal strength in fingerprinting as stated by Dempster, Binghao & Quader (2008). The sample point $x_{ij}$ is now regarded as vector

$$\vec{x}_{ij} = (rss^{k}_{ij}, \theta^{k}_{ij}) \quad k \in 1 \ldots q \tag{2.6}$$

where each element $k$ of the vector $\vec{x}_{ij}$ corresponds to the radio field of a dedicated access point AP$_k$ at a maximum count of $q$ APs. All that vectors are part of the radio map, defined by the matrix $X$:

$$X = (\vec{x}_{ij}) \quad i \in 1 \ldots m,\ j \in 1 \ldots n \tag{2.7}$$

Thereby, each element of matrix $X$ represents a sample of the $q$-dimensional radio map and refers to a geographical position by row $i$ and column count $j$ respectively. Note that each sample point $\vec{x}_{ij}$ is generated at the centre of a matrix cell in $X$. Figure 2.5 points out the multi-dimensional character of the RM matrix $X$.

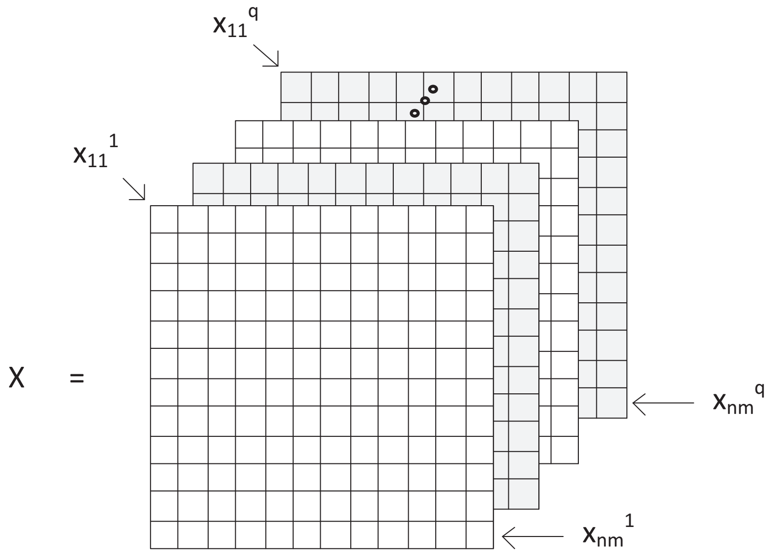


Figure 2.5.: Mathematical definition of $q$-dimensional radio map

Each layer in that matrix represents a radio map dedicated to an AP. Even more than one map layer per AP is feasible, in the case if parameter $\theta$ is used for the MUs' orientation.

Finally, after this pure mathematical definition of the radio map, it might be also regarded as a coverage of the electro-magnetic field strength in space. Each coverage cell has got its geographical coordinate, in its defined geographic reference system, with certain spatial extent. That spatial extent of the raster cell is generally known under the term coverage resolution, often measured in meter. This resolution implicitly defines the sampling width of the discretisation process of the continuous field phenomena - the radio signal strength in space. The following subsection gives an overview on possible modelling processes of the continuous field phenomenon.

### 2.2.2. Field Modelling Approaches

In order to establish a (radio) map of a continuous field phenomenon, the modelling process of the field have to be conducted first. For that a bunch of different approaches exist. In scope of this thesis, an overview on a few approaches, identified in the literature (Honkavirta et al. 2009), (Parodi et al. 2006), (Li et al. 2005) and (Sen et al. 2008) as most relevant for radio map modelling, will be given.

For the purpose of structure, the identified modelling methods are first of all classified into their basic mathematical description: *deterministic* and *probabilistic*. A graphical outline of that classification is depicted in figure 2.6.

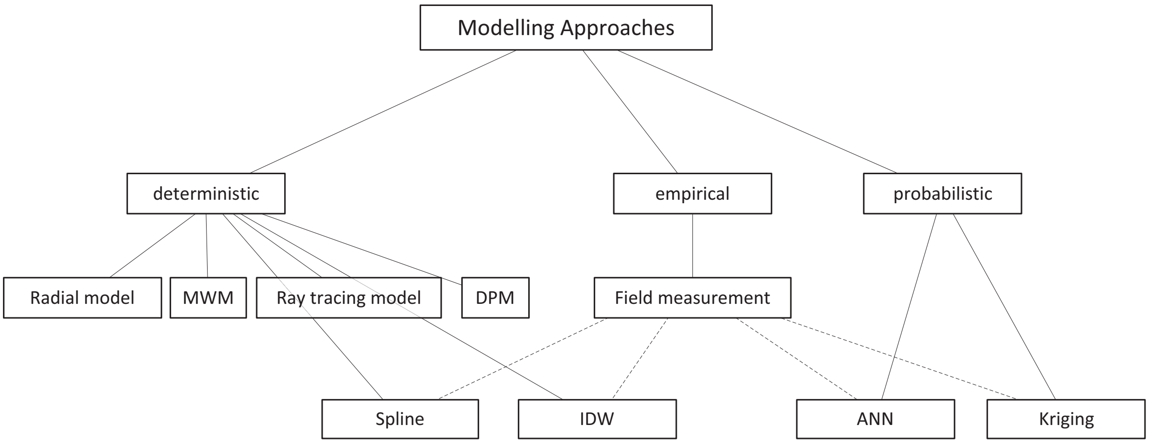


Figure 2.6.: Radio map modelling approaches

The deterministic branch covers the *radio* model, MWM (Multi Wall Model), *ray tracing* and DPM (Direct Path Model) model. The deterministic interpolation methods *spline* and IDW (Inverse Distance Weight) in the context of continuous field modelling are regarded as *semi-deterministic* modelling approaches; since these methods are based on empirical a priori field measurement samples. The probabilistic branch covers ANN (Artificial Neural Network)s and the *Kriging* interpolation method, whereas ANNs are not regarded further in scope of this thesis. While in deterministic models all variables are exclusively dependent on uniquely determined parameters, in probabilistic models the parameters are described by probability distributions. Though, basically, the ANN and Kriging based models are as well regarded as semi-probabilistic modelling approaches as they are again based on field measurements.
Let's discover in more detail the currently regarded modelling approaches.

### Radial Model

The radial model is the most straight forward field modelling approach based on parameter. The radial symmetry in electro-magnetic field distribution, as shown in subsection 2.1.1 Principles and Concepts, is utilised to model the continuous field between APs and MU. For that purpose equation 2.4 is taken to compute the field distribution. For the case that the transmitted power $P_t$ of the access points is not known, Parodi et al. (2006) suggests for initialisation each AP measures the RSS to $N-1$ neighbouring APs and determine $P_t$ by least square fit. In the case of position reference points, this could be even done directly by MUs' RSS measurements.
In positioning environments highly influenced by multipath effects, as for instance indoor environments, the radial model is very approximate, compared to the real field distribution (Parodi et al. 2006).

### MWM Model

The MWM considers the influence of walls to the RSS on the direct path between AP and MU. Basically, that approach is based on the radial model equation 2.4:

$$P_r(d) = 10\log_{10}(P_t) + 10\log_{10}(d)^2 + C + \sum WAF \quad [\mathrm{dbW}] \tag{2.8}$$

WAF (Wall Attenuation Factor) is a parameter dependent on physical characteristics of walls, intersecting on the direct propagation path. This parameter is dependent on the thickness of walls and its dielectric properties. Furthermore, the orientation of walls in reference to the direct propagation path plays a crucial role in the MWM, as it implicitly defines the path's length through the wall. All that parameter influences have to be considered in the WAF. The summation symbol in equation 2.8 formulates the possibility that more than one wall might be intersected on the direct propagation path. In order to determine the WAF parameter in indoor environments, a BIM (Building Information Model) could be utilised to obtain orientation, physical characteristics and thickness. For further information on BIMs please see (Schlueter & Thesseling 2009). The main drawback of the MWM is that it only considers the direct path between AP and the point of interest on the map. This path does not necessarily correspond to the strongest influence of field strength at the point of interest.

**Ray Tracing Model**

The ray tracing model is based on ray-optical propagation modelling. In that approach all possibilities of optical ray propagation between AP and MU are taken into account. Basically, this model is in accordance with multipath signal considerations discussed in subsection 2.1.1 Principles and Concepts. In 3D environments, each propagation path reflected by ground, walls, natural or artificial obstacles have to be considered. As shown by Il-Suek et al. (2007) each reflection causes different ray scattering and diffraction, dependent on physical material constants and surface roughness. Additionally, the ray tracing model considers signal attenuation through walls of the MWM. Although, nowadays 3D vector models of buildings in the context of BIMs provide high accuracy, they often lack in surface material definition, leading to significant errors in optical ray-modelling (Wölfle, Wahl, Wertz, Wildbolz & Landstorfer 2005). For instance at a frequency of 5GHz at the upper ISM band for WLAN communication, a wavelength of 6cm is given. This would claim a BIM accuracy in the sub-centimetre range, to provide reasonable RSS model accuracy.
All that considerations result consequently in massive modelling and processing effort. To overcome the computational burdens of the ray tracing model, the DPM will be introduced next.

**DPM Model**

The DPM lies between both models, the MWM and ray tracing model. Wölfle et al. (2005) states that in most cases 2 or 3 dominant rays of the ray tracing model are contributing to more than 95% of the overall energy, at a dedicated point on the radio map. The DPM meets the requirements of a radio map model, that is independent of each mirco detail of an underlaying BIM, while focusing on dominant propagation paths only, to reduce computational effort. For

determination of the dominant path between AP and MU located in different rooms, Wölfle & Landstorfer (1998) presents a algorithm, utilising a building's inherent topology of neighbouring rooms, walls and other building elements. The algorithm determins the dominant path by utilising an a-priori processed room-structure tree. This tree might be dereived from a BIM. Based on that tree, out of the neighouring room information the dominanat path is given. In the case that AP and MU are located in the same room, though hided around a corner where no LoS (Line of Sight) condition is given, a topology tree of convex corners of a room helps to finde the dominant path. This algorithm for determination of dominant paths can lead to more than one solution. Finally, the solution with lowest attenuation along the path is selected.
The approach of DPM is mainly considered for indoor field modelling. Though, this approach could be applied for outdoor positioning environments, where several natural and artificial obstacles are located within or around the positioning field.

A comparative summary of previous discussed modelling techniques is given in figure 2.7. This should be considered as conceptual sketch only, as the geometrical proportions of propagation paths are methodologically chosen.

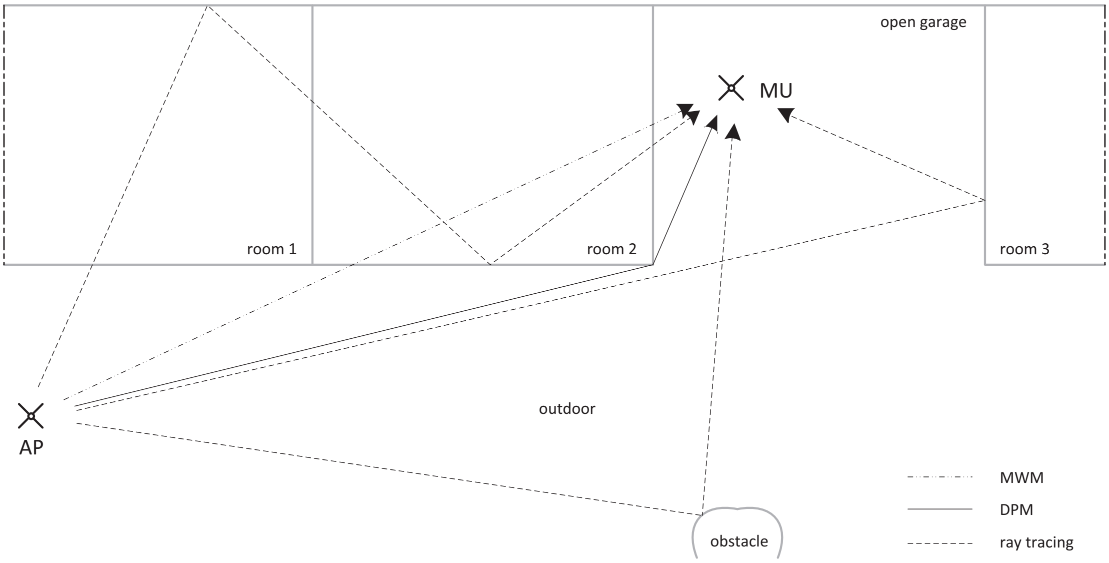


Figure 2.7.: Map modelling approaches based on BIM information

For a clear arrangement of the concepts behind that models the techniques are depicted in 2D space. Though, all methods are assignable to 3D space. Where the ray tracing model takes all reflecting propagation paths into accout, the DPM focuses on the dominant path component. The MWM exclusively considers the direct path between AP and MU. If there is a LoS between AP and MU, all WAF in equation 2.8 are zero and the MWM corresponds to the radial model.

The *semi-deterministic* modelling approaches are based on empirical a-priori measurements during the so called training phase in wireless network positioning. This phase could be seen as a system calibration phase of the radio map generation process. For that, geographically known RP (Reference Points) should be available for measuring the calibration samples. Out of that samples, measurement values can be processed for each RP, constituting the basis for interpolation.
Deterministic interpolation methods can be considered as two different groups: *local* and *global* interpolation techniques. Whereas local methods only consider a certain amount of measurement points in the neighbourhood of the prediction point, global methods take the entire data set into account. Furthermore, deterministic interpolation techniques can be either *exact*, where the resulting surface passes through the measurement points, or *approximative*, where points on the interpolation surface are different to the data set.

### IDW

The IDW interpolation method is exact and points of the interpolation surface are determined on the extent of similarity. IDW is based on the assumption, things that are close together are more alike than others far apart. This is achieved by a distance weighted interpolation function:

$$\hat{Z}(x_0) = \sum_{i=1}^{N} \omega_i(x_0) Z(x_i) \tag{2.9}$$

$\hat{Z}(x_0)$ is the predicted value of the interpolation surface for the location $x_0$. For the predicted value, $N$ measurement points $x_i$ of the data set $Z$ are considered. $\omega_i$ is the weight function of the predicted values defined as following:

$$\omega_i(x_0) = d^{-p}(x_0, x_i) \,/\, \sum_{i=1}^{N} d^{-p}(x_0, x_i), \qquad \text{where} \quad \sum_{i=1}^{N} \omega_i(x_0) = 1 \tag{2.10}$$

$d(x_0, x)$ is the geometrical distance between interpolation value and measurement value. The sum of $N$ weight factors needed for computation of $\hat{Z}(x_0)$ is equal to one due to the normalisation of $\omega_i(x_0)$. The power $p$ of the distance influences the weight of neighbouring measurement to the predicted value. The higher $p$, the less the influence of measurement values in far distance to the predicted value $x_0$. The optimal $p$ value is found at the minimum of the root-mean-square prediction value, that is calculated from cross-validation (Johnston & Institute 2004).

### Spline

Spline is another interpolation technique. It is assigned to the group of radial basis functions and is an exact interpolation technique, where the prediction surface must include each measurement value. Spline uses mathematical functions that minimise the overall curvature of the surface. This functions are based on the assumption, that the approximation surface should pass on the one hand data points and on the other be as smooth as possible (Mitasova, Mitas, Brown, Gerdes, Kosinovsky & Baker 1995). The term spline comes originally from the flexible spline devices used by shipbuilders to draw smooth shapes.

Basically, there are two spline methods: *regularized* and *tension*. In the regularized technique values of the approximation surface are smooth and gradually changing and may be outside the interval of the sample data range. The tension method generates a less smooth surface, where the stiffness is controlled according to the physical character of the modelled phenomenon.

The basic problem can be stated in $d$-dimensional space as follow:

$$S(x^{(j)}) = z^{(j)},\ j = 1, \ldots, N. \tag{2.11}$$

The function $S(x)$ of the interpolation surface at $N$ discrete measured points $x^{(j)} = (x_1^{(j)}, x_2^{(j)}, \ldots, x_d^{(j)})$ is equal to the studied phenomenon $z^{(j)}$. Now a interpolation function $S(x)$ has to be found that fulfils the condition

$$\|S(x)\| = \quad \text{minimum}. \tag{2.12}$$

The unique solution of that problem is stated in (Mitasova & Mitas 1988) by

$$S(x) = T(x) + \sum_{j=1}^{N} \lambda_j R(x, x^{(j)}), \tag{2.13}$$

where

$$T(x) = \sum_{l=1}^{M} a_l f_l(x)) \tag{2.14}$$

and $f_l(x)$ is a set of linearly independent functions which have zero SS (Smooth Seminorm) so that

$$\|f_l(x)\| = 0,\ l = 1, \ldots, M. \tag{2.15}$$

The *trend* function $T(x)$ is dependent on the option selected for spline interpolation. The generating function GF (Generating Function) $R(x, y)$ for the general d-dimensional case is given by

$$R(x, x^j) = \sum_{\alpha} \frac{g_\alpha^*(x) g_\alpha^*(x^j)}{\|g_\alpha\|^2} \tag{2.16}$$

where

$$\|a\| = \sum_{m=1}^{d} \alpha_m, \tag{2.17}$$

denoting a multiindex $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d)$ with non-negative integer components. The function $g(x)$ is given, where the constraint of smoothness is applied to the interpolation. The coefficients $\lambda$ can be found by the solution of a linear set of equations.

This general formulation of the interpolation problem is now adjusted to the above introduced techniques of regularized spline and spline with tension in the two dimensional space.
For the regularized option the trend function is

$$T(x) = a_1 + a_2 x_1 + a_3 x_2, \tag{2.18}$$

where the coefficients a can be found by the solution of the linear set of equations.

$$R(r) = \frac{1}{2\pi} \left\{ \frac{r^2}{4} \left[ ln\left(\frac{r}{2\tau}\right) + c_E - 1 \right] + \tau^2 \left[ K_0\left(\frac{r}{\tau}\right) + c_E + ln\left(\frac{r}{2\pi}\right) \right] \right\},\ d = 2 \tag{2.19}$$

$r$ is the distance from the point $x$ to $x^j$, defined over $r = [(x_1 - x_1^j)^2 + (x_2 - x_2^j)^2)]^{\frac{1}{2}}$ for $d = 2$. The weight parameter $\tau$ defines the weight of the third derivatives of the surface $g(x)$ in the curvature minimization expression. The third derivative of $g(x)$ is a measure of smoothness, implying small weight changes to the interpolation function in close neighbourhood of data points only. $K_0$ is the modified Bessel function of zeroth order and $c_E = 0.577215...$ is the Euler constant.
For the tension option the trend function is given by

$$T(x) = a_1 \tag{2.20}$$

and

$$R(r) = -\frac{1}{2\pi\varphi^2} \left[ ln\left(\frac{r\varphi}{2}\right) + c_E + K_0(r\varphi) \right],\ d = 2. \tag{2.21}$$

In spline with tension, $\varphi$ is the weight parameter for tension. This weight parameter determines the influence of the first derivative of $g(x)$ in the surface minimization. The parameter $\varphi$ influences the character of the interpolation from a simple spline ($\varphi \to 0$) to a rubber-sheet surface($\varphi \to \infty$).

For sufficiently large data sets, spline interpolation allows segment based processing (Mitasova & Mitas 1993). This claims an interpolation function with local behaviour. In global spline interpolation the computational effort is proportional to $N^3$ where $N$ is the maximum count of interpolation values. Mitasova & Mitas (1993) propose an algorithm that reduce the computational effort to $N$ by segment based interpolation.

**Kriging**

Kriging is a geostatistical interpolation method producing an interpolation surface, incorporating the statistical properties of the measured data. Besides the resulting prediction surface, the Kriging techniques produce an error surface, indicating the statistical residuals of the predictions. The Kriging process is divided into two main tasks, the quantification of spatial structure of data and producing of a prediction. In the structural quantification, known as variography, a spatially dependent model is fitted to the data. For the prediction of unknown values at specific location, Kriging utilises that spatial model, the spatial data configuration and the values of measured data around the prediction location. The configuration describes spatial autocorrelation amongst

measured points around the prediction location. Dependent on the fitted model, Kriging is either an exact or inexact interpolator. If the regression model in variography starts at the origin of the coordinate system the Kriging prediction is exact. That means the error in measurement data is zero.
The basic idea of Kriging is expressed according to Isaaks & Srivastava (1990) by a simple mathematical formular:

$$Z(s) = \mu(s) + \epsilon(s) \tag{2.22}$$

The variable of interest $Z$ at location $s$ represents the prediction surface. It is composed of a deterministic trend $\mu(s)$ and random, autocorrelated errors from $\epsilon(s)$ (Johnston & Institute 2004). The trend $\mu$ will not predict perfectly. For that case the assumption is made that $\epsilon$ is randomly distributed and expected to be zero in average. Further, the autocorrelation between $\epsilon(s)$ and $\epsilon(s + \vec{h})$ is not dependent on the actual location $s$ but on the vector shift $h$ in a certain distance and direction.

Basically, Kriging distinguishes different methods where the most prominent and relevant for continuous field modelling will be shortly revised:

- In *Ordinary Kriging* $\mu(s)$ is a unknown constant $m$. It is hypothetically constant over the entire observation area and therefore sensitive against trends in the sample data set.
- In *Simple Kriging* $\mu(s)$ is again constant, however the assumption is made that $m$ is a known parameter. Thus, the values of $\epsilon(s)$ are also exactly known. This allows better estimation of autocorrelation in $\epsilon(s)$. $\mu(s)$ might be based on a physical model were the trend is known.
- In *Universal Kriging* the trend variable $\mu(s)$ can be arbitrarily modelled. If in explorative data analysis a global trend is found e.g. by regression, this trend can be eliminated from the measurement data set. After removing the trend, conceptually the autocorrelation is now modelled from the random errors $\epsilon(s)$.
- *Co-Kriging* is a multi-variant extension of the revised Kriging methods, where several variables are taken into account. Thereby, the autocorrelation of the main variable $Z_1$ is regarded as well as cross-correlations between $Z_1$ and all other variables $Z_n$ in order to achieve better predictions.

The more theoretical and mathematical background on Kriging is now shown on the example of Ordinary Kriging. In literature, Ordinary Kriging is associated with the acronym B.L.U.E for "best linear unbiased estimator"(Isaaks & Srivastava 1990). It is "best" because it aims at minimizing $\tilde{\sigma}_R^2$, the variance of errors. The process is linear because its estimates are weight linear combination of the available measurement data. "Unbiased" since the mean residual or error is tried to have equal to 0. In Ordinary Kriging the estimated value $\hat{Z}(s)$ is a weight linear combination of the random variables $Z(s)$ at the available measurement locations:

$$\begin{aligned}\hat{Z}(s_0) &= \sum_{i=1}^{n} \omega_i Z(s_i) \\ \text{where } &\sum_{i=1}^{n} \omega_i = 1,\end{aligned} \tag{2.23}$$

The fact that the sum of $\omega$ is equal to 1 is based on the constrained of the unbiasedness condition, that the sum of $\epsilon(s)$ is equal to 0. For any solution of estimated locations, the model is a stationary random function, implying that the separation of random variables depends only on the distance and not on their location. As we do not know the constant $\mu$ in this approach and thus the estimator of $\epsilon$, the original problem is transferred into a corresponding model problem. This is the minimization of the modelled error variance $\tilde{\sigma}_R^2$ (Isaaks & Srivastava 1990).
In Kriging, the minimum estimation error is produced in order to determine the optimal weights. Minimizing the function of the error variance, as shown in detail by (Isaaks & Srivastava 1990), yield in the following equation:

$$\sum_{j=1}^{n} \omega_j \gamma_{ij} + \lambda = \gamma_{i0} \quad \forall\ i = 1, \dots, n, \tag{2.24}$$

where $\gamma_{ij}$ expresses the estimated covariation between the measurement data set of the interpolation and $\gamma_{i0}$ the estimated covariation between prediction value $\hat{z}(s)$ and measurement value $z(s)$. The Lagrange parameter $\lambda$ is used to obtain an unconstrained minimization problem of $\tilde{\sigma}_R^2$, that the sum of weights $\omega_i$ is equal to 1. Equation 2.24 shows that the weight $\omega$ is a function of the spatial correlation amongst the neighbouring measurement values of the predicted value, as well as the correlation between measurement values and the predicted value. This considers that redundant information of neighbouring measurements, known as measurement clusters, is eliminated by giving less weight in the estimation.
Equations 2.24 can be written as a system of equations in matrix notation as

$$\begin{gathered}\Gamma \cdot \Omega = D \\ \begin{pmatrix} \gamma_{11} & \cdots & \gamma_{1n} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma_{n1} & \cdots & \gamma_{nn} & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \omega_1 \\ \vdots \\ \omega_n \\ \lambda \end{pmatrix} = \begin{pmatrix} \gamma_{10} \\ \vdots \\ \gamma_{n0} \\ 1 \end{pmatrix}\end{gathered} \tag{2.25}$$

By the rules of linear algebra the system can be solved for the weights $\omega$:

$$\Omega = \Gamma^{-1} \cdot D \tag{2.26}$$

The correlation values $\gamma$ are known estimators that are modelled by a semivariogram function. The semivariogram model is determined during the statistical process of EDA (Exploratory Data

Analysis). According to the first law of geography from Waldo Tobler, that near things are more related than distant things, the variance of data values increases with distance. This statistical variances $E[(Z_i - Z_j)^2]$ across the measurement samples will be constructed in the empirical semivariogram. Analytical regression of the empirical data delivers the theoretical semivariogram, modelling the estimated variance over the geometrical distance between considered points in the test area. That functional model is the basis for the computation of all covariances $\gamma$ in the Kriging process.
A further strength of Kriging is that every prediction on the interpolation surface has a corresponding Kriging standard deviation:

$$\sigma_R^2 = \sum_{i=1}^{n} \omega_i \gamma_{i0} + \lambda \tag{2.27}$$

Kriging is a method to model spatial distributions by statistical distance rather than geometrical distance e.g. as kown from IDW interpolation. The inherent statistical process of Kriging allows de-clustering of the available dataset in terms of information redundancy. The utilisation of a model of spatial continuity, describing the statistical distance of the spatial data set configuration, allows flexibility during the process of EDA. That yields into a flexible estimation procedure to a qualitative prediction surface of the observed phenomenon.

This section has given the basic mathematical definition of a multi dimensional RM. A variety of deterministic and probabilistic methods have been highlighted for continuous field modelling based either on parameter models or empirical measurement.

## Conclusions

The illuminated principles and concepts of this chapter have given a first introductory overview on the wide topic of electro-magnetic field modelling. Although it is not the core of this thesis and does not correspond to basic research questions, the revision of wireless network position techniques has pointed out the advantages of RSS positioning against others. For given WLAN infrastructure with present multipath initiating obstacles, in the area of low cost off-the-shelf technology, RSS positioning is still the only reasonable solution. However, this positioning approach needs a carefully modelled RM, which will be further investigated in this thesis. For that modelling, many approaches have been found in literature. In conclusion of this section either an extended parametric radial field model or the probabilistic Kriging interpolation approach seems to be the most promising to follow up in the on-going research. As they are global modelling techniques, they could be best suitable as a solid groundwork for coverage mapping of electromagnetic field phenomena. However, first the available data sources, their modelling and analysis will be considered in the upcoming chapter.

# 3. Data Sources, Modelling, and Analysis

The term "data" plays a dominant role in GIS (Geographic Information System) and constitutes the foundation of any spatial analysis, statistics or display of geographically related information. This is supported by Goodchild (1992*b*)'s originally proposed GIS research agenda, where data collection and measurement, data capture, data modelling and theories of spatial data and data structures, algorithms, and processes are part of. The fact that four of eight agenda points take care about data issues, points out the significance of data within the research framework of GIS. The availability of data in spatial coverage, quality, and distribution in the time-space domain plays a crucial role, for further on applied modelling strategies and methods. Therefore, the following chapter is exclusively dedicated to data sources, modelling, and analysis.
The first data source section is divided into static infrastructure and dynamic measurement data. It figures out the origin of different data sets, their inherent structure and necessary pre-processing for the following steps of data modelling and integration. In data modelling a object relational data structure is proposed, for seamless integration into a database. This constitutes the groundwork for the integration of data; along the way intensive processing for the purpose of data interpolation and correlation will be reflected. This chapter ends by spatio-temporal analysis of dynamic data sets and propose filter mechanisms for extraction of measurement reference values.

## 3.1. Data Sources

All data sources used for field modelling in scope of this thesis, have been provided out of the project context of SESAAM (Geo-Spatially Enhanced Situational Awareness for Airport Management) (Bretz et al. 2011), hosted by Z_GIS University of Salzburg. In this project all data sources for the purpose of WLAN (Wireless Local Area Network) field modelling are spatially limited to a testbed area, spatially delimited by the main apron area of the Airport Salzburg. The available data have different characteristics in dimension, space and time scale, accuracy and their origin or data source. This section gives an overview on available spatial and non-spatial data for the purpose of geographical orientation and field modelling, around and inside of the testbed area.
In order to obtain a clear arrangement of data and corresponding sources this section is divided into two main parts: static infrastructural data and dynamic measurement data. While the first one deals with vectorised infrastructural GIS data and WLAN AP (Access Point) parameters of

the positioning infrastructure, the latter cares about measurement data of the electro-magnetic field phenomena.

### 3.1.1. Static Infrastructure Data

Static infrastructure data covers on the one hand spatial GIS vector data sets and on the other positioning infrastructure specific non-spatial parameters. In this thesis project, static data is defined as data that vary neither in spatial nor in time domain. The use case diagram in figure 3.1 shows the available data sets and their data provider or data source respectively.

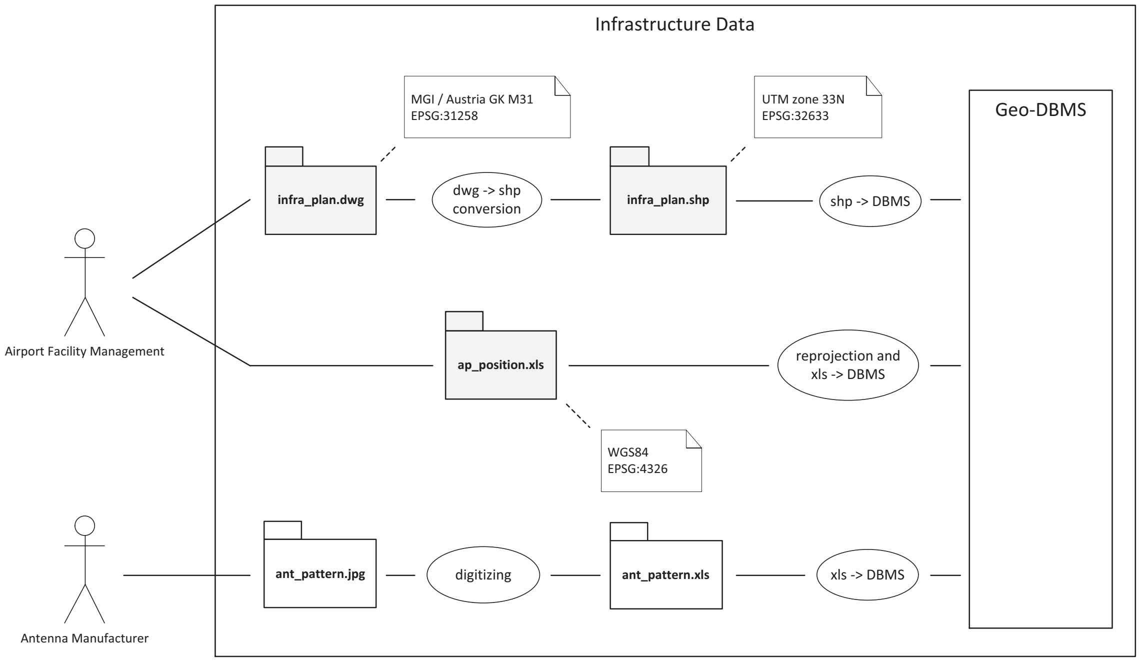


Figure 3.1.: Use case diagram for infrastructural data; boxes coloured in light grey indicate spatial data sets

The airport facility management provides the infrastructure plan in DWG (Drawing) file format, to the SESAAM project. Basically, the DWG file format has its origin in the CAD (Computer Aided Design) domain, where it acts as the standard exchange format for the AutoCAD suite and was licensed by Autodesk inc. in 1982. The DWG format is capable to store two and three dimensional drawings and metadata (Gopi 2010). The `infra_plan.dwg` file contains two dimensional polyline vector data, over the entire airport infrastructure and is geo-referenced to the Cartesian MGI (Military Geographic Institute) Austria BMN (Bundesmeldenetz) M31 Gauß Krüger coordinate system, referenced as EPSG (European Petroleum Survey Group)-Code 31258.
In order to pace from the CAD to the GIS domain, the DWG file is converted into the SHP (Shape) file format, acting as (mostly) open quasi standard data exchange format in the GIS world, developed and regulated by ESRI inc. (ESRI 1998). Two reasons exists for the conversion

into the SHP file format; (i) the conversion allows reprojection of spatial data into the targeting spatial reference system, which is in the available case WGS84 UTM (Universal Transverse Mercator) zone 33N EPSG-Code 32633. This saves unnecessary processing costs due to on the fly projection of spatial datasets in GIS systems. (ii) most geo-databases provide SHP-file data loader, as for instance Oracle's 11G or PostGIS 2.0, to insert spatial data sets and their attributive information to a dedicated databases. This is done in the next step as depicted in figure 3.1. The reason for choosing UTM as geographical base reference system is the fact that in scope of SESAAM, all aviation specific positioning information has to be based on the same reference ellipsoid WGS84. Furthermore, a Cartesian reference system is most convenient for geometrical computations, as done in the following chapter 4 Field Modelling.
The metadata structure of the original `infra_plan.dwg` file is shown in the following table 3.1:

| attribute | data type |
|---|---|
| Entity | char |
| Layer | char |
| Color | int |
| Linetype | char |
| Elevation | int |
| LineWt | int |
| RefName | char |
| BEZ | char |
| HOEHE | float |
| Shape_Leng | float |

Table 3.1.: Metadata structure of the spatial infrastructure data set of Salzburg Airport

The attributive metadata information of the DWG file shows rather CAD related than GIS relevant information as for instance color, linetype or line width. During the SHP-file conversion, geometry related information e.g. linetype or shape length is stored explicitly or implicitly within the geometry. Other information like color or line width is not needed for the purpose of pure geometry storage in the geo-database and can be defined separately in a GIS viewer. Hence, the target SHP-file carries only the `Layer` information as relevant attributive information.

Another data set, provided by the Salzburg Airport Facility Management, is the location of WLAN AP around the main apron of Salzburg Airport, as depicted in figure 3.1. The three-dimensional AP positions are relevant for later upcoming research on field modelling approaches in scope of this thesis. For field modelling, the exact position and spatial orientation of the antenna's phase-centre is relevant rather than the AP position itself. The AP position was originally measured by a hand-held GPS (Global Positioning System) receiver and afterwards precisely geo-referenced with aid of a desktop-GIS. Thus, the resulting excel file holds spatial point information in WGS84 coordinates.
The structure of the `ap_positions.xls` excel sheet, including point geometries and metadata of APs, is shown in following table 3.2:

| attribute | data type |
|---|---|
| X coordinate | float |
| Y coordinate | float |
| Height | float |
| Antenna_az | float |
| AP_MAC | character |

Table 3.2.: File structure of the AP data set of Salzburg Airport

The first three items in table 3.2 describe the AP's spatial position in `X`, `Y` and `Z` in WGS84 coordinates. The `antenna_az` attribute represents the horizontal orientation of the antenna phase-centre in reference to geographical north. `AP_MAC` attribute describes the physical MAC (Media Access Control) address of the AP radio interface and constitutes a unique identifier for the networking segment. This address enables the exact matching between received WLAN signals or messages and their transmission source.

Both spatial data sets, `infra_plan.shp` and `ap_positions.xls` are loaded to a GeoDBMS (Geographical Database Management System), while the latter one is reprojected first into the UTM reference system. For now, the GeoDBMS will be considered as "black-box" system. The structure of the database model will be considered in the upcoming section, while details about the conceptual loader framework, algorithms and implementation will be given in chapter 6.2 Processing Framework Architecture.
An overview on the final spatial static infrastructure dataset can be gathered from figure 3.2.

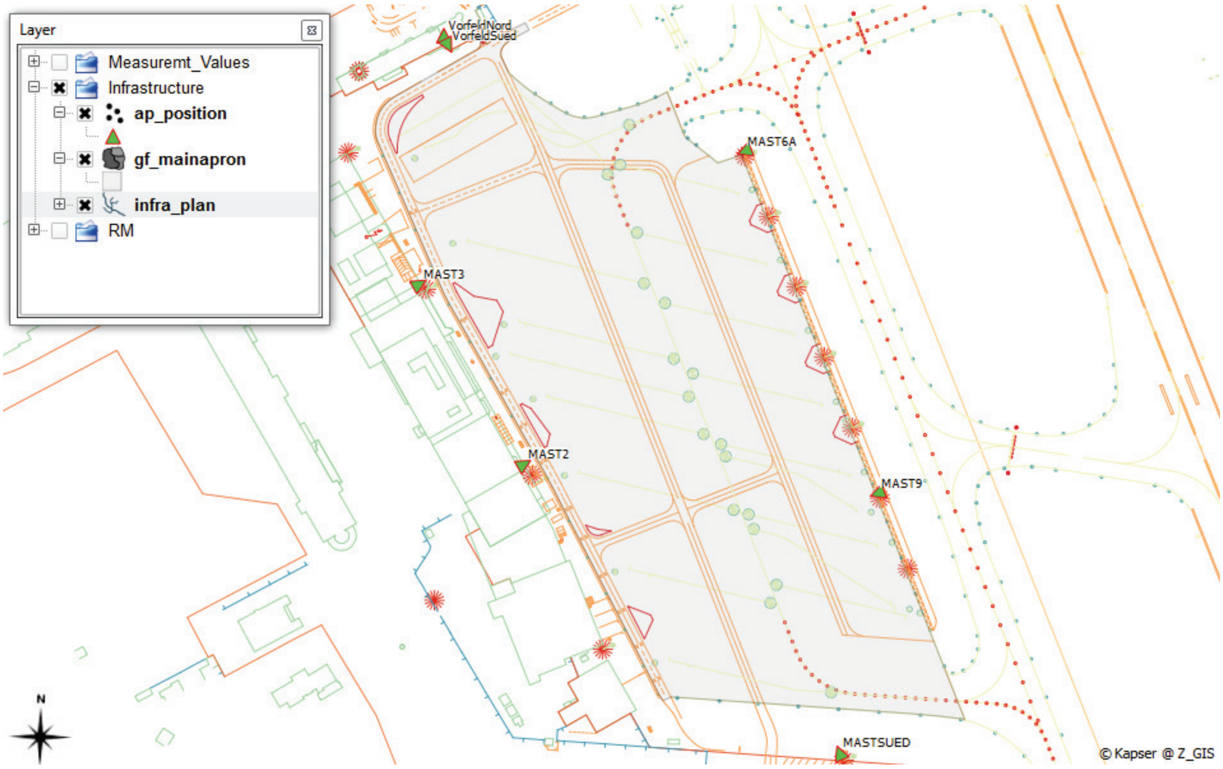


Figure 3.2.: Overview on spatial static infrastructure data, in QGIS

The overview on spatial static infrastructure data is delimited to the main apron area, constituting the testbed area. As depicted in figure 3.2 the main apron is delimited by a polygon shape in light grey and covers an area of ∼108280 $m^2$ or 10.828 ha. A detailed layer description of the infrastructure plan will not be considered further, as the static infrastructure data acts merely for general orientation.
The `ap_position` layer includes the AP positions, depicted by green triangles. Note that the horizontal orientation of these triangles indicates the boreside direction of the AP antenna, defined by the `antenna_az` parameter (cp. table 3.2). Thereby, the sharp angle of the triangle points at antenna boreside direction, towards the test-bed area. The influence of the antenna boreside direction together with its pattern will be discussed soon. The APs as part of the communication infrastructure literally surround the main apron for the purpose of WLAN coverage.

The last item of static infrastructure data is the AP antenna pattern, delivered by the antenna's technical specification (Systems 2005). This is usually provided by the antenna manufacturer. Basically, the antenna pattern describes the characteristic of radiated field intensity, as a function of polar direction in reference to the boreside direction as shown in figure 3.3. It is measured in dBi and presents the antenna gain in relation to an isotropic radiator, radiating the same intensity in all directions. This antenna characteristic is valid under far field condition, where distance $R \geq 2D^2/\lambda$ and $D$ is the largest dimension of the radiating antenna's aperture (Haykin 2001). The antenna pattern is usually measured in E-plane and H-plane, in accordance with the electromagnetic field theory (Haykin 2001). Therein, E- and H-vector of the electro-magnetic wave are standing perpendicular to each other and their cross product indicates the propagation direction.

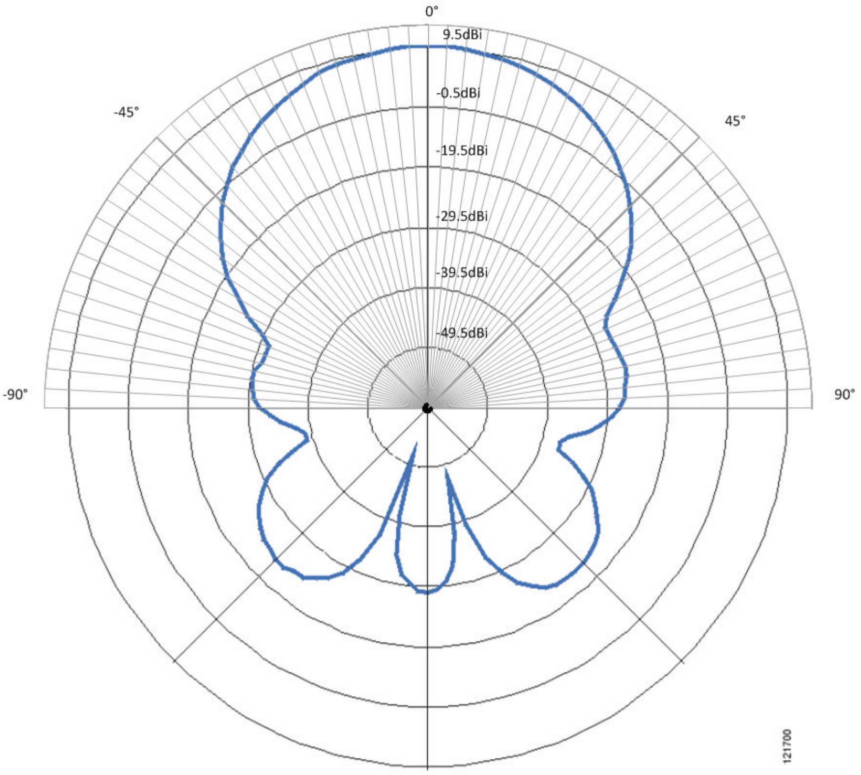


Figure 3.3.: Antenna pattern Cisco Aironet 9.5-dBi Patch Antenna (AIR-ANT5195P-R) in H-plane

Per definition, the H-plane is horizontally aligned and therefore relevant for two dimensional propagation intensity across the testbed. For that the assumption is made that antenna's vertical influence of field propagation is negligible, which is legitimate since the vertical antenna offset of few meters is minor in relation to horizontal test-bed dimensions of several hundred meters.
The plotted antenna characteristic of the available patch antenna in figure 3.3 shows, that the coverage area of the antenna is focused on the area in boreside direction at an angle of aperture of approximately 90°. The antenna has got an antenna gain of 9.5 dB. In other words, the antenna gain of 9.5 dBi means that the radiated field intensity in boreside direction is 8.91 times higher in relation to an isotropic radiator.
The antenna pattern is only available in printed format as jpeg-picture, hence is has to be digitised first as already shown in the use case diagram in figure 3.1. For the purpose of digitalisation the antenna pattern is equidistantly segmented in 3° steps as shown in figure 3.3. Since all relevant AP antennas around the testbed have been directly aligned towards the testbed (cp. figure 3.2), the antenna pattern +/- 90° around the antenna's boreside direction has been digitised only. For each of the constructed segments, the antenna gain is captured manually to the excel file `ant_pattern.xls`. That excel file is again loaded to the GeoDBMS in prearrangement to further processing.

The illumination of static infrastructure data in this section has shown, diverse spatial and non-spatial data have been provided within the project context. This data can be either utilised for field modelling or simply for display and orientation purpose. For re-usability of the data-framework the data integration process has been neatly drawn. Next the focus will be laid on description and integration of dynamic measurement data.

### 3.1.2. Dynamic Measurement Data

Dynamic measurement data comprise all available data with inherent dynamic characteristic either in time, space or the combination of both domains. It covers purely non-spatial WLAN measurement data on the one hand, MU (Mobile Unit) and aircraft positioning data on the other. Figure 3.4 shows the use case diagram to the process of data delivery and conversion.

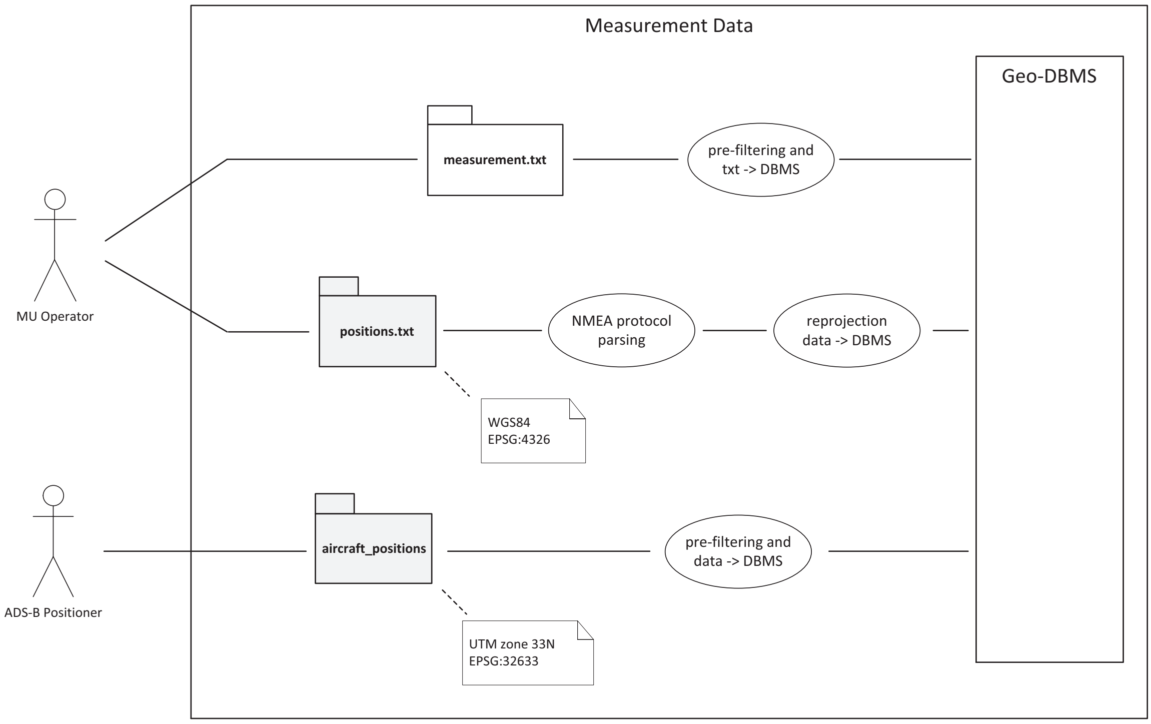


Figure 3.4.: Use case diagram for measurement data; boxes coloured in light grey indicate spatial data sets

Let's start with a detailed consideration of the measurement data as such. This non-spatial data set is delivered by the MU operator, Technical University of Vienna research group Geodesy, and project partner in context of SESAAM. The MUs are basically flight operation and logistic support vehicles, permanently moving on the main apron of Salzburg Airport and testbed area. On selected vehicles a WLAN interface plus antenna and EGNOS (European Geostationary Navigation Overlay Service) capable GPS receiver have been installed. While EGNOS allows more precise differential GPS positioning. These MUs permanently acquire the signal strength of surrounding WLAN APs in reach and record their geographic position via GPS. It is to emphasis that data recoding rates of WLAN and GPS interfaces are different and direct synchronisation between both data streams is not given. However, the clocks of both interfaces have been obtained the same time base and running synchronous.

The incoming measurement data have basically the following file structure, available as comma separated string format:

| attribute | data type |
|---|---|
| WLAN-adapter ID | integer |
| status flag | boolean |
| timestamp | timestamp |
| AP MAC address | character |
| channel | integer |
| signal strength | integer |
| package type | character |
| SSID | character |

Table 3.3.: File structure of the measurement data set of MUs

The `WLAN-adapter ID` acts as a unique identifier of the WLAN interface and as such as unique identifier for certain vehicles. The `status flag` indicates the validity of the WLAN message and has to be considered in further processing. The `timestamp` counts time and date and enables synchronisation of measurements and positions later on. `AP MAC address` helps for the identification of the WLAN data package or signal source. The MAC address can be easily correlated with the AP name. `Channel` indicates the selected WLAN channel for package transmission. The `signal strength` constitutes the core message of this block and carries the received signal power of the incoming signal, measured by the WLAN interface. That power is measured in whole-number dB steps. `Package type` indicates the WLAN protocol package type and SSID (Service Set Identification) is used for identification of the WLAN setup.
Following, the `measurement.txt` file is loaded into the GeoDBMS as shown in figure 3.4. However, in advance the data set is filtered in order to save database storage. The filter is focused on status flag and package type. All data blocks have been loaded to the GeoDBMS where its status flag is "vaild" and package type is set to "BEACON". The beacon message is periodically transmitted by APs and announces the availability of a WLAN network. As the Beacon frame includes a fixed count of data fields, defined by the IEEE 802.11x standard, this message type is ideally suited for power measurements.

The second part of dynamic measurement data includes the positions of MUs. This continuously recorded GPS EGNOS positions produce consequently a spatial data set, referenced in a geographic coordinate system with WGS84 datum. The `position.txt` file is provided by the MU operator and holds recorded NMEA (National Marine Electronics Association)[1] protocol messages from the GPS receiver. This messages are encoded in GPGST and PUBX sentence format, both available each full second. While the GPGST sentences provides GPS pseudorange noise statistics, the PUBX is a proprietary sentence format from the GPS receiver manufacturer

[1] www.nmea.org

uBlox, holding the position solution data (ublox 2008).
For each position fix a GPGST and PUBX sentences is provided. Thus, both NMEA protocol sentences have been parsed and composed to one message format as shown in the following table 3.4:

| attribute | data type |
|---|---|
| timestamp | timestamp |
| lat | float |
| lon | float |
| alt_ref | float |
| nav_stat | character |
| hacc | float |
| vacc | float |
| speed_og | float |
| course_og | float |
| hdop | float |
| vdop | float |
| tdop | float |
| rms_dev_pseudo | float |
| lat_dev_pseudo | float |
| lon_dev_pseudo | float |

Table 3.4.: Including NMEA protocol data of GPGST and PUBX sentences

The `timestamp` holds the UTC (Coordinated Universal Time) time of the position fix and helps for correlation of positions and measurement values later on. The position is given in geographic latitude (`lat`) and longitude (`lon`) coordinates in degrees and minutes. `Alt_ref` indicates the altitude above the datum ellipsoid WGS84. The naviagion status is given by `nav_stat` and `hacc` and `vacc` indicates the horizontal and vertical accuracy estimate of the GPS position fix in meters. Speed and course over ground is given by `speed_og` and `course_og` in km\h and degrees. The DOP (Dilution of Precision) value is described in horizontal, vertical and time by `hdop`, `vdop` and `tdop`. All these data is originally encoded into the PUBX sentences. `Rms_dev_pseudo`, `lat_dev_pseudo` and `lon_dev_pseudo` are differential GPS specific standard deviations in ranges and position out of the GPGST sentences format.
After message parsing, positions are reprojected into Cartesian UTM coordinates and loaded into the GeoDBMS. Details about parsing, reprojection and database loading can be find in section 6.2.3 MU Data Feeder Application.

The aircraft positioning data is again provided by the project context of SESAAM. Therein, the position of aircraft carrying ADS-B (Automatic Dependent Surveillance Broadcast) enabled transponders have been collected by an ADS-B receiver. This enables aircraft tracking around the airport and especially on the main apron, constituting the project's testbed area.
The aircraft position data is basically transmitted from the ADS-B receiver in Eurocontrol's ASTERIX[2] cat 21 encoding standard for the purpose of aircraft surveillance data exchange. This

[2] http://www.eurocontrol.int/services/asterix

protocol format is parsed and stored into an external database. The available aircraft_position data set is simply a copy of the stored ADS-B data frames, where each frame has got the following structure:

| attribute | data type |
|---|---|
| mode3a_code | character |
| flightno | character |
| data_source_ident_id | integer |
| target_add | character |
| timestamp | datetime |
| proc_pos_cart_x | float |
| proc_pos_cart_y | float |
| proc_pos_z | float |
| cat | integer |
| position | geometry |

Table 3.5.: File structure of the aircraft_position data set

The mode3a_code is an aviation specific identification code with an address range of $8^4$. This code is still used for aircraft identification, however due to its minor address range replaced by 24-bit SSR (Secondary Surveillance Radar) Mode S addresses. The flightno is transmitted by the SSR Mode S Message and includes the ICAO (International Civil Aviation Organization) flight identification call sign. This field is consequently used as unique aircraft identifier within the context of this thesis. The data_source_ident_id is a unique ADS-B receiver station identifier and is utilised in distributed surveillance systems. This however is not applicable on the locally delimited scale of this project. The timestamp is again in UTC timeformat and allows the correlation to other positioning information. Proc_pos_cart_x and proc_pos_cart_y carries the already reprojected geometric UTM zone 33N aircraft position in plain text. Proc_pos_z carries the aircraft's altitude and will be used for position filtering later on. The cat field indicates the ASTERIX protocol category, applicable if different positioning sensors would be used simultaneously. The position holds a GeoDBMS specific geometry format, allowing a simple database insert by a copy instruction.
As already mentioned, the aircraft position data set can be simply inserted into the GeoDBMS by a copy instruction. However in advance, it is spatially filtered to the extend of main apron area on ground. Otherwise, all aircraft positions would be taken into account at a radius of approx. 50 km around the airport.

The introduction of dynamic measurement data shows, although WLAN measurement and GPS position data have been delivered from the same data source, the MU processing unit, there is no direct correlation available between both data sets. However, the timestamps of both data sets are synchronised. For that purpose a correlation strategy in time domain has to be found, in order to connect measurement data to spatial information. That method will be elaborated in the upcoming section.

Concluding, the data source section has shown that different spatial and non-spatial data have been made available for the target of electro-magnetic field modelling. Static infrastructure data is employed, for the purpose of topological orientation and geo-referencing. Non-spatial static data about the antenna characteristic is provided for electro-magnetic field modelling later on. Dynamic spatio-temporal GPS positioning and WLAN measurement data acquired by mobile sensors have been provided as well. All that data is integrated into a GeoDBMS for the advance of clear data organisation and further data processing/mining capabilities. The topic of data integration and modelling will be deliberately outlined in the upcoming section.

## 3.2. Data Modelling

A central question the in handling of spatial and non-spatial data is their modelling. From a GIS point of view the term data model is considered as the mathematical representation of geographic objects either in discrete or continuous form. Typical mathematical models meeting that requirements are vector models for continuous and cell matrix models for discrete representation. The term "data modelling" however describes the process behind. Goodchild (1992*a*) defines data modelling as following:

> *Data modelling is defined as the process of discretizing spatial variation, but may be confused with issues of data structure, and driven by available software, rather than by concern for accurate representation.*

Despite Goodchild's strict confinement of the term "data modelling", this section covers data(base) modelling and data integration. While the first one deals with modelling of data within a object relational geo-database, the latter considers data integration, including the processing steps of data filtering, interpolation and correlation between spatial data and attributive information.

### 3.2.1. Data(base) Modelling

In context of the project frame, a massive load of data has been made available by continuously recorded WLAN measurement and GPS positioning values. Moreover, data is provided from different sources and domains, as shown in the previous chapter. In order to reflect the relationship of data to real world objects and model the spatial and non-spatial relationships between them, the object relational data modelling paradigm is applied. A modelling paradigm is per-se independent of any software related data structure or implementation platform. Though, for the implementation of the data model a object relational GeoDBMS is selected due to following points:

- A object relational geo-database enables seamless integration of the proposed object relational modelling paradigm.

- Databases allow scalability in terms of storage and processing capacities and have thus the capability to store the available load of data.
- Databases offer internal mechanisms to check data integrity.
- Available spatial data types allow discretising of real world geographical objects.

Summarized, these strengths of a GeoDBMS helps in handling the available load of data in a structured and reliable way.

The data modelling approach of available static and dynamic, spatial and non-spatial data is presented in figure 3.5 as object relational data model. All of previously in section 3.1 introduced data are loaded to that model, settling within the GeoDBMS. The model itself is structured into five logical groups, implemented as different schemas in the database model, orientated on thematically different data sources. Each group holds relevant data for inherent objects, representing real world objects.

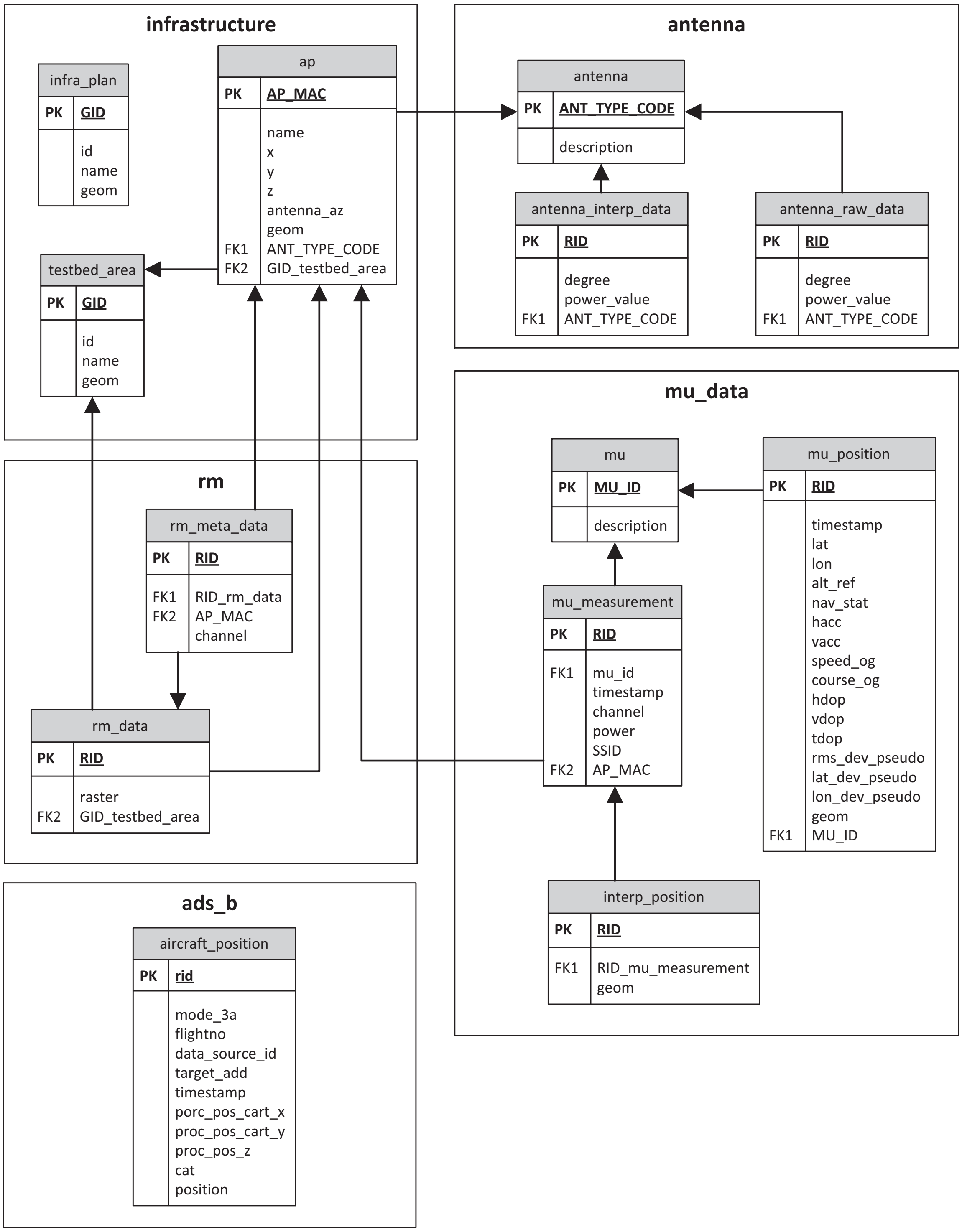


Figure 3.5.: Object relational data model

Let's start with infrastructure group. It holds the classes `infra_plan`, `testbed_area` and `AP`. The first class infra_plan represents objects of the infrastructure plan, as introduced in the previous section, by its spatial geometry (geom). Since that plan has information purpose only, in the context of electro-magnetic field modelling, it has not been further integrated in the data model yet. However, it is suggested that in future relevant infrastructure objects for field modelling purpose, should be integrated into the model, based on that class. The testbed_area class describes the spatial extend of the testbed area by a polygon shape (geom). In context of the project that area is delimited to the main apron of Salzburg Airport (cp. figure 3.2), though that model allows the extension to multiple testbed areas. Each testbed area is related to a subset of WLAN APs, enabling WLAN availability in these areas. The ap class describes available APs as part of the positioning infrastructure by an unique MAC address of the radio interface, name, antenna orientation, antenna type and position as point geometry. The antenna type is modelled in the antenna group.

The antenna group holds all antenna related data and includes exclusively non-spatial data. The `antenna` class describes the antenna type. For each antenna type digitised `antenna raw data` exists, as already discussed above in section 3.1. Within the data integration process, elaborated in the upcoming section, these data will be interpolated further and loaded to the `antenna_interp_data` class. This class is consequently again referenced to the `antenna` class. Although, within the context of this thesis only one antenna type is used, that modelling approach allows high flexibility for extension of additional antenna types.

The mu_data group holds all dynamic MU related data. Each MU or sensor platform object is basically described by the `mu` class. Measurement and position data, acquired by different sensors, are directly related to its sensor platform. The `mu_measurement` class hold each WLAN field strength measurement with its timestamp, channel, measured power value, SSID and MAC address. Thereby, the MAC address enables the matching between received WLAN-signal and physical AP. Thus, in that model `mu_measurement` and `ap` class are connected over the AP_MAC address. The `mu_position` class holds positioning related information for each position object, as previously discussed in section 3.1; whereas, each position is modelled as point geometry (geom). As no direct synchronisation between positioning and WLAN measurement exists, the `mu_position` class is exclusively related to the `mu` class. The correlation of positions and WLAN measurements is done in the time domain and will be elaborated in the upcoming data integration section. The correlation process results in an interpolated position for each single measurement object, described by the `interp_position` class. That is consequently related to the `mu_measurement` class about its RID and again related to the `mu` class.

The rm group holds the RM (Radio Map) as result of the field modelling process. It describes the RM as spatial raster data set over the extent of a certain testbed area. The `rm_data` class holds the spatial raster dataset, including a raster of multiple channels for each AP, and the testbed area identifier. As each RM is related to a testbed area, a direct relation between `testbed_area` class and `rm_data` class is established. Note that one RM might be compound of multiple raster tiles, justifying multiple objects per testbed area. The assignment between raster channels and APs is modelled by the `rm_meta_data` class. Each object of this class records the raster channel matching to a certain WLAN AP. Consequently, the `rm_meta_data` class is related to both, the `rm_data` and `ap` class.

Finally, the ads_b data set represents the spatially filtered aircraft positions to the extend of the testbed area. This information will be exclusively used as validity criterion for MU measurements later on. Therefore, interpolation on the aircraft's track will not be applied to this data set.

The data model shows what complex relationships between real world objects exists and how they can be transferred onto a model. That generic object relational model constitutes the basis for data integration into a GeoDBMS. Basically, the model already implicitly presents given cardinalities by already stated primary/foreign key notations. The given model comply with the logical database design, as tables and relations are already given by its class definition. For more details about the integration platform please have a look at 6 Research Methods Appendix.

### 3.2.2. Data Integration

Although according to previously discussed processes all raw data has been made available in the GeoDBMS, figured out by use cases in section 3.1, the previously introduced data model has not been entirely filled yet. The intention of the following section is to close the gap between available raw data from different sources and thereof derived interpolated or timely correlated data. To approach that target, first a general overview on how to obtain interpolated data out of available raw data is given. Next, it goes into more detail about the actual interpolation methods applied, starting with spline interpolation of the antenna pattern. Following, the parametric spline interpolation method for MU positions will be elaborated. After the interpolation of positions the correlation strategy in the time domain between positions and WLAN measurement values will be shown.

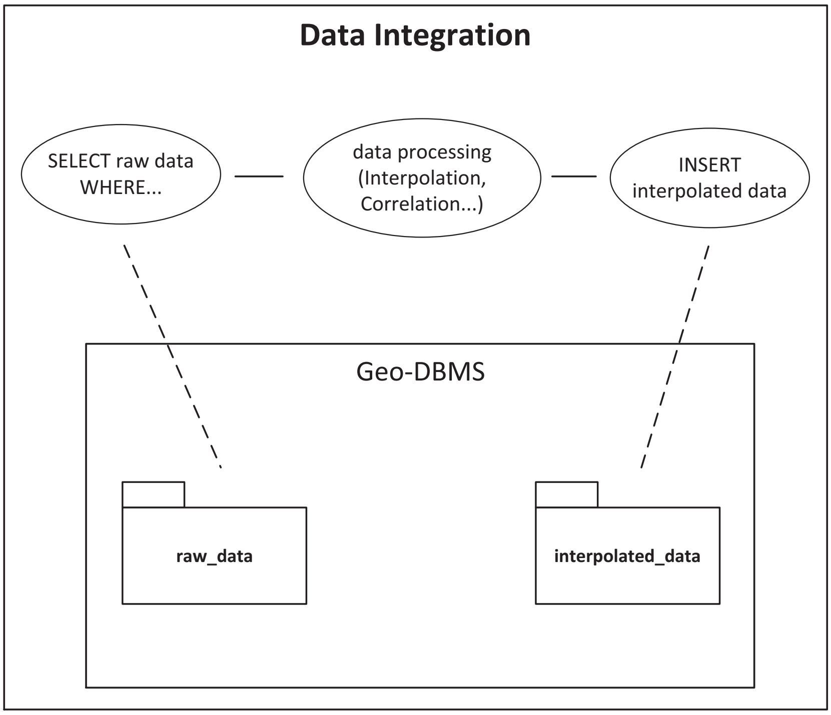


Figure 3.6.: UML use case diagram of the data integration process

The general process of data integration is depicted by a use case diagram in figure 3.6. Please note that in the upper part of that diagram the three use cases are stated outside of the GeoDBMS, in order to obtain more processing flexibility during the prototyping phase. However, this use cases could be integrated to the GeoDBMS either by internal functional database processes. AT the use case diagram, available raw data from different data sources is firstly selected from the GeoDBMS. Thereby, database internal filter mechanisms enables pre-filtering of raw data, formulated by the "WHERE" clause in the SQL "SELECT" statement. After data has been retrieved from database, data processing is conducted. This includes interpolation, correlation and other methods depending on the targeting integration strategy. After the processing has finished, data is again inserted into the database, this time into the interpolated data section.

**Antenna Pattern Integration**

Firstly, let's have a look on the integration process of the antenna pattern. As already mentioned in previous section 3.1, digitised antenna gain values of the antenna pattern have been made available in the database. These values are equally distributed over a range from -90° to +90° at steps of 3°. In order to obtain a finer resolution of power gain values in polar direction, they have to be interpolated accordingly. For that purpose the deterministic spline interpolation method, as amongst others introduced in section 2.2.2, for the one-dimensional case is applied. Spline interpolation was selected (i) having an exact interpolation result, including all interpolation points and (ii) in order to allow segment wise interpolation, reducing computation effort at high counts of interpolation points.

Let's shortly revise the mathematical background of spline functions for the example of a cubic spline, according to (Späth 1995). A function $f(x)$ is given by discrete count of sampling points $x_i(i = 1, ..., n)$ and their basic values $f_i = f(x_i)$.
The functional description of the interval between two sampling points is interpolated by a polynomial of the third degree:

$$P_i(x) = f_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3;\ x \in [x_i, x_{i+1}],\ i = 1, ..., n - 1 \tag{3.1}$$

The interpolation function $P_i(x)$ must be *twice* continuously differentiable at the interval boundary:

$$\begin{aligned} &P_i(x_i + 1) = f_i + 1\ ,\ i = 1, ..., n - 1; \\ &\dot{P}_i(x_i + 1) = \dot{P}_{i+1}(x_i + 1) \\ &\ddot{P}_i(x_i + 1) = \ddot{P}_{i+1}(x_i + 1)\ ,\ i = 1, ..., n - 2. \end{aligned} \tag{3.2}$$

The polynominal coefficients can be solved accordingly by following equations for $c_i$:

$$\begin{gathered} c_1 = \ddot{f}_i/2;\ c_n = \ddot{f}_n/2 \\ h_{i-1}c_{i-1} + 2(h_{i-1}+h_i)c_i + h_i c_{i+1} = \frac{3(f_{i+1}-f_i)}{h_i} - \frac{3(f_i - f_{i-1})}{h_{i-1}} \quad i = 2, ..., n-1. \\ \text{where } h_i = x_{i+1} - x_i \end{gathered} \tag{3.3}$$

$b_i$:

$$b_i = \frac{f_{i+1}-f_i}{h_i} - \frac{h_i(c_{i+1}+2c_i)}{3} \quad i = 1, ..., n-1. \tag{3.4}$$

$d_i$:

$$d_i = \frac{c_{i+1}-c_i}{3h_i} \quad i = 1, ..., n-1. \tag{3.5}$$

The coefficients of equation 3.3 can be expressed as system of three linear equations, that can be solved e.g. by Gaussian elimination. Thereafter, $b_i$ and $d_i$ can be solved according equations 3.4 and 3.5.

For the applicable case the most commonly used basis spline (B-spline) is applied. For computation of the actual one-dimensional spline curve, Python wrapped Fortran routines splrep[3] and splev[4] of Fitpack were utilsed.

[3] http://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.splrep.html#scipy.interpolate.splrep

[4] http://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.splev.html#scipy.interpolate.splev

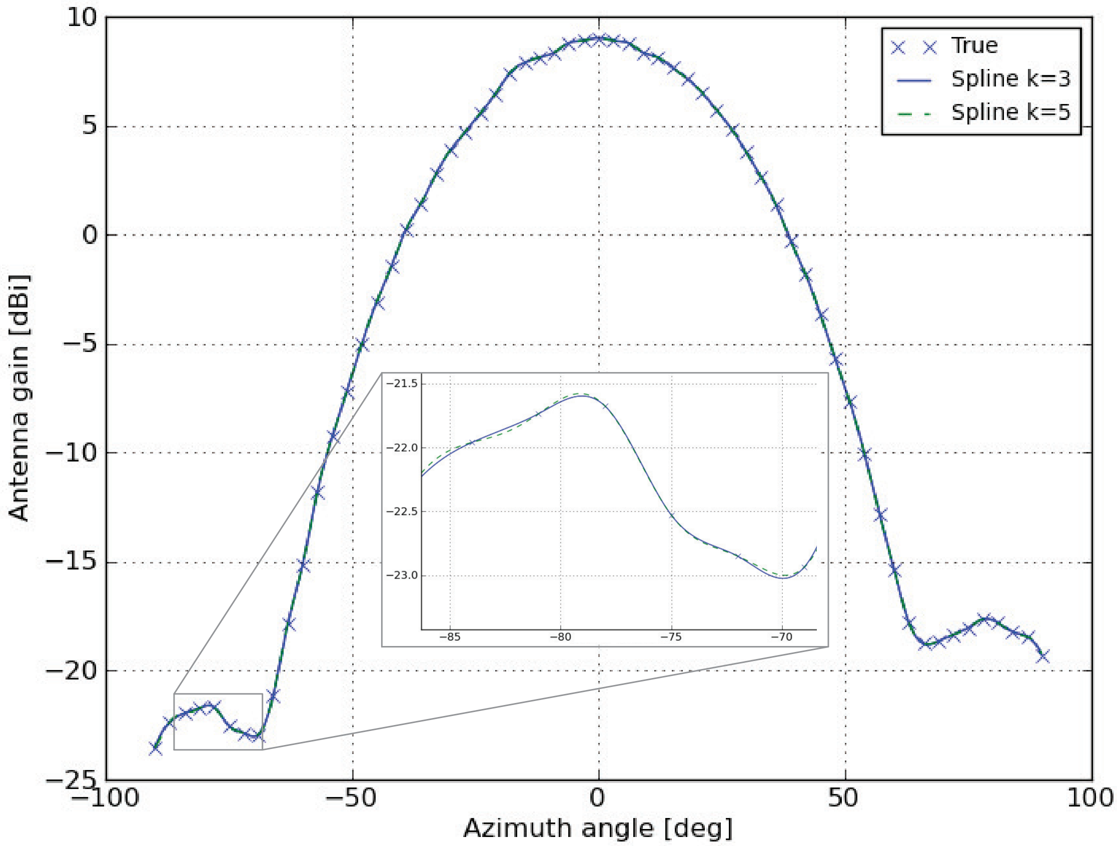


Figure 3.7.: Spline interpolation of the antenna pattern

Figure 3.7 shows the interpolation results of a cubic spline (k=3) and a spline with a order of k=5 at a resolution of 0.1°. The "true" values represent the originally digitised interpolation points. The detail window of the interpolation result shows, the natural spline results in a smoother, less oscillating curve than at a spline order of five. Thus, the interpolation order is selected thereafter to three for the natural spline. Details about exact parametrisation of the interpolation algorithm can be gathered from A Annex 1.

### Measurement Data Integration

The integration of already available GPS positioning and WLAN measurement data does raise the main question about the correlation strategy of these data sets. Figure 3.8 presents both data sets in their domain; positioning data are distributed over X/Y coordinate in space and measurement data $m_1, m_2, ..., m_n$ are fixed in time. As all vehicle movements in altitude are constraint to the apron's surface, the Z coordinate delivered by GPS is not considered. Though, position data carry inherent time information from the GPS position fix, that can be utilised for later correlation of both data sets over time.

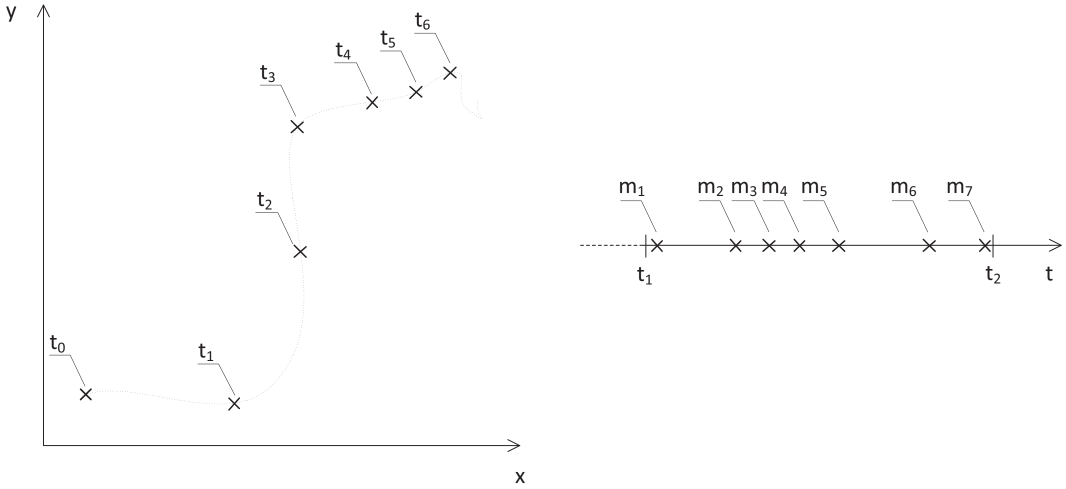


Figure 3.8.: Schematic spatial distribution of recorded positioning, on the left hand, and measurement data, on the right hand side

Note that the GPS position fix is done on discrete points in time $t_0, t_1, ..., t_n$ at an interval of 1 second. Depending on the vehicle's motion profile the positions are more or less separated in space, as depicted in figure 3.8 where the vehicle's velocity on the begin of the track is higher than at the end. The dashed line in figure 3.8 left hand side indicates the true track of the vehicle. In preparation to correlation of positions and measurements over time an interpolation of positions in two dimensional space is applied. A finer resolution of discrete tracking points allows more accurate correlation between positions and measurement values over time.

For interpolation of a n-dimensional curve, parametric spline interpolation is selected again due to above stated reasons of exact interpolation and segment wise computation capability. Therefore, the GPS track has to be as regarded as parametric curve. This curve $r(\tau)$ should now be interpolated segment wise by spline interpolation polynomials $P_i(\tau)$. At this approach each component of the curve coordinate is interpolated by a spline function. For each sampling point, or GPS position respectively, of the curve $r(\tau)$ the position vector $\vec{x}_i$ is known at sampling points $\tau_i$ $(i = 1, .., n)$. For cubic interpolation follows:

$$r(\tau) = \vec{P}_i(\tau) = \vec{x}_i + \vec{b}_i(\tau - \tau_i) + \vec{c}_i(\tau - \tau_i)^2 + \vec{d}_i(\tau - \tau_i)^3;\ \tau \in [\tau_i, \tau_{i+1}],\ i = 1, ..., n - 1 \quad (3.6)$$

where the parameter $\tau$ can be considered as continuous curve parameter. Equation 3.6 has basically the same form as equation 3.1 for one-dimensional interpolation. It can be solved by equations 3.3, 3.4 and 3.5 for vectorised polynomial coefficients for the case of open curves, as applicable for GPS track interpolation. For computation of the actual two-dimensional basis spline polyominals, again Python wrapped Fortran routines splprep[5] and splev of Fitpack were utilised.

For interpolation of the position track a bunch of data per mobile unit is available. Each MU records 3600 positions per hour, yielding in 43.000 positions in 12 hours. This would generate massive processing effort for continuous interpolation over the entire 43.000 position fixes per half a day. Moreover, in the applied interpolation libraries no possibility was found to adjust the outer left and right boundary conditions of the spline polynomials. In order to overcome that issue, a moving window strategy for segment wise interpolation is applied.

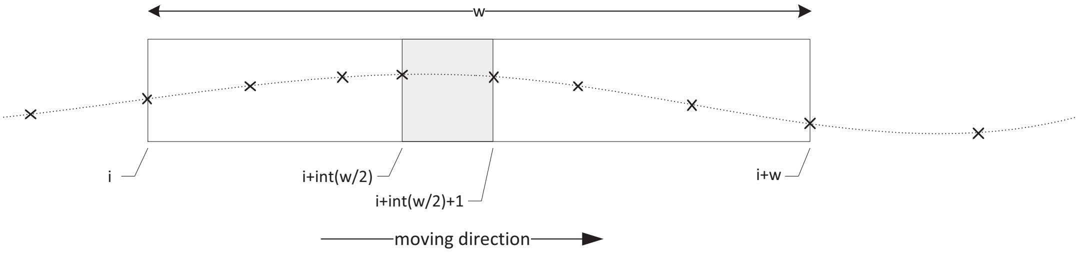


Figure 3.9.: Moving window strategy for segment wise spline interpolation

Figure 3.9 shows the formal definition of the moving window. The index $i$ for each interpolation node is running from $i = 0, ..., n$ and the window width $w = 3, 5, 7, ..., n$ if $n$ is uneven, otherwise $w = 3, 5, 7, ..., n - 1$. All included points of the window are used for one interpolation cycle, whereas only the centralised segment of the window (cp. figure 3.9 grey shadowed zone) is regarded for the interpolation result. The centralised segment is defined by its boundaries $i + int(w/2)$ on the left and $i + int(w/2) + 1$ on the right hand side for $i \geq int(w/2)$. For $i < int(w/2)$ the left hand boundary is defined by $i$ and the right hand side by $i + 1$. Note that the $int()$ function cuts off the decimal values from divisions.
Figure 3.10 shows exemplarily the result of the defined moving window interpolation process of seven segment wise interpolation steps, from $i = 0, 1, ...6$ at a window width of 7 segments.

[5] http://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.splprep.html#scipy.interpolate.splprep

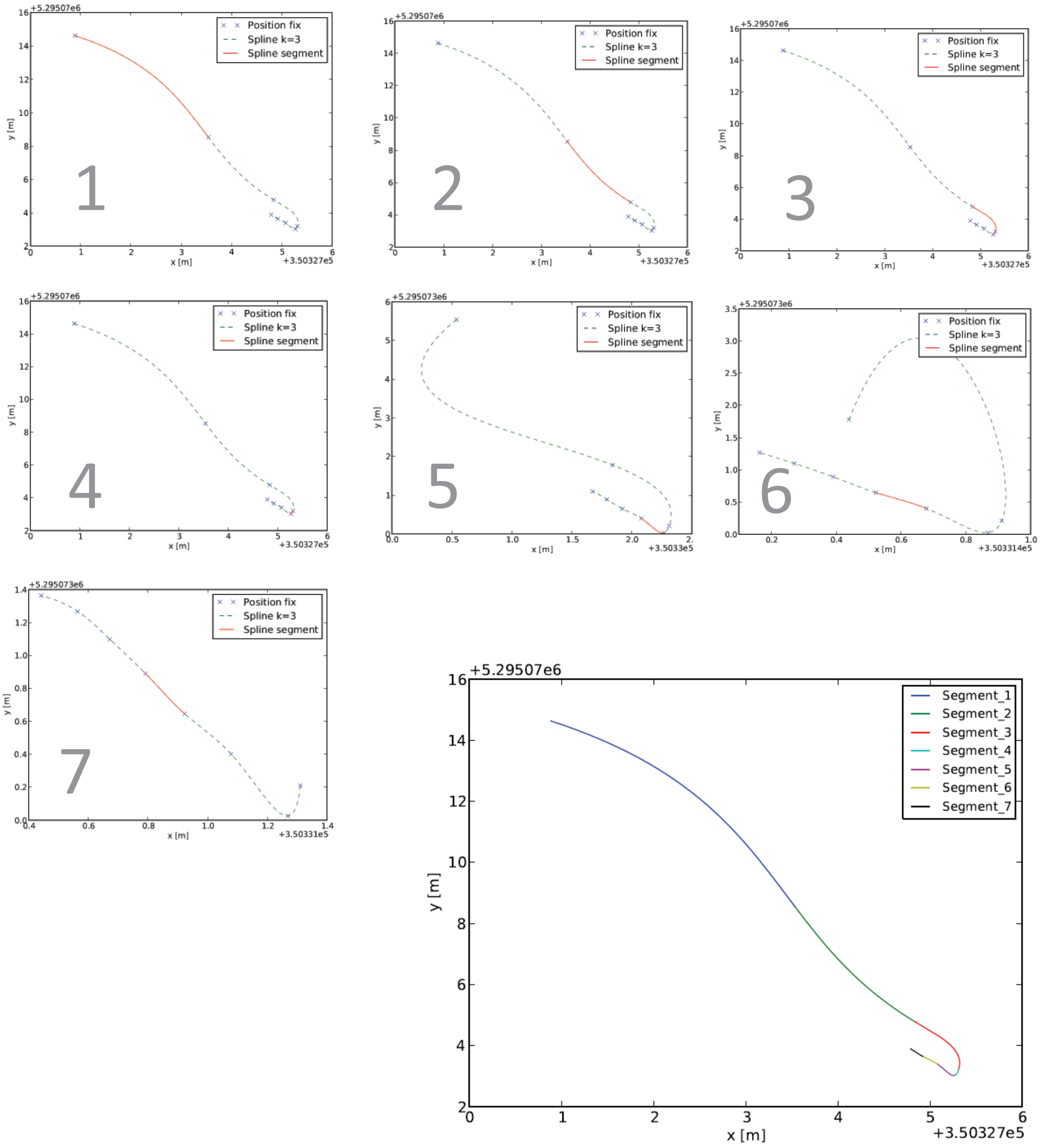


Figure 3.10.: Exemplary moving window interpolation process from step 1 to 7

Note that at the beginning of the moving window process the interpolation segment has not been centred yet. If index $i$ is greater than 3 the window starts to move. It has shown that a window width of 7 is sufficient to obtain a smooth interpolation of the position track. The segmented overall result of that process is summarised in figure 3.10 at the bottom line. The result shows a smooth curve, where the first derivation of neighbouring segments have identical values.

The continuous interpolation parameter $\tau$ in that process is configured from $\tau = 0, 1, ...25000$, resulting in 25001 interpolated points within the frame of the interpolation window. As the parameter $\tau$ implicitly represents the distance between two neighbouring points of the discretised interpolation polynominal, the maximum count of $\tau$ is a measure of the bin size. In the example of $\tau_{max} = 25000$ at a continuous speed of 200km/h, that would yield in sampling point separation of approx. 1.5cm.

For correlation between the interpolated track of GPS positions and the WLAN measurements, the interpolated track is transformed into the time domain. This is done segment wise by firstly computing the arc length $s$ of a line segment from $\tau_{min}$ to $\tau_{max}$. In discrete representation this is fulfilled by following equation:

$$s = \sum_{\tau=\tau_{min}}^{\tau_{max}-1} \left\| \vec{P}(\tau+1) - \vec{P}(\tau) \right\|_2 \tag{3.7}$$

Form the GPS position fix interval it is known, one interpolation segment reclines within a fixed time interval of 1 sec. Hence, for each discretised point of the interpolation polynomial a time representation is given, simply by the linear relation between segment length $s$ and individual path length of discretised polynomial points. Note at this simplified approach the moving object's change in acceleration within the one second interval is not considered. However, it is implicitly taken into account for each segment of the correlation process as the segment separation changes at positive or negative acceleration.
On the basis of two vectors, WLAN measurements and interpolated position track data, both in the time domain, a simple nearest neighbour matching is done for correlation of measurements and spatial information. Figure 3.11 shows an exemplary result of this correlation process, based on the segment wise interpolation process discussed above.

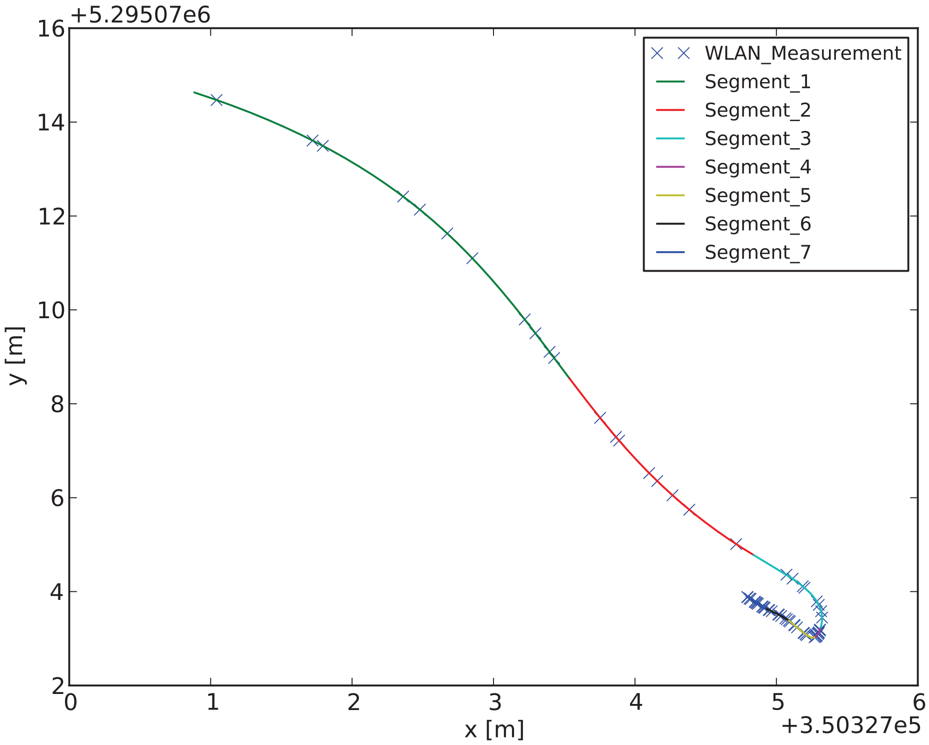


Figure 3.11.: Correlation of measurement values and position track

Each cross on the track indicates the spatial representation of a correlated WLAN measurement to a registered AP of the positioning infrastructure.

The correlation process is applied for the available GPS positioning data and WLAN measurements, yielding into approx. 220.000 position/measurement tupels for one vehicle per day.

Figure 3.12 shows the result of the correlation process for one AP over the testbed area at Salzburg Airport. The result includes 57.097 spatial point datasets for one vehicle over a timespan of approx. 14 hours. The point cloud shows a densely covered testbed area, with limitation on the lower right corner, where the antenna pattern has low gain (cp. figure 3.7), thus the MU is going out of the AP's WLAN coverage.

The detail view in figure 3.12 shows that measurement values are sometimes clustered in spatial distribution. This is the case when the MU stops and holds its position for a certain amount of time. This sort of clusters would allow averaging over space, time and measurement domain in order to obtain more accurate reference values for further field modelling purpose.

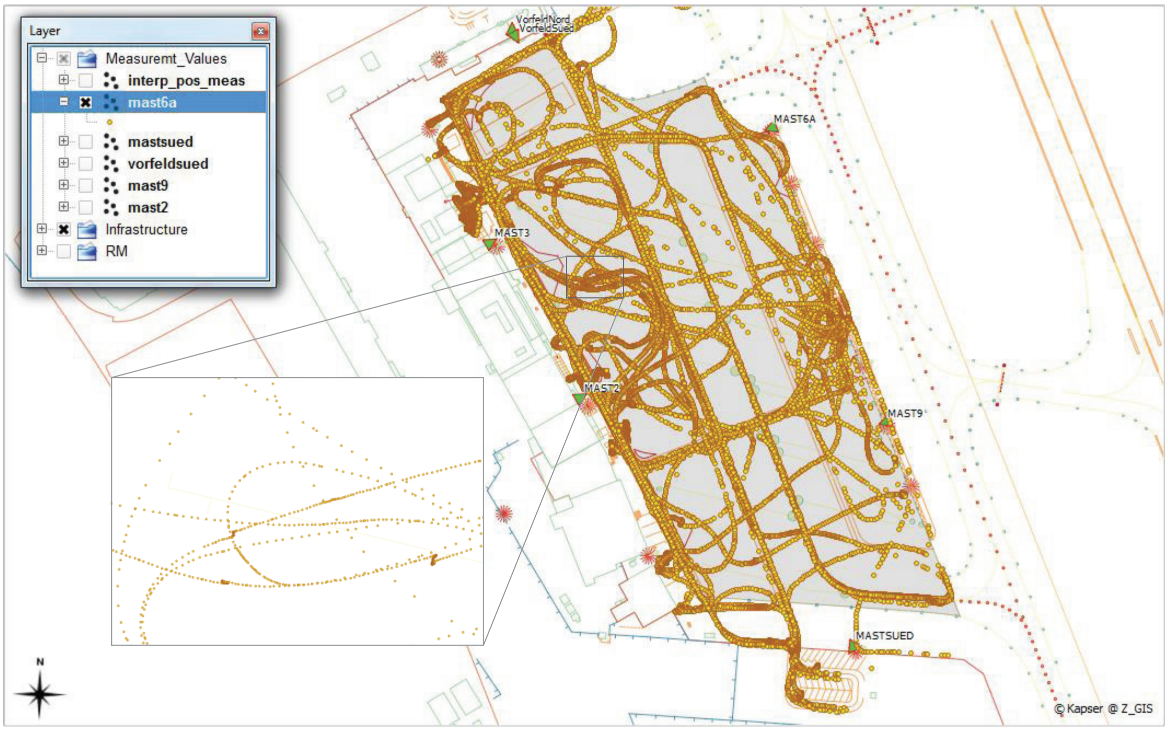


Figure 3.12.: Spatial WLAN measurements to AP mast6a at main apron of Salzburg Airport

This section has shown that data integration of non-spatial and spatial data generates rather much effort in data handling, processing and storage. By means of interpolation, a big pile of data has been made available, constituting the groundwork for field modelling. This raises the question:"What data is actually applicable or useful for field modelling?" This should be answered in the upcoming section, where the focus is laid on explorative data analysis.

## 3.3. Data Analysis

Isaaks & Srivastava (1989) states: “One of the things that distinguishes earth science data sets from most others is that the data belongs to some location in space.” This is definitely applicable for the case of available georeferenced measurement data of MUs and aircraft positions, although the task of electro-magnetic field modelling is not related to earth science in a broader sense. Another aspect is given by the time-variant characteristic of the sampled electro-physical phenomenon, that each data sample belongs additionally to certain point in time. The main characteristic of time-variant systems is that the system’s output depends exclusively on time. This however is not the case in the electro-magnetic field modelling, as the field distribution is dependent on location as well. This would give the overall phenomenon a spatio-temporal characteristic.

The toolbox of spatial (geo)statistics provides a bunch of methods for the spatial description of data sets. This section will explore the available measurement and position data in spatio-temporal scale. The main target of this section is the exploration of data in terms of spatio-temporal distribution. Moreover, it introduces a spatio-temporal buffer method for isolation of measurement reference values.

### 3.3.1. Spatio-Temporal Distribution

Currently, through figure 3.12 in section 3.2.2 Measurement Data Integration, we have already obtained a first impression of the spatial distribution of interpolated measurement positions. This measurement track was recorded on a time window of approx. 18 hours. However, the intensive data volume and small scale of spatial distribution, clustering of measurement values is hard to analyse. This raises intuitively the question:“How does the distribution look like on a spatio-temporal scale?”

In order to find an answer to that questions the open source software SGeMS (Standford Geostatistical Modeling Software)[6] was utilised to explore the collected data in a three dimensions: X, Y, and time. SGeMS was selected at this stage as it allows easy access to multidimensional data exploration, wherein most of the available GIS software systems are still lacking behind.

[6] http://sgems.sourceforge.net/old/index.html

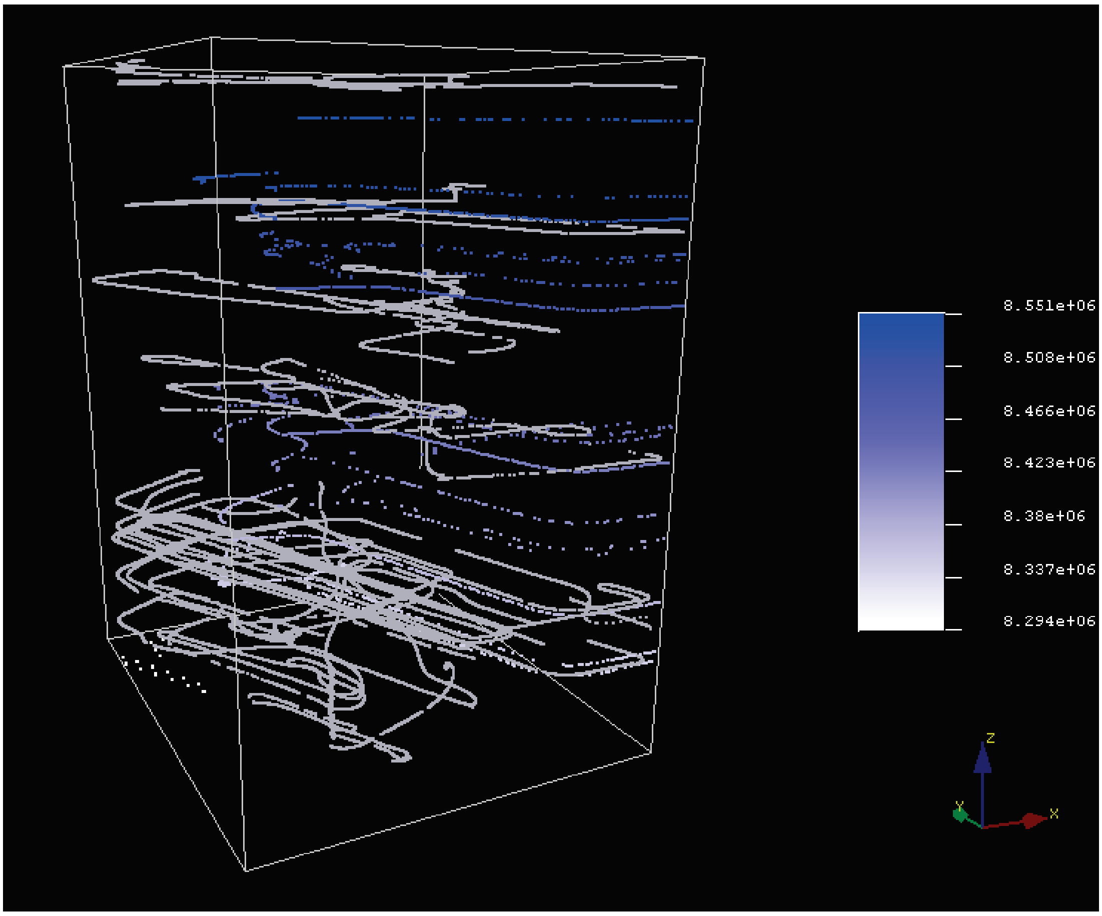
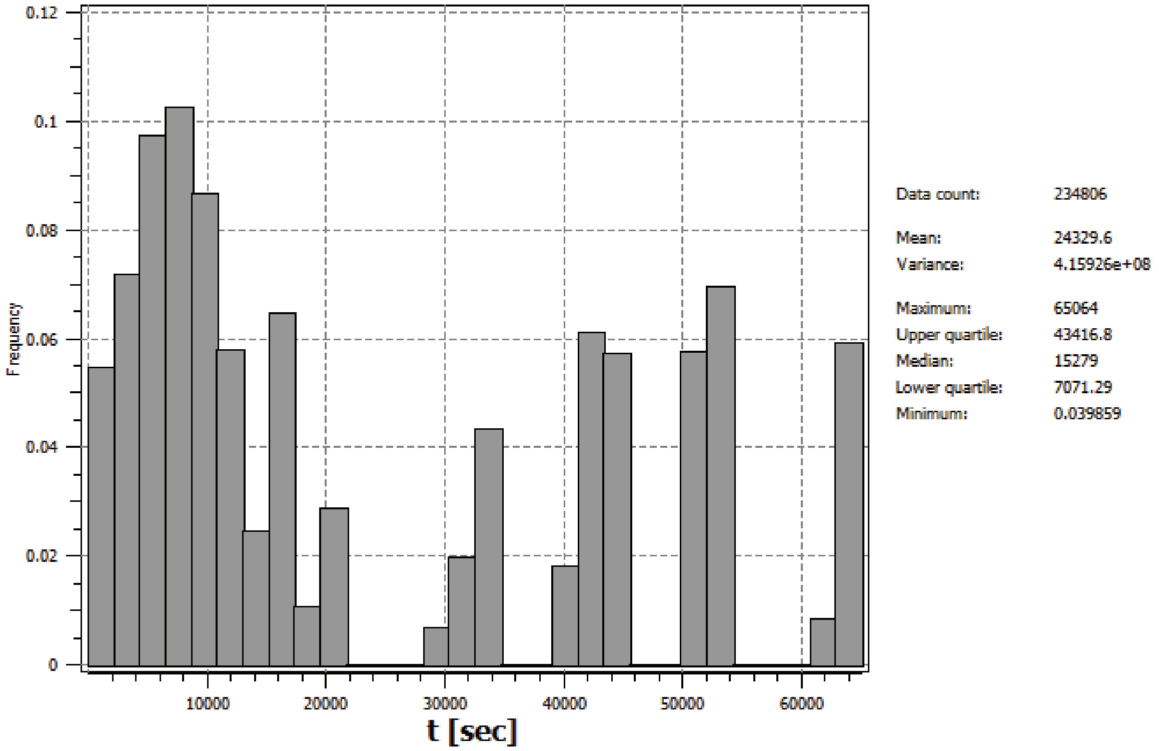


Figure 3.13.: Spatio-temporal distribution of measurement and aircraft postitions

Figure 3.13 shows the spatio-temporal distribution in a time-space cube, where the z coordinate denotes time of the MU's georeferenced WLAN measurement data in grey and aircraft position coloured in blue. The blue colour simply indicates a continuous index of aircraft positions and helps to distinguish between measurement data and aircraft positions. The recording of the presented data set was done by one MU on 1st of February 2012, over a period of 18 hours from 3:58h to 22:23h UTC. Both data sets are spatially delimited by the testbed area; the WLAN data set counts 234806 points, whereas the aircraft positions count 4064 data tuples. This is due to the fact of higher data acquisition rates of the WLAN measurement unit and higher movement frequency of the MU in relation to aircraft. Note that not all aircraft are enabled to broadcast their position in ADS-B. This depends on aircraft's status of avionic technology and the activation of ADS-B message broadcast on ground.

The data overview in figure 3.13 shows that positions are not equally distributed over time and space. This depends on characteristic routings of aircraft and MU in time and space. In the case of the MU the unit stops recording measurement data approx. 15 min. after ignition stop by the driver. At the outer boundaries in x-y dimension the MU track is sometimes interrupted. This is caused by the spatial join of testbed area and position data. Data outside the testbed area are not considered in this analysis.

The distribution of WLAN measurements over time is plotted as histogram in figure 3.14, with general statistical parameter of the data set on the right hand side.



Data count: 234806
Mean: 24329.6
Variance: 4.15926e+08
Maximum: 65064
Upper quartile: 43416.8
Median: 15279
Lower quartile: 7071.29
Minimum: 0.039859

Figure 3.14.: Histogram of WLAN measurement data

The histogram depicts the entire time frame of the data set recordings about approx. 18 hours. The histogram as such and the median of 15279, cutting the precentile at 50%, shows that major recording activity of the MU takes place in the first third of the time frame. In this context the recording activity is not necessarily equal to movment activity, as position recording is solely dependent on ignition activity. The upper boundary of the timeframe's first third is approx. at 9:48h UTC.

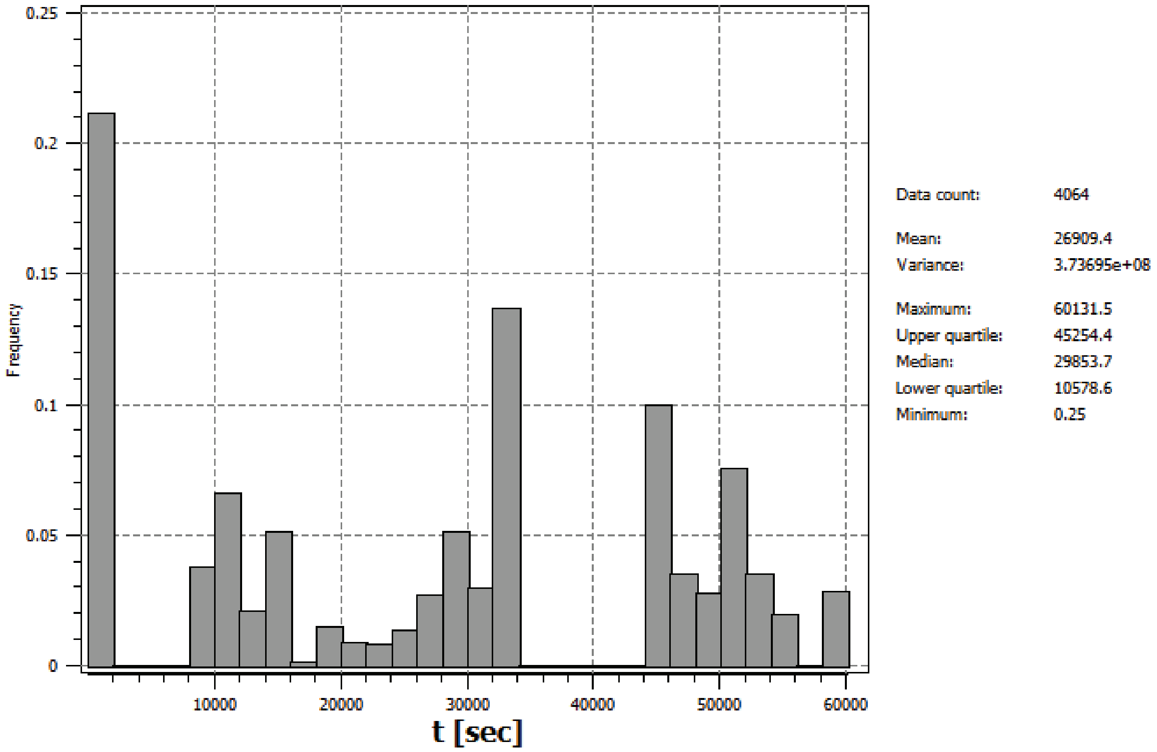


Figure 3.15.: Histogram of aircraft postitions

In contrast to the MU recording activity, figure 3.15 shows the histogram of aircraft positioning within the testbed area. Note that this graph represents the positioning activity of all ADS-B equipped aircraft on the main apron. The median at 29853 in relation to lower quantile at 10578 and upper at 45254 shows, the aircraft positioning activity is more equally distributed over time than the MU's recoding activity. The frequency of aircraft position activity in figure 3.15 on the left is low, in comparison to the MU graph. This is (i) due to the lower position recording rate of ADS-B of approx. 1s, whereas the WLAN interface provides up to 15 measurements per second, (ii) aircraft positions are not interpolated to higher recording rates as necessary at MU measurements and (iii) that not all aircraft are equipped with ADS-B capable positioning technology. However, commercial aircraft are generally equipped with ADS-B capable transponders. This means that especially aircraft positions of general and private aviation have not been acquired.

The expected overlap of MU data and aircraft positions in time is shown by comparison of both histograms, depicted in figure 3.14 and 3.15. Thus, spatial closeness between MU measurement positions and aircraft can be assumed. This assumption has to be considered at further investigations, as interfering multipath influences on WLAN measurements caused by aircraft metal wings and fuselage will be expected. Next, a strategy will be elaborated extracting measurement cluster values out of recorded measurement data.

### 3.3.2. Measurement Reference Value Extraction

In general, measurement data of WLAN field strengths recoded by consumer electronics tends to be noisy. This is (i) due to low measurement resolutions, laying in the available data at 1 dB and (ii) the time-variant characteristic of the overall system. As introduced in section 2.1.1 Principles and Concepts, electro magnetic fields underlay the physical phenomenon of multipath. This is valid for static and dynamic environments, where for the latter the effect is also known as signal fading. Consequently, in the case that measurements take place in close neighbourhood to a taxing aircraft, the measurement value could change dramatically over time.
In order to overcome or at least to reduce that issue, measurement values will be averaged over time at a fixed location. Thereby, measurement values are taken into account where a certain distance criterion to aircraft is given. At this first approach, the assumption is made that recorded field strength is independent on small aircraft and apron vehicles. This is due to the fact that at current state only position data of commercial aircraft have beev made available. For further field modelling approaches precise reference values of field strengths have to be taken into account. The main task of the following subsection is the extraction of that reference values, out of the bunch of recorded measurement values.

Basically, for the generation of field strength reference values several measurement values have to be considered at fixed location. Over these measurement values the actual measured power value can be averaged over time, resulting in a more precise field strength reference value. In order to obtain measurement values at fixed location, a filtering process was elaborated enabling spatio-temporal cluster detection.
Due to the enormous size of available measurement data (234806 within 18 hours!) an overall cluster analysis is not practicable in terms of computational effort. To overcome that issue, a simple filter strategy on the MU's ground speed, delivered by the on-board GPS receiver, is applied. Additionally, the positions's DOP value was considered, in order to consider reliable positions only. Figure 3.16 points out the measurement cluster generation process as implemented in scope of this thesis.

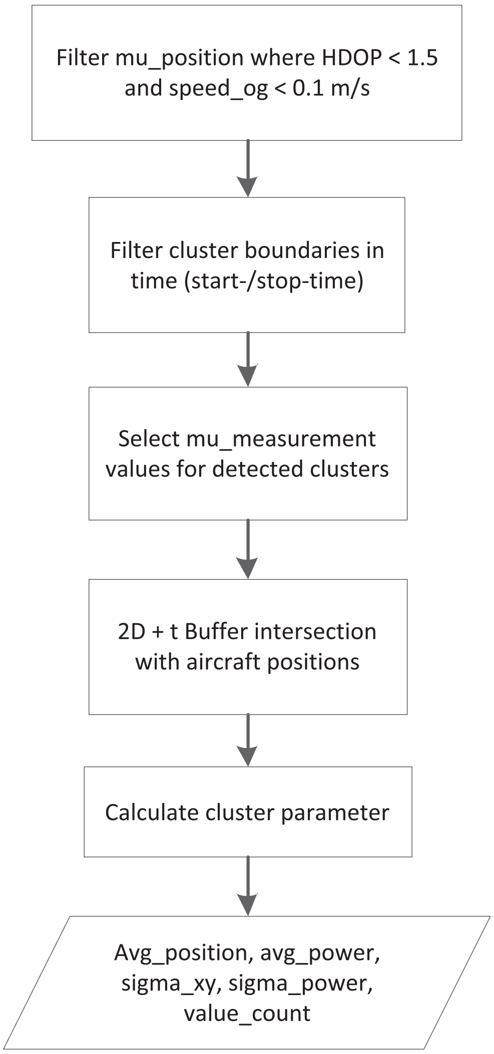

Filter mu_position where HDOP < 1.5 and speed_og < 0.1 m/s
Filter cluster boundaries in time (start-/stop-time)
Select mu_measurement values for detected clusters
2D + t Buffer intersection with aircraft positions
Calculate cluster parameter
Avg_position, avg_power, sigma_xy, sigma_power, value_count

Figure 3.16.: Measurement cluster generation process

The process starts with filtering the available measurement values by speed over ground and HDOP. Thereby, the maximum ground speed is adjusted to 0.1 m/s in order to consider noise effects in ground speed computation by the GPS receiver. Next, the filter algorithm divides the selected measurement values in clusters, based on a special timing criterion. Thereby, all non-moving measurement values are clustered, which are separated less than a certain amount of time. For the actual clustering process the time gap was selected to 3 seconds in order to reduce the sensitivity on recording gaps in available position values. Thus, the process determines for each cluster a start and stop time. On the basis of start and stop time, the interpolated position measurement values of MUs will be selected.

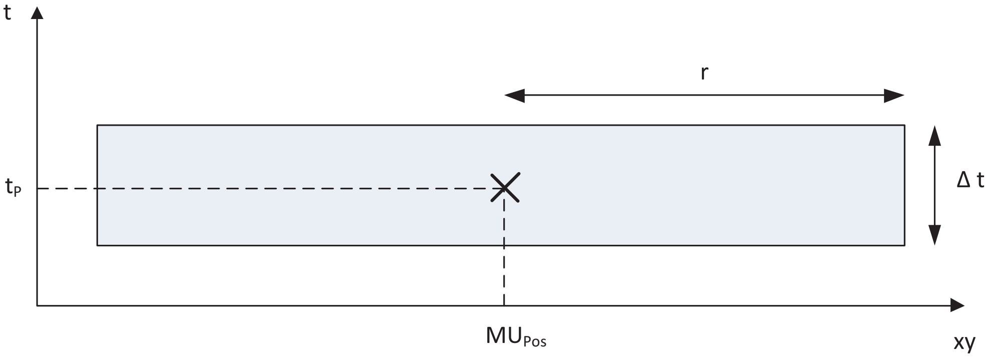


Figure 3.17.: Cross section of time space buffer concept

In the following step the selected position measurement values will be filtered, if close spatio-temporal relationship to aircraft positions does not exist. This is required, as aircraft and MU are moving in space and time. In order to achieve that, a three-dimensional buffer was constructed around each interpolated MU position, as depicted in figure 3.17. Around the actual $\text{MU}_{Pos}$ a radius $r$ was constructed in the space domain and a buffer hight of $\Delta t$ was selected in the time axis. This results in a flat cylinder buffer shape, allocated in time and space. This buffer is now intersected with available aircraft positions. If intersection occurs, the $\text{MU}_{Pos}$ is not considered further for the cluster calculations. In the actual filtering, the buffer radius $r$ was selected to 100m, resulting in 92,61 dB signal attenuation at loss-free signal reflection, determined by equation 2.3 at a total signal path of 200m. The buffer hight $\Delta t$ was adjusted to 2 seconds, in order to guarantee intersection to aircraft at a position acquisition rate of 1 Hz.
After cluster creation and filtering, the actual cluster parameters can be determined. In this case, value count, average power of the received signal and standard deviation $\sigma_{xy}$ in xy direction and for the received signal power $\sigma_P$ is computed. The standard deviation sigma is determined by following equation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \qquad \text{where } \mu = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{3.8}$$

$N$ denotes the number of measurements per cluster and $x_i$ the discrete cluster measurement values. The process ends finally with these five resulting cluster parameters.

The computational result of the above described process yields in a count of 184 clusters over 234806 measurement values and a recording time of 18 hours. Figure 3.18 shows the exemplary result of the cluster generation process for AP mast6a. Note that the circle diameter of the cluster symbol correlates with processed parameter $\sigma_P$, the standard deviation of power measurement values. Therefore, the lower the diameter of the cluster circle, the more precise the power measurement in terms of variance.

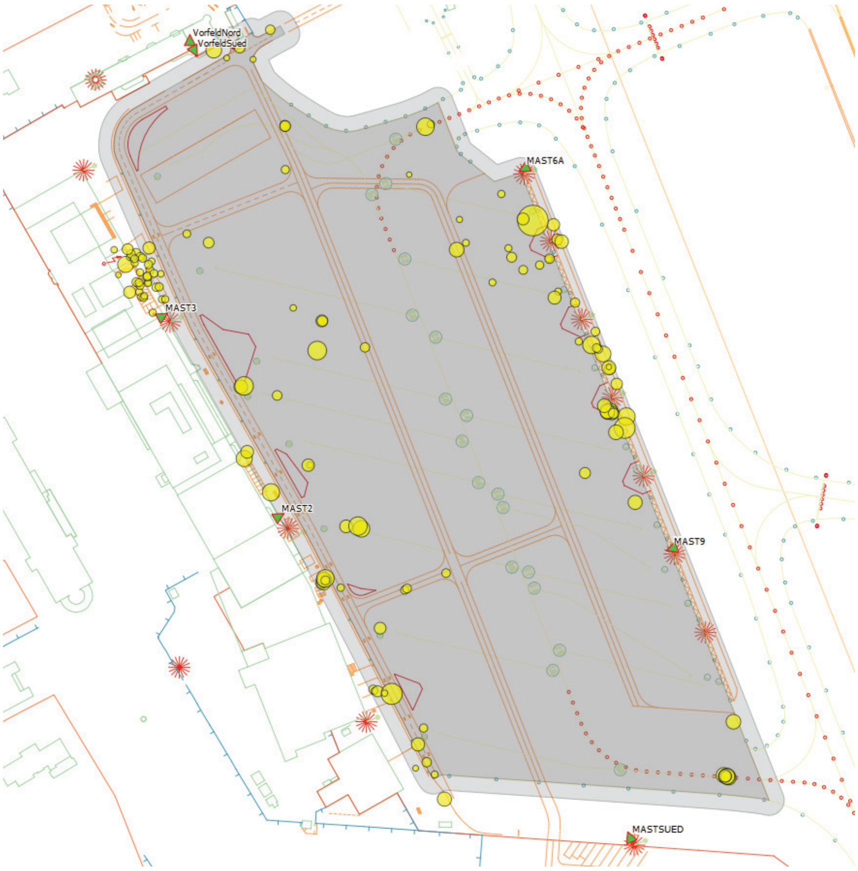


Figure 3.18.: Spatial distribution of cluster generation process result for AP mast 6A

The spatial distribution of cluster location is concentrated on the east and west boundary of the main apron. This depends on the moving profile and typical holding positions of the MU.
A summary on the statistics of calculated cluster parameter is shown in table 3.6.

| parameter | min | max | mean | median |
|---|---|---|---|---|
| $\sigma_P$ | 0 | 4.1751 | 1.3594 | 1.2359 |
| $\sigma_{xy}$ | 0 | 66.50 | 1.3143 | 0.4034 |
| count | 1 | 5885 | 278.51 | 39.0 |

Table 3.6.: Cluster parameter statistics of AP mast6a

A standard deviation minimum of 0 occurs if either all measurements have the same value or the cluster count is equal to one. Note that $\sigma_{xy}$ maximum of 66.50 is a typical statistical outlier. This is emphasised by the median parameter of 0.4034, separating the higher and the lower half of the sample.
Generally, it is to discover that standard variation $\sigma_{xy}$ and decreases and $\sigma_P$ increases with higher value count of the cluster. This might be occurred on higher influence of multipath effects to WLAN power measurements. The longer the recording of measurement values at a static position at the main apron takes, the higher the probability of signal fading and multipath effects, caused by passing by vehicles.

For that reason a second filter was implemented, focusing spatially on the inner test-bed area, depicted in figure 3.18 as underlying polygon in dark grey. Furthermore, it takes the measurement clusters' standard deviation of power and position into account. In this filter for $\sigma_P$ a maximum value of 1.5 is accepted, a maximum value of 0.3 for $\sigma_{xy}$ and a minimum cluster count of 5 measurements per cluster.

| parameter | min | max | mean | median |
|---|---|---|---|---|
| $\sigma_P$ | 0 | 1.496 | 1.034 | 0.999 |
| $\sigma_{xy}$ | 0.023 | 0.226 | 0.123 | 0.112 |
| count | 5 | 144 | 34 | 18 |

Table 3.7.: Cluster parameter statistics AP mast6a of final measurement reference values

The results of the final filtering in table 3.7 shows, the cluster's maximum standard deviation $\sigma_P$ and $\sigma_{xy}$ could be fairly reduced. This filtering result counts now 24 clusters and acts as a reasonable basis in size and precision for further model calibration purpose.

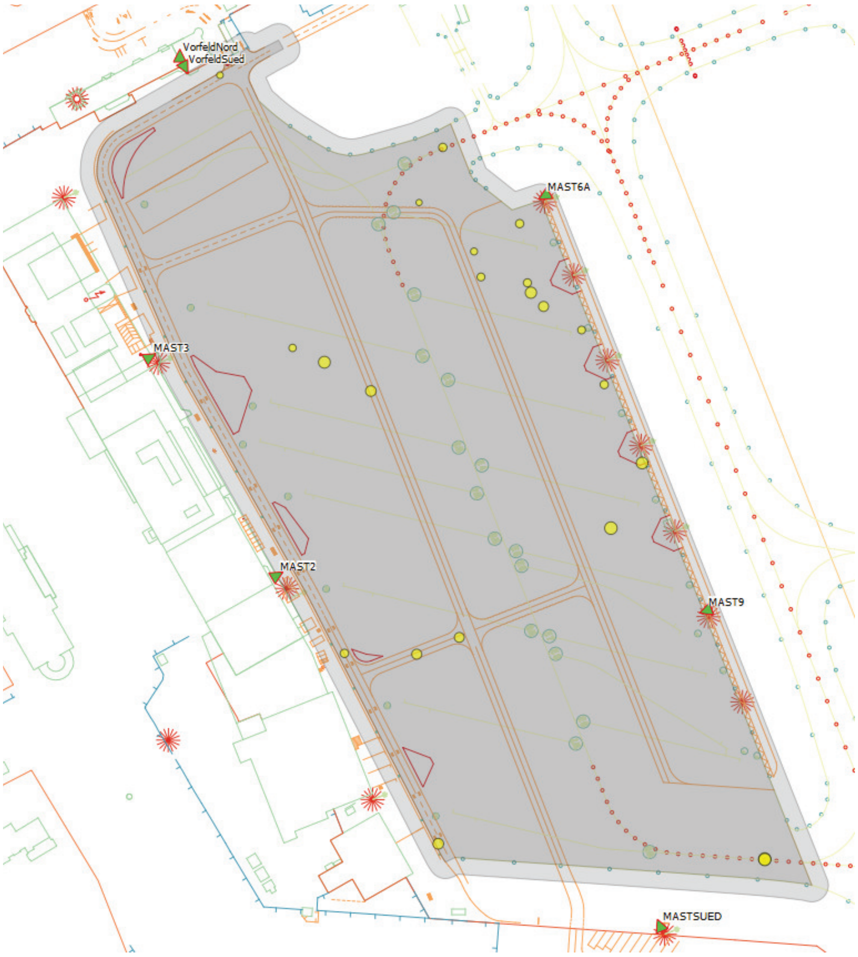


Figure 3.19.: Final measurement reference value distribution for AP mast 6A

Figure 3.19 shows the processing result of the final measurement reference value extraction process, spatially delimited to the actual testbed area, the main apron of Airport Salzburg.

The process of measurement reference value extraction has shown that cluster extraction is a possible method to condense huge data sets. However, it is always a trade-off between processing effort and choice of filter parameter. The consideration of aircraft positions has shown a minor influence on the filtering process, as to less MU clusters (2!) were affected by intersection of the defined spatio-temporal filter criterion. This might be justified by low density of available aircraft positions. A higher density of available aircraft positions, even if small aircraft and mobile units would be considered, would increase the scientific evidence of this research.

## Conclusions

The thesis project involves many different data sources of either static or dynamic nature, in time and/or space domain. For structuring of these data, an appropriate data model is essential. This chapter has proposed a object relational data model, aggregating dynamic and static data in a generic way. That constitutes the groundwork for further on-going processing and research of this and other projects, related to field modelling in general. The two dimensional interpolation of dynamic positioning data, has opened a strategy for correlation of position data with attributive electro-magnetic field strength information. Without that solution, dynamic data from different and not synchronised sensors platforms could not have been merged together. This is actually a very common problem in data acquisition of multiple, not synchronised data sources, as it is often the case e.g. in sensor networks or meshed geo-applications. The spatio-temporal characteristic of available dynamic data sources has been shown by analysis. On that basis a three-dimensional buffer concept was introduced in the time-space domain, for separation of positioning categories. These all together, have been resulted in a process definition for measurement reference value extraction. This process definition has laid the basis for algorithmic implementation, allowing automised computation of measurement reference values. Nevertheless, a higher density of available aircraft positions would have led to more representative result of this analysis.
The next chapter of this thesis builds upon these measurement reference values, with different field modelling approaches and utilises the introduced data model for RM processing.

# 4. Field Modelling

As the thematic part of the thesis title, “Modelling of Continuous Fields”, already states; field modelling is one of the central questions of this research. In connection to WLAN (Wireless Local Area Network) infrastructures, this raises the question: What field model is robust enough to cope with the highly sensible electro-magnetic field? This chapter tries to give an answer by implementation and analysis of deterministic, probabilistic and combined modelling approaches. First of all, the parameter model is shortly revised, followed by the implementation of an extended radial model approach, considering antenna characteristics. The second part deals with the implementation of probabilistic models, based on geo-statistical Ordinary Kriging interpolation. Therein, a new concept of mixing the implemented parameter model with probabilistic interpolation will be analysed. This chapter closes by qualification of implemented modelling methods against a reference track compound of field measurements.

## 4.1. Parameter Model

In the introduction to modelling approaches in section 2.2 Methods for Radio Map Generation a series of deterministic electro-magnetic models as the radial model, MWM (Multi Wall Model) model, ray tracing model and DPM (Direct Path Model) model have been introduced. All that models have been considered as parameter based, whereas the latter three are mostly utilised in in-door positioning environments; thereby, accurate infrastructural building information is necessarily required. This research elaborates a extended radial modelling approach, taking beyond fairly simple isotropic radiation the directivity characteristic of applied patch antennas into account. That results in an ideal electro-magnetic field distribution model for obstacle free outdoor environments, constituting the basis for additional model calibration.
First, the conceptual implementation of the parameter model is introduced, in accordance to the available antenna pattern. That illuminates the modelling approach, resulting in a first preliminary RM (Radio Map), available as geo-referenced raster data set. The second part of this section proposes an empirical method for model calibration within the testbed.

### 4.1.1. Model Implementation

The radial model is based on the radial symmetry in electro-magnetic field distribution. Thereby, the assumption is made that the field strength of an isotropic radiator is exclusively dependent on the distance $d$ between receiver and transmitter antenna, at a given wave length $\lambda$, (Haykin 2001).

Atmospheric influences have not been further considered in that approach, as they have minor influence at close receiver transmitter distances. The concept of path loss models the physical key concept of the radial model and is given by equation 4.1, already derived in section 2.1.1 Principles and Concepts.

$$PL(d) = -10\ log_{10}(G_t G_r) + 10\ log_{10}\left(\frac{4\pi d}{\lambda}\right)^2 \quad [\text{db}] \tag{4.1}$$

A model based on that equation implies omnidirectional receiver and transmitter antenna characteristic. Otherwise, the antenna's gain directivity has to be taken in to account. This is the case in the given testbed scenario, where highly directional antennas have been installed. Consequently, equation 4.1 has to be adjusted:

$$PL(d,\delta) = -10\ log_{10}(G_t(\delta) G_r) + 10\ log_{10}\left(\frac{4\pi d}{\lambda}\right)^2 \quad [\text{db}] \tag{4.2}$$

The function $G_t(\delta)$ describes the transmitter's antenna directivity in dependence of the angle $\delta$ in reference to the antenna's boreside direction. The course of that function is given by previously processed antenna directivity data, in section 3.2.2 Data Modelling. The receiver antenna gain $G_r$ remains constant due to omnidirectional antenna characteristic.
On basis of the model equation 4.2 a preliminary RM will be implemented. The RM itself is represented by a multi-channel raster coverage, as already discussed in section 2.2.1 Definition of the Radio Map. Thereby, each raster cell of the n-dimensional RM matrix $X$ corresponds to a discrete sampling vector $\vec{x}_{ij} = ({rss_{ij}}^k)$ of the field model. Thus, the RM is compounded of $k$ "layers", one for each AP (Access Point) of the testbed's communication infrastructure. The parameter $\theta_{ij}^k$ of the generic RM model, introduced in section 2.2.1 Definition of the Radio Map by equation 2.6, is not applicable for the implementation of this parameter model approach and therefore eliminated.

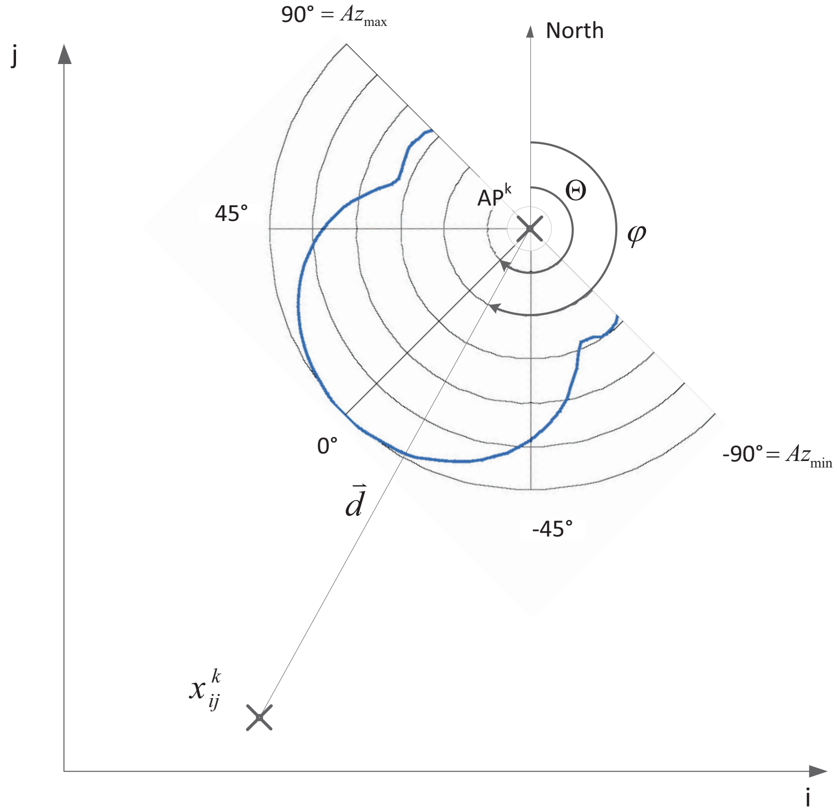


Figure 4.1.: Geometrical relationships for RM computation

The geometrical relationships for RM creation are depicted in figure 4.1. For each column $i$ and row $j$ of the raster matrix coverage $X$, the field model is iteratively sampled for $k$ APs. In order to start the computation process, the location and horizontal orientation of the AP antenna has to be retrieved from the DBMS (Database Management System). This is given by the geographical position $AP^k$ and the antenna boreside direction $\Theta$ in reference to geographical north. In the following iterative computation process the vector $\vec{d}$ is determined for each raster cell centre $x^k_{ij}$:

$$\vec{d} = x^k_{ij} - AP^k \tag{4.3}$$

The Euclidian norm $||\vec{d}||_2$ gives the geometrical distance $d$ between raster cell $x^k_{ij}$ and the antenna phase centre. Out of that result the PL (Path Loss) could be computed by equation 4.1, without taking the antenna characteristic into account.

For consideration of the antenna characteristic $G_t(\delta)$, the angle $\varphi$ in reference to geographic north has to be determined first. This is archived by:

$$\begin{aligned}
\varphi &= \tan^{-1}(d_x/d_y) && \forall \quad x \geq 0 \land y \geq 0 \\
\varphi &= \tan^{-1}(d_x/d_y) + 180^\circ && \forall \quad y < 0 \\
\varphi &= \tan^{-1}(d_x/d_y) + 360^\circ && \forall \quad x < 0 \land y \geq 0
\end{aligned} \tag{4.4}$$

In this case, the arctangent is calculated by $x/y$ in order to determine the angle $\varphi$ in reference to geographic north. On that basis the corresponding angle $\delta$ for antenna gain determination can be computed accordingly:

$$\delta = \Theta - \varphi \tag{4.5}$$

With the resulting values of the variables $d$ and $\delta$ the field model can be computed by equation 4.2 over the testbed area, for each of the available APs.
For the resulting raster coverage a 16-bit, six channel raster space is allocated in the spatial DBMS. This allows to store of a six layer RM with a numerical quantisation of $2^{16}$ steps per cell value. For the prototype RM a cell value resolution of 0.01 dB was selected, yielding into at a value range from 0 to 655.35 dB.

Figure 4.2 and 4.3 shows the exemplary raster coverage result of path loss computation over the testbed area. For the purpose of fast computation, a fairly coarse raster cell size of 3m x 3m was selected, however sufficient for discovery of the modelled phenomenon. Additionally, in the process of model computation, a spatial filter criterion was introduced, computing only values within the testbed area. Consequently, the resulting raster coverage is shaped according to the testbed area.
The resulting map of the electro-magnetic field distribution, according to the applied path loss modelling, shows the significant influence of the antenna directivity on the field distribution. The field distribution's characteristic is far beyond a radial symmetry around the antenna's phase centre. The main and side lobe influences of the antenna pattern in figure 4.1 are explicitly recognisable in the field distribution. The comparison of figure 4.2 and 4.3 shows, the applied antenna is basically more suitable for horizontal narrow directed than broad field coverage.

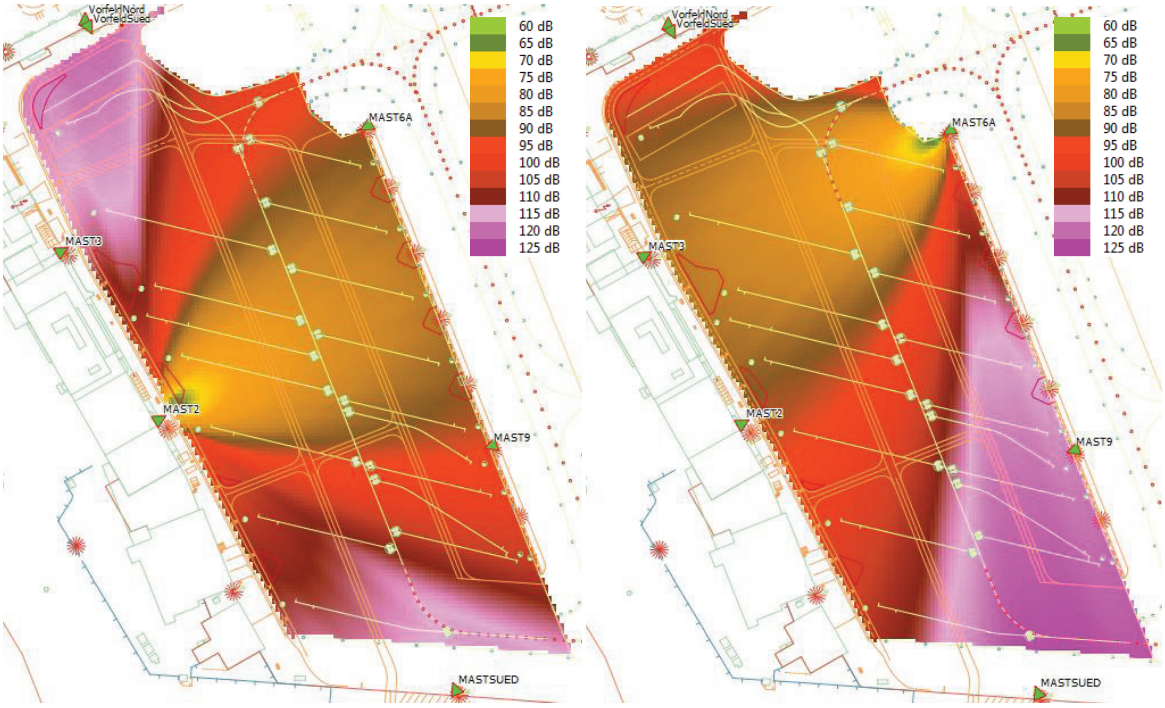


Figure 4.2.: RM of APs MAST2A on the left and MAST6A on the right hand side

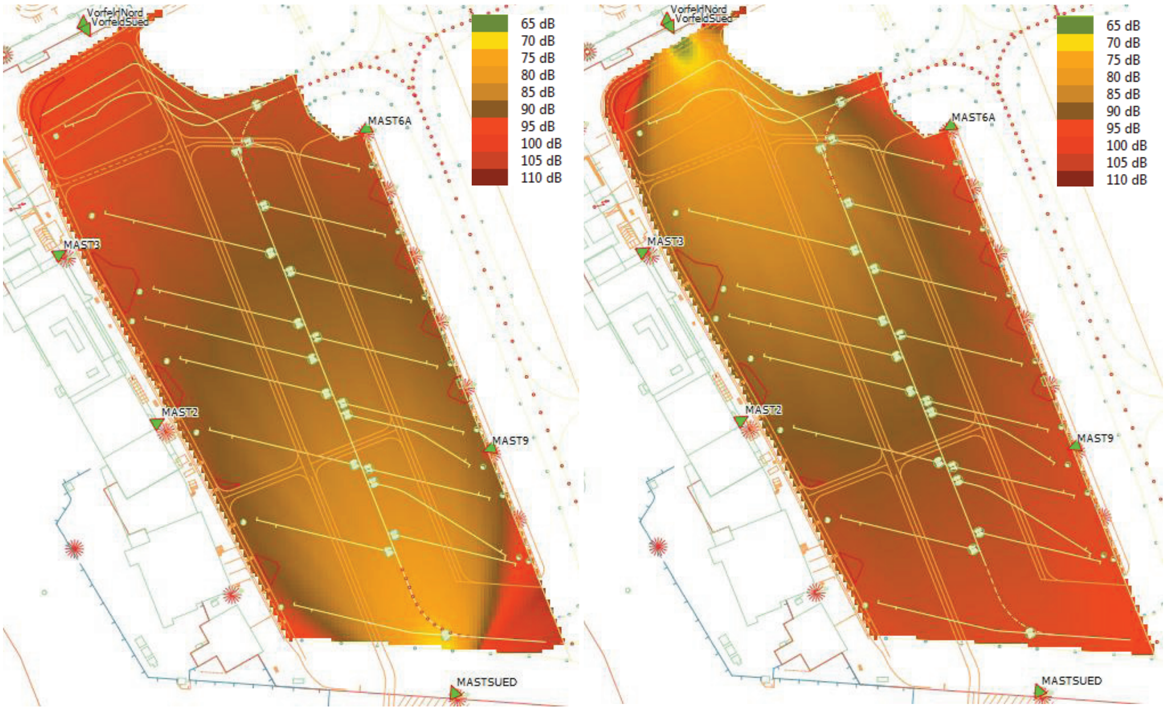


Figure 4.3.: RM of APs MASTSUED on the left and VORFELDSUED on the right hand side

Although the modelled raster coverages present the path loss, they do not reflect the physically received signal strength at the MU (Mobile Unit)'s receiver side. This is due to the fact of unknown transmitted power of APs, receiver antenna gain and attenuation of the receiver insertion network. In order to obtain a signal strength surface of the received power, the model has to be calibrated with reference measurement values. This will be elaborated in the upcoming section, model calibration.

### 4.1.2. Model Calibration

The focus of this section is laid on the calibration of the previously composed parameter model of electro-magnetic field distribution. The introduced calibration process is based on measurement reference values, which has been gathered in section 3.3.2 Measurement Reference Value Extraction. The main target of this section is the extraction of a constant field model calibration parameter out of processed measurement reference values. Consequently, the calibration parameter has to be located in the power domain. For that the signal strength value $p_{avg}$ of measurement reference values will be utilised, referring to the overall average power of a measurement cluster.

The calibration process is compounded of several stages. Right at the beginning the power delta $\Delta p$ between the RM field model and the measurement reference values has to be determined at their dedicated reference value location. This is simply realised by raster value point requests at the geographical position of measurement reference values. Consecutively, the average power value $p_{avg}(x, y)$ is subtracted from the gathered RM value $rss_{ij}$. Applying that process, again to the example of AP mast6a, this yields in 24 values of $\Delta p$. Note that RM values $rss_{ij}$ are discrete raster values, whereas $p_{avg}(x, y)$ are vectorised and regarded as continuous in the space domain. The failure imposed by the discrete raster data set is dependent on the raster cell size, however that issue will not be further investigated within the scope of this thesis.

This raises the question of how to obtain a unique calibration parameter, out of several $\Delta p$ values, for the above determined electro-magnetic field model. This calibration parameter should meet the best suitable calibration of the field model in the power domain. In order to tackle that goal, least square approximation for overdetermined systems is applied, originally introduced by Carl Friedrich Gauß (Bjõrck 1996). That approach corresponds to the maximum likelihood criterion if the errors of the experiment are normally distributed. The linear model that should be approximated in this calculation is fairly simple the calibration parameter $a$. This parameter should be best fitted, which is the case if the sum of squared residuals, the error function $E(f)$, is minimum:

$$E(f_i, a) = \sum_{i=0}^{N} (\Delta p_i - a)^2 \tag{4.6}$$

Note that in this case the function $f$ represents the discrete delta values $\Delta p_i$. In order to obtain the solution of $|E(f_i, a)|_{min}$ this problem is solved numerically.
Additionally, a weighted criterion is considered, depending on the count $c_i$ of originally underlying measurement values per cluster. This allows a direct proportional weighting of measurement

reference values as a function of cluster impact. For that function $E(f_i, a)$ is supplemented by the weight $w_i$:

$$E(f_i, a, w_i) = \sum_{i=0}^{N} (\Delta p_i - a)^2 w_i \tag{4.7}$$

where

$$w_i = \frac{c_i}{\sum c_i} \tag{4.8}$$

The graphical plot of the numerically solved error function $E(f_i, a, w_i)$ is shown in figure 4.4. Note that both functions are normalised within the depicted range from 10 to 30 dB.

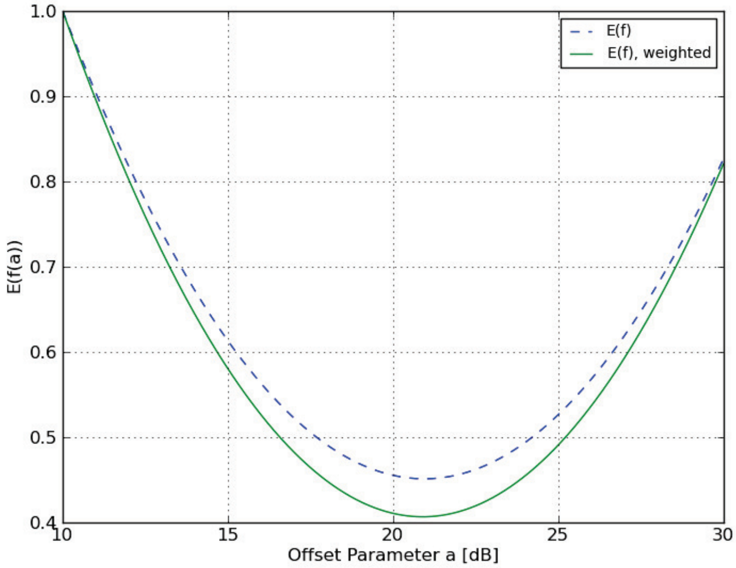


Figure 4.4.: Numerical solution of error function $E(f_i, a)$

The numerically exact solution for $|E(f)|_{min}$ is 20.891 dB for $E(f_i, a, w_i)$ and 20.924 dB for $E(f_i, a)$. In this processing example of AP mast6a, there exists only a minor deviation in the difference of both error function minimums. Nevertheless, the absolute minimum of the weighted error function, 20.891 dB, is considered as resulting offset parameter $a$ of the approximation process.

The final result of the calibrated RM for mast6a is shown in figure 4.5. The RM represents the estimated signal attenuation at certain location in dB, which correlates to the negative path loss value. Furthermore, figure 4.5 shows positive deviation of reference measurement values to the calibrated RM coloured in green and negative deviations in red. It shows inside the region of the antenna main lobe, deviations of the RM are mainly negative and outside positive. This indicates that the antenna directivity is significantly lower in practical applications than specified by its data sheet.

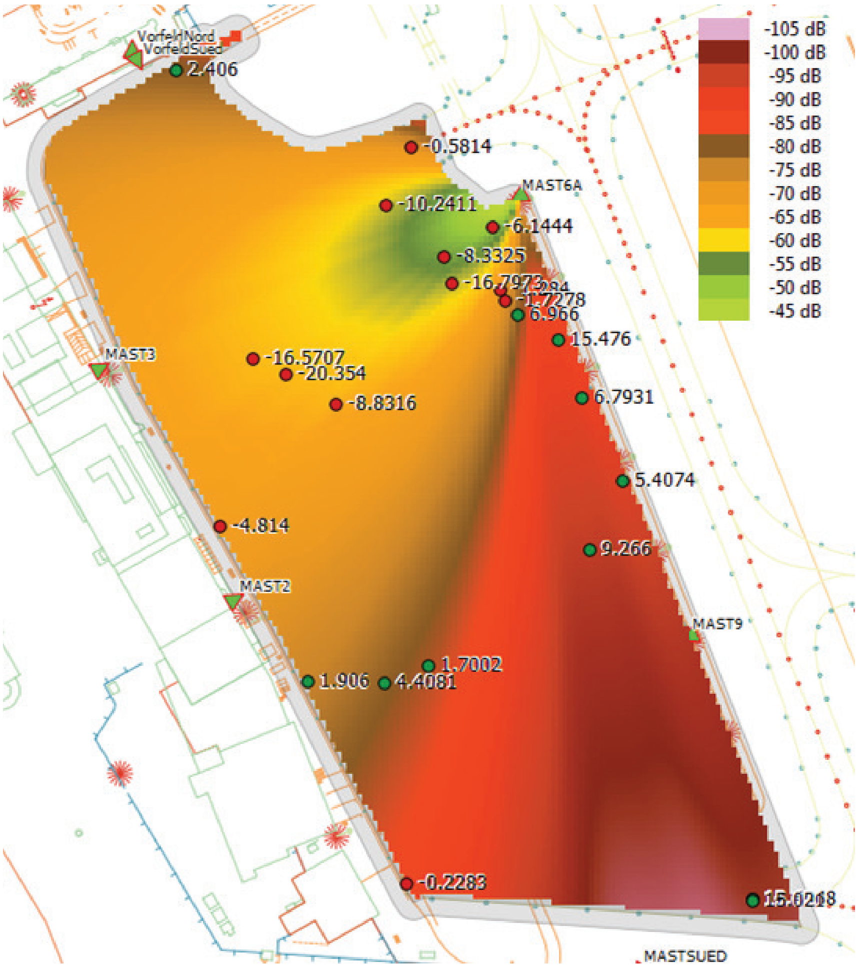


Figure 4.5.: Calibrated radio map of AP mast6a with deviations to reference measurements

The modelling approach of the parameter model has shown that the result is highly dependent on the quality of the underlying antenna pattern. The applied antenna pattern provided by the antenna manufacturer does by nature not consider the installation environment. This however affects the resulting antenna characteristic significantly. The applied least square approximation method for RM model calibration does not regard that effect. In order to consider that influence, it is recommended to calibrate the antenna pattern as well. Further research on a method for in-situ calibration of the antenna pattern would significantly enhance that modelling approach.

## 4.2. Probabilistic Interpolation

The last section about calibration of the parametric field model has taught that the ideally computed field model deviates partially a lot from the actual reference measurement clusters' values. This is based on the fact that we have globally shifted the field model in its power axis, approximated best for reference measurements across the testbed area. In order to overcome that limitation of the model's inflexibility, the probabilistic Ordinary Kriging interpolation method is

applied in this section.
This will be done in two different approaches. At first, the Ordinary Kriging method is exclusively applied to measurement reference values. On a second approach, the same interpolation technique is applied to interpolation points compound of measurement reference values mixed with values of the calibrated parameter model of the previous section. This measure should reduce expected oscillations on the surface boundaries. Following, both variants will be illuminated in terms of data exploration, interpolation and error surfaces. In this case, the applied approach to both variants will be discussed in parallel, in order to obtain the direct comparison.

### 4.2.1. Data Distribution

The entire interpolation point basis is shown in figure 4.6 by black crosses. While interpolation points based on measurement reference values are highlighted in yellow. The processing of the measurement reference values has been already explicitly described in section 3.3.2 Measurement Reference Value Extraction.

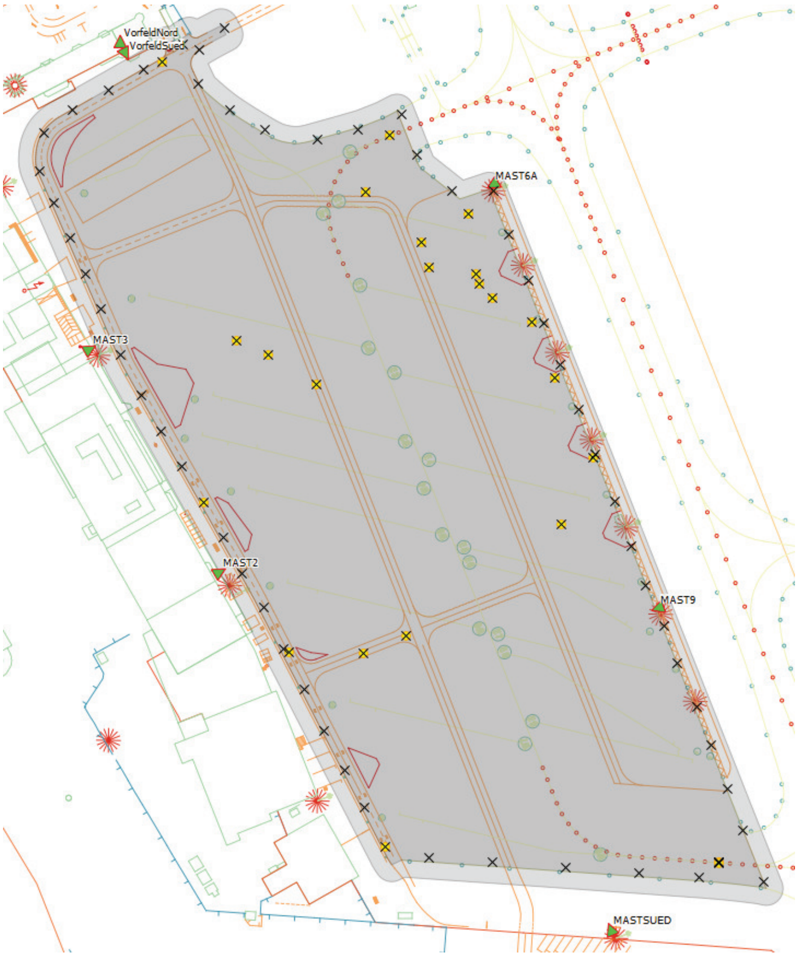


Figure 4.6.: Interpolation point distribution of measurement reference values (highlighted in yellow) and artificial boundary points

Without explicitly applied point pattern analysis it is recognisable; the cluster centroids are not randomly distributed over the testbed area as clearly shown in figure 4.6. The sample data set is spatially clustered around the upper right part of the testbed area next to AP mast6a. This is mainly dependent on the moving pattern of the WLAN sensor and varies over time. In order to overcome the expected bad influence of clustering on the boundaries of the interpolation surface, two interpolation approaches will be applied. Whereas the first interpolation approach exclusively uses the measurement reference points, for the second boundary values of the calibrated parameter model, will be incorporated. These boundary values are equally distanced by approx. 25m on the border of the testbed area.

Next, the explorative data analysis will be proceeded by comparison of the second order trend analysis function of both interpolation approaches. The trend analysis functions shown in figure 4.7 were processed with ESRI's ArcGIS Geostatistical Analyst[1] toolbox. Both trend functions show in common the same tendency of the electro-magnetic field distribution. The field strength declines outgoing from the radiation source AP mast6a in the upper right corner. Additionally, the expected antenna directivity characteristic can be recognised in both figures. This can be derived from the unsymmetrical field characteristic towards the right hand boundary of the trend function, where the field strength declines. Both trend functions compared to each other show, the antenna boreside direction is slightly different. The measurement cluster trend function points more towards geographic south than the other. Moreover, at the testbed area boundaries, the trend function of the measurement cluster parameter model combination counts lower values. This is due to the influence of the calibrated parameter model, where far from the power source distanced locations are rated lower in terms of field strength.

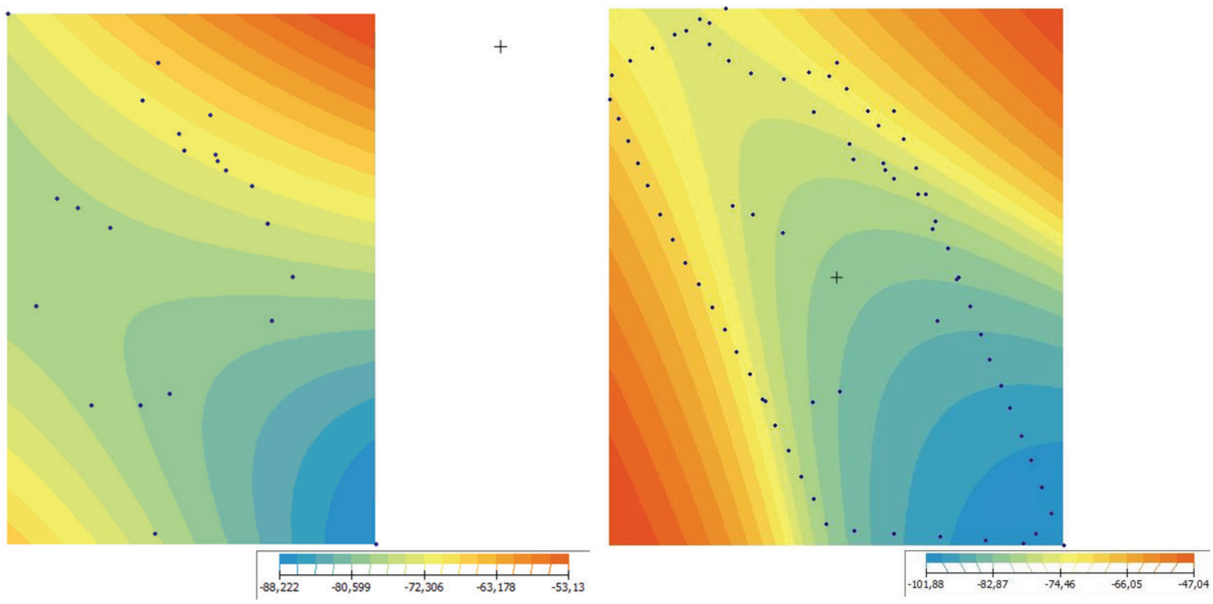


Figure 4.7.: Second order trend surface for measurement cluster centroids (left hand side) and with supplemented boundary points (right hand side)

[1] http://www.esri.com/software/arcgis/extensions/geostatistical

For the analysis of spatial variance within both datasets, the semivariance is modelled in figure 4.8.

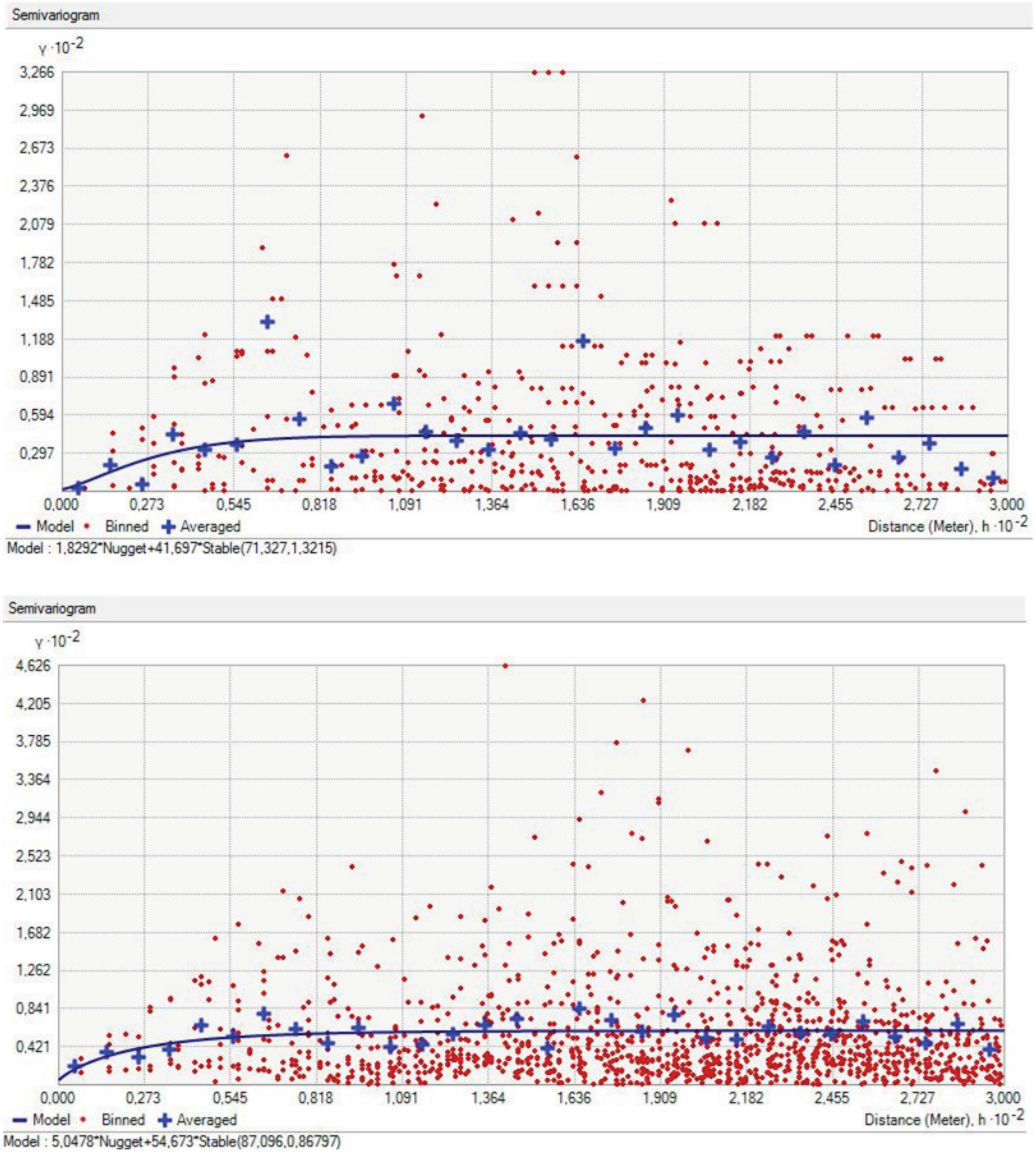


Figure 4.8.: SemiVariogram of measurement cluster centroids (at the top) and with supplemented boundary points (at the bottom)

For the semivariogram the parameter were chosen according to the following rule-of-thumb:

$$\frac{1}{2}d_{max} = \quad \text{lag}_{size} \cdot \quad \text{lag}_{num} \tag{4.9}$$

The maximum distance $d_{max}$ between points inside the testbed area is measured to approx. 600 m. As local variances should be considered in these variogram analysis the lag$_{size}$ parameter was adjusted to 10 m and the lag$_{num}$ to 30 accordingly. For the actual semivariogram model the exponential type was chosen, fitting best for electromagnetic field strength models according to (Konak 2010), as power of electromagnetic waves attenuates significantly within the first few meters. According to (Konak 2010) the exponential model is defined as,

$$\gamma(h) = C_0 + (C_1 - C_0)exp\left(1 - \frac{3h}{R}\right) \tag{4.10}$$

where $C_0$ is the nugget effect, $C_1$ is the sill parameter and R is the range parameter. At distances beyond $R$ correlation between two points is not any more significant. The nugget effect represents variability at very low distances including measurement errors and other noise parameters. The sill parameter represents the maximum value of the semivariogram model, which is approached asymptotically at exponential models.
The actual semivariogram model parameter values of both interpolation sources are shown in table 4.1.

| interpolation source | $C_0$ | $C_1$ | $R$ |
|---|---|---|---|
| measurement reference points | 1.829 | 41.697 | 71.327 |
| mensurement reference and param. model points | 5.048 | 54.673 | 87.096 |

Table 4.1.: Parameter of the exponential semivariogram model

The direct comparison of both models show that measurement reference values as interpolation basis yield in less nugget effect, maximum model variance and range. This is due to the additionally implied effect of spatial autocorrelation at the calibrated field model. As this model is based on the inherent cubic propagation function of electro-magnetic fields, higher variance between value pairs is given at far distance.

In general, it is to notice that interpolation points derived from measurement reference clusters are not equally distributed over the testbed area. One reason for that is the comparatively low data acquisition time of approx. 16.5 hours. Moreover, the applied data aggregation strategy, reducing the effect of temporal variance in field strength measurements, reduces the number of available interpolation points enormously. This effect could be either reduced by expanding the WLAN data acquisition time or by increasing the number of mobile WLAN sensors, without changing the data aggregation strategy.

### 4.2.2. Ordinary Kriging Interpolation

Based on both previously discussed interpolation data sources, Ordinary Kriging interpolation and its corresponding error surfaces will be illuminated in this section. The theoretical background to the following applied Ordinary Kriging interpolation has been revised in section 2.2.2 Field Modelling Approaches.

The interpolation is based on the exponential model parameters in each case of interpolation data sources, as elaborated in previous section 4.2.1 Data Distribution. In both cases trend elimination is applied of the second order trend surface, as shown in the previous section. Figure 4.9 shows the Ordinary is applied interpolation surface of both interpolation data sources next to each other.

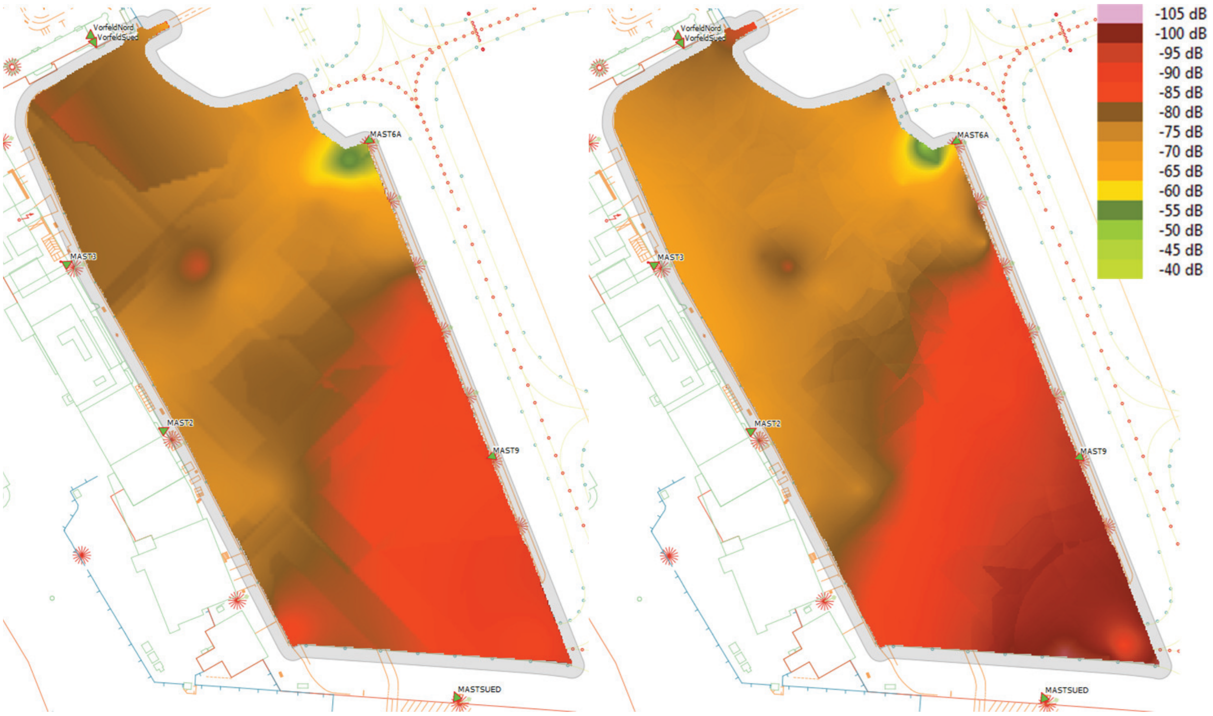


Figure 4.9.: Ordinary Kriging interpolation surfaces over cluster centroids (left hand side) and with supplemented boundary points (right hand side)

The left hand side of figure 4.9 shows the interpolation result exclusively calculated over measurement cluster centroids. In this surface the antenna directivity is clearly recognisable. The radiation source centre can be allocated well and fits with the known radiation location of mast 6a. Due to the spatially clustered distribution of interpolation points, the quality of the interpolation surface across the testbed is different. At locations with reasonable density of interpolation points, as in the near distance of AP mast6a, the surface shows a rather homogeneous characteristic. The antenna main lobe directivity is met quite well in this area. At far distance from the radiation source, especially on the testbed boundary, where density of interpolation points is low, discontinuities within the interpolation surface occur.
This is different in the result of interpolation, taking measurement reference and parameter model points into account, shown on the right hand side of figure 4.9. In this case, the boundary of the interpolation surface is strained to the calibrated parameter model values. This affects the interpolation result on the testbed boundary, resulting in a more homogeneous surface boundary distribution. On the other hand, it does negatively affect the antenna's directivity and main lobe shape nearby the radiation source, where high density of measurement reference values is given. Furthermore, the interpolation surface is strained down to lower signal values than actually measured on the lower right corner of the testbed area. This shows the direct comparison of both interpolation surfaces.

The corresponding error surfaces to that interpolation are shown in figure 4.10. As due to the nature of interpolation on clustered point distributions already expected, the error surface counts high values where density of interpolation points is low.

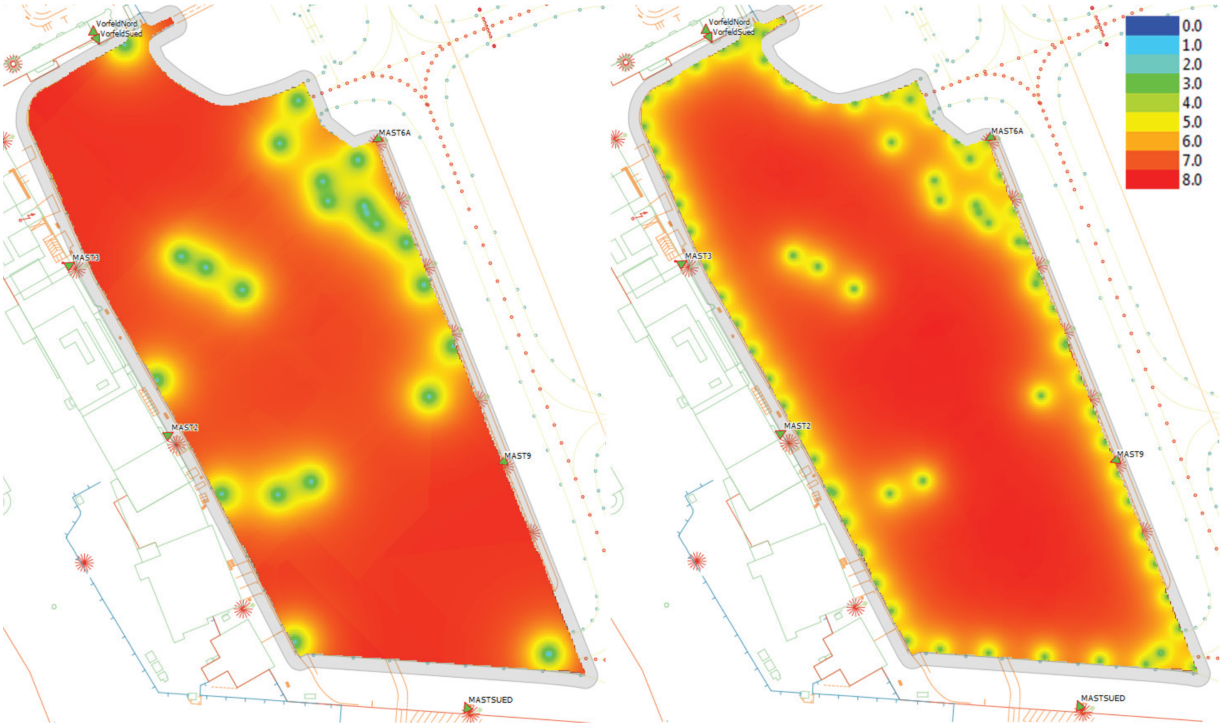


Figure 4.10.: Ordinary Kriging error surfaces over cluster centroids (left hand side) and with supplemented boundary points (right hand side)

The error surface shows that in the case of measurement reference value interpolation, the error of interpolation is lower at locations where reasonable density of interpolation points is given. However, the overall error seems to be lower at the alternative interpolation case, taking the parameter model values into account.
Note that this consideration should not be regarded as an absolute qualification of the interpolated radio map, as this would require the assumption that all interpolation points rely on the same acquisition technique and equivalent data source respectively.

The resulting Ordinary Kriging interpolation surfaces of this section show different characteristics in terms of continuity and boundary behaviour. As both underlying interpolation sources are partially established from different source and accuracy, the Kriging error surface or statistical cross validation techniques would not be meaningful in terms of accuracy estimation and qualification of the produced radio map. Therefore, the upcoming section approaches the issue of surface qualification empirically, by cross validation with reference track measurements.

## 4.3. Model Qualification

This chapter about field modelling discusses basically three methods of field modelling: (i) the calibrated parameter model approach, (ii) Ordinary Kriging applied on measurement reference values and (iii) Ordinary Kriging applied on interpolation points compounded of measurement reference and calibrated parameter model values. All of these approaches have obtained partially different or combined data sources, with different characteristics of inherent acquisition errors. However, basic laws of statistics do not allow to compare applied error or cross validation results with different underlying failure distribution amongst each other.

In order to come across that issue of comparing all three modelling approaches in terms of quality, the way of empirical validation by reference track measurements is taken. For that purpose, at first a single reference track of electrical field strength measurements to AP mast6a is isolated out of the interpolated WLAN measurements, elaborted in section 3.2.2 Data Integration. Second, the actual model evaluation by empirical validation of modelling approaches against the measurement reference is conducted. In this analysis pure statistical parameters and spatial distribution of model deviation will be compounded.

### 4.3.1. Measurement Reference Track

Due to the fact that this thesis does only consider an exemplary field modelling approach based on WLAN measurements to AP mast6a, consequently the spatial location of the measurement reference track has to be adjusted according to the geographical location of mast6a. Figure 4.11 shows the selection of extracted measurement values out of the interpolated WLAN measurements.

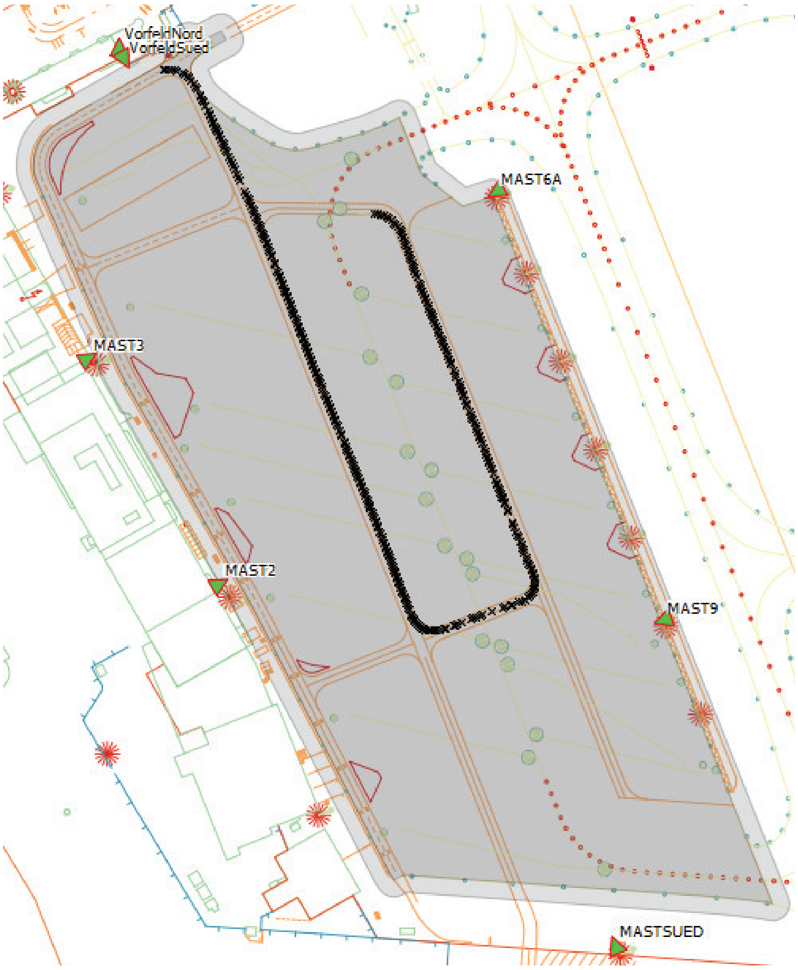


Figure 4.11.: Measurement reference track of geo-referenced WLAN measurement values to mast6a

These timely variant organised electrical field strength measurement values, based on records of the MU movements to AP mast6a, constitutes the measurement reference track. Due to constraints in the spatial availability of the WLAN signal to AP mast6a, the measurement reference track is located in the northern part of the testbed area, where consistent signal availability is given. This signal availability is essential for position determination algorithms, based on the final radio map, which is however out of the thesis's scope.

This measurement reference track will be utilised next, in order to qualify the previously introduced field strength models against empirical measurements.

### 4.3.2. Model Evaluation

As last step of the model qualification, each of the introduced modelling approaches is qualified against the previously discussed measurement reference track. For that, the gathered measurement track will be statistically compared to the locally corresponding field strength values of different field models. This will result in an absolute empirical qualification of the applied field models against the recorded reference measurement track.

In order to obtain a more differentiated basis for model qualification, the testbed area was divided into three sub-areas A, B and C. This allocation is basically done in reference to (i) the geographical location of the AP and (ii) the antenna's field directivity characteristic. These sub-areas divide the testbed in a main-lobe area (Area B), in reference to antenna AP mast6a, and two side-lobe areas (Area A, Area C). The main-lobe area in this case is configured to ±40° in reference to the antenna main lobe direction. This is basically derived from the antenna's H-pattern 3-dB beamwidth of 50°, as stated in (Systems 2005), with an additional margin of 15° in each direction. The sub-areas define on the one hand areas of higher or lower field strength intensity and on the other areas underlying different influences by surrounding buildings. These buildings can have major contribution to multipath influences on the WLAN field stregth as discussed in section 2.1.1 Principles and Concepts. While the measurement reference track is influenced most by surrounding buildings inside of Area A, Area C is least affected. The geographical location of these areas is shown in figure 4.11.

The measurement reference track in figure 4.11 shows basically the power deviation $\Delta p(x, y)$ between the different field modelling approaches and recorded measurements. This was easily achieved by a spatial join of the RM data sets with the measurement track. Consecutively, the field strength component of the measurement reference value $p_{meas.}(x, y)$ was subtracted by the power value $p_{RM_n}(x, y)$ of the applicable RM, where the subscript $n$ denotes the different models:

$$\Delta p(x, y) = p_{meas.}(x, y) - p_{RM_n}(x, y) \tag{4.11}$$

This spatial power difference $\Delta p(x, y)$ was determined for all modelling approaches and encoded in colours as shown in figure 4.11.

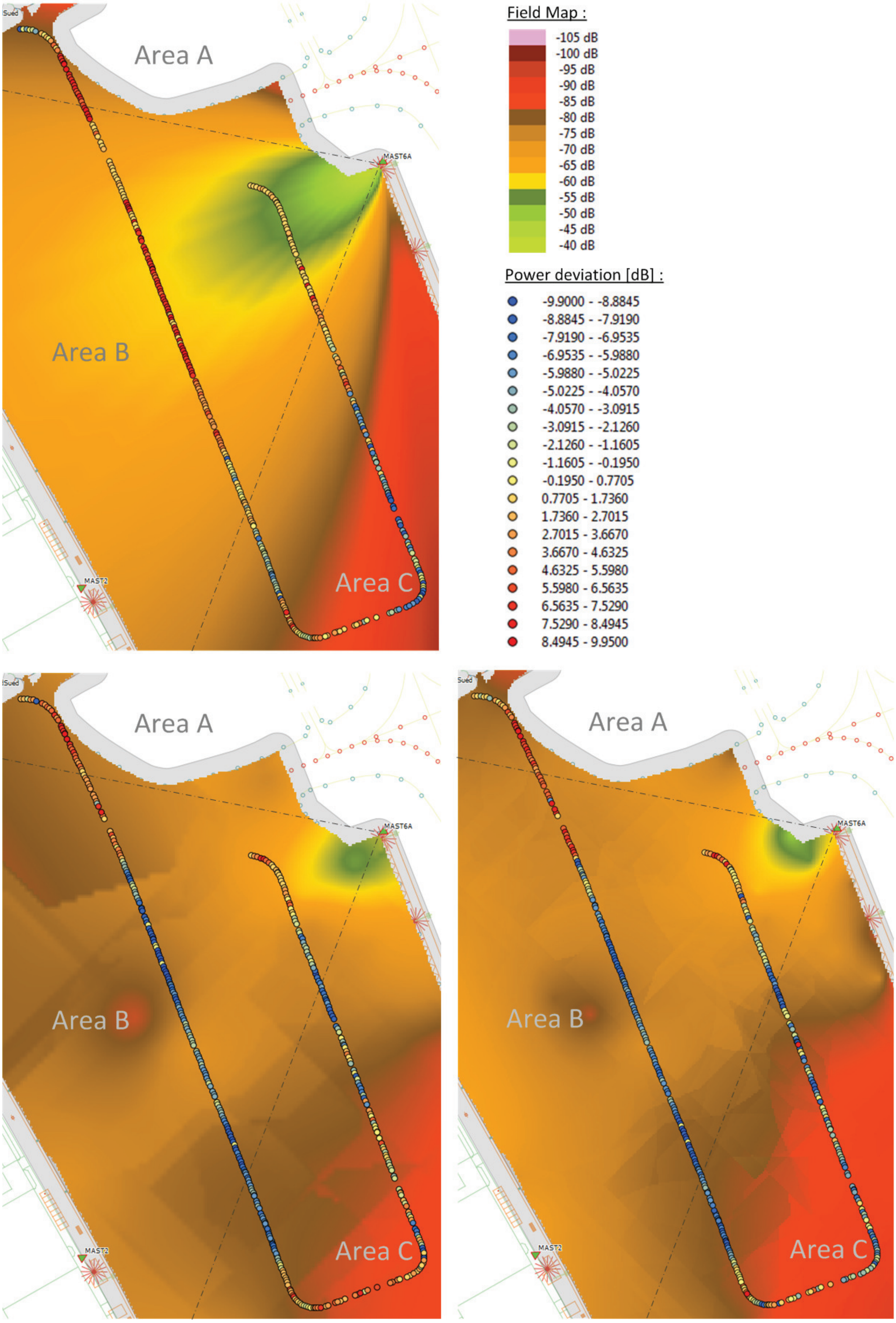


Figure 4.12.: Measurement track vs. field models; (i) upper left cal. parameter model, (ii) bottom left Ord. Kriging model, (iii) bottom right Ord. Kriging param. model

As the a visual interpretation of the power deviation is hard to quantify, it is additionally underpinned by statistical analysis for each of the different field models. Thereby, each analysis is done over the different exploration areas A, B and C as well as for the test-bed area as a whole. For the statistical analysis the maximal positive and negative deviation of $\Delta p$, the mean value and the standard deviation is taken into account. The mean value $\mu$ in combination with the standard deviation $\sigma$ gives a measure of quality, as $\sigma$ shows how much variation exists around the average. For the mathematical definition of these statistical parameter please see equation 3.8 in section 3.3.2 Measurement Reference Value Extraction.
The following qualification will take spatial and statistical explorations into account. First the areal analysis for each model will be discussed before considering the overall analysis.

The spatial distribution of $\Delta p$ in figure 4.12 compared amongst all models show, the calibrated parameter model has the biggest deviation on the very upper end of the track, where the multipath effect by surrounding buildings come into account. In contrast, the Ordinary Kriging model and Ordinary Kriging parameter model shows significantly better performance in that region. Near the border to Area B, the Ordinary Kriging approaches seems to meet the physical phenomena slightly better as well. The statistical figures out of table 4.2 underline these observations. Although the calibrated parameter model has got the lowest mean value, the high variation of $\Delta p$ relativise this advantage.

| | Cal. Param Model | Ord. Kriging | Ord. Kriging + Param. Model |
|---|---|---|---|
| $\Delta p_{min}$ | -9.00 | -8.60 | -3.17 |
| $\Delta p_{max}$ | 9.65 | 9.46 | 9.34 |
| $\mu$ | 4.3021 | 4.5361 | 4.3866 |
| $\sigma$ | 4.5482 | 3.4536 | 3.1496 |

Table 4.2.: Statistical analysis of Area A

In the main lobe area of the reference track, Area B, we can distinguish in figure 4.12 between the near field and the far field of the antenna. In the near field area the calibrated parameter model shows a rather positive result with small variation and low deviation from the measurement reference track. The Ordinary Kriging model meets the measurement track partially even better, however with slightly higher variance on the near field boundary areas. The Ordinary Kriging parameter approach affiliates the negative influences of both models and counts high variations on the boundary areas with high deviations inside the centre of the near field. In the far field centre area, the calibrated parameter model significantly over-estimates the electro-magnetic field, whereas the Ordinary Kriging models show the opposite phenomena. Nevertheless, the calibrated parameter model meets the reality of the field phenomena rather well at the boundary areas of the far field. All this observations can be underlined by the result of statistical analysis in table 4.3. While the calibrated parameter model shows positive mean and low variance values, the Ordinary Kriging approaches count negative mean and high variance values.

| | Cal. Param Model | Ord. Kriging | Ord. Kriging + Param. Model |
|---|---|---|---|
| $\Delta p_{min}$ | -9.10 | -9.85 | -9.80 |
| $\Delta p_{max}$ | 9.95 | 8.68 | 9.87 |
| $\mu$ | 3.6961 | -3.3377 | -3.1735 |
| $\sigma$ | 3.3334 | 4.4714 | 4.8735 |

Table 4.3.: Statistical analysis of Area B

In Area C, the side lobe area in the south side of the test-bed, the calibrated parameter model in figure 4.12 shows high variation in model deviation close to the northern boundary of Area C. In the south-eastern part of the measurement reference track, that model clearly underestimates the electro-magnetic field. In this area the Ordinary Kriging approach succeeds with lowest deviation from the electro-magnetic field. The Ordinary Kriging parameter model tends to underestimate the electro-magnetic field as well in this area. This interpretation is again underlined by the statistical analysis results in table 4.4. All three models have got a negative mean value, as they are underestimating the electro-magnetic field. Ordinary Kriging shows the lowest deviation, however combined with high variation.

| | Cal. Param Model | Ord. Kriging | Ord. Kriging + Param. Model |
|---|---|---|---|
| $\Delta p_{min}$ | -9.23 | -8.48 | -9.90 |
| $\Delta p_{max}$ | 8.20 | 7.96 | 8.80 |
| $\mu$ | -1.7243 | -1.3466 | -3.4877 |
| $\sigma$ | 3.9102 | 4.2824 | 3.6228 |

Table 4.4.: Statistical analysis of Area C

After visual interpretation and statistical analysis of different testbed sectors, table 4.5 gives the statistical values of the entire testbed area. Therein, the calibrated parameter model shows the best performance in terms of absolute deviation from the measurement data set in combination with lowest variance. Both Ordinary Kriging approaches clearly underestimate the electro-magnetic field with higher variation in field deviation.

| | Cal. Param Model | Ord. Kriging | Ord. Kriging + Param. Model |
|---|---|---|---|
| $\Delta p_{min}$ | -9.23 | -9.85 | -9.90 |
| $\Delta p_{max}$ | 9.95 | 9.90 | 9.87 |
| $\mu$ | 1.4059 | -1.8218 | -2.6791 |
| $\sigma$ | 4.6029 | 4.8153 | 4.7509 |

Table 4.5.: Overall statistics of field models

Despite of the statistical overall results, the areal analysis above has shown that each model has strengths and weaknesses in certain areas against different influences. This shows that the evaluation of field models cannot simply strapped down statistical overall analysis. In order to take the areal analysis into account a result matrix, shown in table 4.6, has been determined. That matrix summarises the areal assessment over considered field models for each of the analysed areas.

| | Cal. Param Model | Ord. Kriging | Ord. Kriging + Param. Model |
|---|---|---|---|
| Area A | − − | − + | + + |
| Area B | + + | − + | − − |
| Area C | − + | + + | − − |
| Result | − + | + + | − − |

Table 4.6.: Result matrix of the visual and statistical areal analysis

The final result of the visual and statistical areal analysis shows, Ordinary Kriging applied exclusively on measurement reference points seems to be the best model approach. This is mostly due to the overall robustness of Ordinary Kriging towards the boundaries of area A and C. In Area A the influence of multipath come into account and in Area C Ordinary Kriging allows to compensate fine field discontinuities best. Especially, in Area C the influence of the calibrated parameter model deteriorates the Ordinary Kriging parameter model by underestimating these field discontinuities. The pure parameter model shows good performance within the main lobe Area A, however, obviously does not take physical influences on the electrical field into account.

## Conclusions

In this chapter different modelling approaches have implemented and qualified against a measurement reference track. Thereby, each model has shown its strengths and weaknesses. The parameter model shows still potential in optimising the calibration process by extending it onto the AP antenna pattern. Basically, the introduced modelling approach and implemented framework is convertible to lots of physical propagation of electro-magnetic characteristic, e.g. sound or light. The combination of parametric and probabilistic modelling approach did not succeed as expected. This is due to the special testbed constitution, where buildings on the boundary imply multipath effects. That effects relativise the originally as advantage considered boundary strapdown of the calibrated parameter model. For higher density of interpolation points, the Ordinary Kriging interpolation applied on measurement reference points is considered as the best method for electro-magnetic field modelling. This is mostly due to consideration of multipath effects. However, for higher interpolation point density either longer data acquisition time is needed or more WLAN sensors, attached to MUs.

# 5. Discussion and Future Directions

This research has proposed a methodological way how to model physical coverages based on dynamic in-situ WLAN (Wireless Local Area Network) measurements. It has introduced theoretical methods and concepts on wireless positioning techniques and has been mainly reviewed methods for radio map generation and field modelling. The set-up materials to this research were intensively discovered and grouped into static and dynamic data sets. In this turn, a object relational data model was proposed for the available set-up material as well as for processed data sets, as for instance the multi-channel RM (Radio Map) raster coverage. This data model has been integrated into a spatial database, turning out as absolutely necessary for handing and analysis of the available volume of data. On the basis of available raw data, substantial processes have been defined, for data interpolation, correlation and extraction of measurement reference values in the spatio-temporal domain. These data structure and processes have been constituted the groundwork for implementation and analysis of deterministic and probabilistic field modelling approaches. Thereby, a radial parameter model has been implemented, considering the physical antenna characteristic. Further, the Ordinary Kriging interpolation has been implemented, based on measurement reference values on the on hand and a mixture of radial parameter model samples and measurement reference values on the other. The implemented models have been qualified by visual and statistical areal analysis against a measurement reference track.

Eventually after this short summary, the main findings will be discussed by picking up the initially stated research questions in the introduction to this thesis.

- What modelling method is suitable for electro-magnetic fields in non-obstacle free environments?

The findings of the last core chapter have shown, Ordinary Kriging based on interpolation points of measurement reference values has the potential to model electro-magnetic fields in non-obstacle free environments, on a global scale, without considering field effects caused by mobile obstacles. This is mostly due to the fact that physical effects of multipath, caused by static obstacles, can be considered. However, this approach depends (i) on the quality of measurement reference values and (ii) the interpolation point density of this data set. Unfortunately, both criterion were only partially given in scope of this thesis project. The results of field modelling could be enhanced, by taking more aircraft positions and positions of smaller aircraft during the measurement reference value extraction process into account. Furthermore, an extension of the recording time-frame would increase the density of measurement reference values and thus the quality of the RM as a whole.

At the current status of the project SESAAM (Geo-Spatially Enhanced Situational Awareness for Airport Management), ASTERIX cat 20 aircraft positions have been made available, gathered by MLAT (Multi-Lateration) technology (Bretz et al. 2011). This allows a higher density of large and even small aircraft positions and would significantly contribute to the enhancement of this modelling approach.

- Is it possible to estimate a static field map out of dynamic in-situ measurements?

The elaboration of the measurement reference extraction process has introduced a strategy in the spatio-temporal domain, for filtering field disturbing mobile objects. This approach allows the consideration of interference free field measurements. Yes, the theoretical process exists, for static field map estimation out of dynamic in-situ measurements. This however, is again dependent on the proportion of positioned objects inside the area of interest. The more objects, that have been made available by their positions, the better the result of static field estimation.

- What data structure is capable for multi-layer field mapping?

The findings of data modelling, together with the actual implementation of the object relational data model, inside a spatial database, have shown, this approach is perfectly suitable for the application of multi-layer field mapping. Especially, the provided geo-referenced spatial data type raster is ideally suited for the multi dimensional layer requirement of the RM. This allows the allocation of separate layers for each WLAN AP (Access Point)s within the same dataset. The export of the RM in GeoTIFF format has helped for seamless display of the radio map in GIS (Geographic Information System) software. Furthermore, the application of spatial indexing allows fast access on data within the RM. This allows for future use, to provide the RM to other interfaces, e.g. the position algorithm, in time critical applications. The decision for utilising a GeoDBMS (Geographical Database Management System) instead of file based data access, has shown further advantages in terms of scalability and the sustainability of the overall solution in general.

This research provides a sustainable groundwork for modelling of continuous fields. Although it has not given a solution to dynamic field mapping, considering spatio-temporal field influences e.g. of mobile objects, it constitutes the basis for that. This point should be regarded from a data structure and management point of view, as well as for future implementation of field modelling approaches. Once the static field model is in place, immune against interfering influences by probabilistic effects, such as moving objects; the spatio-temporal influences could be modelled on top. And this points out the author's recommended direction of future research in this area. This framework would allow the extraction of field influences caused by mobile objects. Once that influences are known, they could be modellised and incorporated in a spatio-temporal field model. This could be done by means of geo-statistical simulation, depending on mobile objects' positions. On that basis a dynamic field map could be provided, e.g. to a positioning algorithm, in near real time by a generic web-service.

# 6. Research Methods Appendix

This research methods appendix constitutes a consciously appended chapter to this thesis. It discloses applied software architectures and implemented software packages, laying the groundwork for this thesis research. Although the realisation of these components has bind major effort of the thesis project, their discussion within previous chapters would have disturbed the development of the main argument.

This appendix does focus on generic software structure and concepts. Although dedicated software components and distributors will be explicitly called, the developed concept behind is independent of platform or programming language.

The following presented software concepts and implementations have been developed inside of the thesis project. The focus of the implemented software was exclusively laid on rapid prototyping for evaluation and validation purpose of the thesis research. It does not fulfil the demand of professional software products in terms of documentation and robustness.

## 6.1. Operating System and Software Component Architecture

The careful selection of a suitable operating system and software component architecture is a crucial point of nearly each research project, related to geo-processing and spatial analysis. The flexibility of such an architecture can decide about success or breakdown of an on-going and naturally growing project. The following section gives comprehensively an overview on utilised tools and architectures, from an operating system and software point of view.

Virtual machines provide several advantages for software development in terms of resource management, transferability and system encapsulation. Thus, the VM (Virtual Machine) technology constitutes an ideal software platform for the thesis project. Figure 6.1 provides a first overview on the interaction of virtual machine and native operating platform. The virtual machine with Linux Ubuntu 10.04 LTE operating system, running on Oracle's Virtual Box[1] software, is embedded within the Windows 7 platform on the physical machine.

[1] https://www.virtualbox.org/

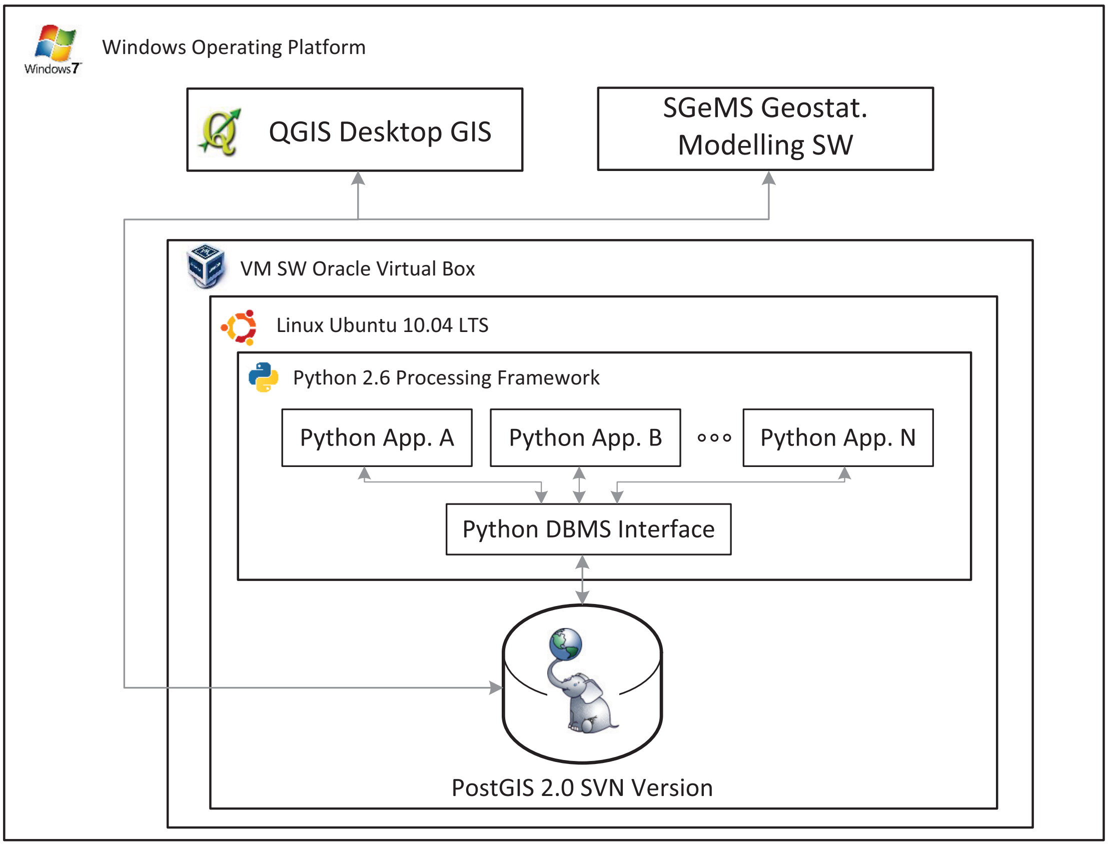


Figure 6.1.: GIS framework and operating system architecture

The core processing framework for data handling, analysis and storage was implemented within the Linux based VM environment. The main reason for choosing Linux as development platform was the seamless availability of binary database sources. This was necessary to work with the latest Open Source PostGIS 2.0 SVN DBMS (Database Management System) version, which had not been available as pre-build version yet, at the project phase of software development. For this project PostGIS 2.0 provides the necessary raster file capability in the world of Open Source DBMS.
As interface to available data sources, e.g. text or excel files, different applications for data parsing, processing and feeding were written in Python 2.6. The programming language Python was selected, as it provides a broad range of programming libraries e.g. for database connectivity, mathematical processing, plotting and much more. Furthermore, Python as powerful scripting language is ideally suitable for rapid prototyping of applications. Basically, within the thesis project, an application package was created for each thematically coherent processing task. These application packages are itemised in figure 6.1 as Python App. A to N.
On the host operating system the QGIS[2] desktop GIS system is utilized for viewing raster and

[2] http://www.qgis.org/

vector data, storing centralized within in DBMS. The communication between desktop GIS and DBMS is done over TCP/IP connection through the virtual machine, as known from distributed network applications. Additionally, SGeMS[3] is provided for advanced 3D data exploration purpose and spatial statistics. SGeMS communicates through an integrated Python interface with the DBMS via physical TCP/IP connection.

After this short introduction of the operating system and software component architecture, next the components of the processing framework will be illuminated.

## 6.2. Processing Framework Architecture

The processing framework is basically divided into four main application packages, realised in the programming language Python. These packages could be considered as thematically grouped toolboxes, supporting the research of this thesis:

- Antenna Interpolation application
- Position Feeder application
- MU (Mobile Unit) Data Feeder application
- Radial Field Model application

All packages together constitute the major development effort of this thesis project. The following section gives a brief introduction to function and architecture of each application package. This will be accomplished by UML (Unified Modeling Language) class diagrams to each package. For more detailed information of programming structure and applied algorithms, the programming code to each application package is attached to this thesis, accessible by ANNEX A to D.

### 6.2.1. Antenna Interpolation Application

The antenna interpolation application supports the interpolation process of the WLAN (Wireless Local Area Network) AP (Access Point) antenna, as theoretically examined in section 3.2.2 Data Integration.
This package includes two main application classes: **Interpol_raw_data** and **load_raw_data**. Both applications utilise the **DB_antenna_interface** as accessible interface towards the underlying database schema, elaborated in section 3.2.1 Data(base) Modelling. The UML class diagram in figure 6.2 shows the functional connections amongst the major components of this package.

[3] http://sgems.sourceforge.net/

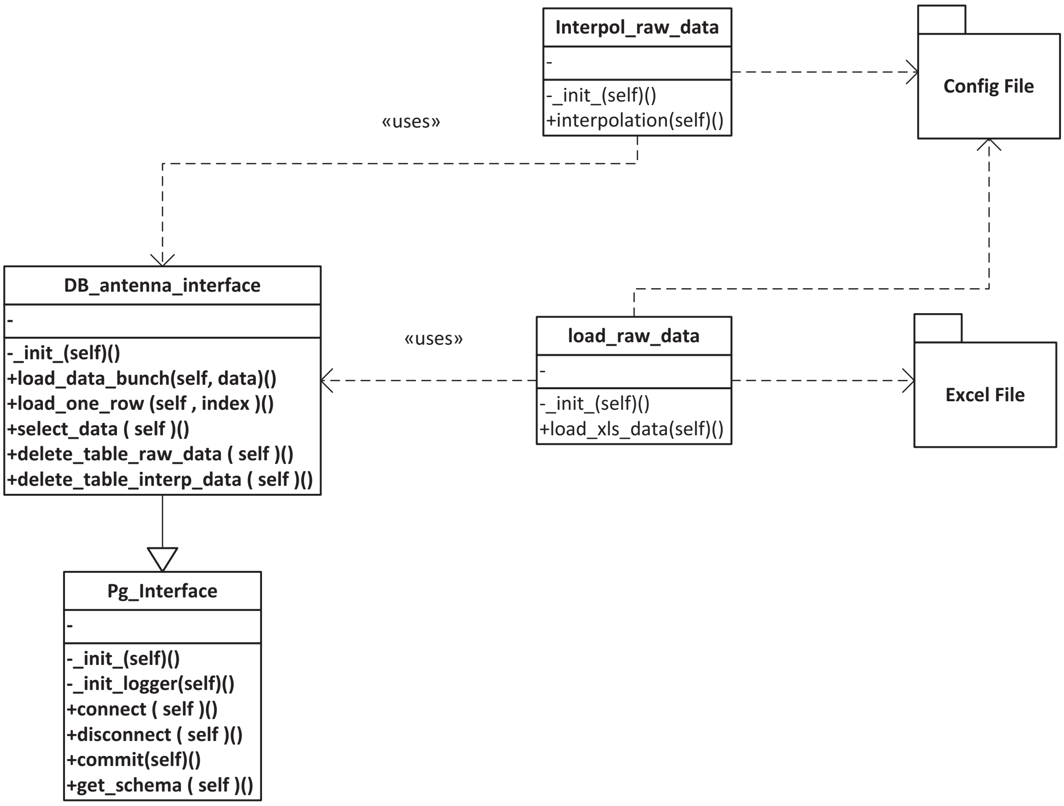


Figure 6.2.: UML class diagram of the antenna interpolation application package

The **load__raw__data** class allows to parse the excel file, including digitised antenna pattern values, and inserts the raw data values into the database. Thereby, the **DB__interpolation __interface** addresses the data flow between database and application. For the data insertion process necessary parameters as e.g. excel file filename and file location will be selected from the config file. After the interpolation raw data has been made available in the database, next the actual interpolation process can be initiated.

This process is implemented in the **Interpol__raw__data** class. First of all, it retrieves the interpolation points from the database through the **DB__interpolation__interface** class. After all interpolation points have been loaded into the main memory, the interpolation will be processed. Finally, the interpolation result is inserted again into the database.

For detailed implementation of the antenna interpolation application's object oriented programming structure and algorithms, please find the programming code in A ANNEX 1.

### 6.2.2. AP Position Feeder Application

The AP position feeder application is a database feeder for geographical positions and metadata of WLAN APs. This application is necessary as APs' geographical positions and metadata has

been externally provided. They had been collected by GNSS position measurements and were consecutively provided in excel file data structure as described in section 3.1.1 Static Infrastructure Data.
The application package includes the **AP__position__feeder** as main class. This utilises the **DB__ap__position__interface** as interface to the database as shown in figure 6.3.

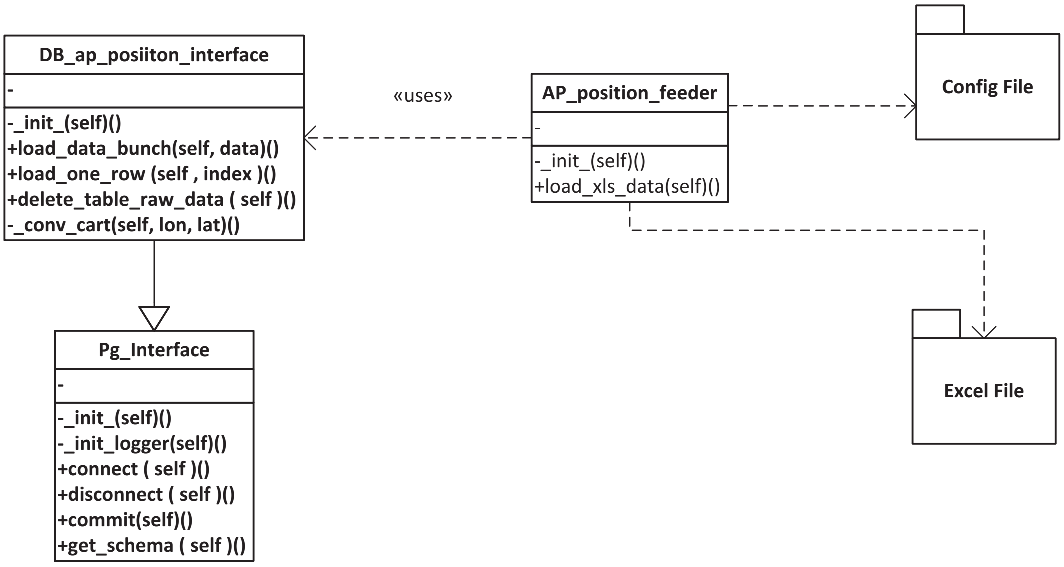


Figure 6.3.: UML class diagram of the AP position feeder application package

The functional design of the AP position feeder application is rather similar to the database insert process, described in the antenna interpolation application section above. The **AP__position __feeder** class reads in the excel file, including AP data, and insert it into the project's database via the **DB__ap__position__interface**. Additionally, an on-the-fly reprojection of geographical WGS84 positions into the UTM (Universal Transverse Mercator) reference system is conducted. For that process configurable parameters have been again provided by the config file.
For detailed implementation of the AP position feeder's object oriented programming structure and algorithms, please find the programming code in B ANNEX 2.

### 6.2.3. MU Data Feeder Application

The MU data feeder application package comprises a comprehensive collection for handing and analysis of MU WLAN measurement and positioning data of MUs and aircraft. A detailed description of data structure and the process behind can be find in section 3.1.2 Dynamic Measurement Data.
The package is compound of the following main classes: **Mu__cluster__analysis**, **Mu__data __position__feeder**, **Aircraft__position __feeder**, **Mu__data__measure__feeder** and **Mu**

**_data_interpolation**. All of these classes utilises **Mu_data_interface** as interface to the project database schema. The UML class diagram in figure 6.4 shows the functional connections of all package components.

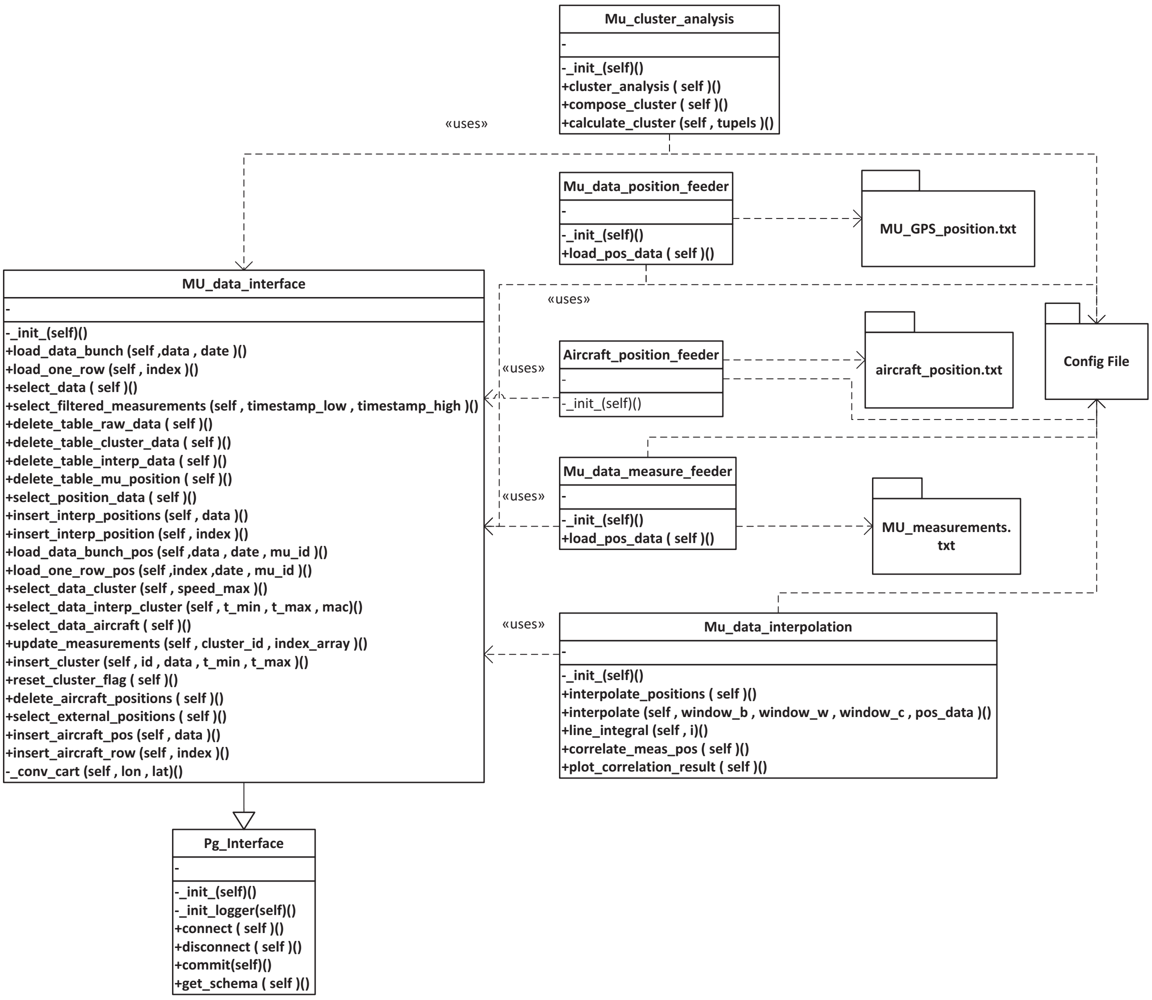

Mu_cluster_analysis
-
-_init_(self)()
+cluster_analysis ( self )()
+compose_cluster ( self )()
+calculate_cluster (self , tupels )()
«uses»
Mu_data_position_feeder
-
-_init_(self)()
+load_pos_data ( self )()
MU_GPS_position.txt
MU_data_interface
-
-_init_(self)()
+load_data_bunch (self ,data , date )()
+load_one_row (self , index )()
+select_data ( self )()
+select_filtered_measurements (self , timestamp_low , timestamp_high )()
+delete_table_raw_data ( self )()
+delete_table_cluster_data ( self )()
+delete_table_interp_data ( self )()
+delete_table_mu_position ( self )()
+select_position_data ( self )()
+insert_interp_positions (self , data )()
+insert_interp_position (self , index )()
+load_data_bunch_pos (self ,data , date , mu_id )()
+load_one_row_pos (self ,index ,date , mu_id )()
+select_data_cluster (self , speed_max )()
+select_data_interp_cluster (self , t_min , t_max , mac)()
+select_data_aircraft ( self )()
+update_measurements (self , cluster_id , index_array )()
+insert_cluster (self , id , data , t_min , t_max )()
+reset_cluster_flag ( self )()
+delete_aircraft_positions ( self )()
+select_external_positions ( self )()
+insert_aircraft_pos (self , data )()
+insert_aircraft_row (self , index )()
-_conv_cart (self , lon , lat)()
«uses»
Aircraft_position_feeder
-
-_init_(self)()
aircraft_position.txt
Config File
«uses»
Mu_data_measure_feeder
-
-_init_(self)()
+load_pos_data ( self )()
«uses»
MU_measurements.
txt
«uses»
Mu_data_interpolation
-
-_init_(self)()
+interpolate_positions ( self )()
+interpolate (self , window_b , window_w , window_c , pos_data )()
+line_integral (self , i)()
+correlate_meas_pos ( self )()
+plot_correlation_result ( self )()
Pg_Interface
-
-_init_(self)()
-_init_logger(self)()
+connect ( self )()
+disconnect ( self )()
+commit(self)()
+get_schema ( self )()

Figure 6.4.: UML class diagram of the MU data feeder application package

The application package includes the three database feeder applications **Mu_data_position _feeder**, **Aircraft_position_feeder** and **Mu_data_measure_feeder**. These allow the first step of inserting positioning and measurement data into the database. For this process externally delivered text files, acting as data container, will be parsed by these application classes and relevant data is inserted into the database.

After positioning and measurement data have been made available in the database, **Mu_data _interpolation** conducts the interpolation of MU positions and correlates them to WLAN measurements over time. The theoretical aspects behind that concept has been elaborated in section 3.2.2 Data Integration. For this process the interpolation of MU's positioning data is conducted first and written back to the database. In a second step the interpolated positioning data is selected from the database, together with WLAN measurement data, and the correlation of both data sets over time is applied. This results in a georeferenced WLAN measurement track.

Finally, the **Mu_cluster_analysis** class allows cluster analysis on the georeferenced WLAN measurement track. The detailed description and theoretical background of this process has been elaborated in section 3.3.2 Measurement Reference Value Extraction.

For detailed implementation of the MU data feeder's object oriented programming structure and algorithms, please find the programming code in C ANNEX 3.

### 6.2.4. Radial Field Model Application

The the radial field model application constitutes the algorithmic implementation of the deterministic radial field parameter model, as theoretically described in section 4.1 Parameter Model. The application package itself is divided into two main classes: **Radial_field_model** and **Regression_Analysis**. Both classes utilise a shared **Radial_model_interface** class as interface to the underlying database. Figure 6.5 shows the functional connection amongst different components of the radial field model application package.

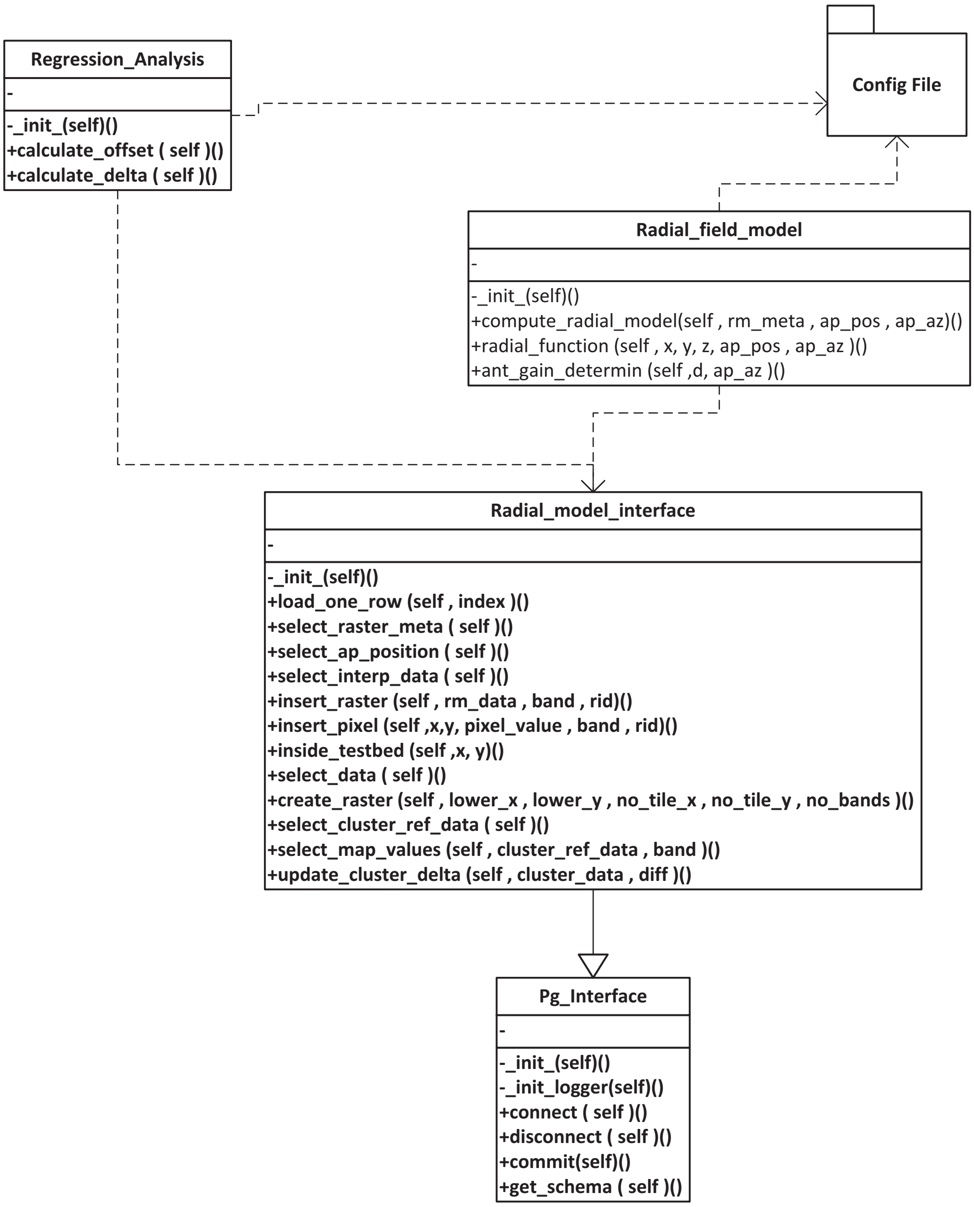

Regression_Analysis
-
-_init_(self)()
+calculate_offset ( self )()
+calculate_delta ( self )()
Config File
Radial_field_model
-
-_init_(self)()
+compute_radial_model(self , rm_meta , ap_pos , ap_az)()
+radial_function (self , x, y, z, ap_pos , ap_az )()
+ant_gain_determin (self ,d, ap_az )()
Radial_model_interface
-
-_init_(self)()
+load_one_row (self , index )()
+select_raster_meta ( self )()
+select_ap_position ( self )()
+select_interp_data ( self )()
+insert_raster (self , rm_data , band , rid)()
+insert_pixel (self ,x,y, pixel_value , band , rid)()
+inside_testbed (self ,x, y)()
+select_data ( self )()
+create_raster (self , lower_x , lower_y , no_tile_x , no_tile_y , no_bands )()
+select_cluster_ref_data ( self )()
+select_map_values (self , cluster_ref_data , band )()
+update_cluster_delta (self , cluster_data , diff )()
Pg_Interface
-
-_init_(self)()
-_init_logger(self)()
+connect ( self )()
+disconnect ( self )()
+commit(self)()
+get_schema ( self )()

Figure 6.5.: UML class diagram of the radial field model application package

The **Radial__field__model** class is basically the implementation of the radial field computation for different APs of the testbed area. It computes consecutively for each AP a spatial electromagnetic field distribution in consideration of their physical antenna characteristic and spatial orientation. This results in a separate RM (Radio Map) raster layer for each AP, while each layer is inserted into the database. For that process necessary control parameters, e.g. raster dimension, are accessible through the config file.

After the RM has been processed and made available in the spatial database, the **Regression__Analysis** class is in charge for the RM calibration process by regression analysis. The

detailed theoretical approach of that calibration is described in section 4.1.2 Model Calibration. Basically, this application retrieves measurement reference values from the database upon which the calibration offset is determined by regression analysis. This offset is mathematically added onto the RM and results in a calibrated radio map.
For detailed implementation of the radial field model's object oriented programming structure and algorithms, please find the programming code in D ANNEX 4.

# Bibliography

Bjõrck, A. (1996), *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics.

Bretz, B., Kapser, J. & Roth, C. (2011), 'Entwicklung eines "complete operational picture" mit standard it-infrastrukturen - fortschritte des ffg-projektes sesaam'.

Carr, J. J. (2000), *The Technician's Radio Receiver Handbook: Wireless and Telecommunication Technology*, Newnes.

Dempster, A. G., Binghao, L. & Quader, I. (2008), 'Errors in determinstic wireless fingerprinting systems for localisation'.

ESRI (1998), 'Esri shapefile technical description - an esri white paper'.

Fu, P. & Sun, J. (2010), *Web GIS: Principles and Applications*, ESRI Press.

Goodchild, M. F. (1992*a*), 'Geographical data modeling', *Computers &amp; Geosciences* **18**(4), 401–408.

Goodchild, M. F. (1992*b*), 'Geographical information science', *International Journal of Geographical Information Systems* **6**(1), 31–45.

Gopi, S. (2010), *Basic Civil Engineering*, Pearson Education.

Groves, P. D. (2008), *Principles of GNSS, Intertial, and Multisensor Integrated Navigation Systems*, Artech House, Inc., 685 Canton Street, Norwood, MA 02062, USA.

Haykin, S. (2001), *Communication Systems*, 4th edn, John Wiley and Sons, Inc.

Honkavirta, V., Perala, T., Ali-Loytty, S. & Piche, R. (2009), 'A comparative survey of wlan location fingerprinting methods'.

Il-Suek, K., Sewoong, K., Jae-Woo, L. & Young Joong, Y. (2007), '3d ray-tracing model including effect of inhomogeneous building surface for characterization of wireless communication channel'.

Isaaks, E. H. & Srivastava, R. (1989), *Applied geostatistics*, Oxford University Press.

Isaaks, E. & Srivastava, R. (1990), *An Introduction to Applied Geostatistics*, Oxford University Press, USA.

Johnston, K. & Institute, E. S. R. (2004), *ArcGIS 9: Using ArcGIS geostatistical analyst*, ESRI. (Redlands, Calif.).

Konak, A. (2010), ‘Estimating path loss in wireless local area networks using ordinary kriging’.

Li, B., Wang, Y., Lee, H. K., Dempster, A. & Rizos, C. (2005), ‘Method for yielding a database of location fingerprints in wlan’, *Communications, IEE Proceedings-* **152**(5), 580–586.

Mansfeld, W. (2004), *Satellitenortung und Navigation*, 2nd edn, Friedr. Vieweg & Sohn Verlag, Wiesbaden.

Miller, H. J. & Bridwell, S. A. (2008), ‘A field-based theory for time geography’, *Annals of the Association of American Geographers* **99**(1), 49–75.

Mitasova, H. & Mitas, L. (1993), ‘Interpolation by regularized spline with tension: I. theory and implementation’, *Mathematical Geology* **25**(6), 641–655.

Mitasova, H., Mitas, L., Brown, W. M., Gerdes, D. P., Kosinovsky, I. & Baker, T. (1995), ‘Modelling spatially and temporally distributed phenomena: new methods and tools for grass gis’, *International Journal of Geographical Information Systems* **9**(4), 433–446.

Mitasova, L. & Mitas, H. (1988), ‘General variational approach to the interpolation problem’, *Computers &amp; Mathematics with Applications* **16**(12), 983–992.

Niculescu, D. & Badri, N. (2003), ‘Ad hoc positioning system (aps) using aoa’.

Pahlavan, K., Xinrong, L. & Makela, J. P. (2002), ‘Indoor geolocation science and technology’, *IEEE Communications Magazine* **40**(2), 112–118.

Parodi, B. B., Lenz, H., Szabo, A., Hui, W., Horn, J., Bamberger, J. & Obradovic, D. (2006), ‘Initialization and online-learning of rss maps for indoor / campus localization’.

Press, E. & Environmental Systems Research, I. (2004), *Using ArcGIS Tracking Analyst: ArcGIS 9*, ArcGIS 9, Esri Press. (Redlands, Calif.).

Roehr, S., Gulden, P. & Vossiek, M. (2007), ‘Method for high precision clock synchronization in wireless systems with application to radio navigation’.

Schlueter, A. & Thesseling, F. (2009), ‘Building information model based energy/exergy performance assessment in early design stages’, *Automation in Construction* **18**(2), 153–163.

Sen, A., Gümüsay, M., Kavas, A. & Bulucu, U. (2008), ‘Programming an artificial neural network tool for spatial interpolation in gis - a case study for indoor radio wave propagation of wlan’, *Sensors* **8**(9), 5996–6014.

Skolnik, M. (1990), *Radar Handbook*, 2nd Edition, McGraw-Hill, Inc.

Späth, H. (1995), *One dimensional spline interpolation algorithms*, A K Peters.

Systems, C. (2005), Cisco aironet 9.5 dbi patch antenna (air-ant5195p-r), Technical report.

ublox (2008), ‘Nmea, ubx protocol specification’.

Vossiek, M., Wiebking, L., Gulden, P., Weighardt, J. & Hoffmann, C. (2003), 'Wireless local positioning - concepts, solutions, applications'.

Waldmann, B., Weigel, R. & Gulden, P. (2008), 'Method for high precision local poitioning radar using ultra wideband technique'.

Widyawan, Klepal, M. & Pesch, D. (2007), 'Influence of predicted and measured fingerprint on the accuracy of rssi-based indoor location systems'.

Wölfle, G. & Landstorfer, F. M. (1998), 'Dominant paths for the field strength prediction'.

Wölfle, G., Wahl, R., Wertz, P., Wildbolz, P. & Landstorfer, F. (2005), 'Dominant path prediction model for indoor scenarios'.

# A. Annex 1

## Interpolate_antenna_raw_data.py

```
import xlrd, logging, psycopg2
from ConfigParser import ConfigParser
from db_antenna_interface import Db_antenna_interface
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate


CONFIGFILE = "antenna_interp.conf"

class Interpol_raw_data():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)

        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db':
            config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter
            ', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), '
            schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema':
            True, 'table_insert':config.get('pg_parameter', 'DB_TABLE_INTER'),'
            table_select':config.get('pg_parameter', 'DB_TABLE_RAW')}
        self.db_scale_factor = float(config.get('plot_scale','db_plot_scale'))
        self.gain = float(config.get('plot_scale','gain'))
        self.sampling_width = float(config.get('plot_scale','sampling_width'))

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s
            : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Db_antenna_interface(self.db_param)

        self.db.select_data()

        self.interpolation()

        self.db.delete_table_interp_data()
        self.db.load_data_bunch(self.data_interp)


    def interpolation(self):
        x = np.zeros(self.db.data.__len__())
        y = np.zeros(self.db.data.__len__())

        for i in range(0,self.db.data.__len__()):
            x[i] = self.db.data[i][0]
            y[i] = self.db.data[i][1]

        # Scaling in dB
        y = (y-100)/self.db_scale_factor+self.gain

        # Interpolation
        tck = interpolate.splrep(x,y,s=0)
        tck_ = interpolate.splrep(x,y,s=0,k=5)
        xnew = np.arange(-90,+90,self.sampling_width)
        ynew = interpolate.splev(xnew,tck,der=0)
        ynew_ = interpolate.splev(xnew,tck_,der=0)

        #save interpolated results to array
        self.data_interp = []

        for i in range(0,ynew.__len__()):
            self.data_interp.append((xnew[i],ynew[i]))

        # Plotting
        plt.figure()
        plt.plot(x,y,'x',xnew,ynew,'b',xnew,ynew_,'--')

        plt.legend(['True','Spline k=3', 'Spline k=5'], prop={"size":10})
        plt.xlabel("Azimuth angle [deg]")
        plt.ylabel("Antenna gain [dBi]")
        plt.grid(True)

        plt.show()

if __name__ == "__main__":
    Interpol_raw_data()
```

### Load_antenna_raw_data.py

```python
import xlrd, logging, psycopg2
from ConfigParser import ConfigParser
from db_antenna_interface import Db_antenna_interface

CONFIGFILE = "antenna_interp.conf"

class Load_raw_data():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.XLS_FILENAME = config.get('xls_parameter', 'XLS_FILENAME')
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_select':config.get('pg_parameter', 'DB_TABLE_RAW'), 'table_insert':config.get('pg_parameter', 'DB_TABLE_RAW')}

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.load_xls_data()

        loader = Db_antenna_interface(self.db_param)
        loader.delete_table_raw_data()
        loader.load_data_bunch(self.data)

    def load_xls_data(self):
        self._logger.info("----- Start Antenna_interp skript! -----")
        self._logger.info("Fetch Data from EXCEL sheet")

        wb = xlrd.open_workbook(self.XLS_FILENAME)
        sheets = wb.sheet_names()
        sh = wb.sheet_by_name(sheets[0])

        self.data = []

        for rownum in range(0,sh.nrows):
                self.data.append(sh.row_values(rownum))

        print self.data

if __name__ == "__main__":
    Load_raw_data()
```

## DB_antenna_interface.py

```python
import xlrd, logging, psycopg2
from PostgresInterf import Pg_Interface
from ConfigParser import ConfigParser

CONFIGFILE = "antenna_interp.conf"

class Db_antenna_interface(Pg_Interface):

    def __init__(self, db_param):
        self.db_param = db_param
        Pg_Interface.__init__(self, db_param)

    def load_data_bunch(self,data):
        self.data = data

        for index in range(0,self.data.__len__()):
            self.load_one_row(index)

        self.commit()

    def load_one_row(self,index):
        print "Load row ", index ,"to ", self.get_schema(),".", self.db_param['table_insert']
        cur = self._connection.cursor()
        cur.execute("INSERT INTO {0}.{1} (degree, power_value) VALUES({2},{3})".format(
                                                    self.get_schema(),
                                                    self.db_param['table_insert'],
                                                    self.data[index][0],
                                                    self.data[index][1]
                                                    ))

    def select_data(self):
        cur = self._connection.cursor()
        cur.execute("SELECT * FROM {0}.{1}".format(
                                                    self.get_schema(),
                                                    self.db_param['table_select'],
                                                    ))

        self.data = []

        for i in cur:
            self.data.append((i[1],i[2]))

    def delete_table_raw_data(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                                                    self.get_schema(),
                                                    self.db_param['table_select'
                                                       ],
                                                    ))

    def delete_table_interp_data(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                                                    self.get_schema(),
                                                    self.db_param['table_insert'
                                                       ],
                                                    ))
```

## PostgresInterf.py

```python
import logging
import psycopg2

class Pg_Interface():

    def __init__(self, db_param):
        self._db_param = db_param
        self._init_logger()

        self.connect()

    def _init_logger(self):
        self._logger = logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

    def connect(self):
        self._connection = psycopg2.connect("host='{0}' dbname='{1}' user='{2}'\
            password='{3}'".format(
            self._db_param['host'],
            self._db_param['db'],
            self._db_param['user'],
            self._db_param['password']))

        self._logger.info("Successfully connected to {0}.".format(
            self._db_param['host']))

    def disconnect(self):
        self._connection.close()
        self._logger.info("{0} Successfully closed.".format(
            self._db_param['host']))

    def commit(self):
        self._connection.commit()
        self._logger.info("Database commited")

    def get_schema(self):
        return self._db_param['schema']
```

# B. Annex 2

### AP_position_feeder.py

```python
import xlrd, logging, psycopg2
from ConfigParser import ConfigParser
from db_ap_position_interface import Db_ap_position_interface

CONFIGFILE = "ap_position.conf"

class Ap_position_feeder():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.XLS_FILENAME = config.get('xls_parameter', 'XLS_FILENAME')
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_insert':config.get('pg_parameter', 'DB_TABLE_POS')}

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.load_xls_data()

        loader = Db_ap_position_interface(self.db_param)
        loader.delete_table_raw_data()
        loader.load_data_bunch(self.data)


    def load_xls_data(self):

  self._logger.info("----- AP_position_feeder skript! -----")
        self._logger.info("Fetch Data from EXCEL sheet")

        wb = xlrd.open_workbook(self.XLS_FILENAME)
        sheets = wb.sheet_names()
        sh = wb.sheet_by_name(sheets[0])

        self.data = []

        for rownum in range(0,sh.nrows):
                self.data.append(sh.row_values(rownum))

        print self.data

if __name__ == "__main__":
    Ap_position_feeder()
```

### DB_ap_position_interface.py

```
import xlrd, logging, psycopg2
from PostgresInterf import Pg_Interface
from ConfigParser import ConfigParser
from pyproj import Proj

CONFIGFILE = "antenna_interp.conf"

class Db_ap_position_interface(Pg_Interface):

    def __init__(self, db_param):
        self.db_param = db_param
        Pg_Interface.__init__(self, db_param)


    def load_data_bunch(self,data):
        self.data = data

        for index in range(0,self.data.__len__()):
            self.load_one_row(index)

        self.commit()

    def load_one_row(self,index):
        cur = self._connection.cursor()

        position = self._conv_cart(self.data[index][1],self.data[index][0])
        print position

        cur.execute("INSERT INTO {0}.{1} (x, y, z, antenna_az, name, ap_mac, position) VALUES({2},{3},{4},{5},'{6}','{7}',ST_GeomFromEWKT('SRID=32633; POINT({8} {9} {10})'))".format(
                                                    self.get_schema(),
                                                    self.db_param['table_insert'],
                                                    self.data[index][0],
                                                    self.data[index][1],
                                                    self.data[index][2],
                                                    self.data[index][3],
                                                    self.data[index][4],
                                                    self.data[index][5],
                                                    position[0],
                                                    position[1],
                                                    self.data[index][2]
                                                    ))

    def delete_table_raw_data(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                                                    self.get_schema(),
                                                    self.db_param['table_insert'
                                                       ],
                                                    ))

    def _conv_cart(self, lon, lat):
        p = Proj(proj='utm',zone=33,ellps='WGS84')
        return  p(lon,lat)
```

# C. Annex 3

## Aircraft_position_feeder.py

```python
import xlrd, logging, psycopg2
from datetime import timedelta
from ConfigParser import ConfigParser
from mu_data_interface import Mu_data_interface
import numpy as np

CONFIGFILE = "Mu_data_feeder.conf"

class Aircraft_position_feeder():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.FILENAME = config.get('input_file', 'FILE_NAME')
        self.FILEPATH = config.get('input_file', 'FILE_PATH')+self.FILENAME
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_interp':config.get('pg_parameter', 'DB_TABLE_INTER_POS'), 'table_select':config.get('pg_parameter', 'DB_TABLE_POSITION'), 'table_update':config.get('pg_parameter', 'DB_TABLE_INTER_POS'), 'table_insert':config.get('pg_parameter','DB_CLUSTER')}
        self.file_date = self.FILENAME[0:8].split('_')
        self.file_date = "20"+self.file_date[0]+"-"+self.file_date[1]+"-"+self.file_date[2]
        print self.file_date

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Mu_data_interface(self.db_param)

        self.db.delete_aircraft_positions()

        self.data = self.db.select_external_positions()
        print self.data[0]
        self.db.insert_aircraft_pos(self.data)

if __name__ == "__main__":
    Aircraft_position_feeder()
```

### MU_cluster_analysis.py

```python
import xlrd, logging, psycopg2
from datetime import timedelta
from ConfigParser import ConfigParser
from mu_data_interface import Mu_data_interface
import numpy as np

CONFIGFILE = "Mu_data_feeder.conf"

class Mu_cluster_analysis():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.FILENAME = config.get('input_file', 'FILE_NAME')
        self.FILEPATH = config.get('input_file', 'FILE_PATH')+self.FILENAME
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_interp':config.get('pg_parameter', 'DB_TABLE_INTER_POS'), 'table_select':config.get('pg_parameter', 'DB_TABLE_POSITION'), 'table_update':config.get('pg_parameter', 'DB_TABLE_INTER_POS'), 'table_insert':config.get('pg_parameter','DB_CLUSTER'), 'mac':config.get('pg_parameter','MAC')}

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Mu_data_interface(self.db_param)
        self.data = self.db.select_data_cluster(0.1)
        self.data_aircraft = self.db.select_data_aircraft()

        self.cluster_analysis()
        self._logger.info("Cluster Count: {0}".format(self.cluster.__len__()))

        self.db.reset_cluster_flag()
        self.db.delete_table_cluster_data()
        self.compose_cluster()

    def cluster_analysis(self):
        print "Data from select on speed query :", self.data.__len__()
        self.cluster = []
        delta_max = timedelta(seconds=3)
        temp = -1
        for i in range(0,self.data.__len__()):
            if i != 0:
                if self.data[i][2] - self.data[i-1][2] >= delta_max:
                    self.cluster.append([self.data[temp+1][2], self.data[i-1][2]])
                    temp = i-1

    def compose_cluster(self):
        for i in range(0,self.cluster.__len__()):
            print self.cluster[i][0], self.cluster[i][1]
            temp = self.db.select_data_interp_cluster(self.cluster[i][0], self.cluster[i][1], self.db_param['mac'])

            # Check if cluster points closely distanced to aircrafts!

      print "Checking distance to aircraft..."
            outliers = []
            for j in range(0,temp.__len__()):
                for pos_aircraft in self.data_aircraft:
                    if temp[j][4] <= pos_aircraft[1]+timedelta(seconds=1) and temp[j][4] > pos_aircraft[1]-timedelta(seconds=1):
                        pos_xy_aircraft = np.array([pos_aircraft[2],pos_aircraft[3]])
                        pos_xy_mu = np.array([temp[j][2], temp[j][3]])
                        dist = pos_xy_aircraft - pos_xy_mu
                        if np.linalg.norm(dist) <= 100:
                            outliers.append(j)
                            break

            print "outliers:", outliers
            for k in range(0,outliers.__len__()):
                del temp[k]

            self.db.update_measurements(i+1, temp)

            cluster_data = self.calculate_cluster(temp)
            if cluster_data[0] != 0:
                self.db.insert_cluster(i+1, cluster_data, self.cluster[i][0], self.cluster[i][1])

            print "ID = :", i+1

    def calculate_cluster(self, tupels):
        power = []
        x_y = []
        x = []
```

```python
        y = []
        for i in tupels:
            power.append(i[1])
            x.append(i[2])
            y.append(i[3])

        np_power = np.asarray(power)

        np_x = np.asarray(x)
        np_y = np.asarray(y)

        return [np_power.__len__(), np.mean(np_power), np.std(np_power),
                np.mean(np_x), np.std(np_x), np.mean(np_y),np.std(y), np.sqrt(np.std(np_x)**2+np.std(np_y)**2)]

if __name__ == "__main__":
    Mu_cluster_analysis()
```

## MU_data_interface.py

```
import logging, psycopg2
from datetime import date
from datetime import time
from PostgresInterf import Pg_Interface
from pyproj import Proj

class Mu_data_interface(Pg_Interface):

    def __init__(self, db_param):
        self.db_param = db_param
        Pg_Interface.__init__(self, db_param)


    def load_data_bunch(self,data, date):
        self.data = data
        self.date = date

        for index in range(0,self.data.__len__()):
            self.load_one_row(index)

        self.commit()

    def load_one_row(self,index):
        cur = self._connection.cursor()

        cur.execute("INSERT INTO {0}.{1} (mu_id, timestamp, mac, channel, power, ssid)\
                     VALUES({2},'{3}','{4}',{5},{6},'{7}')".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            self.data[index][0],
                            self.data[index][2],
                            self.data[index][3].lstrip(' '),
                            self.data[index][4].lstrip(' '),
                            self.data[index][5].lstrip(' '),
                            self.data[index][7].lstrip(' ').rstrip('\n')
                            ))

    def select_data(self):
        cur = self._connection.cursor()
        cur.execute("SELECT * FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param['table_select'],
                            ))

        self.data = []

        for i in cur:
            self.data.append((i[1],i[2]))

    def select_filtered_measurements(self, timestamp_low, timestamp_high):
        cur = self._connection.cursor()

        cur.execute("SELECT * FROM {0}.{1} WHERE timestamp >= '{2}' AND timestamp <= '{3}'".format(
                            self.get_schema(),
                            self.db_param['table_measurements'],
                            timestamp_low,
                            timestamp_high
                            ))

        self.data = []

        for i in cur:
            self.data.append((i[0],i[2]))
        return self.data


    def delete_table_raw_data(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param['table_select'],
                            ))

    def delete_table_cluster_data(self):
        self.delete_table_interp_data()

    def delete_table_interp_data(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            ))

    def delete_table_mu_position(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            ))

    def select_position_data(self):
        cur = self._connection.cursor()
        cur.execute("SELECT  ST_X(position), ST_Y(position), speed_og, course_og
            , timestamp, hacc FROM {0}.{1} where hdop < {2}".format(
                            self.get_schema(),
                            self.db_param['table_select'],
                            2
                            ))

        self.data = []

        for i in cur:
            self.data.append(i)

        return self.data

    def insert_interp_positions(self,data):
        self.data = data

        for index in range(0,self.data[0].__len__()):
            self.insert_interp_position(index)

        self.commit()

    def insert_interp_position(self,index):
        cur = self._connection.cursor()

        cur.execute("INSERT INTO {0}.{1} (rid_mu_measurement, position)\
                     VALUES({2},ST_GeomFromEWKT('SRID=32633; POINT({3} {4} {5})
                         ')".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            self.data[0][index][0],
                            self.data[0][index][1][0],
                            self.data[0][index][1][1],
                            500
                            ))

    def load_data_bunch_pos(self,data, date, mu_id):
        self.data = data
        self.date = date

        for index in range(0,self.data.__len__()):
            self.load_one_row_pos(index, date, mu_id)

        self.commit()

    def load_one_row_pos(self,index,date, mu_id):
        cur = self._connection.cursor()
        x, y = self._conv_cart(str(float(self.data[index][4][0:3])+float(self.
            data[index][4][3:11])*100/60/100), str(float(self.data[index
            ][2][0:2])+float(self.data[index][2][2:11])*100/60/100))
        lon, lat = str(float(self.data[index][4][0:3])+float(self.data[index][4][3:11])*100/60/100), str(float(self.data[index][2][0:2])+float(self.data[index][2][2:11])*100/60/100)
        cur.execute("INSERT INTO {0}.{1} (mu_id, timestamp, lat, lon, alt_ref, nav_stat, hacc, vacc, speed_og, course_og, hdop, vdop, tdop, position)\
                     VALUES ({2},'{3}',{4},{5},{6},'{7}',{8},{9},{10},{11},{12},{13},{14},ST_GeomFromEWKT('SRID=32633; POINT({15} {16} {6})'))".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            mu_id,
                            date+" "+self.data[index][1][0:2]+":"+self.data[index][1][2:4]+":"+self.data[index][1][4:9],
                            lat, lon,
                            self.data[index][6],
                            self.data[index][7],
                            self.data[index][8],
                            self.data[index][9],
                            self.data[index][10],
                            self.data[index][11],
                            self.data[index][14],
                            self.data[index][15],
                            self.data[index][16],
                            x, y
                            ))

    def select_data_cluster(self, speed_max):
        cur = self._connection.cursor()
        cur.execute("SELECT * FROM {0}.{1} WHERE speed_og <= {2} and hdop < 4".format(
                            self.get_schema(),
                            self.db_param['table_select'],
                            speed_max
                            ))

        self.data = []

        for i in cur:
            self.data.append(i)

        return self.data

    def select_data_interp_cluster(self, t_min, t_max, mac):
        cur = self._connection.cursor()
        cur.execute("SELECT rid, power, ST_X(position), ST_Y(position), timestamp, mac FROM {0}.{1} WHERE timestamp >= '{2}' and timestamp <= '{3}' and mac = '{4}'".format(
                            self.get_schema(),
                            self.db_param['table_interp'],
                            t_min,
                            t_max,
                            mac
                            ))

        self.data = []

        for i in cur:
            self.data.append(i)

        return self.data

    def select_data_aircraft(self):
        cur = self._connection.cursor()
        cur.execute("SELECT rid, timestamp, ST_X(position), ST_Y(position) FROM {0}.{1}".format(
                            self.get_schema(),
                            'aircraft_position_testbed'
                            ))

        self.data = []

        for i in cur:
            self.data.append(i)

        return self.data

    def update_measurements(self, cluster_id, index_array):
        cur = self._connection.cursor()

        for element in index_array:
            cur.execute("UPDATE {0}.{1} SET cluster_id = {2} WHERE rid = {3}".format(
                            self.get_schema(),
                            self.db_param['table_update'],
                            cluster_id,
                            element[0]
                            ))

        self.commit()

    def insert_cluster(self, id, data, t_min, t_max):
        cur = self._connection.cursor()

        cur.execute("INSERT INTO {0}.{1} (cluster_id, count, power_avg, time_start, time_end, sigma_power, sigma_x, sigma_y, sigma_xy, position)\
                     VALUES({2}, {3}, {4}, '{5}', '{6}', {7}, {8}, {9}, {10} , ST_GeomFromEWKT('SRID=32633; POINT({11} {12} {13})'))".format(
                            self.get_schema(),
                            self.db_param['table_insert'],
                            id,
                            ata[0],
                            data[1],
                            t_min,
                            t_max,
                            data[2],
                            data[4],
                            data[6],
                            data[7],
                            data[3],
                            data[5],
                            500
                            ))

        self.commit()

    def reset_cluster_flag(self):
        cur = self._connection.cursor()

        cur.execute("UPDATE {0}.{1} SET cluster_id = 0".format(
                            self.get_schema(),
                            self.db_param['table_update']
                            ))

        self.commit()

    def delete_aircraft_positions(self):
        cur = self._connection.cursor()
        cur.execute("DELETE FROM {0}.{1}".format(
                            self.get_schema(),
                            'aircraft_position',
                            ))

        self.commit()

    def select_external_positions(self):
        cur = self._connection.cursor()
        cur.execute("SELECT position_id, proc_date, timestamp, flightno, proc_pos_cart_x, proc_pos_cart_y, proc_pos_z FROM {0}.{1} WHERE cat = 21".format(
                            self.get_schema(),
                            'external_position'
                            ))

        self.data = []
```

```python

        for i in cur:
            self.data.append(i)

        return self.data

    def insert_aircraft_pos(self,data):
        self.data = data

        for index in range(0,self.data.__len__()):
            self.insert_aircraft_row(index)

        self.commit()

    def insert_aircraft_row(self,index):
        cur = self._connection.cursor()

        cur.execute("INSERT INTO {0}.{1} (rid, timestamp, flightno, position)\
                     VALUES({2},'{3}','{4}',ST_GeomFromEWKT('SRID=32633; POINT ({5} {6} {7})')))".format(
                self.get_schema(),
                                'aircraft_position',
                                self.data[index][0],
                                date.isoformat(self.data[index][1])+" "+time.isoformat(self.data[index][2]),
                                self.data[index][3],
                                self.data[index][4],
                                self.data[index][5],
                                self.data[index][6]
                                ))

    def _conv_cart(self, lon, lat):
        p = Proj(proj='utm',zone=33,ellps='WGS84')
        return  p(lon,lat)
```

## MU_data_interpolation.py

```python
import xlrd, logging, psycopg2, datetime
from ConfigParser import ConfigParser
from mu_data_interface import Mu_data_interface
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate

CONFIGFILE = "Mu_data_feeder.conf"

class Mu_data_interpolation():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.FILENAME = config.get('input_file', 'FILE_NAME')
        self.FILEPATH = config.get('input_file', 'FILE_PATH')+self.FILENAME
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_measurements':config.get('pg_parameter', 'DB_TABLE_MEASUREMENT'),'table_select':config.get('pg_parameter', 'DB_TABLE_POSITION'), 'table_interp_pos':config.get('pg_parameter','DB_TABLE_INTER_POS')}

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Mu_data_interface(self.db_param)

        self.position_raw_data = self.db.select_position_data()

        self.interpolate_positions()

        self.correlate_meas_pos()

        self.plot_correlation_result()

    def interpolate_positions(self):
        self.interpolation_result = []

        #input data set pre-filtering
        filtered_position_data = []

        for i in range(0,self.position_raw_data.__len__()):
            if i == 0:
                filtered_position_data.append(self.position_raw_data[i])
            else:
                if self.position_raw_data[i][0] != self.position_raw_data[i-1][0] and self.position_raw_data[i][1] != self.position_raw_data[i-1][1]:
                    filtered_position_data.append(self.position_raw_data[i])

        #Window interpolation

        window_w = 7
        window_c = int(window_w/2)

        for i in range(0, 7):#filtered_position_data.__len__()):
            if i < window_c:
                self.interpolation_result.append(self.interpolate(0, window_w, i, filtered_position_data))
            else:
                self.interpolation_result.append(self.interpolate(i-window_c, window_w, window_c, filtered_position_data))
            self.line_integral(i)

    def interpolate(self, window_b, window_w, window_c, pos_data):
        print "window_b, window_w, window_c :", window_b, window_w, window_c
        res_interp = 0.01

        x = np.zeros(window_w+1)
        y = np.zeros(window_w+1)
        v = np.zeros(window_w+1)

        for i in range(window_b ,window_b + window_w+1):
            x[i-window_b] = pos_data[i][0]
            y[i-window_b] = pos_data[i][1]
            v[i-window_b] = pos_data[i][2]

        # Interpolation position
        tck,u = interpolate.splprep([x,y],s=0)

        unew = np.linspace(0,1,25000)
        out = interpolate.splev(unew,tck,der=0)

        # Search for Segment in out
        index_start = 0
        index_end = 0

        for i in range(0, out[0].__len__()):
            if abs(np.linalg.norm(out[0][i]-x[window_c])) < 0.001 and abs(np.linalg.norm(out[1][i]-y[window_c])) < 0.001:
                index_start = i
            elif abs(np.linalg.norm(out[0][i]-x[window_c+1])) < 0.001 and abs(np.linalg.norm(out[1][i]-y[window_c+1])) < 0.001:
                index_end = i

        if index_end == 0:
            print("Index end not found, interpolation step presumably not big enought!")
            return

        print
        print "Interp_section Index: ", index_start, index_end

        # Interpolation velocity
        t = np.arange(0, u.__len__(),1)
        t_new = np.arange(0, u.__len__(), res_interp)

        tck_v = interpolate.splrep(t,v, s=1)
        out_v = interpolate.splev(t_new,tck_v,der=0)

        # Plotting
        """
        plt.figure()
        plt.plot(t, v, ’x’, t_new, out_v)

        plt.title(’Spline interpolation antenna h-plane pattern’)
        plt.show()
    """


    """
        plt.figure()
        plt.plot(x,y,’x’,out[0],out[1],’--’,out[0][index_start:index_end],out[1][index_start:index_end])
        plt.legend([’Position fix’,’Spline k=3’, ’Spline segment’])
        plt.xlabel("x [m]")
        plt.ylabel("y [m]")

        plt.show()
        """

        return [pos_data[window_b+window_c][4],pos_data[window_b+window_c+1][4],out[0][index_start:index_end],out[1][index_start:index_end], out_v[index_start:index_end]]

    def line_integral(self, i):
        line_sum = 0
        line_seg = []

        for k in range(0, self.interpolation_result[i][2].__len__()):
            if k != 0:
                x_1 = np.array([self.interpolation_result[i][2][k-1], self.interpolation_result[i][3][k-1]])
                x_2 = np.array([self.interpolation_result[i][2][k], self.interpolation_result[i][3][k]])
                line_sum += np.linalg.norm(x_2 - x_1)
                line_seg.append(line_sum)

        self.interpolation_result[i].append(line_seg)

    def correlate_meas_pos(self):
        self.result_correlation = []
        for segment in self.interpolation_result:

            if str(segment[1]-segment[0]) != "0:00:01":
                self._logger.warning("Interpolation Segment is wider than 1 sec, it is {0} !!!".format(segment[1]-segment[0]))
            else:

                timestamp_low = segment[0]
                timestamp_high = segment[1]

                print timestamp_low, timestamp_high

                measurements = self.db.select_filtered_measurements(timestamp_low, timestamp_high)

                total_sum_seg = segment[5][(segment[5].__len__()-1)]

                # Correlation
                index_pos_array = []
                for element in measurements:
                    sec = element[1].microsecond/1000000.0

                    dist = total_sum_seg*sec
                    for k in range(0,segment[5].__len__()):
                        if k != 0:
                            if segment[5][k-1] <= dist and segment[5][k] > dist: # search for appropriate interval
                                if abs(dist-segment[5][k-1]) < abs(dist-segment[5][k]):
                                    index_pos = k-1
                                else:
                                    index_pos = k
                                index_pos_array.append([element[0],(segment[2][index_pos], segment[3][index_pos])]) # RID_measurement, (x,y)
            self.result_correlation.append(index_pos_array)

    def plot_correlation_result(self):
        x_interp_pos = []
        y_interp_pos = []

        for i in range(0,self.result_correlation.__len__()):
            for tupel in self.result_correlation[i]:
                x_interp_pos.append(tupel[1][0])
                y_interp_pos.append(tupel[1][1])

        plt.figure()
        plt.plot(x_interp_pos, y_interp_pos, 'x',
                 self.interpolation_result[0][2],self.interpolation_result[0][3],
                 self.interpolation_result[1][2],self.interpolation_result[1][3],
                 self.interpolation_result[2][2],self.interpolation_result[2][3],
                 self.interpolation_result[3][2],self.interpolation_result[3][3],
                 self.interpolation_result[4][2],self.interpolation_result[4][3],#,self.interpolation_result[1][2],self.interpolation_result[1][3])
                 self.interpolation_result[5][2],self.interpolation_result[5][3],
                 self.interpolation_result[6][2],self.interpolation_result[6][3]

                 )
        plt.legend(['WLAN_Measurement','Segment_1','Segment_2', 'Segment_3','Segment_4','Segment_5','Segment_6','Segment_7'], prop={"size":10})
        plt.xlabel("x [m]")
        plt.ylabel("y [m]")
        plt.show()

if __name__ == "__main__":
    Mu_data_interpolation()
```

## MU_data_measure_feeder.py

```
import xlrd, logging, psycopg2
from ConfigParser import ConfigParser
from mu_data_interface import Mu_data_interface

CONFIGFILE = "Mu_data_feeder.conf"

class Mu_data_measure_feeder():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.FILENAME = config.get('input_file', 'FILE_NAME')
        self.FILEPATH = config.get('input_file', 'FILE_PATH')+self.FILENAME
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_insert':config.get('pg_parameter', 'DB_TABLE_MEASUREMENT')}
        self.file_date = self.FILENAME[0:8].split('_')
        self.file_date = "20"+self.file_date[0]+"-"+self.file_date[1]+"-"+self.file_date[2]

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.load_pos_data()

        loader = Mu_data_interface(self.db_param)

        loader.delete_table_mu_position()
        loader.load_data_bunch(self.file_data, self.file_date)

    def load_pos_data(self):
        # 1st fetch data from txt File

    self._logger.info("----- Start Mu_data_measure_feeder skript! -----")
        self._logger.info("Fetch Data from TXT File")

        fh = open(self.FILEPATH,'r')

        self.file_data = []

        for line in fh:
            temp = line.split(';')
            if temp.__len__() == 8:
                if temp[1] != "0" and temp[6] == " M/BEACON":
                    self.file_data.append(temp)

        self._logger.info("Fetch Data from TXT File finished!")

if __name__ == "__main__":
    Mu_data_measure_feeder()
```

## MU_data_position_feeder.py

```
import xlrd, logging, psycopg2
from ConfigParser import ConfigParser
from mu_data_interface import Mu_data_interface

CONFIGFILE = "Mu_data_feeder.conf"

class Mu_data_position_feeder():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.FILENAME = config.get('input_file', 'FILE_NAME')
        self.FILEPATH = config.get('input_file', 'FILE_PATH')+self.FILENAME
        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'), 'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'), 'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,'table_insert':config.get('pg_parameter', 'DB_TABLE_POSITION')}
        self.file_date = self.FILENAME[0:8].split('_')
        self.file_date = "20"+self.file_date[0]+"-"+self.file_date[1]+"-"+self.file_date[2]
        print self.file_date

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.load_pos_data()

        loader = Mu_data_interface(self.db_param)

        loader.load_data_bunch_pos(self.file_data, self.file_date, 0)

    def load_pos_data(self):
    # 1st fetch data from txt File

    self._logger.info("----- Start Mu_data_position_feeder skript! -----")
        self._logger.info("Fetch Data from TXT File")

        fh = open(self.FILEPATH,'r')

        self.file_data = []
        data = []

        for line in fh:
            temp = line.split(',')
            if temp[0] != "$GPTXT":
                data.append(temp)
        print "data.__len__() :", data.__len__()

        for i in range(0,data.__len__()/2):
            if data[i*2][1] == data[i*2+1][2]:
                self.file_data.append(data[i*2+1][1:21]+data[i*2][1:9])

if __name__ == "__main__":
    Mu_data_position_feeder()
```

# D. Annex 4

### Radial_field_model.py

```python
import logging, psycopg2
from ConfigParser import ConfigParser
from radial_model_interface import Radial_model_interface
import numpy as np

CONFIGFILE = "Mu_data_feeder.conf"

class Radial_field_model():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)

        self.db_param = {'host': config.get('pg_parameter', 'DB_HOST'), 'db': config.get('pg_parameter', 'DB_DB'),
                         'user': config.get('pg_parameter', 'DB_USER'), 'password': config.get('pg_parameter', 'DB_PWD'),
                         'schema': config.get('pg_parameter', 'DB_SCHEMA'), 'force_schema': True,
                         'table_raster':config.get('pg_parameter', 'DB_TABLE_RASTER'),
                         'schema_antenna':config.get('pg_parameter', 'DB_SCHEMA_AP')}

        self.band_list = config.get('rm_bands','AP_NAMES').split(' ')

        self.offset_og = config.get('mu', 'OFFSET_OG')

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Radial_model_interface(self.db_param)

        self.db.create_raster(349992, 5294800, 1, 1, self.band_list.__len__())

        self.db.select_raster_meta()
        self.antenna_interp_data = self.db.select_interp_data()
        self.antenna_interp_data_res = self.antenna_interp_data.__len__()/180

        self.ap_position = self.db.select_ap_position()
        self.ap_np_position = []
        self.ap_az = []
        self.band_name_list = []

        for ap in self.ap_position:
            if ap[4].strip(" ") in self.band_list:
                self.band_name_list.append(ap[4].strip(" "))
                self.ap_np_position.append(np.array([float(ap[0]),float(ap[1]), float(ap[2])]))
                self.ap_az.append(ap[3])

        print "ap_np_positions: ", self.ap_np_position[0]
        print "ap_az: ", self.ap_az[0]

        for k in range(0,self.ap_np_position.__len__()):
            #Loop for processing of each RM_Band for configured AP_Positions
            self._logger.info("AP "+self.band_name_list[k]+" will be processed!")
            for i in range(0, self.db.raster_meta.__len__()):
                #Loop for processing of each raster tile
                self.compute_radial_model(self.db.raster_meta[i], self.ap_np_position[k], self.ap_az[k])
                self.db.insert_raster(self.rm, k+1, self.db.raster_meta[i][0])

        # print out of band list

        no = 0
        print "Band List :"
        print "----------------------------------------"
        for band in self.band_name_list:
            no+=1
            print no, band

    def compute_radial_model(self,rm_meta, ap_pos, ap_az):
        print "rm_meta", rm_meta
        upperleftx = 1
        upperlefty = 2
        width = 3
        height = 4
        scalex = 5
        scaley = 6
        numbands = 10
        offsetx = rm_meta[scalex]/2
        offsety = -rm_meta[scaley]/2

        self.rm = []

        for k in range(int(rm_meta[upperlefty]),int(rm_meta[height]*rm_meta[scaley]+rm_meta[upperlefty]), int(rm_meta[scaley])):
            self.rm.append([])
            for i in range(int(rm_meta[upperleftx]),int(rm_meta[width]*rm_meta[scalex]+rm_meta[upperleftx]), int(rm_meta[scalex])):
                self.rm[int((k-rm_meta[upperlefty])/rm_meta[scaley])].append(self.radial_function(offsetx+i, offsety+k, self.offset_og, ap_pos, ap_az))

    def radial_function(self, x, y, z, ap_pos, ap_az):
        g_t = 1
        g_r = 1
        f_c = 5.8
        lambda_wlan = 30/f_c*0.01

        mu = np.array([float(x), float(y), float(z)])
        ap = ap_pos
        d = np.linalg.norm((mu-ap))

        ant_gain = self.ant_gain_determin(mu-ap, ap_az)

        if ant_gain != 99:
            PL = (-ant_gain - 10*np.log10(g_r)+10*np.log10(((4*np.pi*d)/lambda_wlan)**2))*100
        else:
            PL = 0
        return PL

    def ant_gain_determin(self,d, ap_az):
        theta_min = round(ap_az,1) - 90
        if theta_min < 0:
            theta_min = theta_min + 360
        elif theta_min > 360:
            theta_min = theta_min - 360

        theta_max = theta_min + 180
        if theta_max < 0:
            theta_max = theta_max + 360
        elif theta_max > 360:
            theta_max = theta_max - 360

        x = d[0]
        y = d[1]

        if x >= 0 and y >= 0:
            phi = np.rad2deg(np.arctan(x/y))
        elif x >= 0 and y < 0 :
            phi = np.rad2deg(np.arctan(x/y)) + 180
        elif x < 0 and y < 0:
            phi = 180 + np.rad2deg(np.arctan(x/y))
        elif x < 0 and y >= 0:
            phi = np.rad2deg(np.arctan(x/y)) + 360

        if theta_min <= 180:
            if theta_min <= phi and phi <= theta_max:
                return self.antenna_interp_data[int((phi-theta_min)*self.
                    antenna_interp_data_res)][1]

            else:
                return 99
        else:
            theta_min = theta_min - 360
            if phi > 180:
                phi -= 360
            if theta_min <= phi and phi <= theta_max:

                return self.antenna_interp_data[int((phi-theta_min)*self.
                    antenna_interp_data_res)][1]

            else:
                print "x, y, phi:", x, y, phi
                print theta_min-180 , phi+180, theta_max+180
                return 99

if __name__ == "__main__":
    Radial_field_model()
```

### RM_regression_analysis.py

```
import xlrd, logging, psycopg2, math
from datetime import timedelta
from ConfigParser import ConfigParser
from radial_model_interface import Radial_model_interface
import numpy as np
import matplotlib.pyplot as plt

CONFIGFILE = "Mu_data_feeder.conf"

class Rm_regression_analysis():

    def __init__(self):
        config = ConfigParser()
        config.read(CONFIGFILE)
        self.db_param = {’host’: config.get(’pg_parameter’, ’DB_HOST’), ’db’: config.get(’pg_parameter’, ’DB_DB’), ’user’: config.get(’pg_parameter’, ’DB_USER’), ’password’: config.get(’pg_parameter’, ’DB_PWD’), ’schema’: config.get(’pg_parameter’, ’DB_SCHEMA’), ’force_schema’: True, ’ref_cluster’:config.get(’pg_parameter’,’DB_CLUSTER’), ’table_raster’:config.get(’pg_parameter’,’DB_TABLE_RASTER’)}

        self._logger =logging.getLogger(self.__class__.__name__)
        self._logger.setLevel(logging.INFO)
        formatter = logging.Formatter("%(asctime)s -%(levelname)s- @ %(name)-11s : %(message)s")
        ch = logging.StreamHandler()
        ch.setFormatter(formatter)
        self._logger.addHandler(ch)

        self.db = Radial_model_interface(self.db_param)
        self.cluster_ref_data = self.db.select_cluster_ref_data()
        print self.cluster_ref_data
        self.map_values = self.db.select_map_values(self.cluster_ref_data, 1)
        print self.map_values

        self.calculate_offset()
        self.calculate_delta()

    def calculate_offset(self):
        diff = []
        for i in range(0,self.cluster_ref_data.__len__()):
            diff.append(self.cluster_ref_data[i][0]-self.map_values[i]*-1)

        print diff

        xnew = np.arange(10,30,0.01)
        y = []
        y_w = []

        e_sum = 0
        e_sum_w = 0
        sum_w = 0

        for u in self.cluster_ref_data:
            sum_w += u[1]*1.0

        print sum_w

        for a in xnew:

            for i in range(0,diff.__len__()):
                e_sum += (diff[i] - a)**2
                e_sum_w += (((diff[i] - a)**2)*(self.cluster_ref_data[i][1]))/sum_w
            y.append(e_sum)
            y_w.append((e_sum_w))
            e_sum = 0
            e_sum_w = 0

        ynew = np.array(y)
        ynew_w = np.array(y_w)

        ynew = ynew/np.max(ynew)
        ynew_w = ynew_w/np.max(ynew_w)

        print "Result not weighted:", np.min(ynew)
        print "Result weighted:", np.min(ynew_w)

        plt.figure()
        plt.plot(xnew, ynew, ’--’, xnew, ynew_w)

        plt.legend([’E(f)’, ’E(f), weighted’], prop={"size":10})
        plt.xlabel("Offset Parameter a [dB]")
        plt.ylabel("E(f(a))")
        plt.grid(True)

    def calculate_delta(self):
        self.min_weight = 20.924

        diff = []
        for i in range(0,self.cluster_ref_data.__len__()):
            diff.append(self.cluster_ref_data[i][0]-self.map_values[i]*-1-self.min_weight)

        self.db.update_cluster_delta(self.cluster_ref_data, diff)
```

```python


if __name__ == "__main__":
    Rm_regression_analysis()
```

### DB_radial_model_interface

```
import logging, psycopg2
from PostgresInterf import Pg_Interface
from pyproj import Proj

class Radial_model_interface(Pg_Interface):

    def __init__(self, db_param):
        self.db_param = db_param
        Pg_Interface.__init__(self, db_param)

    def load_data_bunch(self,data, date):
        self.data = data
        self.date = date

        for index in range(0,self.data.__len__()):
            self.load_one_row(index)

        self.commit()

    def load_one_row(self,index):
        cur = self._connection.cursor()

        cur.execute("INSERT INTO {0}.{1} (mu_id, timestamp, mac, channel, power, ssid) VALUES({2},'{3}','{4}',{5},{6},'{7}')".format(
                  self.get_schema(),
                  self.db_param['table_insert'],
                  self.data[index][0],
                  self.data[index][2],
                  self.data[index][3].lstrip(' '),
                  self.data[index][4].lstrip(' '),
                  self.data[index][5].lstrip(' '),
                  self.data[index][7].lstrip(' ').rstrip('\n')
                  ))

    def select_raster_meta(self):
        cur = self._connection.cursor()

        cur.execute("SELECT rid, (foo.md).* FROM (SELECT rid, ST_MetaData(rast) As md FROM {0}.{1}) As foo".format(
                  self.get_schema(),
                  self.db_param['table_raster']
                  ))
        self.raster_meta=[]

        for i in cur:
            self.raster_meta.append(i)

    def select_ap_position(self):
        cur = self._connection.cursor()

        cur.execute("SELECT ST_X(position), ST_Y(position), ST_Z(position), antenna_az, name  FROM antenna.ap_position")
        self.ap_meta=[]

        for i in cur:
            self.ap_meta.append(i)
        return self.ap_meta

    def select_interp_data(self):
        cur = self._connection.cursor()

        cur.execute("SELECT degree, power_value FROM antenna.antenna_interp_data")
        self.interp_data=[]

        for i in cur:
            self.interp_data.append(i)
        return self.interp_data

    def insert_raster(self, rm_data, band, rid):

        cur = self._connection.cursor()

        cur.execute("SELECT rid, ST_NumBands(rast) As numbands FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param[’table_raster’]
                            ))

        self.raster_bands=[]

        for i in cur:
            self.raster_bands.append(i)

        print "[rid, bands] :", self.raster_bands

        if self.raster_bands[0][1] < band:
            count_missing_bands = band - self.raster_bands[0][1]
            for i in range(0,count_missing_bands):
                cur.execute("SELECT rid, ST_NumBands(rast) As numbands FROM {0}.{1}".format(
                            self.get_schema(),
                            self.db_param[’table_raster’]
                            ))

        for row in range(0,rm_data.__len__()):
            print "row number :", row
            for column in range(0,rm_data[row].__len__()):
                if self.inside_testbed(column+1, row+1):
                    self.insert_pixel(column+1, row+1, rm_data[row][column],band, rid)

        self.commit()

    def insert_pixel(self,x,y, pixel_value, band, rid):

        cur = self._connection.cursor()

        cur.execute("UPDATE {0}.{1} SET rast = ST_SetValue(rast,{2},{3},{4},{5}) WHERE rid = {6}".format(
                                self.get_schema(),
                                self.db_param['table_raster'],
                                band, x, y,int(pixel_value),rid
                                ))

    def inside_testbed(self,x, y):
        x = self.raster_meta[0][1]+ (x * self.raster_meta[0][5]) - self.raster_meta[0][5]
        y = self.raster_meta[0][2]+ (y * self.raster_meta[0][6]) - self.raster_meta[0][6]

        #print "x, y :", x, y
        cur = self._connection.cursor()
        cur.execute("SELECT ST_INTERSECTS(ST_GEOMFROMTEXT('Point({0} {1})', 32633), (SELECT the_geom FROM Infrastructure.testbed_area WHERE gid = 1))".format(
                                x,y
                                ))

        for i in cur:
            result = i

        return result[0]

    def select_data(self):
        cur = self._connection.cursor()
        cur.execute("SELECT * FROM {0}.{1}".format(
                                self.get_schema(),
                                self.db_param['table_select'],
                                ))

        self.data = []

        for i in cur:
            self.data.append((i[1],i[2]))

    def create_raster(self, lower_x, lower_y, no_tile_x, no_tile_y, no_bands):
        cur = self._connection.cursor()

        factor = 6

        tile_size_x = 77*factor/no_tile_x
        tile_size_y = 92*factor/no_tile_y

        raster_res = 6/factor

        cur.execute("SELECT DropRasterTable ('{0}','{1}')".format(
                            self.get_schema(),
                            self.db_param['table_raster'],
                            ))

        cur.execute("create table {0}.{1} (rid serial, constraint antenna_intp_pk PRIMARY KEY(rid))".format(
                            self.get_schema(),
                            self.db_param['table_raster'],
                            ))

        cur.execute("SELECT AddRasterColumn('{0}', '{1}', 'rast', 32633, ARRAY['16BUI','16BUI','16BUI','16BUI'], false, true, '{3}',{2},{2},100,100, null);".format(
                            self.get_schema(),
                            self.db_param['table_raster'],
                            raster_res,
                            '{0,0,0,0}'
                            ))

        for x in range(0,no_tile_x):
            for y in range(0,no_tile_y):
                cur.execute("INSERT INTO {0}.{1} (rast) VALUES(ST_MakeEmptyRaster({4}, {6}, {2}, {3}, {5}, {5}, 0, 0, 32633)) RETURNING rid".format(
                            self.get_schema(),
                            self.db_param['table_raster'],
                            lower_x+(tile_size_x*x),
                            lower_y+(tile_size_y*y),
                            tile_size_x,
                            raster_res,
                            tile_size_y
                            ))

                rid = cur.fetchone()[0]
                for i in range(0,no_bands):
                    cur.execute("UPDATE {0}.{1} SET rast = ST_AddBand(rast,'16BUI') WHERE rid = {2}".format(
                            self.get_schema(),
                            self.db_param['table_raster'],
                            rid
                            ))

    def select_cluster_ref_data(self):
        cur = self._connection.cursor()
        self.cluster_data=[]
        cur.execute("SELECT power_avg, count, cluster_id FROM mu_data.{0}".
            format(self.db_param['ref_cluster']))

        for i in cur:
            self.cluster_data.append((i[0],i[1],i[2]))
        return self.cluster_data

    def select_map_values(self, cluster_ref_data, band):
        cur = self._connection.cursor()

        map_values = []
        for entry in cluster_ref_data:

            cur.execute("SELECT ST_Value(rast, {2}, (SELECT position FROM
                mu_data.cluster_mast6a_testbed WHERE cluster_id = {3}))FROM
                {0}.{1}".format(self.get_schema(),
                            self.db_param['table_raster'],
                            band,
                            entry[2]
                            ))

            for i in cur:
                map_values.append(i[0]/100)

        return map_values

    def update_cluster_delta(self, cluster_data, diff):
        cur = self._connection.cursor()

        for i in range(0,cluster_data.__len__()):
            cur.execute("UPDATE mu_data.cluster_data_mast6a SET delta_to_map =
                {1} WHERE cluster_id = {2}".format(
                            self.get_schema(),
                            diff[i],
                            cluster_data[i][2]
                            ))

        self.commit()
```