DISS. ETH No. 14283 2001

Algorithms and Software for Radio Signal Coverage Prediction in Terrains

A dissertation submitted to the

Swiss Federal Institute of Technology (ETH) Zürich

for the degree of Doctor of Technical Sciences

presented by

Christoph Stamm
Dipl. Informatik-Ingenieur ETH/HTL

born December 17, 1968 in Schleitheim, Switzerland

accepted on the recommendation of

Prof. Dr. Peter Widmayer, Examiner
Prof. Dr. Werner Bächtold, Co-Examiner

Acknowledgments

I thank all the people who have helped me in realizing this thesis, and with whom it was a pleasure to work, including those not explicitly named here.

First of all, I express my thanks to my advisor Prof. Dr. Peter Widmayer for his guidance, support, the freedom of choice of research topics, and the excellent working environment.

I am grateful to Prof. Dr. Werner Bächtold for being my co-examiner. I could benefit from his capacity as an innovative researcher in electromagnetic fields and microwave electronics.

I also thank all partners and sponsors of my research project, i.e., Prof. Dr. Peter Stucki and Dr. Michael Beck from the Multimedia Laboratory at the University of Zurich, Helmut Köchler and Edouard Dervichian from Swissphone Telecom, the Swiss National Science Foundation, the Swiss Federal Office of Topography, the Swiss Federal Statistical Office, and Swissphoto AG.

I take this chance to thank many people and friends with whom I spent time, academically and socially. I thank Dr. Stephan Eidenbenz, Prof. Dr. Renato Pajarola and Gábor Szabó for their helpful work in the RA$_3$DIO project, Roland Ulber for his fruitful discussions on C++ topics, and Urs Keller for his tenacious discussions on OO topics. I also thank Dr. David Taylor and Zsuzsanna Lipták who generously spent their time checking the English of the thesis. Furthermore, I am thankful to all other (former) members of the research group for their cooperation and for sharing the precious time with me at the ETH: Mark Cieliebak, Daniele De Giorgi, Dr. Shi Fei, Barbara Heller, Dr. Brigitte Kröll, Dr. Sonia Mansilla, Dr. Gabriele Neyer, Dr. Paolo Penna, Conrad Pomm, Peter Remmele, Dr. Thomas Roos, Konrad Schlude, Dr. Ulrike Stege, and Prof. Dr. Roger Wattenhofer.

I am grateful to all the external researchers and students who contributed in part to the several thousand lines of source code of the RA$_3$DIO project; in particular to Thomas Giger, Guido Riedweg, Martin Schneider, and Rolf Spuler.

Last but not least I am very grateful to my parents for helping me reach my goals and to my great love Birgit for her cordial friendship and her love.

Contents

Abstract **8**

Kurzfassung **9**

1 Introduction **11**

1.1 Outline of Thesis 13

2 Basic Concepts **15**

2.1 GIS 15

2.1.1 Introduction in GISs 16

2.2 VR-GIS 19

2.2.1 Terrain Models 20

2.2.2 Dynamic Scene Management 21

2.2.3 LOD 22

2.2.4 The Tiling Concept 23

2.2.5 Terrain Visualization 25

2.3 Public Mobile Communications Networks 26

2.3.1 The Cellular Radio Concept 27

2.3.2 GSM and DCS 1800 29

2.3.3 GPRS and HSCSD 29

2.3.4 UMTS 29

2.3.5 WLL 30

2.3.6 Paging 31

2.4 Cellular Network Design 32

2.4.1 Design Objectives 32

2.4.2 Analytical Design Approach 33

2.4.3 Integrated Design Approach 34

2.5 The Propagation Channel 35

3 Aspects of Terrain Models **37**

3.1 The Earth 38

3.1.1 The Shape of the Earth 38

3.1.2 Geographical Coordinates 40

3.1.3 Spherical and Spheroidal Terrain Models 41

3.2 Hierarchical Triangulations 43

3.2.1 The Longest Side Bisection 44

3.2.2 Modified Longest Side Bisection 45

3.2.3 Progressive Mesh Refinement 49

3.3 Texture Data 51
3.3.1 Progressive Image Refinement 52
3.3.2 Progressive Graphics File (PGF) 53
3.3.3 PGF Results 57

3.4 Geomorphologic Data 62
3.4.1 Raster and Vector Format 62
3.4.2 Storage, Retrieval, and Visualization 63

3.5 Conclusions 64

4 Wave Propagation Prediction 65

4.1 Introduction 66
4.1.1 Antenna Characteristics 66
4.1.2 Free-Space Path Loss 69
4.1.3 Link Budget Concept 70

4.2 Empirical Models 71
4.2.1 The Okumura Model 72
4.2.2 The CCIR Hata Model 74
4.2.3 The COST231 Hata Model 74
4.2.4 Modified Hata Models 75

4.3 Terrain Visibility 76
4.3.1 Point-to-Point Visibility 79
4.3.2 A Divide-and-Conquer Approach for Vertex Visibility 82
4.3.3 Point-to-Region Visibility 87

4.4 Physical-Geometrical Models 90
4.4.1 The Walfish-Ikegami Model 90
4.4.2 Path Profile Models 92
4.4.3 Ray-Tracing Based Models 96

4.5 Conclusions 97

5 Optimization 99

5.1 Minimum Antenna Placement 100
5.1.1 Minimum Set Cover Problem 101
5.1.2 Data Structures 102
5.1.3 Approximation Algorithms 103
5.1.4 Tests and Results 105

5.2 Interference Minimization 106
5.2.1 Delay-Spread Minimization 107
5.2.2 Analysis of the Delay-Spread Problem 108

5.2.3 Simulated Annealing 111
5.2.4 Tests and Results 113
5.3 Future Extensions 115
6 RA$_3$DIO 117
6.1 System Design and Requirements 120
6.1.1 Precision and Correctness 120
6.1.2 Efficiency 122
6.1.3 Extensibility and Reducibility 124
6.1.4 Graphical User Interface (GUI) 125
6.2 System Overview 125
6.3 Pre-Processing 127
6.4 Run-Time System 128
6.4.1 Scene Graph 129
6.4.2 Terrain Manager 131
6.4.3 User Interface and Navigation 132
6.5 Database 134
6.6 The XGobi Link 135
6.6.1 XGobi 136
6.6.2 Distributed Computing Environment (DCE) 137
6.6.3 The RA$_3$DIO/XGobi Link 137
6.6.4 Future Work 141
6.7 Conclusions 141
7 Conclusions and Outlook 143
Color Plates 147
Copyrights 154
References 155
Curriculum Vitae 166
Index 168

Abstract

In this thesis we investigate the components of a radio signal coverage prediction framework based on a geographical information system with virtual reality capabilities (VR-GIS). The framework contains tools for personal communications network definition, analysis, and optimization. It allows the user to explore terrain and simulation data in a virtual reality manner. We focus on both the GIS and telecommunications aspects of our framework.

In contrast to GISs, VR-GISs require improved concepts and techniques such as dynamic scene management and different levels of detail to process huge amounts of terrain data. Terrain data may include terrain surface elevation data, geomorphologic data, and texture data. A data structure for a digital elevation model that supports extraction of triangulated surfaces at an arbitrary level of detail is called a 'multiresolution triangulation model.' We present a new triangle refinement rule to create such a multiresolution model based on the 'longest side bisection' triangulation. Our refinement rule improves the lower bound of the smallest occurring angle from one half to two thirds of the smallest angle in the initial triangulation. In case of texture data, we present a new image file format, called 'Progressive Graphics File' (PGF), which serves efficient storage and very fast progressive loading of different levels of detail. PGF greatly increases coding speeds by allowing a slight degradation in quality; for instance, the decoding of a PGF is six times faster than the decoding of a corresponding JPEG 2000 file.

In the telecommunications part of this thesis we explain in detail new algorithms to efficiently compute the wave propagation prediction of antennas placed on digital terrain models. These algorithms combine well-known empirical wave propagation models with different concepts of visibility. Some of these visibility concepts are frequency-dependent and therefore, computing is more complex. Furthermore, we discuss optimization problems and approximation algorithms related to cost and interference minimization. For instance, we explain a variant of the base station location problem. Since this optimization problem is *NP*-hard, we also discuss implementations and results of some greedy heuristics. In case of quasi-synchronous or simulcast networks, delay-spread is a major interference problem. We present an exact definition of the delay-spread problem and discuss a solution based on Simulated Annealing.

Finally, we present our experimental signal coverage prediction framework. RA$_3$DIO (Radio Antenna placement with 3D Interactive Optimization) supports exploratory interaction, placement of antennas, computation of essential personal communications network characteristics, and interactive optimization of delay-spread and antenna locations. We discuss the system architecture and explain how the design and development of such a framework is a good deal more than merely combining several system pieces which have been solved optimally.

Kurzfassung

In dieser Dissertation untersuchen wir die einzelnen Komponenten eines Frameworks zur Berechnung und Vorhersage der Signalabdeckung in Mobilfunk-Netzwerken. Das Framework baut auf einem geografischen Informationssystem mit „Virtual-Reality“-Fähigkeiten (VR-GIS) auf. Es enthält Werkzeuge zum Aufbau, zur Analyse und zur Optimierung von Netzwerken und gestattet zudem der Benutzerin die interaktive Erkundung des Geländes. Wir präsentieren sowohl Aspekte geografischer Informationssysteme (GIS), als auch der Telekommunikation.

Im Gegensatz zu konventionellen GIS benötigen VR-GIS verbesserte Konzepte und Techniken zur interaktiven Handhabung grosser Mengen von Geländedaten. Dabei spielen verschiedene Auflösungsstufen und dynamisches Szenenmanagement eine wichtige Rolle. Unter Geländedaten verstehen wir nicht nur digitale Geländemodelle, sondern auch geomorphologische Daten und diverse Arten von Texturen. Die Datenstruktur, welche die adaptive Triangulation eines digitalen Geländemodells gemäss einer beliebigen Auflösung unterstützt, bezeichnet man als „Multiresolution“-Triangulationsmodell. Wir zeigen ein neues Verfeinerungsverfahren zur Erzeugung eines solchen Triangulationsmodells, welches auf der Halbierung einer längsten Seite eines Dreiecks basiert. Es verbessert die untere Schranke für den kleinsten Winkel in einer Triangulation von einem Zweitel auf zwei Drittel des ursprünglich kleinsten Winkels. Für Texturen präsentieren wir ein neues Bilddateiformat, genannt „Progressive Graphics File“ (PGF), um verschiedene Auflösungsstufen effizient abzuspeichern und schnell und progressiv zu laden. PGF nimmt kleine Qualitätseinbussen in Kauf, um die Geschwindigkeit der Codierung zu erhöhen; die Decodierung eines PGF-Bildes ist etwa sechsmal schneller als die Decodierung eines entsprechendes „JPEG 2000“-Bildes.

Im Telekommunikationsteil dieser Dissertation stellen wir neue Algorithmen zur effizienten Berechnung der elektromagnetischen Wellenausbreitung in einem digitalen Geländemodell vor. Diese Algorithmen verknüpfen bekannte empirische Ausbreitungsmodelle mit verschiedenen Sichtbarkeitskonzepten. Einige dieser Sichtbarkeitskonzepte sind frequenzabhängig und somit schwieriger zu berechnen. Des weitern diskutieren wir Optimierungsprobleme und Approximationsalgorithmen zur Minimierung der Kosten und Interferenzen in einem Netzwerk. Eines dieser Optimierungsprobleme ist eine Variante des Antennenplatzierungsproblems. Da dieses Problem *NP*-schwer ist, präsentieren wir Implementierungen und Resultate von möglichen Greedy-Heuristiken. In quasi-synchronen Netzwerken kann ein spezielles Interferenzproblem auftauchen. Wir definieren dieses „Delay-Spread“-Problem und diskutieren verschiedene Lösungsansätze und einen Algorithmus basierend auf „Simulated Annealing“.

Im letzten Teil der Dissertation präsentieren wir unsern Prototypen des Frameworks. RA$_3$DIO (Radio Antenna placement with 3D Interactive Optimization) ermöglicht interaktiv das Gelände zu erkunden, Antennen zu platzieren, Ausbreitungscharak-

teristiken zu berechnen und Antennenpositionen und Interferenzen zu minimieren. Wir diskutieren die Systemarchitektur und führen aus, dass der Entwurf und die Entwicklung eines solchen Frameworks weit mehr ist, als die Zusammenführung einzelner optimal gelöster Komponenten.

„Die Computertechnologie ist nicht nur die Erfüllung der postmodernen Ästhetik, sondern sie sorgt auch dafür, dass diese Ästhetik in der breiten Öffentlichkeit, aber auch in den Hörsälen allgemeine Resonanz findet. Computer verkörpern die Theorie der Postmoderne und holen sie auf den Boden der Wirklichkeit."

Sherry Turkle

1 Introduction

The ongoing deregulation of the telecommunications markets all over the world as well as increased global exchange of personal and commercial information have a significant impact on the design of new personal communications networks. Telecommunications companies are faced with an ever increasing complexity of both radio network infrastructure and radio network planning. The need for adequate models for new radio network analysis, design, management, and optimization frameworks[1] is pressing and plays a key role in successful competition. Research and development activities in radio network planning frameworks have recently increased rapidly.

New sophisticated radio network planning frameworks, based on three-dimensional geographical information systems with virtual reality capabilities (VR-GISs), offer the potential to facilitate the planning of their network infrastructure. The reason for this is that virtual reality can overcome typical shortcomings of today's information systems such as limited space for information presentation and limited information exploration abilities. Unfortunately, it is by no means obvious how to design and develop such a sophisticated radio network planning framework. Questions of the following type arise: How can we efficiently handle the huge amount of terrain data? How can we efficiently compute the signal coverage prediction of a set of antennas? How can we simplify the synthesis of radio networks?

Motivated by these questions, the topic of this thesis is the analysis, design and development of a part of a radio network planning framework based on a VR-GIS. Our goal is to develop a framework which contains tools for network definition, analysis, and optimization. Because of its limited functionality in contrast to a sophisticated radio network planning framework we call it a *signal coverage prediction framework*. We discuss both the GIS and telecommunications elements of our signal coverage prediction framework.

Current network planning frameworks are based on GISs without virtual reality capabilities. They sometimes use two dimensional (2D) maps as information basis,

---

[1] We use the term *framework* as a structure containing some components and tools

instead of three dimensional (3D) terrain data. Often, the missing third dimension is a real drawback: first, it makes it impossible to run simulations depending on topography, and second, the data visualization is usually limited to an orthographic map view. Though 3D-GISs without virtual reality capabilities overcome the first drawback, they are usually still limited to orthographic map and profile views. In particular, they do not allow to explore the terrain in a virtual reality manner. This means that aspects and configurations only seen with the dynamic movement of the viewpoint cannot be displayed. For instance, the visible parts of a terrain viewed from a given antenna location can be computed and visualized in a 3D-GIS even without virtual reality capabilities, but the user cannot move her viewpoint to the antenna location and look around.

In contrast to GISs, VR-GISs require improved concepts and techniques such as dynamic scene management and different levels of detail to process huge amounts of terrain data. The terrain data may include terrain surface elevation data, geomorphologic data, and texture data. Large terrain databases call for efficient access methods to provide fast query processing. We present new solutions for storing and retrieving different levels of detail for both digital elevation data and texture data.

From the designer's point of view, the telecommunications part of our signal coverage prediction framework can be seen as one application module of a VR-GIS. Of course, there are many other conceivable application modules for such a VR-GIS, e.g. water, smoke, dust, snowslides, and avalanches simulation for rural areas or light and noise analysis for urban areas. All of these application modules offer a wide variety of interesting algorithmic problems. Some of them are related to computational geometry, because they work on terrains, while others are more related to computational optimization. We present new algorithms to efficiently compute the wave propagation of antennas placed on digital terrain models. We also discuss some optimization problems and approximation algorithms related to cost and interference minimization in radio networks.

To understand in detail the needs of both the VR-GIS and the application module to collaborate with each other, we exemplarily developed an efficient and professional prototype of a VR-GIS with different telecommunications modules for network analysis, design, and optimization. It is important to note here that the design and development of such a framework is a good deal more than merely combining several system pieces which have been solved optimally. The reason is that the efficiency of each piece depends on its own data structure and design decisions. A general data structure for all pieces together as a whole is always a compromise between generality and efficiency. There are many important system design decisions necessary to build a good system. The experience and the results of our analysis, design and development of the signal coverage prediction framework are summarized in four main chapters in this thesis.

1.1 Outline of Thesis

Before we present important details in the four main chapters (3 to 6) of this thesis, we introduce in Chapter 2 basic concepts needed to understand the ideas and terms being used. Since one of our objectives is writing a monograph which can be read like a textbook, we also introduce some general terms and concepts.

In the first part of Chapter 2 we introduce geographic information systems in general which are the conceptual basis of VR-GISs. We schematize the kernel of our VR-GIS that is able to handle large amounts of terrain and texture data in an efficient way. We then introduce triangulated terrain models, explain the concepts of dynamic scene management and level of detail (LOD), and give more detail about the tiling concept. In the second part, we introduce the general structure of wireless personal communications networks and we explain the cellular radio concept as an example of a wireless personal communications network in greater detail. Then we discuss some cellular radio network design objectives which need to be supported in our framework and contrast two different network design processes. We end this preliminary chapter with a few words about the propagation channel. This is necessary to understand the basic approaches of the wave propagation prediction models discussed in Chapter 4.

Chapter 3 deals with the GIS part of our framework. We investigate LOD constrained terrain access and surface triangulation for interactive terrain visualization. In LOD constrained access, an additional query parameter to the range denotes the precision at which the requested terrain is to be constructed. Efficient handling of these LOD requests is often resolved with multiresolution terrain models. We discuss multiresolution models, particularly hierarchical triangulations, in greater detail and present a new result about a triangulation refinement method, called ‘longest side bisection’. LOD constrained access influences not only the terrain model, but also texture data mapped onto the terrain surface. In general, texture bitmaps are a set of equally spaced ‘height’ fields; because of their relationship to terrain models, LOD concept variants can be used. Often, an efficient LOD constrained texture access is based on a bitmap storage format, which serves progressive loading. The efficiency depends on the loading time; we are therefore interested in a storage format which primarily serves very fast progressive loading and secondarily, image quality. Usually, the order of priorities is reversed. Finally, we discuss a new bitmap storage format, called ‘Progressive Graphics File’ (PGF), which serves very fast progressive loading of the different levels of detail.

Chapters 4 and 5 deal with the telecommunications part of our framework. This telecommunications part manages different phases of the personal communications network design process. In the analysis phase of this process, the wave propagation prediction of base stations plays a major role. It involves map and terrain data, antenna characteristics, and a wave propagation prediction model.

In Chapter 4 we introduce different wave propagation prediction models. While some of them are terrain based and use the whole terrain information provided by the underlying terrain, others are empirical and use more limited information. We start our investigation with the modeling of three dimensional antenna characteristics. A three dimensional antenna characteristic is needed to simulate the antenna gain for all points in the terrain. The discussion of wave propagation models begins with well-known empirical models, based on statistical measurements done in the late sixties. They are often based on 2D terrain information, e.g. land-usage types, city types, etc. A typical representative of this group is the model by Okumura and Hata. Because the accuracy of these models also depends on line-of-sight between base and mobile station, we introduce visibility and discuss several visibility concepts. We define the term 'radio visibility' and explain algorithms capable of computing several visibility concepts. We also present new results about visibility algorithms for terrain models supporting the concept of 'dynamic scene management.'

In Chapter 5 we discuss some possible optimizations during the network design phases. In the network definition phase for example, where the network designer looks for the best antenna positions, network infrastructure costs are a natural optimization criteria. Because infrastructure costs mainly depend on the number of base stations, we investigated the following optimization problem: 'Minimize the number of base stations while the entire terrain is covered by the antennas of these base stations'. We show the relationship of this problem to the minimum set cover problem. Since this optimization problem is *NP*-hard, we discuss some greedy heuristics. Some of the other optimization possibilities, for instance interference, depend on the network type: while in cellular networks the co-channel interference needs to be minimal, in simulcast networks the area with delay-spread needs to be minimal. The formulation, an algorithm, and practical results of these optimization problems are also presented in this chapter.

Chapter 6 contains a detailed description of our framework, called RA$_3$DIO. The acronym RA$_3$DIO stands for *Radio Antenna placement with 3-Dimensional Interactive Optimization*. We present the system architecture, based on the kernel of a VR-GIS, give an overview of the functionality, and discuss some design principles used in the design process. While most of the underlying concepts of the different components of RA$_3$DIO are introduced in the other sections, the bidirectional data exchange concept based on remote procedure calls is discussed in Chapter 6.

Finally, an outlook and conclusions are presented in Chapter 7.

„Alle Menschen streben nach Wissen; dies beweist die Freude an den Sinneswahrnehmungen, denn diese erfreuen an sich, auch abgesehen von dem Nutzen und vor allem anderen die Wahrnehmungen mittels der Augen ...“

Aristoteles

2 Basic Concepts

The purpose of this chapter is to introduce basic concepts and terms heavily used in this thesis. The first part of this chapter refers to geographical information systems and the second part to radio network issues.

In Section 2.1 we introduce geographical information systems (GISs) which are the conceptual basis of GISs with virtual reality capabilities (VR-GISs). In Section 2.2 we schematize the kernel of such a VR-GIS which is able to retrieve, store, and visualize large amounts of terrain and texture data in a virtual reality manner. In particular, we introduce triangulated terrain models, explain the concepts of tiling, dynamic scene management and level of detail (LOD), and round off with some remarks on terrain visualization.

For cellular radio networks, we introduce the general structure of wireless personal communications networks and we explain the cellular radio concept as an example of a wireless personal communications network in Section 2.3. Several personal telecommunications services based on the cellular radio concept, e.g. GSM, GPRS, UMTS, are only touched to round off the section. Then we discuss some major cellular network design objectives and two different network design processes which try to achieve the objectives in Section 2.4. Finally, in Section 2.5 we complete this chapter with some characteristics about the propagation channel in mobile communications. These characteristics are important to understand the basic approaches of the wave propagation prediction models discussed in Chapter 4.

2.1 GIS

During the last decade, *Geographic Information Systems* (GISs) have become very popular in a variety of application domains [LT92], for instance in environmental and civil engineering, land surveying, cartography, coverage and network planning, etc. According to M. F. Goodchild: ‘A GIS can be seen as a system of hardware, software and procedures designed to support the capture, management, analysis, modeling and display of spatial-referenced data for solving complex planning and management problems.' With the grown popularity of GISs the need for a solid theoretical background and for high performances in geometric reasoning has become urgent. This has made GISs a field of primary importance for application of computational geometry.

In this thesis, we primarily consider issues related to representation and processing of the geometric aspects of geographic data, with special emphasis on the application of computational geometry techniques. We follow a broad classification of classical geometric data in a GIS into *map data* and *terrain data*. Such classification is not standard in the GIS community, but it is convenient here to identify data characterized by different spatial dimensionality. Map data are located on the surface of the earth and are basically 2D, i.e., they are points, lines, and polygonal regions, which are combined together to form either subdivisions or arrangements, sometimes organized into layers. We call geographic information systems only handling map data *classical GISs*. Terrain data are related with the 3D configuration of the surface of the earth. The geometry of a terrain is modeled as a 2½D surface, i.e., a surface in 3D space described by a bivariate function (Subsection 2.2.1). The term 2½D refers to terrain models that are not 'true' 3D objects, because most part of modeling and reasoning is done in their 2D domain.

Before we discuss the consequences of using terrain data instead of map data we need to understand the concept of GISs in general. Therefore, we give a short introduction in GISs in the next subsection.

2.1.1 Introduction in GISs

The purpose of a GIS is to represent the real world in the form of a data model [BFS]. By transforming and processing simulated data in such a model world, processes affecting the environment can be investigated. Particularly interesting in a country as widely varied as Switzerland is the possibility of combining statistical and survey data such as topography. The results can be a basis for administrative, economic and scientific decisions. The spatial data in GISs covers on one hand:

- the atmosphere
- the surface of the earth
- the soil

and on the other hand:

- population and economy
- technical and administrative installations such as buildings, industrial plants, infrastructure, etc.
- other economical and ecological aspects.

The data contained in GISs can be systematically registered and processed based on a uniform spatial reference system. Common spatial reference systems are either directly based on a reference ellipsoid and three geocentric translation parameters (*Geodetic*

Reference System, GRS) or on a projection of a reference ellipsoid onto a flat surface. The reference ellipsoids are approximations of the shape of the Earth in a specified year, e.g. WGS 1984 and Bessel 1841. WGS-84 for instance, approximates the shape of the Earth in the year 1984. At that time, the equatorial and the polar radius were approximated with 6378137.00 respectively 6356752.31 meters [Col], [WGS].

Several kinds of projections can be used [Mal92]: azimuthal projections using a plane tangent to the reference ellipsoid in the center of the area to be mapped, conical projections, cylindrical projections using a cylinder tangent to the reference ellipsoid along the equator *(Mercator projection)*, or along a line of longitude *(Transverse Mercator projection)*, or along a geodesic *(Oblique Mercator projection)*. A *geodesic* of an ellipsoid is a curve similar to the great circle of a sphere. It is described by the intersection of the boundary of the ellipsoid with a plane through two opposite points on the equator and a third (not collinear) point on the ellipsoid. This third point is called the reference point of the projection. For instance, the Swiss coordinate system is based on an Oblique Mercator projection of the Bessel ellipsoid with Bern as the reference point.

Both, the GRS and the projection based reference system have their own advantages and disadvantages. While the GRS exactly mirrors the reference ellipsoid the geodetic projections are only approximations of the ellipsoid. Geodetic projections may preserve the angles (conformality), the areas (equivalence), or the distances in one direction (equidistance). The three properties (conformality, equivalence, equidistance) are mutually exclusive to one another. However, for small parts of the ellipsoid good approximations with both small errors in angles and areas are possible. In contrast to GRS, where surface distances on ellipsoids cannot be expressed in an analytical formula, reference systems based on projections are often Cartesian and make it simple to compute distances on the surface and angles between points lying on the ellipsoid. However, in GRS a significant simplification in distance computing is possible: numerical integration methods needed to compute distances can be avoided if the shortest elliptic arc between two points on the ellipsoid is approximated by a circular arc. The Swiss coordinate system, for instance, is based on a conformal projection.

Real objects are modeled in a GIS by defining their spatial position (geometry) and their specific characteristics (attributes). Spatial relationships (topology) between the objects and their surroundings are also defined. Objects are the smallest elements within a GIS which can be associated with a spatial position and with attributes:

- Geometrical data define the absolute and relative positions of objects, based on their coordinates within a standardized coordinate system;
- Attribute data describe the thematic, non-geometrical characteristics of objects. Usually, attribute data are stored separately and linked with the geometrical data by a coding scheme.

The efficiency and capacity of a GIS depend on its capability of systematically linking various data and deriving and displaying results from such combinations as clearly as possible. A typical example of data linking is the radio wave emission investigation in a rural or urban area. In order to find out how many local inhabitants would be affected specifically by the radiated power of base stations of a mobile phone network, a system compares computed power emission ranges with data taken from a national census. Often, the geometrical allocation of population data is based on digitized coordinates for all residential buildings registered by the census. Finer differentiation is possible if the necessary attribute data is available, such as selected age groups, professions, etc.

The geometrical data of a GIS are of different types, depending on the dimension of the data items. Most of the GISs only handle up to two dimensional geometrical data. Geometrical data in higher dimensions suffers from the lack of representation on a two dimensional screen. Therefore, higher dimensional data is often projected onto the earth surface and visualized in two dimensions. Usually, the following geometric data types are provided by a GIS:

- *Point data*, defined by spatial coordinates, e.g. the position of a base station of a personal communications network.
- *Point data in an (equally) spaced grid*, also referred to as a dot matrix or a lattice, e.g. the official Swiss land use statistics. Normally, the attributes are allocated to the cell represented by the point and not to the point, e.g. population per hectare.
- *Line data*, defined by the coordinates of their origins and ends (nodes) and of intermediate points (vertices), e.g. river or road networks.
- *Polygon data*, defined by the coordinates of their boundary lines, e.g. lakes, base-station-free areas. Attributes are allocated to the entire polygon, e.g. the area of the polygon.

Normally, two dimensional area information is either represented by a lattice or by a set of polygons. The accurate representation mainly depends on the distribution of the attributes over the covered area. If the distribution is not a simple function on the spatial position, then the function is discretized and represented by point samples in a spaced grid. A good example is the Swiss land-usage statistics which are based on point samples located at grid intersections. Information on the actual type of land-usage is identified and stored for each point. Each sample point thus classified with an attribute statistically represents the area of one grid cell, in the case of the Swiss land-usage statistics one hectare. The error involved in this kind of point sample data depends on the number of points for each attribute within a given evaluation area, as well as on the distribution of these points. Small or linear objects (such as buildings and roads) are represented only in an incomplete and inconsistent way, while large, contiguous areas of land use (such as forests or lakes) are represented with greater precision. This must be taken into account for data analysis and interpretation.

2.2 VR-GIS

Starting from classical 2D cartography [Tom90], today GISs are combined with 3D terrain visualization for environmental and civil engineering, and for simulation applications (see also [HU94]). Unfortunately, just adding 3D terrain visualization capabilities to a 2D-GIS leads to a number of problems. A first (easy solvable) problem occurs if objects of GISs are represented only with 2D coordinates (which is the normal case in a classical 2D-GIS). 2D points have to be located in map data and the third dimension, the height of the point, is approximated by the surrounding terrain points. A bit more difficult is the dimension boost of line objects. Because line objects are normally defined by both control points and 2D functions between each two adjacent control points, new control points have to be inserted if the terrain height varies non-linearly between two adjacent control points.

Another problem is the heavily increased need of main memory to represent and visualize the terrain and all GIS objects. Computing/loading a 3D extension of a main memory-optimized 2D scene either overloads the main memory of the computer or more often the (3D) graphics subsystem. An overloaded 3D graphics subsystem results in a very poor frame rate and reduced interactivity. Because of these and other problems, a GIS with fully integrated 3D terrain visualization capabilities requires improved concepts and techniques to process huge amounts of terrain data. A prototype of such a 3D terrain visualization system with the appropriate real-time surface visualization techniques (LOD, Dynamic Scene Management) has been presented in [KLR+95], [Paj98]. This kind of 3D terrain visualization system can be seen as a kernel of a VR-GIS.

Adding ‘true’ three dimensionality to GISs is a more recent research area. 3D or even 4D capabilities in a GIS are required for instance in geosciences and environmental modeling for representing the structures of either earth, or atmosphere (e.g. levels of rocks in geology (3D); air flow, temperature and pressure in meteorology (4D)). 3D geographic data can be handled with techniques inherited from 3D geometric modeling and CAD systems.

A major difference between classical 2D- and 3D-GISs is the need of an *object carrier surface* in a 3D-GIS. While in a 2D-GIS the geometric objects form by its own the geographic information, a 3D-GIS normally needs an explicit surface carrying the geometric objects. This carrier surface allows surface visualization from an arbitrary viewpoint in a perspective or a orthographic (parallel) projection without visual chaos, because the opaque surface hides invisible objects. In contrast, in a 2D-GIS either a cartographic map or a plane (with the reference grid drawn) may represent such a carrier surface, but there is no need for it, because it does not hide any objects. To simplify discussion, we assume for the rest of this monograph (when nothing else stated) that the carrier surface is equal to the terrain surface. Of course, this is not a necessary assumption, but it does not restrict our results.

2.2.1 Terrain Models

A *terrain* can be regarded as the image of a real bivariate function f defined over a connected domain D in the Euclidean plane. A *Digital Elevation Model* (DEM) is a model of one such terrain built on the basis of a finite set of digital data. Terrain data consist of elevation measures at a set of points $S \subseteq D$; points in S can either be scattered, or form a regular grid. A DEM built on S represents a surface that interpolates the measured elevations at all points of S.

A *triangulation* T is a tessellation of a polygonal region of the plane into triangles that is *conforming*, i.e., for each two triangles of T their intersection is either empty, or it is coincident with a vertex, or an edge of both triangles.

An important class of DEMs are *Triangulated Irregular Networks* (TINs). A TIN is defined by a triangulation of the domain D having its vertices at the points of S. Function f is defined piecewise as a linear function over each triangle. Thus, the surface described by a TIN consists of planar faces. TINs show good capabilities to adapt to terrain features, because they can deal with irregularly distributed data sets and may include surface-specific points and lines. Vertices in TINs may describe nodal terrain features, e.g. peaks, pits or passes, while edges depict linear terrain features, e.g. break, ridge or channel lines [Lee91]. Often, a *Delaunay triangulation* [Del34] is used as a domain subdivision for a TIN, because of its good behavior in numerical interpolation. A triangulation is a Delaunay triangulation if and only if the circumcircle of each triangle does not contain any other vertex inside.

The most common input format of elevation data is a 2D matrix of altitude values, also called the *height field*. These values represent a sampling of the surface's altitude, the z-value, at the vertices of a (regularly spaced) 2D grid in the parallel projection onto the x–y plane. In regularly spaced grids the distance between adjacent grid lines is called the *resolution* of the height field. The resolution in x- and y-direction needs not be the same. The regular height field has a very space efficient representation. It is completely defined by its origin in the x–y plane, the grid resolutions and an ordered sequence of the z-values of all elevation points. Regular grid surface polygonalizations are often used as terrain and general surface approximations. One of the simplest surface polygonalizations built with a regular height field is a triangulation, where each rectangle bounded by two adjacent x- and y-grid lines is represented by two triangles. This triangulated surface model represents the height field exactly. Other surface approximation representations include techniques such as wavelet transforms [GGS95] and methods that meet application specific criteria, such as preserving important terrain features [Dou86], [Sou91].

The triangulation model is the core structure for most of the terrain surface visualization systems. It must carefully be adapted and integrated into the VR-GIS kernel. In order to provide a concise representation, the triangulation should only use as much triangles as necessary. To do this without negative impact on the accuracy of the surface approximation, the triangulation should be adaptive to the terrain structure. This means that high frequency elevation changes in the surface are modeled with more triangles per area unit than low frequency surface regions. Furthermore, this triangulation model must also provide means to extract surface representations at variable precisions. This is needed to enable visualization using multiple LODs. Such a model is also called a *multiresolution* triangulation [DeFMP96].

Given the input dataset, there are two basic techniques for constructing multiresolution triangulations:

- *refinement* methods start from a coarse TIN and produce more refined components by adding vertices progressively in order to increase resolution;
- *simplification* methods start from the reference TIN and produce less refined components by reducing the number of vertices progressively.

A survey about some important concepts, problems, and algorithms on terrain data can be found in [vKre97].

2.2.2 Dynamic Scene Management

[Paj98] Large-scale terrain databases are usually too large to be displayed as a whole, even when using multiple LODs. For instance, the digital elevation model of Switzerland at a 25 meter grid-resolution consists of more than 60 million points, or over 120 million triangles. A very fast graphics accelerator renders only a few million triangles per second. Therefore, only a fraction of such an extensive geometric model can be rendered in real-time at an interactive frame-rate of more than twenty frames per second. This partial scene, however, must be updated dynamically according to changes of the viewpoint and view direction. Maintaining such a triangulated scene which dynamically changes with time involves a *dynamic scene update* mechanism. This update mechanism refers to the periodic reloading and discarding of partial regions of the visible scene. The *scene manager* has to decide upon certain parameters: when and which parts of a scene will be discarded, updated (redefined) or new loaded from the database.

A windowing concept to dynamically move over the surface is a simple and efficient implementation of the dynamic scene management. This is similar to panning and roaming used in image viewing. Only the surface data within the window is kept in main memory. The rest of the data is held and managed on external storage, for instance in a set of files or in a database. The window always represents the visible scene. Any change of the user’s view coordinates likely incorporates an update of the visible scene and therefore an update of the surface data in the main memory. Such a dynamic scene management involving data access on secondary storage is often called a paging mechanism. This paging requires that the secondary storage provides spatial range query functionality to allow fast updates.

Whenever the visible scene window *W* changes to represent a new view *W'*, the outdated data *W\W'* has to be discarded and the newly visible region *W'\W* must be loaded from secondary storage. However, it is not very efficient to perform such an update for every small window variation. Moving the visible window in discrete steps allows fewer, more economical updates. Therefore, a partitioning of the visible scene into a matrix of rectangular *tiles* as shown in Fig. 1, efficiently supports discrete scene updates. Whenever the observer crosses a tile boundary the scene matrix has to be adjusted. Thus the data reloads can be composed from fixed sized range-queries, one range query for each newly visible tile. Subsection 2.2.4 goes into more detail about the tiling concept.

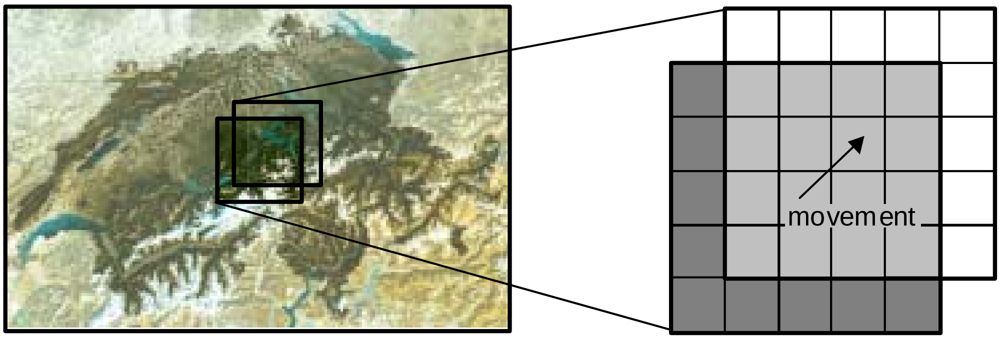


Fig. 1. Dynamic scene management: the dark gray tiles are discarded and white tiles new loaded.

The 2D matrix of terrain tiles of the visible scene is also called the *scene map*. Such a dynamic scene map enhances automatic culling of invisible terrain tiles prior to rendering. Of the current scene map, only tiles which have a non-empty intersection with the view frustum are kept in the display list. Therefore, the per frame rendering operations have to process fewer data because rough culling is already performed, which improves display performance. Additionally, the scene manager keeps track of the LODs at which the different tiles have to be displayed.

2.2.3 LOD

[Paj98] The best way to take full advantage of a given rendering performance is to reduce as much as possible the complexity and the number of geometric elements used to display a scene. However, the simplification of the scene complexity should not result in an inferior visual representation on screen. The concept of *level of detail* (LOD) corresponds nicely to the human's visual system which does perceive objects at different resolutions based on distance and angle of view. Accordingly, instead of showing everything in full detail, objects or parts of a scene are displayed in lower resolution – with higher approximation errors – the farther away they are from the viewpoint and view direction.

If we aim at a continuous LOD rendering instead of a set of fixed precomputed LODs, then a surface visualization should consider following three aspects:

- representation of a scene in an almost unlimited number of different LODs
- display different parts of a scene at different resolutions without discontinuities in between
- smooth changes between different LODs of the same scene

The first aspect is handled by a multiresolution triangulation that can efficiently extract an ε-approximation for any given error tolerance ε. The first part of the second aspect can be handled by different LODs for each tile in the scene map. This LOD is defined by an application dependent visibility priority calculation (see also Fig. 2). The second part of the second aspect is much harder to manage for multiresolution triangulations (see also [dBer97], [dBD95], [DeFP95]). However, it is efficiently solved for grid-based models by the *restricted quadtree* triangulation [LKR+96], [Paj98]. At last, the third aspect may be solved by morphing between different LODs of the same tile [Hop96].

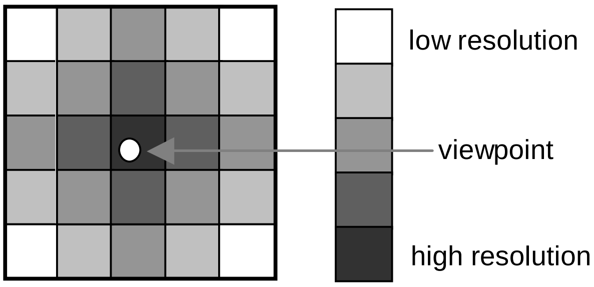


Fig. 2. Scene map with LOD distribution.

2.2.4 The Tiling Concept

We already introduced the tiling concept as a simple and efficient implementation of a dynamic scene management in Subsection 2.2.2. We also presented in Subsection 2.2.3 its elegance in solving some aspects of a continuous LOD handling. Here we discuss the size of the terrain tiles (measured in the number of vertices) and the consequences to data structures and algorithms based on the (visible) scene in main memory.

The size n of a tile is a crucial point of the tiling concept. While the area A_t of each tile is equal, the number of vertices depends on the LOD and thus varies between the tiles in the scene map. Therefore, we need to distinguish between the minimum number of vertices in a tile n_m, the maximum number n_M, and the medium increase of two consecutive LODs Δn. n_M is given by A_t and the resolution. Because n_M depends on A_t it determines the number of tiles in a scene map for a given scene area A_S and hence, it also determines the minimum number of initial range queries to load a scene map. The larger n_M the smaller the number of initial range queries. For a given A_S, a small n_M results in a large number of tiles in a scene map and therefore in a large number of vertices on the four borders of a tile. Because border vertices are part of at least two tiles (assume there is no land's end), they might be redundantly loaded. Nevertheless, the increased expense neatly stitches adjacent terrain tiles together.

n_m is given by the type of multiresolution triangulation and influences the duration of the data transfer initiated by an initial range query. The larger n_m the larger the duration. Δn lies in the range between 1 and $n_M - n_m$. It mostly depends on the number of tiles in a scene map, because in a typical view with direction parallel to the surface the full range of LODs is covered. Thus Δn depends on n_M or a given A_S. Δn influences the duration of consecutive range queries in a similar way as n_m influences the duration of the initial range queries, and it also influences the smoothness of consecutive LODs.

The scene manager uses an appropriate data structure to handle the entire scene map and for each terrain tile a data structure manages the LOD dependent triangulation. Apart from these scene manager data structures an application may contain additional map and terrain data structures, e.g. a terrain dual-graph, visibility map, etc. These terrain application data structures are either based on independent terrain tiles or on the whole scene map and should always reflect the current state of the terrain in the scene map. The application data structures should be updated whenever a tile data structure has changed because of create and LOD-update operations. To reduce the total maintenance effort application data structures should be locally updateable. This means that modifications in the underlying scene map can be efficiently brought forward without recreation of application terrain data structures. In cases where they are not locally updateable each terrain tile should contain its own application data structure and a border data structure to resolve efficient connections across tile borders.

Modifications in application terrain data structures often require reprocessing these data structures. For instance, a visibility map containing all visible parts of the terrain has to be updated whenever the underlying scene map changes. Therefore, a program computing the visibility map must rerun at least for the changed terrain parts. In case the application contains a terrain data structure for the entire scene map, modifications in the scene map require to rerun the program on the whole terrain data structure. Flags indicating the modified parts rarely reduce the time complexity of the algorithm. To overcome this drawback we advocate separate application data structures for each terrain tile if algorithms and their data structures fulfill the following three necessary conditions:

- *Locality:* Local modifications in a scene map must result in local changes in data structures. Or in other words: a data structure of a single tile can be built without the data structures of the other tiles. This allows separate building of the data structures of new loaded terrain tiles while the visible scene window changes.

- *Compatibility:* Separate data structures of adjacent tiles can be neatly and efficiently stitched together. This makes it possible to efficiently build a data structure for a larger part of the entire scene from the data structures of separate tiles.
- *Order:* There exists an order over all tiles such that the data structures of all separate tiles can be built separately. This allows separate building of all data structures in one pass while the initial terrain tiles of the scene map are loaded.

The separate tile data structures are also helpful when using parallel algorithms, where each parallel thread can process a single tile data structure.

2.2.5 Terrain Visualization

One of the fundamental tasks any GIS has to perform is the visualization of geographic data. How to visualize such data depends on the representation used for the terrain, that is, on the DEM used. The basics of terrain visualization and a discussion about hidden surface removal can be found in [dBer97].

In a 3D-GIS with VR behavior a rendering pipeline is usually used to process and visualize the geometric objects (e.g. polygons, lines, B-spline faces, etc.) of a terrain model. The geometric objects are kept together with transformation nodes, grouping structures and light sources in a scene graph. The exact structure of the scene depends on a particular graphics toolkit providing the scene graph and the rendering pipeline. For each defined viewpoint and associated window the scene graph is new rendered in every passage of the rendering pipeline. Such a passage is often called a frame. The number of rendered frames per second (fps), the frame rate, is a measure of speed in VR systems. For smooth movements like in movies one aspires a value of 24 fps or more.

Each geometric object consists of spatial coordinates, vertex and polygon normals, and vertex and/or polygon material properties and maybe associated texture information. The material properties basically define color parameters (e.g. ambient color, diffuse and specular reflection, light emission, etc.) and transparency. Terrain surfaces are often colored in dependence on their height above or below sea level. Such height colors are usually called hypsometric colors. Several schemes for different map scales and height profiles are known from classical cartography. In case of directed light sources the surfaces are additionally shaded according to the angle between a light ray and the normal of the surface. The exact method depends on the chosen shading algorithm, e.g. Phong shading, Gouraud shading. Shading increases the plasticity of the terrain. In classical cartography, the more sophisticated relief maps are often hand drawn shadings. A Gouraud shaded scene is depicted in Fig. 29 on page 147.

Graphics toolkits provide a lot of functionality for real-time visualization of static scenes. However, dynamic changes in the scene graph must be managed by the application itself. This can be done by altering the scene graph's content whenever the scene must be changed. Loading new data from external memory to update the currently visible scene, and providing the geometric objects that can be rendered on screen is the task of the dynamic scene manager of the application. These scene updates are done according to the user's movements as already introduced in Subsection 2.2.2. The scene manager has also to take care of the correct LOD assignment for every terrain tile.

2.3 Public Mobile Communications Networks

A communications network is a spatially distributed arrangement of hardware and software allowing users to exchange information. It consists of a set of nodes that are interconnected to permit the exchange of information. These nodes are distinguished into terminal and communication nodes. The terminal nodes provide the interface to the user. A connection starts and ends at terminal nodes. The communication nodes play the role of relays and establish connectivity among nodes. They are connected by links which are capable to transmit information. A simplified example of such a communications network is depicted in Fig. 3. The terminal nodes, i.e., the mobile stations, are connected via radio links to the communication nodes, i.e., the base stations.

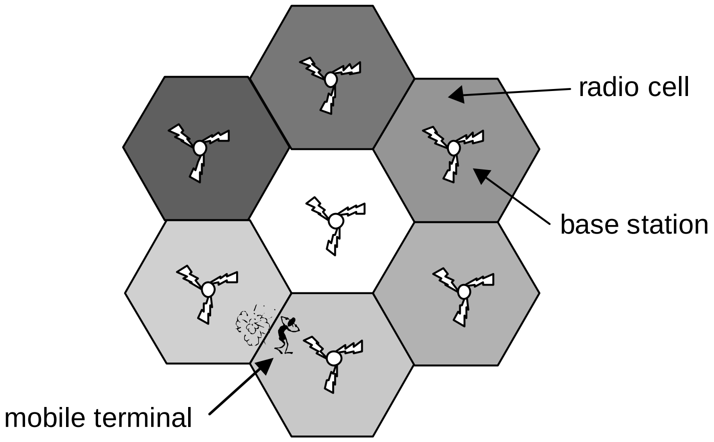


Fig. 3. Cellular radio concept.

The main purpose of mobile communications networks is the provision of wireless communications services at any place at any time. Mobile communications systems consist of two main components: the radio network and the wired/wireless transport subsystem. The radio network provides the wireless access to the mobile stations using a low power radio communications link between a mobile station and a grid of base stations. The transport subsystem is responsible for relaying the communication service either trough a conventional wired network or through a radio link network to its final destination. The endpoint of a connection can either be a customer in a wired system or another wireless subscriber.

In the last ten years, almost all participants in the mobile communications arena have tried to link their services to the personal concept. What was once known as *mobile communications* has become *personal communications* [Mac96]. Several mobile telecommunications services may be described as the precursors, rather than the providers, of personal communications, ranging from simple paging to digital public cordless service.

In radio based systems it is important to use the limited radio spectrum as fully as possible. A typical measure of efficiency is traffic (in Erlangs) per megahertz per square kilometer. High spectral efficiency relies on utilizing small areas, reusing frequencies in nearby areas, etc. This efficiency aspect led to the development of the *cellular radio concept* [McD79]. The basic idea of the concept is the partitioning of the service area into radio cells, typically one to twenty kilometers across. Subsection 2.3.1 goes into more detail about the cellular radio concept.

A fixed personal communications network can provide one element of personal mobility in that a user can be reached at any termination point of the network. A radio terminal introduces additional mobility by providing total movement across the cells of a cellular radio network. Personal communications services may be provided by cordless telephones, pagers, private mobile radio, and satellite links, as well as cellular radio based systems. Cellular techniques are dominant, because they most fully meet the requirement of the mass market in particular, giving benefits of mobility throughout urban environments. However, satellite personal communication is the most effective solution for wide area, low population density environments and paging gives highly reliable one way communication over wide areas.

2.3.1 The Cellular Radio Concept

[Mac96] The key to mass market personal radio systems is frequency reuse. The cellular radio system concept is to use frequencies regularly in a grid of cells across the required coverage area. This is the concept of *frequency reuse* which refers to the use of radio channels, on the same carrier frequency, to cover different areas which are separated from one another by sufficient distances so that destructive interferences are manageable.

Radio cells, under ideal conditions, have almost circular shape, so they are commonly approximated by hexagons. A group of neighboring cells (*a cluster*) uses the same set of carrier frequencies, but there is no reuse within the same cluster. Each cell in a cluster uses another subset of carriers, called a *frequency group*. Assuming idealized equal cell sizes and thus congruent hexagons leads to a regular cell plan with an N cell frequency reuse pattern. N denotes the number of cells in a cluster and thus the number of frequency groups. It can be seen that the frequency set reuse distance D is dependent upon N. The larger N, the greater is D. If R is the radius of the cells, simple hexagonal geometry of the cells gives the following relationship:

$$\frac{D}{R} = \sqrt{3N}\,. \qquad (1)$$

Because of frequency reuse, a mobile communicating on a carrier frequency of its cell, receives not only the wanted signal but also unwanted signals of interferers sending on the same carrier frequency. The larger the frequency reuse distance D the weaker the signals of the interferers. The ratio of the wanted signal power divided by the sum of the signal powers of the interferers is the *Co-channel Interference* (C/I). Considering the first ring of interferers (base stations in neighbor clusters with the same carrier frequency) only results in applicable approximation of C/I. It is clear that C/I improves as N is increased, but because the number of channels per cell is equal to the total number of channels divided by N, the capacity of the network falls. New digital standards for cellular radio use modulation and coding techniques which possibly tolerate lower values of C/I, so that frequency reuse can be improved. In a sectored cell, separate directional antennas are used for each sector, instead of one omni-directional antenna. Therefore, a mobile station is only exposed to a fraction of the number of interfering signals thus allowing these to be brought closer and hence enabling a lower value to be used for N.

Channel allocation is often made on a frequency division basis. For uniform traffic distribution, the channel allocation may follow one of the well-known reuse patterns depending on C/I requirements. In contrast, channel allocation for non-uniform traffic distribution can be optimized using graph coloring heuristics. Furthermore, for irregular terrain and highly built-up areas, non-standard frequency assignments may have to be made. It should, however, be noted that irregular frequency planning can lead to severe problems when a new base site is inserted into the existing network.

In a nation-wide practical cellular radio network there are two distinct regions:

- co-channel interference limited;
- coverage area limited.

Urban areas are generally C/I limited as due to the high user density, small cells are used. Conversely, rural area cells are made as large as the propagation loss will allow, because the required channels per square kilometer is much lower. Comparing 900 and 1800 MHz networks it is found that cell sizes are similar in urban areas, but in rural areas an extra ten decibel propagation loss for the latter halves the diameter of radio cells. Not only different cell sizes, but also variable heights, and terrain features, complicate the situation. Therefore, propagation modeling and simulating tools are needed to plan the frequency reuse and to predict the interference levels between cells.

2.3.2 GSM and DCS 1800

GSM (*Global System for Mobile communication*) is a widely spread digital cellular radio standard. It offers a broad range of data, messaging, and telephony services to the mass market, and enables subscribers to roam internationally between networks belonging to different operators. This enables customers to roam across Europe and beyond without changing their handsets. To achieve this, the GSM specification defines in great detail the use of the radio interface, enabling mobile equipment to communicate with any GSM network. It also defines the major network interfaces, enabling networks to exchange information related to individual subscribers. GSM was designed to be the mobile part of ISDN (*Integrated Services Digital Network*), and offers many equivalent services, including a two-way, acknowledged messaging service, SMS (*Short Message Service*).

DCS (*Digital Cellular Systems*) 1800 is an adaptation of the 900 MHz GSM standard for operation in the 1800 MHz band.

2.3.3 GPRS and HSCSD

Today’s GSM is not very attractive for data transfer, because it is mainly designed for voice. There is a trend to more data transfer between mobiles stations using phones, PDAs, and laptops. Network providers plan several GMS extensions to overcome these drawbacks and to bridge the transition toward future third-generation mobile systems. Two of these extensions are *High Speed Circuit Switched Data* (HSCSD) and *General Packet Radio Service* (GPRS). HSCSD uses software improvements and increases the transfer rate from 9.6 kbps to 14.4 kbps. With four coupled channels for data transfer it almost reaches ISDN throughput. In contrast, GPRS handles its data transfer by cutting up data into packets and using the Internet Protocol (IP) for sending and receiving them. Therefore, the net is only loaded when packets are transferred. The users are always online and the troublesome reconnecting in GSM WAP is not needed anymore.

2.3.4 UMTS

The first generation standards of personal communications networks were diverse and based on analogue technology. Second generation systems are based on digital communication techniques. A standard for an *Universal Mobile Telecommunication System* (UMTS) aims to provide an enabling capability to meet all requirements of the third generation within the frequency range 1.88–2.2 GHz, i.e., low cost, high capacity, personal mobility services. It will offer broadband multimedia-oriented personalized communications to the mass market, regardless of location, network, or terminal. The Special Mobile Group within the European Telecommunications Standards Institute (ETSI) coordinates these standardization activities.

UMTS will combine new technologies with the evolution of existing networks (ISDN and GSM). Also, it will have high interworking capabilities with these networks, especially with enhanced GSM.

UMTS will have both terrestrial and satellite components that will enable service access in a very wide range of radio environments from megacells (satellites) through to macro, mini, micro, to picocells. As a consequence, UMTS must offer universal coverage; that is, it must have connectivity capacity over large geographic areas. Universality also implies the availability of UMTS services in multiple environments (rural, residential, indoor, and business areas). The terminal in this future communications system must adapt automatically its technical characteristics to the propagation conditions found in the different operational scenarios and as function of the services (voice, data, messaging) demanded by the user requires.

2.3.5 WLL

Wireless Local Loop (WLL) subsumes radio communications techniques used to connect the local telephone central offices with the affiliated telephone subscribers. The distance between them is also called the *last mile*. WLL is a point to multi-point technique based on directional microwave radio. A sender or a base station serves up to several hundred subscribers, the receivers. The main advantage should be – the operators say – cheap connection costs and short installation times, because no new cables are buried on the last mile. Between transmitter and receiver must be *line-of-sight* (LOS), which means the transmitter and receiver 'see' each other. This is one of the biggest handicaps of the WLL technique. Therefore, planning of WLL networks with only as much transmitters as needed asks for very efficient radio network planning tools (for different frequency bands) based on high-resolution terrain models for urban and rural areas.

In Switzerland, WLL will transmit on two different carrier frequencies: 3.4 and 26 GHz. The 3.4 GHz radio band allows transfer rates up to 3 Mbps over ten kilometers. The 26 GHz radio band reaches higher transfer rates, up to 40 Mbps, but the maximum distance is only five kilometers. The 3.4 GHz band is more practical in rural areas and for private subscribers, because the smaller transfer rates should be enough. The 26 GHz band is better for urban areas and medium and larger companies, because of their need for higher transfer rates. Big enterprises with a demand of transfer rates up to 155 Mbps are also potential WLL subscribers, because base stations can bundle several channels.

2.3.6 Paging

Radio paging is a mobile radio messaging system which allows a user to be continuously accessible whenever or wherever he/she goes. The pager is a small pocket size radio, which on receiving a call from the paging transmitter, gives a beep. After receiving the beep, the person with the pager takes out the pager and switches on the display to read the message.

Paging is a quasi-synchronous radio technology. This means the pager may receive the same signal from different base stations with different delays and powers. Because the pager tries to synchronize with the dominant signal, the case of several different delayed signals usually does not lead to a problem. When the power difference between the two strongest signals is small and the delay time difference between them is large, then the pager gets into trouble and cannot properly receive the signal. In such a situation the pager is in a non-capture area. These areas are also sometimes called *delay-spread* areas.

Conventional paging technology includes *POCSAG* (Post Office Code Standard Advisory Group) paging systems and *ERMES* (European Radio Messaging System). POCSAG is originally a British system. It generally uses frequency modulation and VHF transmission. ERMES is a pan-European VHF multi-channel, wide-area alphanumeric paging network developed by a technical committee at ETSI. It was launched in 1995 across Europe (Denmark, Sweden and Switzerland first) and it has quietly disappeared into the great meringue of paging technologies available. ERMES devices may eventually be able to receive common frequencies across the US, Asia and the Pacific region. The European commission allocated 400 kHz of spectrum between 169.4125 and 169.8125 MHz for ERMES, and other frequencies around 800 MHz have been reserved in the VHF band. It has a true throughput-rate of about 6.25kbps on 25kHz channels.

The major drawback of conventional Radio Paging is that essentially it has been a one-way messaging system. It was generally perceived simply as ‘the bleep on the belt’, a one-way communication system of limited functionality compared to technologies such as cellular telephony. Due to the surging success of cellular, the paging market around the world took a battering and was in serious decline in the early and mid 90s. However, due to some recent development, Radio Paging industry has not only survived the crisis, it has found a new lease of life. The most important of these developments is a technology called FLEX[2]. By adding a response channel to a traditional paging system, FLEX allows carriers to provide highly cost-effective two-way messaging. The FLEX-technology based protocols are flexible and asymmetric, and can support multiple transmission options. This allows the operators to tailor their

---

[2] FLEX is a registered trademark of Motorola, Inc.;
Internet: http://www. motorola.com/MIMS/MSPG/Products/OEM/FLEXDecoding/

forward and reverse coverage and capacity to most effectively meet their specific needs. FLEX technology has established a commanding lead over its rivals because of the approval by the ITU (International Telecommunication Union) for its inclusion in the ITU-R ‘Recommendation on Codes and Formats for Radio Paging’.

Development such as FLEX has led to a dramatic reversal of fortunes for radio paging, but it needs to further develop and evolve to provide a competitive solution to other wireless technologies.

2.4 Cellular Network Design

The engineering and architecture of large cellular radio networks is a highly complicated task. During the early stages of network engineering, a full assessment of the system design problems is difficult for the network planner. Some of the design challenges only arise due to the collaboration of the network components with each other and can hardly be anticipated. Other engineering problems are postponed to later design stages since their analysis would consume too much time. In particular, efficiency issues are often regarded of minor importance at early network deployment phases, because teletraffic is usually low in new systems. There are several good textbooks, e.g. [Far96], [Stü96], [Lee97], [HP99], which address the different stages of cellular network engineering.

Conventional cellular radio network design methodologies only partially address the complexity of systems. In addition, they still require a lot of personal experience and manual interaction by the network designer. The conventional ‘analytical design approach’ is reviewed in Subsection 2.4.2. New design procedures for the engineering of cellular radio networks need to focus early and equally on all of the major design objectives [Tut99]. Tutschku’s new ‘integrated design approach’ to cellular planning is introduced in Subsection 2.4.3.

2.4.1 Design Objectives

Since cellular radio networks are large scale engineering objects and consist of numerous technical entities and represent high financial investments, they need a systematic design approach using precisely stated network design objectives and requirements. Unfortunately, there exists a large number of technical and economical objectives. Moreover, these design requirements are often contrary to each other.

Beside the primary RF (*Radio Frequency*) objective of providing a reliable radio link at every location in the planning region, state-of-the-art network design has to ensure a high quality-of-service (degree of satisfaction of a user of a service) while considering the aspects of cutting the cost of engineering and deploying a radio network. The design objectives can be separated in three groups:

- *RF design objectives:* The RF design objectives are usually expressed in terms of radio link quality measures. A good link design has to ensure a sufficient radio signal level throughout the planning region and minimized signal distortion by channel interferences among other RF design objectives.
- *Capacity and teletraffic engineering objectives:* Since the available frequencies (channels) are extremely limited, the network design has to predict the teletraffic and therefore the number of channels required in a cell as precisely as possible. To increase the total system capacity, the design has to enforce a large frequency reuse factor.
- *Network deployment objectives:* The network deployment objectives mainly address the economic aspects of operating and engineering a radio network. Deploying a new network or installing additional hardware has significant costs. Therefore, an efficient network design has to minimize hardware costs, for instance by using as few base stations as possible.

2.4.2 Analytical Design Approach

The conventional design procedure for cellular systems is based on the *analytical design approach*. This approach is mainly focused on the determination of the transmitter parameters, like position, antenna type, or transmitting power. It obeys the RF objectives described in Section 2.3.6, but neglects the capacity and the network deployment objectives during the engineering process. This analytical approach is the base for most of today's commercial cellular network planning tools, e.g. Planet[3] and Aircom Enterprise[4].

The analytical approach consists of four phases: radio network definition, propagation analysis, frequency allocation, and radio network analysis. These four phases are iteratively passed in several turns until the specified design objectives and requirements are met. The network improvements are derived from the analysis of the current configuration, so this approach is also denoted as a reverse engineering process.

During the *radio network definition* phase, an (human) expert chooses the cell sites and the transmitter parameters. Usually, the popular concept of distributing the transmitters on a hexagonal grid is used, because it helps to reduce difficulties in the following steps.

---

[3] Planet is a registered trademark of Mobile Systems International (MSI);
Internet: http://www.msi-us.com/planet.htm

[4] Aircom Enterprise is a trademark of AIRCOM International Ltd;
Internet: http://www1.aircom.co.uk/aircomwebsite/

The *propagation analysis* phase evaluates the radio coverage by field strength prediction methods. Usually, several wave propagation prediction models are implemented but the planning tools offer little support in choosing the appropriate model. If the coverage by the already placed transmitters is not sufficient enough, new transmitters and/or new transmitter locations have to be chosen and the propagation has to be analyzed again.

The radio network capacity issues are addressed in the third step, the *frequency allocation*. At first, the teletraffic distribution within the planning region is derived based on rough estimates on the land-usage of the area. Then the (medium/total) traffic per cell is determined and the required number of channels and frequencies are computed by common capacity planning techniques. When, for a given C/I and frequency reuse pattern, all cells can be supplied with the required number of channels, then the process proceeds to the last phase. Otherwise it starts all over again.

The fourth and the last phase, the *radio network analysis*, calculates the quality-of-service values with regard to blocking and hand-over dropping probabilities. Stochastic channel characteristics as well as user demand estimates are used to calculate the network performance. If quality-of-service specifications are met, the process is complete, otherwise it has to be restarted.

The major disadvantages of this analytical approach are its preference of RF design objectives, its isolated design steps, and hence trade-offs between the design objectives. Overall optimization is not feasible, because the reverse reasoning technique complicates the application of algorithmic optimization methods for the generation of synthetic networks.

2.4.3 Integrated Design Approach

The *integrated approach* to cellular network planning overcomes some of the shortcomings of the conventional approach by using forward engineering procedures. The cellular design objectives and constraints are organized in four basic modules: radio transmission, mobile subscriber, system architecture, and resource management. For the synthesis of a cellular configuration the structured set of input parameters to the modules is used. The network is derived from the synthesis of high-level requirements specified by system planners or by other decision makers, so this approach is denoted as a forward engineering process.

Due to the equal and parallel contribution of all the basic modules to the network design, the integrated concept can obey the interactions and dependencies between the objectives. In particular, the capacity and deployment objectives can be addressed early and in an appropriate way. Therefore, the integrated approach is able to find a trade-off between contradictory objectives and achieves optimized network configurations.

The core technique of this approach is the representation of the spatial distribution of the demand for teletraffic by discrete points, denoted as *demand nodes*. These demand nodes constitute a (static) population model for the description of the mobile subscriber density and thus form the common basis of all modules. The base station locating task is reduced to a maximum covering location problem, where the locations of the transmitters are determined in such a way that the number of demand nodes within the permitted service range is maximized.

The demand node concept facilitates not only the mobile user characterization, it also simplifies the resource allocation task. Since the demand nodes are distributed according to the expected service demand, the expected teletraffic in a specified cell can be obtained from the number of nodes in a cell. A potential base station site can be verified, whether it fulfills the teletraffic and hardware constraints or not. This verification can enforce, for instance, the deployment of small and cheap transmitters instead of large and heavily loaded macro cells.

The integrated approach can be very powerful, but needs well-defined objectives from the start. Therefore, this design approach should be used when system planners already have sufficient experience in network engineering and when they are able to transfer their knowledge into general, applicable rules.

2.5 The Propagation Channel

Man-made structures, such as buildings or small houses in suburban areas with sizes ranging from a few meters to tens of meters, exert a decisive influence on mobile communications. In urban environments, the size of structures is even greater. In rural environments, features such as isolated trees or groups of trees may have similar dimensions. These environmental features are similar or greater in size than the transmitted wavelength (VHF: 30–300 MHz; UHF: 300–3000 MHz) and may both block and scatter radio signals, causing specular and/or diffuse reflections. These contributions may reach the mobile receiver via multiple paths in addition to that of the direct signal. In many situations, these echoes are responsible for a certain energy reaching the receiver, thus making communications possible, especially when the direct signal is blocked by environmental features found in the transmitted signal path.

The frequencies used in mobile communications are above 30 MHz, and the maximum length of the links does not exceed 25 kilometers. It must be taken into account that mobile communications are often bidirectional and that the uplink (mobile to base station) is power-limited. Furthermore, mobile radio coverage ranges are short due to the shadowing effects on the terrain and buildings in urban areas. This makes frequency reuse possible at relatively short distances.

Two extreme propagation channel scenarios may be considered:

- *LOS:* A strong direct signal, referred usually as a line-of-sight (LOS) signal, is available together with a number of weaker multi-path echoes. This case occurs on open land or in very specific spots in city centers, in places such as crossroads or squares with good visibility of the transmitter. These situations may be modeled by a Rice distribution for variations in the received RF signal envelope. In this case the received signal is strong and quite steady, with small slow and fast fading due to shadowing and multi-path effects.
- *NLOS:* A number of weak multi-path echoes are received and no line-of-sight (NLOS) signal is available. This case is typically found in highly built-up urban areas. This is the worst of all situations, because the received signal is weak and subject to marked variations due to shadowing and multi-path effects. This kind of situation may also occur in rural areas where the signal is obstructed by trees and woods. The received signal amplitude variations in this situation can normally be modeled with a Rayleigh distribution.

Based on these two extreme scenarios several different wave propagation prediction models have been developed (see also Chapter 4). A deeper insight into modeling the propagation channel in mobile communications is found in [HP99], [Lee97].

„Der Gelehrte studiert die Natur nicht, weil das etwas Nützliches ist; er studiert sie, weil er Freude daran hat, und er hat Freude daran, weil sie so schön ist. Wenn die Natur nicht so schön wäre, so wäre es nicht der Mühe wert, sie kennenzulernen.“

Henri Poincaré

3 Aspects of Terrain Models

Our signal coverage prediction framework requires terrain models for urban and rural areas. The choice between conventional or integrated network designs only slightly influences the requirements of terrain models. In contrast to the design approach, the kind of radio network affects the terrain resolution and thus the system requirements. For instance, planning of optimized WLL networks in urban areas requires high-resolution terrain models containing buildings, whereas planning of paging networks in rural areas can be done with a less accurate terrain model. The higher the terrain resolution, the bigger the storage and performance requirements.

Today, new terrain models are available with resolutions down to a few meters. Especially in urban areas, high-resolution terrain models with resolutions of better than two meters are a good value, because they integrate man-made structures into the terrain surface. Conventional terrain models with appropriate resolutions but without integrated man-made structures are often available for entire countries.

Fast and efficient exploration of huge high-resolution terrains in a virtual reality (VR) manner requires different level of details (LODs) to speed processing of terrain parts in background and outside the view frustum. Thus, the triangulation refinement problem has become an important issue in processing terrain data.

As long as terrain models are only a coarse approximation of the earth surface, a lot of interesting information in between the terrain vertices is lost if it is not modeled either by additional vectorial data or by texture data. While texture data in general uses more storage space than vectorial data, texture data directly mapped on the faces of a triangulation is often better supported by the graphics hardware than vectorial data.

The sections of this chapter contain important aspects of huge high-resolution terrain models. First of all, in Section 3.1 we introduce new definitions for spherical/spheroidal terrains and for spherical/spheroidal triangulations, because spherical/spheroidal terrains are more accurate models of the earth than classical terrain models even if the extent of the regarded terrain part is just a few kilometers. We also discuss the consequences of using spherical or spheroidal terrain models instead of planar ones.

Efficient handling of LOD requests is often resolved with multiresolution terrain models. We discuss in Section 3.2 multiresolution models in greater detail, particularly hierarchical triangulations, and present a new result about a triangulation refinement method, called longest side bisection.

LOD constrained access influences not only the terrain model, but also texture data mapped onto the terrain surface. In general, texture images are a set of equally spaced 'height' fields. Because of their relationship to digital elevation models (DEMs), LOD concept variants can be used. We introduce in Section 3.3 a new image file format, which serves very fast progressive loading of different levels of detail and which reaches compression ratios comparable to JPEG.

Wave propagation prediction not only depends on topography but also on geomorphology. Urban or wooded areas, for instance, influence the propagation of ultra high frequency signals drastically, because the wave length is smaller than the extent of the obstacles. Therefore, we need a way to describe, store, retrieve, and visualize geomorphologic data. The problems of geomorphologic data are discussed in Section 3.4., and Section 3.5 concludes the chapter.

3.1 The Earth

The idea that the Earth is a sphere dates from the Greek geometers of the sixth century BC. Towards the end of the seventeenth century, Newton demonstrated that the concept of a truly spherical Earth was inadequate to explain the equilibrium of ocean surface. He argued that because the Earth is a rotating planet, the forces created by its own rotation would tend to force any liquids on the surface toward the equator. He showed, by means of a simple theoretical model, that hydrostatic equilibrium would be maintained if the equatorial axis of the Earth were longer than the polar axis. The 3D body which corresponds is called an ellipsoid of rotation. The amount of polar flattening f may be expressed by

$$f = (a - b)/a, \tag{2}$$

where a and b are the lengths of the major and minor semi-axes of the ellipse. For the Earth f is close to 1/298. We now know that the difference in length between the two semi-axes is approximately 11.5 kilometers. Since the ellipsoid of rotation approximates a sphere so closely it may be called a *spheroid*.

3.1.1 The Shape of the Earth

Normally, three different ways are used in which the shape and size of the earth are defined. These are in order of increasing mathematical difficulty:

- a plane which is tangential to the earth at some point (*plane assumption*);
- a perfect sphere of suitable radius (*spherical assumption*);
- a spheroid of suitable dimensions and ellipticity (*spheroidal assumption*).

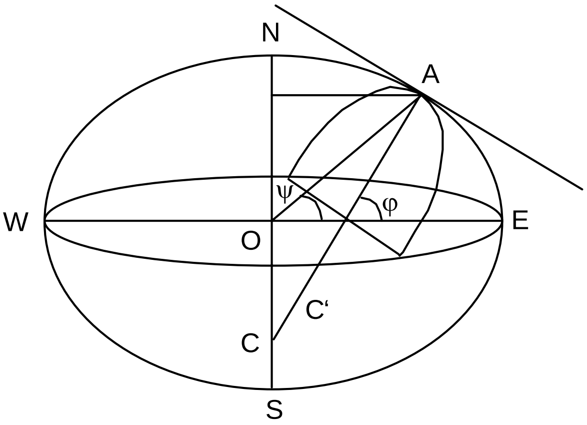


Fig. 4: An ellipsoid in section and the definition of latitude on the spheroid.

The tangent plane has the considerable advantage of simplicity. We cannot state categorically that all angles are truly represented, that great circles are straight or that the particular scales behave in a definite fashion without converting equations into those for a specific projection, so it is desirable to establish practical limits to the size of a survey which can be undertaken using the plane assumption without any significant loss of accuracy arising from this cause. In a typical cartographic survey map of scale 1:5000 the width of the thinnest line is about 0.1 mm. Therefore, the maximum accuracy of such a map is about half a meter. Using half a meter as the maximum accuracy of distance one should not exceed a 35 kilometers extent from the center of the tangential plane. Already at a distance of fewer than four kilometers from the center, the *curvature of the earth* results in a height difference between observed and real terrain height of one meter. It follows that high-accurate terrain models, which are sometimes used for line-of-sight computations (see also Section 4.3), must take into account a height correction Δh depending on the earth curvature. The correct height correction at distance d from a given viewpoint is

$$\Delta h(d) = \frac{R}{\cos(d/R)} - R\,, \tag{3}$$

where R denotes the radius of the earth. Usually, the following approximation of Δh for small values of d is accurate enough:

$$\Delta h(d) = \frac{d^2}{2R}\,. \tag{4}$$

In case of a spherical assumption, of course the radius of the earth curvature is for every point and direction identical. In contrast, in case of a spheroidal assumption this is not true, because the radii of curvature of an ellipsoid are defined as the lines perpendicular to the tangent plane at any point on the curved surface. In Fig. 4 (p. 39) the normal to the tangent plane at point *A* is $\overline{AC'C}$. The normal $\overline{AC'C}$ does not pass through the geometrical center of the ellipse, *O*, except where the normal lies on one of the semi-axes of the ellipse. It follows that the radii of an ellipsoid vary with the position of point *A*. Furthermore, we may distinguish two separate radii at point *A*. One of these is the radius of the arc *NAE*; the other is the radius of the arc which is perpendicular to the arc *NAE* at *A*. The radii are represented in Fig. 4 by the lines $\overline{AC'}$ and $\overline{AC}$ respectively.

We omit the formulas of the two radii, because there are a number of good textbooks about coordinate systems and map projections, e.g. [Mal92], [BS95], [YST00]. For the moment it is enough to know that in general, the arc length of an ellipsoid cannot be expressed in a simple algebraic formula without integrals. Therefore, numeric integration methods or approximations based on power series are used to calculate the distance between two points on a ellipsoid. In case of a spheroid the lengths of the two radii are close which allows us to approximate the spheroid locally by a sphere to compute small arc lengths.

We have already commented upon the fact that the lengths of the two semi-axes of the spheroid differ by only a few kilometers. At a scale of 1:100 million, the difference is about the width of the lines needed to draw the axes. This implies that the main use of the spherical assumption occurs in the use of comparatively small format visualizations showing large parts of the earth's surface, such as maps of the entire world, a hemisphere, a continent or even a very large country. The question, to what maximum scale one can use the spherical assumption, is explained for instance in [Mal92]. This subject was tackled theoretically and in a number of experiments. Here, we only state that in practical cartography the limit is usually taken to be at scale of 1:5 million or thereabouts.

3.1.2 Geographical Coordinates

Another interesting issue is the definition of geographical coordinates. Using the spherical assumption, we normally use longitude and latitude to specify the position of any point *A* on the earth. The *longitude* of point *A* is the angle measured in the plane of the equator between the plane of the meridian through *A* and the plane of some other meridian selected as datum. The choice of a datum meridian for measurements of longitude is arbitrary. Normally, the meridian through the former site of the Royal Observatory at Greenwich is used. The *latitude* of point *A* is the angle measured at the center of the sphere between the plane of the equator and the radius drawn to *A*. In case of the spheroidal assumption, the definition of longitude is the same but we may distinguish between three major types of latitude:

- *geocentric latitude* ψ is the angle, measured at the geometrical center of the spheroid, between the plane of the equator and the straight-line to A;
- *geodetic latitude* φ is the angle between the equatorial axis of the spheroid and the normal to the tangent plane at A, measured at the point of intersection of the normal with the equatorial plane;
- *auxiliary latitudes* are used to map the spheroid to an auxiliary sphere according to certain mathematical principles: conformality, equivalence, or equidistance.

The difference between the geocentric and the geodetic definition varies with latitude and is greatest in latitude 45° where it amounts to nearly 12′ of arc. Geodetic latitude is the more important quantity, and this is the variable which is normally used in GISs and in terrain models.

A point in general position on or above a sphere/spheroid can be represented by three coordinates (λ, φ, z), where λ is the longitude, φ the (geodetic) latitude, and z the positive height above the sphere/spheroid. The height z is always measured orthogonal to the surface.

3.1.3 Spherical and Spheroidal Terrain Models

We already introduced in Subsection 2.2.1 the concept of terrain models. The definition of terrain in Subsection 2.2.1 uses the Euclidean plane as domain, so we have to take special care about the spherical and spheroidal earth shape assumptions. In case of the plane assumption the earth shape corresponds exactly to the definition of a terrain and the use of that definition has no further consequences.

In cartography or in classical GISs the shape of the earth or a part of it has to be mapped or projected onto a flat surface, e.g. plane, cone, or cylinder. This flat surface is usually the paper in cartography on which the maps are printed, and the computer screen in classical GISs. In contrast, in a VR-GIS, which is able to handle real 3D data in a VR manner, we do not need this intermediate terrain data mapping/projection step. We may model the shape of the earth in a more realistic way, either as a sphere or as a spheroid. Therefore, we have to slightly modify the definitions of a terrain and a triangulation from Subsection 2.2.1.

A *spherical (spheroidal) terrain* can be regarded as the image of a real bivariate function f defined over a connected domain D on the surface of a sphere (spheroid). By the image $z = f(\lambda, \varphi)$ of the function f of a point (λ, φ) we mean the point (λ, φ, z) in geographical coordinates.

A *spherical (spheroidal) triangulation* is a conforming tessellation of a region of the surface of the sphere (spheroid) into triangles. If the tessellation is defined on the entire sphere (spheroid), then the spherical (spheroidal) triangulation is a special case of a orientable 2-manifold. In any case the (spherical or spheroidal) triangulation is a polyhedral surface.

Of course, a spherical or spheroidal terrain model is only useful if there are some advantages in having a spherical/spheroidal domain. We assume that a DEM is given in geographical coordinates. This is the normal case if terrain data for entire countries or even continents are processed. If we use a spherical/spheroidal terrain model, then one advantage is that we can omit otherwise required preprocessing steps to project the given DEM on a flat surface. Such projection procedures are sometimes very time consuming, especially when we use the spheroidal assumption. A second advantage is the direct use of geographical coordinates: the terrain is now the image of a bivariate function $f(\lambda, \varphi)$, where λ is the geographical longitude and φ is the geodetic latitude, so we do not need coordinate transformations to process coordinates, provided that no other coordinate system is used in the user interface of the GIS. A third and main advantage is the superfluous height correction for earth curvature, which has to be applied, for instance, in terrain visibility algorithms to get correct visibility maps of the earth.

Among some other advantages there are also some disadvantages. In a terrain model defined in the Euclidean plane we can simply apply the Euclidean distance to measure distances and angles. In case of a spherical (spheroidal) terrain T we have to distinguish between two kinds of distance between two points P and Q on or above T: the *base distance* is the great circle (geodetics) arc length between P' and Q', where P' and Q' are the projected points P and Q with $f = 0$, respectively; the *space distance* is equal to the Euclidean distance between P and Q if T is embedded in $\mathbb{R}^3$. While the base distance on a sphere can be expressed in a simple algebraic formula without integrals in general, the base distance on a spheroid cannot. The computation of the space distance in both cases is based on a simple trigonometric conversion from geographical coordinates to Cartesian coordinates. The computation of azimuths between two points on the spheroid is also more complicate than the same computation on a sphere. Formulas for the different distance and azimuth expressions are found in [Mal92]. Furthermore, terrain algorithms have to be revised to ensure they do not refer to a planar domain. For the plane and the sphere several hierarchical spatial data structures are available, for instance sphere quadtrees [Fek90], and have been demonstrated to be useful for global GISs [GS92]. Weighting these advantages against additional computational complexity costs, the appropriate reference model has to be chosen carefully.

Before we end this section, we should note that all concepts and techniques introduced in the rest of this chapter are also valid for spherical/spheroidal terrains and triangulations unless stated. We use the terms terrain and triangulation in a more general way not restricted to planar domains.

3.2 Hierarchical Triangulations

Multiresolution triangulation models offer the possibility of visualizing a terrain at different degrees of resolution: a coarse representation can be used in areas far from the observer, while high resolution can be used close to the point of interest. A multiresolution triangulation model is effective if its storage cost does not introduce serious overhead with respect to a simple triangulated terrain model at the maximum precision, and if its access and manipulation algorithms are kept efficient [DeFP95].

A large subclass of multiresolution triangulation models proposed in the literature are characterized by a (nested) subdivision of the domain. Such models are usually called *hierarchical* (introduced by De Floriani et al. in 1984). The hierarchical triangulation guarantees that the difference in elevation of any location between terrains described by the resulting multiresolution TIN and the original DEM never exceeds a predefined precision level.

If we restrict ourselves such that one triangle is exactly partitioned into a set of triangles in each step, then we call the hierarchical triangulation *strict*. A strict hierarchical TIN can be effectively described by a tree where nodes are the fragments (local TINs), and arcs correspond to containment of a TIN into a triangle of another TIN. Hierarchical TINs rely on a top-down refinement process, driven by various criteria, e.g. random or accuracy-driven strategies for the insertion of points, Delaunay or heuristic triangulation, etc. The continuity of the surface is guaranteed through a consistent refinement of edges. *Adaptive Hierarchical Triangulation* [DeFMP96], *Hierarchical Delaunay Triangulations* [DeFMP96], and *Longest Side Bisection* [RS75], [SEP98] are three examples of such strict hierarchical triangulation models.

In more general hierarchical triangulation models, the spatial interference between two fragments does not necessarily reduce to a containment of one fragment into the other. The first proposal is the *Delaunay pyramid* [DeFMP96], which encodes fragments from a sequence of Delaunay-TINs describing a terrain at a sequence on increasing resolutions. Such model does not rely on a special construction technique and can be built by simplification as well as by refinement. Interference links are stored between pairs of consecutive triangles which have a proper intersection. The model proposed by [dBD95] is built by iterative simplification on a Delaunay-TIN. At each step, a set of independent vertices (i.e., vertices that are not endpoints of the same edge) of small degree is removed and the ‘holes’ left by those vertices are re-triangulated using the Delaunay triangulation. Interference links are maintained between the triangles incident to a removed vertex and those created to fill the hole.

Many criteria have been proposed as to what constitutes a ‘good’ triangulation. The criteria are either of topographical or geometrical characteristics. Typical geometrical characteristics are: maximizing the smallest angle or minimizing the total edge length. The Delaunay triangulation satisfies the property of maximizing the smallest angle, but there is no refinement process known, which leads to a strict hierarchical

Delaunay triangulation. In our applications (visualizing large terrains) we are interested in triangles without small angles, because small angles have some drawbacks in visualization, e.g. long, disturbing edges, flickering, and shading artifacts.

Hierarchical triangulation models can be built both by refinement and by simplification methods. A typical triangular mesh refinement method is the vertex split operation, while a typical triangular mesh simplification method is the edge collapse operation. Good overviews of mesh simplification methods can be found in [PS97], [HG97]. Because we discuss in the next subsections a new result of a specific refinement technique, we concentrate on refinement methods. The triangulation refinement problem can be formulated as follows [Riv93]: given a non-degenerate triangulation, construct a locally refined triangulation with a desired resolution or a desired maximum error such that the smallest angle is bounded.

Analogous to the criteria for 'good' triangulations there are several criteria for a 'good' refinement strategy. For instance, minimizing the number of refinement steps to reach the desired resolution/maximum error, minimizing the number of subtriangles in one refinement step (to gain a maximum adaptivity), or leading to a strict hierarchical triangulation. If the aim is a strict hierarchical triangulation with maximum adaptivity in terms of the number of triangles, then the 'longest side bisection' is a possible refinement strategy.

3.2.1 The Longest Side Bisection

The *longest side bisection* of a triangle t is the partition of t by the straight-line segment from the midpoint of its longest edge to the opposite vertex. The neighbor of triangle t is the neighboring triangle t' which shares with t a longest side (the candidate for bisection) of t. Two triangulations τ_i, τ_j are said to be *adjacent* along a straight-line segment l if their domains intersect only along l; τ_i, τ_j are said to be *matching* along l if they are adjacent along l and if each vertex on l is both a vertex of τ_i and a vertex of τ_j; a triangulation τ is said to be *matching* if all adjacent triangulations τ_i, $\tau_j \in \tau$ are matching. The longest side bisection triangulation is a strict hierarchical matching triangulation, where each refined triangle is subdivided using the method of longest side bisection.

A triangle subdivision based on the longest side bisection solves the triangulation refinement problem with maximum adaptivity, because in each refinement step only two new subtriangles are constructed. It also leads to a strict hierarchical triangulation, because every single triangle can be subdivided into a pair of smooth TINs whose geometrical properties only depend on the initial triangulation [Riv93]. Furthermore, because the point location in a triangulation of size N takes $O(\log N)$ time and the work for one point insertion uses only constant time, the insertion of k points can be performed in $O(k \log N)$ time.

However, this recursive mesh refinement can produce non-matching triangulations. In order to make the triangulations matching, the local subdivision of a given triangle t involves a refinement of its neighbor t'. We bisect t and its neighbor t' and continue this process iteratively until the last two triangles share the same longest side. The same idea has to be applied in order to match the set of non-matching vertices generated in the inverse order in which they were created. This triangulation refinement process is sometimes called *Rivara refinement* [Riv93].

If the initial polygonal region is a square, which is split into two right angular, isosceles triangles, then the described refinement method leads to a restricted quadtree triangulation structure which is thoroughly described in [vHB87], [SS92], [Paj98a]. The restricted quadtree triangulation is a regular hierarchical triangulation and the longest side bisection triangulation is a generalization of the restricted quadtree triangulation with irregular angles.

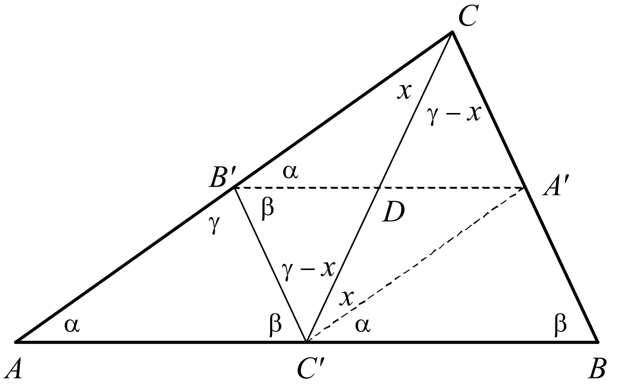


Fig. 5. Bisections of a triangle ΔABC.

3.2.2 Modified Longest Side Bisection

Let ΔABC (see Fig. 5) be a given triangle with interior angles α, β, and γ located at A, B and C, respectively. If ΔABC is bisected into two triangles $\Delta A_iB_iC_i$ with interior angles α_i, β_i and γ_i, $i = 1, 2$, we use both the notations

$$(\alpha,\beta,\gamma) \longrightarrow (\alpha_i,\beta_i,\gamma_i) \text{and} (\alpha_i,\beta_i,\gamma_i) \longleftarrow (\alpha,\beta,\gamma).$$

(α, β, γ) denotes a similarity class of triangles with interior angles α, β, and γ and '$\rightarrow$' is a binary relation on the set of all these similarity classes. We also use the notation $\overline{MN}$ to denote the line segment between the points M and N and $|\overline{MN}|$ to denote its Euclidean distance.

Let A', B' and C' be the midpoints of $\overline{BC}$, $\overline{AC}$ and $\overline{AB}$, respectively. Without loss of generality we assume that $0 < \alpha \le \beta \le \gamma$. Because the sizes of the edges of

ΔABC are in the same relation as the opposite angles, it follows $|\overline{BC}| \le |\overline{AC}| \le |\overline{AB}|$. $|\overline{BC}| \le |\overline{AC}|$ yields $|\overline{AC}| > |\overline{CC'}|$. It follows in combination with $|\overline{AC}| > |\overline{AC'}|$ that $\overline{AC}$ is the longest side in $\Delta AC'C$. From Fig. 5 we obtain

$$\begin{tikzcd}
(\alpha,\beta,\gamma) \arrow[r] \arrow[d, shift right] & (\alpha+x,\beta,\gamma-x) \\
(\alpha,\beta+\gamma-x,x) \arrow[r] \arrow[u, shift right] & (\alpha+\beta,\gamma-x,x)
\end{tikzcd} \quad . \tag{5}$$

These relations are valid in general.

LEMMA 3.1. If

$$|\overline{BC}| \ge |\overline{CC'}| \ge |\overline{BC'}|, \tag{6}$$

then all three edges of triangle ΔABC would be bisected in two consecutive refinement steps.

PROOF. $\overline{AB}$ is the longest side of triangle ΔABC. Thus in the first step ΔABC is bisected into two triangles $\Delta AC'C$ and $\Delta BCC'$ (see Fig. 5, p. 45). We also know that $\overline{AC}$ is the longest side of $\Delta AC'C$ and is therefore bisected in the second refinement step. Because of Equation 6 $\overline{BC}$ is a longest side of $\Delta BCC'$ and therefore, it will be bisected in the second refinement step too. □

The *modified longest side bisection* of a triangle ΔABC is equal to the longest side bisection of ΔABC, except when Equation 6 holds. In cases where Equation 6 holds, ΔABC is subdivided into four equal triangles (they are all similar to ΔABC) using line segment $\overline{A'B'}$ instead of $\overline{CC'}$. This particular subdivision is often called *quaternary triangulation* [GG79].

$T(A,B,C)$ is an infinite family of triangles, formed by iteratively applying the modified longest side bisection to triangle ΔABC and to its descendants.

THEOREM 3.2. *Let α be the smallest interior angle of triangle ΔABC. If Δ is any triangle in T(A,B,C), and θ is an interior angle of Δ, then θ ≥ 2α/3.*

The proof of Theorem 3.3 consists of two parts. In the first part we prove that only one refinement step applying the modified longest side bisection does not falsify the theorem. In the second part we prove that the theorem is also true if we iteratively apply the modified longest side bisection to triangle ΔABC and its descendants.

PROOF (Part 1). If Equation 6 is valid (see region ① in Fig. 6), then the precondition of the modified longest side bisection holds, and the smallest interior angle $\forall \Delta \in T(A,B,C)$ is by definition equal to α, because of the quaternary triangulation applied in this case. Otherwise, if Equation 6 is not satisfied, we have to check two different cases: $\gamma \geq \pi/2$ (see region ② in Fig. 6) and $\gamma < \pi/2$ (see region ③ in Fig. 6). Before we can do this, we need some intermediate results.

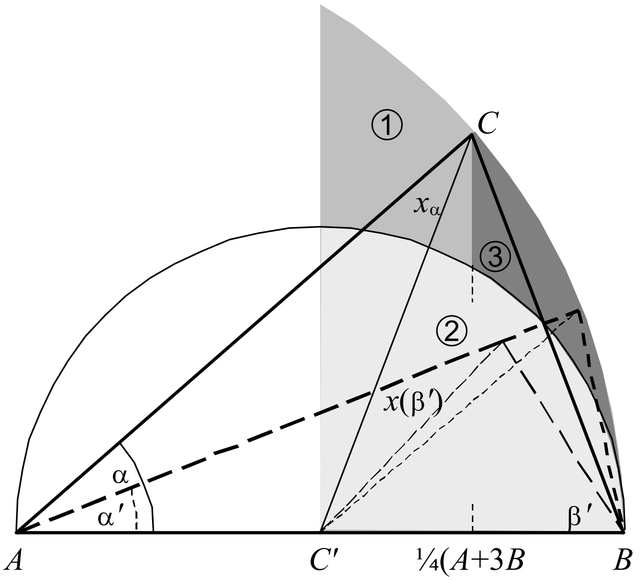


Fig. 6. Various values of the angle *x*.

We assumed without loss of generality that $0 < \alpha \leq \beta \leq \gamma$ and because $\alpha + \beta + \gamma = \pi$, it follows that

$$\alpha \leq \frac{\pi}{3} \leq \gamma < \pi \quad \text{and} \quad \beta < \frac{\pi}{2} . \tag{7}$$

The relations $|\overline{B'C}| = ½|\overline{AC}| \geq ½|\overline{BC}| = |\overline{A'C}|$ yield

$$x \leq \gamma - x. \tag{8}$$

Because also $\beta \geq \alpha$ and $|\overline{AC}| + |\overline{BC}| > |\overline{AB}|$, it follows $|\overline{AC}| > |\overline{AC'}|$ and

$$\gamma - x + \beta \geq \pi/2 . \tag{9}$$

In Fig. 5, $\gamma - x + \beta \geq \pi/2 > \alpha$, $x + \alpha > \alpha$, and $\pi - \gamma = \alpha + \beta > \alpha$. Thus, the only candidates for angles less than α are x and $\gamma - x$.

1. $\gamma \geq \pi/2$: If $\gamma \geq \pi/2$, then $|\overline{AC'}| \geq |\overline{CC'}|$, therefore

$$x \geq \alpha \Leftrightarrow \gamma \geq \pi/2. \tag{10}$$

If follows from Equation 8 and Equation 10 that $\gamma - x \geq x \geq \alpha$, and hence all interior angles of $\Delta AC'C$ and $\Delta BCC'$ are greater or equal to α. Thus, the modified longest side bisection of ΔABC does not increase the smallest interior angle in the two subtriangles.

2. $\gamma < \pi/2$: If $\gamma < \pi/2$, then $|\overline{AC'}| < |\overline{CC'}|$, and because Equation 6 is not satisfied, it holds that

$$|\overline{BC}| < |\overline{CC'}|, \tag{11}$$

and hence $\alpha + x < \beta \leq \gamma$. It follows that

$$\gamma - x > \alpha\,. \tag{12}$$

Therefore, the only remaining angle that can be smaller than α is x. We are interested in a lower bound for x/α, so we first have to find the minimum of x for a fixed α. Let x_α be this minimum. With reference to Fig. 6 (p. 47), we fix A, B, and α, and change β between $\beta_{min} = \alpha$ and $\beta_{max} = \gamma$. Clearly, $x = x(\beta)$ is a decreasing function of β in the region $\beta_{min} \leq \beta \leq \beta_{max}$. Thus, with $\beta + \gamma = \pi - \alpha$ and $\beta \leq \gamma$, x is minimal for a fixed α if $\beta = \gamma = (\pi - \alpha)/2$. With a little trigonometry [RS75], we obtain

$$\tan x_\alpha = \frac{\sin\alpha}{2 - \cos\alpha}\,. \tag{13}$$

Note, that x_α is an increasing function of α in the region $0 < \alpha \leq \pi/3$ (see Fig. 6, p. 47), which can be easily verified by computing the derivative of x_α, using Equation 13. Because vertex C lies in region ③, we are only interested in α in the range $0 < \cos\alpha \leq \tfrac{3}{4}|\overline{AB}|/|\overline{AC}|$. If α is small, x_α is a better lower bound than $2\alpha/3$, since

$$\lim_{\alpha \to 0} \frac{x_\alpha}{\alpha} = 1\,.$$

Finally, we get our lower bound for the quotient x_α divided by α if we set $\beta = \gamma$ and hence $|\overline{AC}| = |\overline{AB}|$ and $\cos\alpha = \tfrac{3}{4}|\overline{AB}|/|\overline{AC}| = \tfrac{3}{4}$:

$$\tan x_\alpha = \frac{\sqrt{1 - \cos^2\alpha}}{2 - \cos\alpha} = \frac{\sqrt{7}}{5}\,, \tag{15}$$

$$\frac{x_\alpha}{\alpha} = \frac{\arctan\frac{\sqrt{7}}{5}}{\arccos\frac{3}{4}} \cong 0.67 > \frac{2}{3}\,. \tag{16}$$

□

Now, we have proven for one refinement step that x is larger than $2\alpha/3$ and that all interior angles of triangle $\Delta BCC'$ are greater or equal to α. Next we have to prove that the iterative refinement of triangle ΔABC does not yield an angle smaller than x.

PROOF (Part 2). Because one refinement step of ΔABC does not yield an angle smaller than x, we know that the further refinement of $\Delta BCC'$ does not yield an angle smaller than x too, because all angles in $\Delta BCC'$ are larger than α. To show that the iterative refinement of ΔABC does not falsify the theorem, we have to prove that the further refinement of a triangle with an interior angle x does not yield an angle smaller than x. Therefore, we focus on triangle $\Delta AC'C$ with point C lying in region ③.

We already know that $\overline{AC}$ is the longest side in $\Delta AC'C$ and is therefore bisected in the second refinement step of this triangle. Because we are only interested in subtriangles with at least one angle smaller than α, we omit $\Delta AC'B'$, because $\Delta AC'B'$ is similar to ΔABC (see Fig. 5, p. 45). To prove that a further refinement of subtriangle $\Delta B'C'C$ does not yield an angle smaller than x, we bisect $\Delta B'C'C$ once more. The relations $|\overline{CC'}| > |\overline{AC'}| = \tfrac{1}{2}|\overline{AB}| \geq \tfrac{1}{2}|\overline{AC}| = |\overline{CB'}|$ yield

$$\alpha + \beta > \gamma - x \,. \tag{17}$$

Because of Equation 8 and Equation 17, $\overline{CC'}$ is the longest side of $\Delta B'C'C$ and is therefore bisected. Let D be the midpoint of $\overline{CC'}$. With reference to Fig. 5, $\Delta DB'C'$ is similar to $\Delta C'BC$ and $\Delta B'DC$ is similar to $\Delta AC'C$. Thus, we obtain

$$\begin{tikzcd}
(\alpha,\beta,\gamma) \arrow[r] \arrow[d, shift right] & (\alpha+x,\beta,\gamma-x) \\
(\alpha,\beta+\gamma-x,x) \arrow[u, shift right] \arrow[r, shift left] & (\alpha+\beta,\gamma-x,x) \arrow[l, shift left] \arrow[u]
\end{tikzcd} \quad . \tag{18}$$

The configuration in Equation 18 is such that arrows going outside of it can originate only at $(\alpha + x, \beta, \gamma - x)$. Because of Equation 12, all angles of $(\alpha + x, \beta, \gamma - x)$ are larger than α. □

We now summarize the proof of the theorem. In each refinement step of triangle ΔABC, we only get interior angles greater or equal to α, except when point C belongs to region ③. In this case we have shown in Equation 16 a new lower bound for x of $2\alpha/3$. Additionally, we have shown in Equation 18 that further refinements of a triangle with angle x does not yield triangles with a smaller angle than x. Therefore, the recursive modified longest side bisection refinement $T(A,B,C)$ does not produce any angles smaller than $2\alpha/3$.

3.2.3 Progressive Mesh Refinement

The modified longest side bisection of a triangle as defined in the previous subsection has a minor drawback, because it sometimes splits the triangle into four instead of only two subtriangles and therefore loses its maximum adaptivity. For that reason, its usage in mesh refinement is limited. To omit this disadvantage, we define the following *modified longest side bisection rule*: Assume triangle t is in the first step refined by applying the longest side bisection method. If Equation 6 holds for t and only if both descendants of t have to be refined, then t is subdivided into four similar triangles by applying the modified longest side bisection method.

A continuous refinement of the triangle mesh τ can efficiently be achieved by an iterative subdivision of a longest edge of all triangles t in τ. For that purpose we maintain all edges of the current triangulation in a heap that has a longest edge at its root, and keeps the smaller ones further down in the heap. The refinement step picks a longest edge from the root of the edge-heap and performs the subdivision on the two incident triangles, taking into account the modified longest side bisection rule. Note that this longest edge of the current triangulation is indeed a longest edge of both incident triangles. The edge selection and the refinement step can be performed in constant time. However, the heap update costs $O(\log N)$ time, because the resulting two new edges from the refinement step have to be inserted into the heap.

A way of refining the triangle mesh more adaptively is to choose the edge that has to be split not because of its length, but based on the largest approximation error of all edges (i.e., distance to the surface that has to be approximated). This adaptive mesh refinement is already described in [Riv93], however, we briefly discuss the *split propagation* behavior.

LEMMA 3.4. If we split the common edge e of two adjacent triangles, then the propagated refinements only split edges that are longer than e.

PROOF. If e is the longest edge in a triangle t, then no split propagation occurs at all in t, because triangle t is correctly refined according to the modified longest side bisection rule. If e is the second longest side of t, then we also have to split the longest side of t because of the modified longest side bisection rule. However, the smallest side of t does not have to be split at all (see also Fig. 5, p. 45). Let $e = \overline{AC}$ be the second longest side of $t = \Delta ABC$. Because of Equation 5 we get a subdivision into three triangles, where the smallest side of t, $\overline{BC}$, is not bisected. If e is the smallest edge, then the split obviously only propagate to longer edges of t. Therefore, the split propagates always to an edge that is longer than e. The same is obviously true if e is the smallest side of t. □

A local mesh refinement can cause splits that propagate to growing edges, and thus to larger triangles or larger angles. As soon as the split propagation arrives at a longest side of a triangle, the propagation stops there. In contrast to the restricted quadtree triangulation we cannot estimate the split propagation in general, however, it is somehow determined by the initial angles of the subdivided triangles.

Note that both discussed mesh refinement methods can locally lead to $\alpha/2$ smallest angles in the first step of the modified longest side bisection rule. Only if both descendants of the first refinement step are further refined, applying the second step of the modified longest side bisection rule, do we get at least $2\alpha/3$ angles.

Progressive meshing can efficiently be achieved by a sequence of refinement events. In contrast to [Hop96], in our case these update events are edge bisections. Each update of splitting two adjacent triangles into four can be performed in $O(1)$ time, and affects the triangulation only locally. Furthermore, mesh morphing can easily be incorporated: the new vertex v' of a bisected edge $e = (v_1,v_2)$ is linearly interpolated to $v' = ½(v_1 + v_2)$ and then smoothly morphed to its final position v using a blending function $f(s)$ that is monotonically increasing for $s = [0, 1]$. Therefore, the current intermediate vertex position using the blending function is: $v_{\text{current}} = f(s)v + (1 - f(s))v'$.

For a given triangulation hierarchy the split propagation can be encoded by dependency relations similar to the restricted quadtree triangulation. Every edge subdivision which is a longest side bisection of triangle t only depends on the opposite vertex of this edge in t. However, the subdivision of a smaller edge e in t depends on the longest side bisection of t. Therefore, each midpoint of an edge e has two dependencies pointing to the opposite vertex or to the midpoint of the longest side of the two adjacent triangles. This dependency relation can be computed during the construction of a longest side bisection triangulation hierarchy.

3.3 Texture Data

Objects and additional information on a triangulated surface are often very small and/or irregular. In both cases modeling these details can be very time and space consuming and therefore ineffective. Sometimes the details are not even available in a form appropriate for modeling. In all of these cases textural data is often used to make up the surface. Normally, textural data is stored in some image data format, for instance in a bitmap format. We call these images containing textural data *texture images* or more general *texture data*.

Texture data used in combination with a triangulated terrain model (e.g. aerial ortho-photos, remote sensing data, digital maps, etc.) can help to better recognize known terrain parts and thus reduces the problem of being lost in space (see Fig. 30, p.147). Texture data often contains information details smaller than the mean triangle size but not too small. The minimum and maximum structure size of these details are in some way related to the triangle size. In cases where the mean size of terrain triangles depends on the LOD, the minimum structure size of texture data should also depend on the LOD. Therefore, LOD constrained access influences not only the terrain model, but also texture data mapped onto the terrain surface. Texture bitmap images are equally spaced ‘height’ fields and because of their relationship to DEMs, LOD concept variants can be used.

Representing texture data as succinctly as possible is a major goal to save storage costs and also processing time. Compression is the process of removing hidden redundancy from the given input data [Sal98]. Digital image compression [NH95] is a specialized field of data compression, dealing with redundancy in 2D images. To handle the vast amount of image data appearing in multimedia applications, medical imaging, and remote sensing several different compression techniques have been developed over the last decade. Besides condensing the exact information content concisely as in data compression (*lossless compression*, [MS95]), image compression also uses elimination of data with low information content (*lossy compression*). Lossy compression methods often use transformation techniques of images into the frequency domain, subsequent quantization of coefficients, and elimination of small coefficients.

Modern graphics hardware in personal computer systems is able to handle large (tens of MByte) texture data efficiently. Whenever texture data is mapped on a surface an appropriate texture filter (e.g. linear, mid-mapped, etc.) is used to match the different texture image and surface sizes. The usage of texture filters can lead to information loss, so a lossless texture compression method is not necessary. Additionally, lossy compression methods are more efficient in terms of compression ratio than lossless methods. Therefore, we primarily consider lossy compression methods in this section.

3.3.1 Progressive Image Refinement

In our LOD constrained texture access we are particularly interested in a lossy compressed image format that can efficiently extract an ε-approximation for any given error tolerance $\varepsilon \geq \varepsilon_0$, with ε_0 equal to the initial compression error. Furthermore, it has to be able to efficiently perform refinement steps. In an efficient refinement step the actual image data is considered and only data to improve the image quality is extracted. Hence, an efficient refinement step is significantly faster than the single extraction of the aimed image quality. Progressive improvements in texture image quality can then be achieved by a sequence of refinement steps. A compressed image format providing progressive refinement is effective if its storage cost does not introduce a serious overhead with respect to a simple compressed image format at the maximum precision, and if its extraction and refinement steps are kept efficient.

In the context of progressive image refinement we should distinguish between two kinds of progressive refinement:

- *with fixed resolution:* The size (in width and height) of the reconstructed image keeps fixed on all progression levels. A reconstruction of a next higher level corresponds to a successive refinement of the pixel values.
- *with scalable resolution:* The size (in width and height) and therefore the quality of the reconstructed image depends on the progression level. A reconstruction of a

next higher level corresponds to an increase of image size and quality. This approach leads to an upside-down pyramid of images. The smallest picture at the bottom of the pyramid is used as the initial image of the progressive refinement process and may be also used directly as a ‘thumbnail’ of the image.

Progressive image refinement methods are not only helpful in LOD constraint image access. They can be used whenever image data has to be transferred through a channel with a low bandwidth relative to the amount of image data, e.g. transferring large images through the internet. The data transferred in one refinement step needs to be as small as possible, so progressive image refinement methods are often coupled with appropriate (lossy or lossless) compression methods. Several different approaches exist, while most of them use transformation techniques, e.g. *Discrete Cosine Transform* (DCT), or *Discrete Wavelet Transform* (DWT). Because of its inner hierarchical structure the wavelet transform is particularly useful for progressive refinement methods. A short introduction to wavelets is found in [Gra95] and deeper insight in [FCD+95].

For progressive image refinement methods with compression there is a trade-off between compression ratio, image quality, and compression/decompression time. For the same image quality methods with a lesser compression ratio tend to be faster. Most of the approaches try to achieve a maximum image quality for a given compression ratio. We call these approaches *quality driven*. In contrast, in our context of progressive texture refinement we are mainly interested in approaches with short decompression time and reasonable quality. We call these approaches *speed driven*.

In the following subsection we discuss a new speed driven progressive image format with scalable resolution, called *Progressive Graphics File* (PGF) format. This image format serves very fast progressive refinement and achieves for the same compression ratios comparable image quality to the most popular JPEG (Joint Photographic Experts Group, [PM92]). The current version of JPEG and MPEG (Motion Picture Experts Group) are based on DCT, while JPEG 2000 is also based on DWT [SEA+00].

3.3.2 Progressive Graphics File (PGF)

PGF is based on a fast integer DWT. Due to the hierarchical structure of the DWT a progressive refinement process with scalable resolution can be integrated naturally. The abandonment of floating point computations results in a crucial speedup, with an often negligible loss of image quality. The construction and reconstruction process chains used in PGF are schematically illustrated in Fig. 7 on page 54.

We assume the input of the coder is a colored bitmap in *RGB* (Red, Green, Blue) format. In case of a grayscale bitmap we can omit the color transformation step. Because the three channels of a natural RGB image contain both spatial and chromatic redundancy, an appropriate channel transformation can result in reduced redundancy and therefore, in a bundling of the energy in one channel. This energy channel is usually called the *achromatic* and the two remaining the *chromatic* channels. Several popular color transformations of this type are known: e.g. *YIQ*, *YUV*, and *YCrCb*, where *Y* always denotes the achromatic channel. In the standard *YUV* color space the achromatic channel collects between eighty and ninety percent of the total energy. Another color transformation with even better energy bundling (> 99%) is in [WGZ97] described. In order of our integer approach, we used the following simple integer version of the *RGB-YUV* forward transformation

$$\begin{aligned} Y &= \left\lfloor \frac{R+2G+B}{4} \right\rfloor - 2^{n-1} \\ U &= R - G \\ V &= B - G \end{aligned} \tag{19}$$

and backward transformation

$$\begin{aligned} G &= Y - \left\lfloor \frac{U+V}{4} \right\rfloor + 2^{n-1} \\ R &= U + G \\ B &= V + G, \end{aligned} \tag{20}$$

where n denotes the number of bits in each R, G, and B channel. While Y occupies also n bits, the chromatic channels U and V need in general n+1 bits.

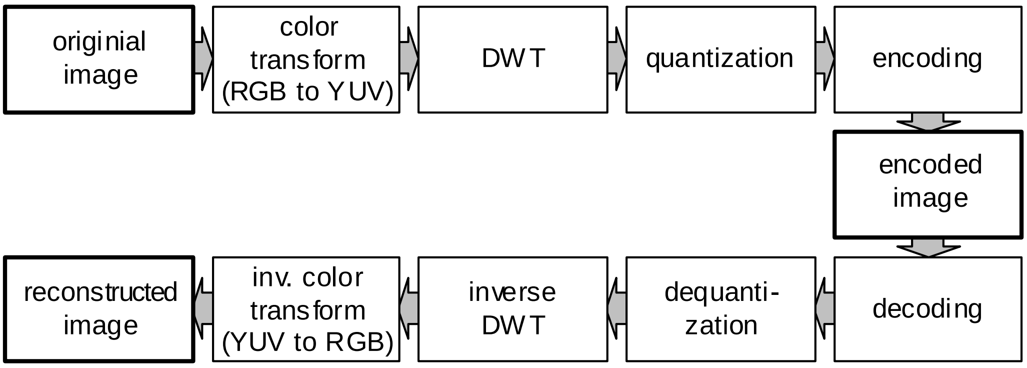


Fig. 7: Steps of a transform based coder and decoder.

The second step in our coder chain is the DWT. A DWT applied on a image produces a pyramid of wavelet transform coefficients. In each transform step it involves applying the 1D transform to the rows and columns of a bitmap. After one step, one ends up with four sub-bands: one average image *LL*, and three detail images *LH*, *HL*, and *HH*, where *L* denotes a low-pass and *H* a high-pass filter step. The next transform step does the same decomposition on *LL*. The crucial two points in the DWT are the choice of an appropriate wavelet and the correct integer implementation, which preserves the precision of the wavelet coefficients.

Compression efficiency is measured for lossless and lossy compression. For lossless coding it is simply measured by the achieved compression ratio for each one of the test images. For lossy coding the *root mean square error* (RMSE), defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - q_i)^2} \qquad (21)$$

is used, as well as the corresponding *peak signal to noise ratio* (PSNR in dB), defined as

$$PSNR = 20\log_{10}\frac{\max_i|p_i|}{RMSE}, \qquad (22)$$

where n is the number of pixels, p_i are the pixels in the original and q_i are the corresponding pixels in the compressed picture. A PSNR of 30 dB corresponds to a low quality image, while 50 dB means a visually perfect reconstruction.

In [VBL95] 4300 biorthogonal wavelet filter banks for image compression have been tested according their impulse and step response and their average PSNR over eight test images using a eight bit quantization and a 16:1 compression. For geographic imaging systems a high impulse response peak may be good, because a lot of artificial reference dots and crosses are introduced into the image in order to indicate absolute location. In case of an integer implementation, the possibility to use a small number of correct integer coefficients is also important. According to these criteria two out of the six best wavelets of this test seemed to be interesting for our PGF. The first candidate is known as the Daubechies 5/3 filter set [Dau88], where five and three denote the length of the low- and high-pass filter, respectively. The coefficients of the filters are $k(-1, 2, 6, 2, -1)$ and $k(-2, 4, -2)$, with $k = 1/(4\sqrt{2})$. The second candidate has a better impulse response but a slightly poorer step response. The filter coefficients of the second candidate are $k(4, 4)$ and $(k/2)(1, -1, -8, 8, 1, -1)$. We decided to use in a first step the Daubechies 5/3 filter set, because we also found in [CF97] a helpful correct integer implementation with precision preservation based on this filter set. The integer implementation uses the fact that the application of the factor of $\sqrt{2}$ in k can be replaced by a normalization operation at the end of the transformation if the low-pass filter is divided by $\sqrt{2}$ and the high-pass filter is multiplied by $\sqrt{2}$. This normalization operation uses only division by 2 and multiplication by 2 and is therefore very fast in a integer implementation. Even better, the normalization can be done at the same time as the quantization.

In theory, both filters are first applied to the rows of a 2D image and then to the columns of the result. In such an implementation with a large image, the application of the filters on the rows needs only the fraction of time it needs on the columns if the image is stored in rows. This is not very remarkable, because CPU caches use the principle of locality in space and time, and this locality is only given in accessing memory locations on the same row. It is important to be aware of this and to implement the filter application in a manner which maximizes space and time locality. Therefore, we first filter only *r* rows at a time, then apply the column filters on the result, and continue with the next rows, where *r* is the minimum number of rows needed to apply the column filters on the result. This concept helps to drastically increase the space and time locality and hence to reduce the transformation time.

The goal of the next step in our coder chain, the quantization of the wavelet transform coefficients, is to reduce the information needed to store the image. This is the only step that introduces information loss. One can distinguish two kinds of quantization: vector and scalar. Vector quantization is in general more powerful than scalar quantization. In case of vector quantization, one replaces a group of coefficients (a vector) with one symbol. The key is to find the right way of grouping the coefficients such that as few symbols as possible are needed. For more details on vector quantization in combination with wavelets we refer to [ABMD92].

In case of scalar quantization, one divides the real or in our case the integer axis in a number of non-overlapping intervals, each corresponding to a symbol s_i. Each coefficient is now replaced by the symbol s_i associated with the interval to which it belongs. The intervals and symbols are generally kept in a quantization table. An even simpler form of a scalar quantization is a uniform quantization with fixed interval length. In this form it is not even necessary to store a quantization table. To store the interval length and the range is just enough, but the missing adaptivity in the uniform scalar quantization could lead to worse image quality. Sometimes, the scalar quantization is combined with a threshold. The idea of the threshold is to get a larger interval in which all wavelet transform coefficients are set to zero and therefore, to reduce a large number of coefficients and to produce a sparse matrix. It is important to omit the threshold in the scalar quantization step of the last *LL* sub-band, because quantization errors in this sub-band lead to very poor image quality.

The quantization used in PGF is a uniform scalar quantization with a threshold. The interval length is restricted to powers of two. This makes it simple to combine the quantization with the normalization factor from the modified filter coefficients. It also increases the speed but it reduces at the same time the number of possible compression rates.

The last step in our coder chain is the encoding of the quantized wavelet transform coefficients, in a reversible way, into a bitstream. In PGF we use the progressive wavelet coder (PWC) presented in [Mal99]. PWC is based on progressive image coding, in which the bitstream is embedded, that is, representations of the image at any rate up to the encoding rate can be obtained simply by keeping the bitstream prefix corresponding to a desired rate. Embedded encoding can be achieved simply by applying the well-known bit-plane encoding technique [SB66] to the scalar-quantized wavelet coefficients. The most significant bit-planes naturally contain many zeros, and therefore can be compressed without loss via entropy coders such as run-length coders.

Although such straightforward bit-plane encoding is not very effective when applied to the original image samples, it can lead to reasonable performance (sometimes even better than JPEG) when applied to quantized wavelet coefficients.

Bit-plane encoding is more efficient if we reorder the wavelet coefficients in such a way that coefficients with small absolute values tend to get clustered together. That translate into longer runs of zeros in the bit-planes, which can be encoded at lower bit rates. An efficient algorithm for achieving such clustering is the embedded zero tree coder [Sha93]. A similar technique to zero trees is used in the set partitioning in hierarchical trees (SPIHT) coder [SP96]. The SPIHT coder is very efficient in clustering zero-valued coefficients at any particular bit-plane; it attains very good compression results even without entropy encoding of the bit-plane symbols. SPIHT is one of the most efficient image compression algorithms reported to date. The PWC in PGF uses also reordering of the bit-planes, but only in a simpler, data-independent (and therefore faster) way.

In the last step of the PWC the bit-planes are encoded by an adaptive run-length/Rice (RLR) encoder [Lan83], although any efficient encoder for asymmetric binary sources would suffice. For instance, adaptive arithmetic coding (AC) can be used instead of the adaptive RLR coder. The RLR encoder is used, because of its simple implementation and its low time complexity. The RLR coder with parameter k (logarithmic length of a run of zeros, which is encoded by the single code word 0) is also known as the elementary Golomb code of order 2^k. In practice the RLR coder is very close to being an optimal variable-to-variable length coder [Fab92].

The performance of the PWC codec has been tested against the SPIHT coder in a series of 25 grayscale images from the Kodak test set. The bit rates have been computed for a PSNR of 40 dB, which leads to almost unnoticeable levels of degradation, for all images. In [Mal99] is reported that the performance of the PWC codec is about the same that of the SPIHT-B codec (with binary encoding), which is about 7% worse than that of SPIHT-A codec (with arithmetic encoding), on average. In PGF this 7% loss in compression rate performance is a small price to pay, considering the lower computational complexity of PWC compared to SPIHT-A.

3.3.3 PGF Results

To prove the quality of our progressive file format PGF we have made three test series. In all three series the algorithms have been evaluated with the first eight images from the Kodak test set (768×512)[5]. The image Lena (512×512) has only been used in the first series and the aerial ortho-photo (1024×1024) only in the two other test series. All these color images have a depth of 24 bit per pixel.

---

[5] The lossless true color Kodak test images in png format are available at: http://sqez.home.att.net/thumbs/Thumbnails.html

The first test series compares the image quality of the current lossy still image standard JPEG [6] (without progressive decoding) with our PGF. The results are listed in Table 3.1.

Image	Compression	JPEG	PGF	Compression	JPEG	PGF
Lena	1:23	32.9	33.8	1:45	31.0	31.9
Kodak 1	1:12	32.8	33.9	1:21	29.3	29.8
Kodak 2	1:24	34.5	35.3	1:51	31.4	32.4
Kodak 3	1:29	36.1	37.1	1:56	32.6	34.0
Kodak 4	1:23	34.7	35.5	1:47	31.5	32.5
Kodak 5	1:11	32.7	33.8	1:19	28.9	29.9
Kodak 6	1:16	33.7	35.1	1:29	30.0	31.2
Kodak 7	1:23	36.0	36.7	1:41	32.3	33.5
Kodak 8	1:10	33.1	34.1	1:18	29.2	30.1
Average		34.1	35.0		30.7	31.7

Table 3.1: PSNR comparison between JPEG and PGF

We see, for both smaller and higher compression ratios the average image quality of PGF is slightly better (ca. 3%) than that of JPEG. These test results are characteristic for natural color images like these in the Kodak test set, but they are not for artificial and computer generated images. The encoding and decoding times are quite similar with a small advantage for PGF during encoding and a small advantage for JPEG during decoding.

In the second and the third test series we used JPEG 2000 instead of the current JPEG standard. JPEG 2000 will be the next ISO standard for still image coding [SEA+00]. It is thought as a complement to the current JPEG standards. It should be used for low bit-rate compression and progressive transmission. JPEG 2000 has more similarities with our PGF as the current JPEG standard it has. It is based on DWT, scalar quantization, context modeling, arithmetic coding, and post-compression rate allocation. The DWT is dyadic and can be performed with either a reversible 5/3 filter, which provides for lossless coding, or a non-reversible biorthogonal 9/7 filter, which provides for higher compression but does not do lossless compression. The quantization is an embedded scalar approach with threshold and is independent for each sub-band. Each sub-band is entropy coded using context modeling and bit-plane arithmetic coding. The generated code-stream is parseable and can be resolution, quality, position or component progressive, or any combination thereof. The complexity of JPEG 2000 is more similar to SPIHT-A than to PGF.

---

[6] JPEG version 6b from the Independent JPEG Group; Internet: http://www.ijg.org/

The second and the third test series have been generated on a PC laptop with a 1 GHz Mobile Pentium^{TM} III processor, 256 KByte of cache and 384 MByte of RAM under Windows 2000. The software implementation of JPEG 2000 used for coding the images comes from the *JasPer* project.[7]

Although we have focused on fast lossy compression with reasonable quality, PGF is usable for lossless compression, too. This is because PGF uses a reversible 5/3 filter and an integer implementation with precision preservation. The results listed in Table 3.2 have been generated with the 5/3 filter version of JPEG 2000.

	compression ratios		encoding time (in s)		decoding time (in s)	
Image	JPEG 2000	PGF	JPEG 2000	PGF	JPEG 2000	PGF
Kodak 1	2.3	2.1	1.53	0.39	1.39	0.32
Kodak 2	2.6	2.4	1.49	0.38	1.36	0.28
Kodak 3	3.0	2.7	1.43	0.34	1.29	0.26
Kodak 4	2.6	2.6	1.76	0.37	1.63	0.27
Kodak 5	2.2	2.0	1.62	0.42	1.46	0.33
Kodak 6	2.5	2.3	1.50	0.38	1.37	0.29
Kodak 7	2.8	2.5	1.44	0.36	1.31	0.28
Kodak 8	2.2	2.0	1.64	0.42	1.47	0.33
Average	2.5	2.3	1.55	0.38	1.41	0.30
Aerial	2.38	2.31	5.31	1.04	4.87	0.82

Table 3.2: Lossless compression ratios of JPEG 2000 and PGF

The lossless compression quality of PGF for natural images is good, but slightly worse than JPEG 2000. On average it is eight percent worse than JPEG 2000, but four to five times faster. For our aerial image the difference is even smaller (3%). The speed gain is due to the integer implementation and the simpler encoding scheme. The SMPG JPEG-LS implementation of the University of British Columbia[8], version 2.2, shows even a better lossless compression ratio on average (16%) than JPEG 2000, but they use a different test set with a mixture of natural and artificial images [SEA+00]. In the same paper they also discuss that the decoding time of JPEG 2000 is four times slower than JPEG-LS. An even better compression ratio and speed (on average) than for

---

[7] The JasPer Project is a collaborative effort between Image Power, Inc. and the University of British Columbia. The objective of this project is to develop a software-based reference implementation of the codec specified in the JPEG-2000 Part-1 standard (i.e., ISO/IEC 15444-1). This software has also been submitted to the ISO for inclusion in the JPEG-2000 Part-5 standard (as an official reference implementation).; Internet: http://www.ece.ubc.ca/~mdadams/jasper/

[8] Internet: http://spmg.ece.ubc.ca

JPEG-LS has been reported for the libpng[9] implementation of PNG[10], version 1.0.3. These results are not totally transferable to our lossless results, because of the different characteristics of the two test sets. In the Kodak test set we used, PGF is 23% better than PNG and 21% better than JPEG-LS.

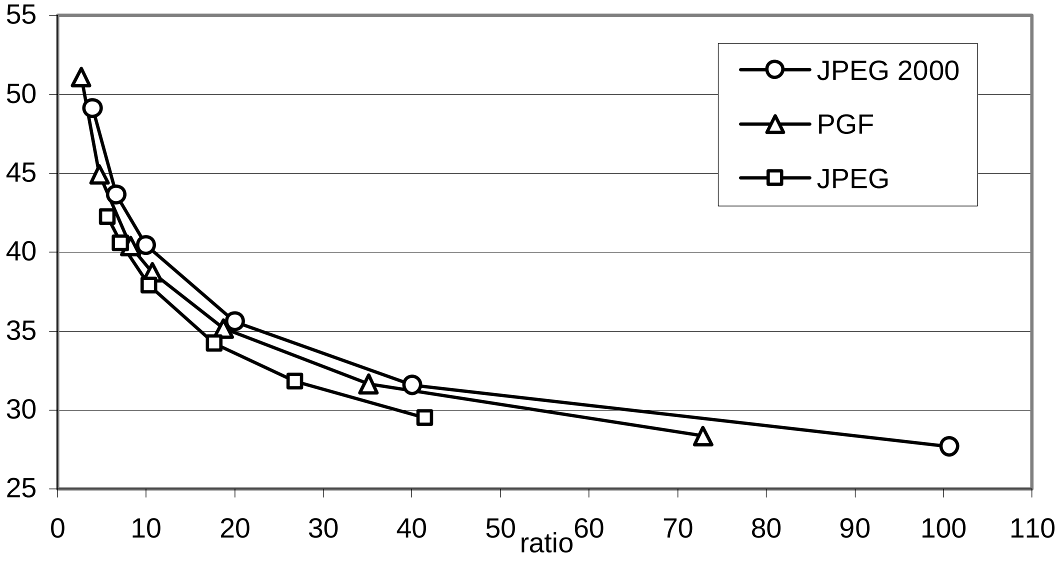


Fig. 8: PSNR of lossy compression.

PGF has been designed to progressively decode lossy compressed aerial images of 3 MByte size in less than a second on a 1 GHz PC processor. A lossless decoding of the same image on the same machine takes usually longer. In the third test series we evaluated the lossy compression qualities of PGF. One of the best opponents in this area is for sure JPEG 2000. Since JPEG 2000 has two different filters, we used the one with the better trade-off between quality and speed. On our test computer the 5/3 filter has a better trade-off than the 9/7 filter. However, the JPEG 2000 has a remarkable good quality (PSNR) for very high compression ratios but has a very poor encoding and decoding speed. Fig. 8 depicts the rate-distortion behavior when fixed (i.e., non-progressive) lossy compression is used for the averages computed over the first eight images of the Kodak test set. The PSNR of the PGF is on average 3% smaller than the PSNR of the JPEG 2000. This is quite substantial but still in the range of natural images. Additional tests have shown that these results are also valid for aerial orthophotos.

---

[9] Internet: ftp://ftp.uu.net/graphics/png

[10] Portable Network Graphics (PNG) is a W3C recommendation for coding still images which has been elaborated as a patent free replacement of GIF.

Because of the design of our implementation we already know that PGF does not reach the quality of JPEG 2000. However, we are interested in the trade-off between quality and speed. Table 3.3 shows the encoding and decoding times in relation to the PSNR of both the JPEG 2000 and PGF. In case of the PGF the encoding time is always slightly greater than the corresponding decoding time. The reason is that the actual encoding phase in the entire conversion process (cf. Fig. 7) takes slightly more time than the corresponding decoding phase. The encoding time contains the steps needed to write a PGF, including closing the file but excluding the time needed to open and read the source image. The decoding time is measured before the PGF has been opened and after the red, green, and blue pixels have been written to a memory buffer.

If we concentrate on the average compression ratios for the first eight images of the Kodak test set in Table 3.3, then we see that for six of seven different compression ratios the PSNR difference between JPEG 2000 and PGF is within 3% of the PSNR of JPEG 2000. Only in the first row is the difference larger (13 dB), but because a PSNR of 50 corresponds to an almost perfect image quality the large PSNR difference corresponds with an nearly undiscoverable visual difference. The price they pay in JPEG 2000 for the 3% more PSNR is very high. The creation of a PGF is five to twenty times faster than the creation of a corresponding JPEG 2000 file, and the decoding of the created PGF is still five to ten times faster than the decoding of the JPEG 2000 file. This gain in speed is remarkable, especially in areas where time is more important than quality, for instance in real-time computation. The test results are characteristic for both natural images and aerial ortho-photos.

		JPEG 2000 5/3			PGF		
Average	Ratio	Enc. [s]	Dec. [s]	PSNR	Enc. [s]	Dec. [s]	PSNR
1	2.7	1.86	1.35	64.07	0.36	0.28	51.10
2	4.8	1.75	1.14	47.08	0.28	0.23	44.95
3	8.2	1.68	1.02	41.98	0.23	0.19	40.39
4	10.7	1.68	0.98	39.95	0.14	0.13	38.73
5	18.7	1.61	0.92	36.05	0.12	0.11	35.18
6	35.1	1.57	0.87	32.26	0.10	0.09	31.67
7	72.9	1.54	0.85	28.86	0.08	0.08	28.37

Table 3.3: Trade-off between quality and speed for the Kodak test set

A different series of PGF images is depicted in Fig. 38 and Fig. 39 on page153. The aerial image on the left in Fig. 38 has been compressed with a compression ratio of 1:7 and stored as PGF. The four pictures on the right are stages of the progressive decoding of the same PGF file. In Fig. 39 you see the original image together with the decoding of three compressed versions. The compression ratios vary from 1:7 to 1:99.

3.4 Geomorphologic Data

Morphologic data describing the form and structure of the earth surface is called *geomorphologic* (or morphographic) data. They are a subset of geographical data and hence a particular type of GIS data. Geomorphologic data is often classified in a number of environment and/or land-usage types, e.g. urban, rural, water, forest, mountain, agriculture, etc. Roads and railways are usually not handled as geomorphologic data, because their function is not to describe the form and structure of the earth surface, but to describe a set of paths and their properties (e.g. number of lanes, quality of the road, etc.) between another set of endpoints (e.g. cities). However, sometimes we are only interested in traffic areas instead of road and railway properties. In this case roads and railways belong to the geomorphologic class of traffic areas. This example of roads and railways shows that point of view and the function of geographical data decides whether geographical data belongs to geomorphologic data or not.

In this thesis we are particularly interested in geomorphologic data, because several wave propagation prediction models (Section 4.2) take it into account to improve their accuracy. Therefore, we are faced with the problem of efficiently accessing, storing, and visualizing geomorphologic data. Geomorphologic data or GIS data in general is either stored in a raster or in a vector format. The differences between these two general data formats are introduced in the next subsection. In the other subsections we discuss the storing, retrieving, and visualizing of geographical and in particular geomorphologic data.

3.4.1 Raster and Vector Format

In the GIS community there is a long-running debate on whether it is better to represent spatial information, for instance geomorphologic data, based on a vector or on a raster model. Neither model appears to be superior in all tasks. It can be differentiated between an entity based view – space is constructed from objects that fill space – and a space oriented view, where each point in space has some properties. The vector and raster approaches seem to correspond to these two alternative views of spatial concepts. The differences between these two approaches are outlined in [DeFMP99].

In a *vector* model, spatial entities are explicitly represented through their geometry and attributes. Either collections or subdivisions may be represented, and topological relations among entities can be stored explicitly. The vector model is well suited to access a spatial database by using spatial entities as search keys. On the other hand, efficient space oriented access may require sophisticated search structures and techniques.

In a *raster* model, the domain is regularly subdivided into a large number of atomic regions, similar to the pixels in a digital image. Each pixel carries information about the portion of space it covers. Points are identified with the pixels they lie in, while lines and regions are obtained as aggregations of pixels on the basis of membership attributes. Therefore, spatial entities are not explicitly represented. The raster model provides direct access to information about a given location or extent of space. It has the disadvantage of providing an approximate geometry, whose accuracy is dependent on the resolution of the grid. On the other hand, its regular structure helps organizing spatial information.

3.4.2 Storage, Retrieval, and Visualization

An efficient geographic database providing spatial access to geographical data is an indispensable back-end for GISs with efficient large scale terrain visualization. The dynamic scene management of a VR-GIS relays on high-performance spatial retrieval of elevation and other geographical data. Moreover, a multiresolution approach calls for spatial access enhanced by a LOD specification. The vast amount of geographical data has to be maintained in external memory, where spatial data structures for points and extended geometric objects [GG97], [NW97], [Sam89] can provide efficient access through spatial indexing and clustering. In case of regular height fields or other geographical data in raster format, the regular structure suggests the use of an equally regular data structure such as the gridfile [NHS84], quadtree [Sam84] or multidimensional hashing [HSW88].

Assuming a polyhedral (multiresolution) terrain model with sufficient resolution, we may store geomorphologic data directly as terrain attributes. As long as terrain attributes are only adherent to terrain vertices, edges, and/or faces they can be stored together with the terrain. This also makes it simple to visualize the geomorphologic data. The color and/or material of terrain vertices and faces can be directly used to reflect the different geomorphologic classes. In cases where terrain faces are much larger than geomorphologic areas or where geomorphologic line structures do not follow terrain edges we need additional internal and external spatial data structures to store, retrieve, and visualize geomorphologic features.

The geomorphologic features may be visualized either by triangle colors, vectorial or texture data. Using triangle colors to visualize geomorphologic features is only appropriate if the features are only adherent to or much smaller than terrain faces. Such a visualization is depicted in Fig. 29 on page 147. As already mentioned in the introduction of this chapter, texture data in general uses more storage space than vectorial data, but directly mapped on the faces of a triangulation it is often better supported by the graphics hardware than vectorial data. Vectorial data directly lying on filled triangles may result in flickering because of the Z-buffer hidden surface removal algorithm used in graphics hardware.

3.5 Conclusions

Nowadays, global geographic information systems with virtual reality capabilities are able to visualize the earth in a large range of different scales. This range of scales passes all three scale groups of classical cartography. Hence, the different assumptions about the shape of the earth for each group can only be conserved with additional effort. A more general approach is to use the most accurate assumption, thus the spheroidal assumption. Because the classical definition of a terrain is based on a Euclidean plane as domain, we have introduced new definitions for spherical and spheroidal terrain models based on spherical and spheroidal domains, respectively.

In addition, we have presented an adaptive hierarchical multiresolution triangulation based on the longest side bisection triangulation. This triangulation has the following nice properties:

- It only produces triangles whose smallest angles are always greater or equal to $\alpha/2$, where α is the smallest angle of the initial triangle.
- All triangles produced belong to a finite number of similarity classes of triangles.
- The refinement always terminates in a finite number of steps with the construction of a matching triangulation.
- It satisfies the following smoothness condition: for any pair of adjacent triangles t_1, $t_2 \in \tau$ (τ is a matching triangulation) with diameters h_1 and h_2 respectively, it holds that $\min(h_1, h_2)/\max(h_1, h_2) \geq \delta > 0$, where δ only depends on the smallest angle of the initial triangulation.

Our modified longest side bisection refinement rule improves the lower bound of the smallest occurring angle to $2\alpha/3$. For interactive visualization of terrain surfaces it is important to avoid small angles and thin triangles because of rendering artifacts. Furthermore, we described progressive meshing based on the longest side bisection triangulation, and examine the split propagation behavior.

We have also implemented a new image file format, called PGF. PGF is based on fast integer DWT and makes use of both lossless and lossy compression. Furthermore, it uses an embedded bit stream that allows to decode the image progressively. At different stages of the progressive decoding it provides images in different resolutions. This is exactly what we need for texture mapping in our research prototype RA_3DIO. The PGF format is more speed driven than quality driven, which allows small quality degradation to increase coding and decoding speeds; the computation time for an additional PSNR of 3% is very high. The creation of a PGF is five to twenty times faster than the creation of a corresponding JPEG 2000 file, and the decoding of the created PGF is still five to ten times faster than the decoding of the JPEG 2000 file. This gain in speed is of key importance in areas where time is crucial, for instance in real-time computation. The lossless compression quality of PGF is good. On average it is eight percent worse than the compression quality of JPEG 2000, but four to five times faster. These results are characteristic for natural images and aerial ortho-photos.

„Das Schönste, was wir erleben können, ist das Geheimnisvolle. Es ist das Grundgefühl, das an der Wiege von wahrer Kunst und Wissenschaft steht. Wer es nicht kennt und sich nicht mehr wundern, nicht mehr staunen kann, der ist sozusagen tot und sein Auge erloschen.“

Albert Einstein

4 Wave Propagation Prediction

In this chapter we present some widely used wave propagation prediction methods from a point of view inspired by GIS implementation. This means we are not interested in the latest details about the wave propagation phenomenon, but more in the combination of well-known and often used propagation models with a 2½D terrain model. Questions of the following type come up: What type of data structures and algorithms are needed to compute the wave propagation between a base and mobile station efficiently? How should we implement these data structures and algorithms? Before we try to answer these questions for several propagation models, we introduce some basic concepts of the wave propagation prediction. This introduction may help to get more familiar with the following sections.

After the introduction, in Section 4.2 we review some widely accepted empirical propagation models. The Okumura and the Hata model are typical representative of this group and are often used in large-cell mobile system planning. Because of the large popularity of the Hata model it has many variations, each especially suited for different parameter ranges. All models we present in that section are implemented and part of our framework.

In Section 4.3 we investigate some fundamental concepts of terrain visibility, which are heavily used in the context of wave propagation prediction. We start our investigation with the simple line-of-sight (LOS) model and discuss different visibility concepts and a variety of visibility algorithms. We also present new results about visibility algorithms for terrain models supporting the concept of dynamic scene management.

Using the results for terrain visibility, we are able to review some physical-geometrical models in Section 4.4. Some of these models are also part of our framework. A popular and widely accepted physical-geometrical model for urban area mobile system planning is the Walfish-Ikegami model. It makes use of terrain visibility and detailed information of the environment, such as the shape and distribution of buildings. Because of the wide acceptance of the Walfish-Ikegami model in the cellular radio network planning community, we are also interested in an integration of this model into our framework. Another group of geometric models computes additional diffraction losses between transmitter and receiver on the path profile between them. They can be used to estimate path losses in terrain parts which are not visible from the antenna. A typical and widely accepted representative of this group is the multiple knife-edge diffraction model by Deygout which is also part of our framework and discussed in greater detail.

4.1 Introduction

In the analysis phase of the personal communications network design process, the wave propagation prediction of base stations plays a major role. The prediction involves several types of map and terrain data as introduced in Chapter 3, antenna characteristics, and a wave propagation prediction model. The antenna characteristics and the wave propagation prediction models are discussed in this chapter.

It is essential to know precisely what propagation models are trying to predict. The study of signal variations with movement is divided into slow or long-term variations (fading) and fast or short-term variations. Three different values of the received signal may be distinguished: first, the instantaneous value, associated with a given mobile position on the route; second, the local mean within a small area; third, the distribution of local means within a larger area. Propagation models normally try to predict the median value of the distribution of local means within a given larger area. Reported sizes for the larger area vary by authors, with values from 50×50 m^2 up to 1000×1000 m^2. Each area should have homogenous characteristics, i.e., should not include two or more land-usage types at the same time.

Instantaneous variations in the received signal are usually characterized by means of a Rayleigh or Rice distribution [Lee97], [HP99]. The most widely used model for describing the slow variations is the log-normal distribution (normal distribution in logarithmic units). The standard deviation of the log-normal distribution is known as *locations variability* and depends on the frequency, the type of environment in which the mobile station is located and the terrain irregularity. A short-term variability (Rayleigh) is superimposed on this long-term variability.

4.1.1 Antenna Characteristics

Base and mobile stations and their antennas may be described by a number of parameters: location (position on terrain), height above ground, carrier frequency, effective power, cable loss, radiation patterns, and tilt (mechanical and electrical) among others. In this subsection we are mainly interested in antenna gain and radiation patterns.

Antenna *radiation patterns* describe the antenna gain as a function of spatial angle. We assume that the gain is relative to an isotropic antenna and measured in dB. Sometimes, the unit dBi is used to indicate the isotropic reference antenna. Usually, antenna suppliers measure for each antenna type two different radiation patterns: the horizontal pattern describes the antenna gain in a horizontal plane through the antenna center, while the vertical pattern describes the gain in a vertical plane through the antenna center and a point in the horizontal pattern with a maximal gain. The horizontal angle for the vertical pattern is chosen under the assumption that the maximum gain of the antenna is at the same horizontal angle as the maximal gain of the horizontal pattern. We define this horizontal angle to be zero. Normally, both patterns are measured at discrete angular intervals, then interpolated and normalized. To *normalize*, the maximum gain is subtracted for all values (see Fig. 9).

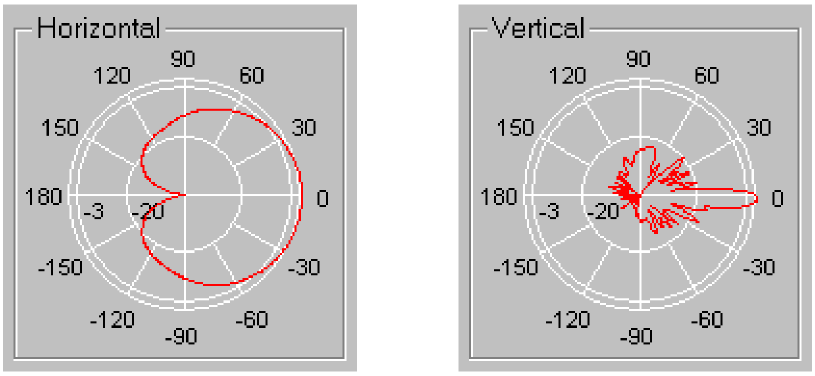


Fig. 9. Horizontal and vertical radiation patterns.

Network or RF engineers use the vertical pattern to determine the electrical tilt and the horizontal pattern to determine the directivity of the antenna. In this context the directivity denotes the horizontal angular interval covered by the antenna. In contrast to omni-directional antennas which cover 360° well, sector antennas cover only a fraction of, e.g. a third or a fourth.

In most of the practical situations these two radiation patterns provide enough information to plan base station sites. Of course, in a VR signal coverage prediction framework where we try to simulate antennas in a 3D space, the two radiation patterns provided by the antenna suppliers are only two orthogonal planes in space and thus do not contain all necessary information. We must expand the 2D antenna radiation patterns to a 3D radiation pattern. It is not clear if the two radiation patterns contain enough information to construct an applicable 3D radiation pattern, but we assume that this is the case if certain conditions are fulfilled. We reduce the problem to finding a suitable 3D radiation pattern with some nice properties, e.g. simplicity, smoothness, and/or pattern preservation.

Let S be the surface of a unit sphere with its center at the position of an antenna. Usually, S is parameterized by two angles α and β, where $\alpha \in [0, 360]$ is the longitude and $\beta \in [-90, 90]$ the latitude. The 3D radiation pattern of an antenna is then a function f: $(\alpha, \beta) \to \mathbb{R}$. A first approach of a 3D radiation pattern is constructing a body of revolution. We either use the horizontal or the vertical pattern and rotate it around the 0°–180° or 90°–-90° axis, respectively (see Fig. 9 on p. 67). In general, neither the horizontal nor the vertical pattern should be used alone to construct a suitable body of revolution. In case of the horizontal pattern, the 0°–180° axis is almost an axis of symmetry, but the intersection of the body of revolution with the vertical plane through the axis of rotation is neither equal nor similar to the vertical radiation pattern. In case of the vertical pattern, usually the 90°– -90° axis is a bad axis of symmetry, but the constructed body of revolution around this axis is at least a better approximation than the one constructed around the 0°–180° axis. However we construct a body of revolution from the 2D radiation patterns, the resulting 3D pattern is a bad approximation of the real.

Let h: $[0, 360] \to \mathbb{R}$ be the horizontal and let v: $[0, 360] \to \mathbb{R}$ be the vertical radiation pattern. The vertical pattern is measured toward the maximum antenna gain ($\alpha = 0$) and both patterns are normalized. Our first and simplest approach to get a suitable 3D radiation pattern is to use the smaller value of both h and v at each direction (α, β): $f(\alpha, \beta) = \min(h(\alpha), v(\beta))$ (see Fig. 32d, p. 149). To check if our approach is reasonable, we compare the output of f with the given input patterns. The function $\min(h(0), v(\beta))$ is equal to $\min(0, v(\beta))$ and hence equal to the right half (first and fourth quadrant) of the normalized vertical pattern $v(\beta)$. If the antenna has no down-tilt ($v(0) = 0$), then $\min(h(\alpha), v(0))$ is equal to $\min(h(\alpha), 0)$ and hence equal to the normalized horizontal pattern $h(\alpha)$. In cases where the antenna has a down-tilt, the minimum function does not exactly preserve the horizontal pattern.

Our second approach is as simple as the first. We add the two normalized pattern values: $f(\alpha, \beta) = h(\alpha) + v(\beta)$. Because h and v are measured in a logarithmic scale (decibel), the addition of the two patterns might be interpreted as a scaling of the horizontal pattern by the vertical pattern. If we check f at $\alpha = 0$, then we get $v(\beta)$ which is equal to the first and the forth quadrant of the normalized vertical pattern. If we check f at $\beta = 0$, then we get the horizontal pattern h, assuming an antenna without down-tilt. In cases where the antenna has a down-tilt, we get $f(\alpha, 0) = h(\alpha) + v(0)$, the horizontal pattern plus a constant.

Another promising approach is to use the horizontal pattern h scaled with the appropriate value of the vertical pattern v or vice versa. Thus, we get the following interpolation formula

$$f(\alpha, \beta) = \frac{(Max + h(\alpha))(Max + v(\beta))}{Max} - Max\,, \tag{23}$$

where *Max* is the maximum antenna gain (see Fig. 32f, p. 149). If we inspect f at $\alpha = 0$, then we get $v(\beta)$ which is equal to the first and the forth quadrant of the normalized vertical pattern. If we inspect f at $\beta = 0$, then we get the horizontal pattern h, assuming an antenna without down-tilt. In cases where the antenna has a down-tilt, we get $f(\alpha, 0) = c_1 h(\alpha) + c_2$, with $c_1 = 1 + v(0)/Max$ and $c_2 = v(0)$, and therefore, f does not exactly preserve the horizontal pattern.

Our fourth approach is to use on one side the right half and on the other side the left half of the vertical pattern and to interpolate between these two pattern halves. Let

$$f(\alpha, \beta) = \frac{\min(\alpha, 360 - \alpha) \cdot v(180 - \beta) + |180 - \alpha| \cdot v(\beta)}{180} \tag{24}$$

be this interpolation function. This function is visualized in Fig. 32e on page 149. Of course, this f preserves the entire vertical pattern. Therefore, this approach only seems to be promising if the interpolation $f(\alpha, \beta = 0)$ looks quite similar to the horizontal pattern, which in fact is often the case.

Obviously, the objective quality of these approaches can be only determined in comparison with measured 3D antenna radiation models. Because of missing 3D antenna characteristics, the objective quality of our and other approaches cannot be determined.

4.1.2 Free-Space Path Loss

A free-space transmission path is a straight-line path in a vacuum sufficiently removed from all objects that might absorb or reflect radio energy. It is well known that the received signal power P'_r decays with the square of the path length d and with the square of the carrier frequency f in free-space. The *free-space path loss* L'_{fs} between two hypothetical isotropic antennas is

$$L'_{fs} = \frac{P'_t}{P'_r} = \left(\frac{4\pi d}{\lambda}\right)^2, \tag{25}$$

where λ is the wave length and d the distance both in meter. Normalized for d in km and f in MHz, the free-space path loss L_{fs} expressed in terms of practical units (dB) is

$$L_{fs} = 10 \log_{10} L'_{fs} = 32.44 + 20 \log_{10} f + 20 \log_{10} d. \tag{26}$$

The presence of the earth modifies the generation and the propagation of radio waves beyond a certain distance d_0 from the antenna. The earth acts as a partial reflector and as a partial absorber, and both of these properties affect the distribution of energy. The distance d_0 is known as the Fresnel zone break point, which is proportional to frequency and antenna height [Xia93]. The LOS path-loss slope within d_0 is similar to free-space path loss, because scattering effects and multipath phenomena generally occur beyond this region, but the received field strength beyond the Fresnel zone breakpoint is ordinarily less than would be expected in free-space. We do not want to go into further detail about the diverse effects influencing the wave propagation. We only state here, among all different scattering effects, the two most important in the VHF and UHF bands are reflection and diffraction.

The relation between the received *field strength* (intensity) E'_r (in V/m) and the available *effective power* P'_r (in W) of the isotropic antenna at the receiver is given by

$$E'^2_r = 120\pi \cdot P'_r \frac{4\pi}{\lambda^2}, \tag{27}$$

where $120\pi\ \Omega$ denotes the wave resistance for vacuum. The field strength is related to the power density of the radio wave at the receiver in the absence of a receiving antenna. Therefore, the received power only depends on the *radiated power* P_t (from an isotropic antenna) and the path loss L. P_t normalized in dBm and L in dB, we can express the field strength at the receiver E_r in terms of practical units (dBμV/m) as

$$E_r = 77.2 + P_t + 20 \log_{10} f - L. \tag{28}$$

4.1.3 Link Budget Concept

To compute the local mean power P_r at the receiver, the path loss L, which is based on a wave propagation model, is used in conjunction with a link budget calculation. The *link budget B* specifies the maximum acceptable path loss between transmitter and receiver that must not be exceeded to maintain a working connection. Besides the hypothetical path loss, other physical parameters such as the radiation power, the cable loss, or the antenna gain are taken into account for the coverage decision. The downlink budget B^{down} (in dB) from a base station to a mobile station is defined as

$$B^{down} = P_{BS} + G_{BS}^{down} - C_{BS} - R_{MS} + G_{MS} - C_{MS} \tag{29}$$

and the reverse uplink budget (in dB), i.e., in the direction from a mobile station to a base station, is

$$B^{up} = P_{MS} + G_{MS} - C_{MS} - R_{BS} + G_{BS}^{up} - C_{BS} \tag{30}$$

with:

P_{BS}	the nominal transmitting power (in dBm) at base station antenna,
P_{MS}	the nominal transmitting power (in dBm) at mobile station antenna,
G_{BS}	the antenna gain (in dBi) at base station due the use of a specific antenna type and its directivity (it may be different for transmission and reception),
G_{MS}	the antenna gain (in dBi) at mobile station due the use of a specific antenna type and its directivity,

C_{BS} the signal deterioration (in dB) in cables and connectors to and from the antenna of a base station,

C_{MS} the signal deterioration (in dB) in cables and connectors to and from the antenna of a mobile station,

R_{BS} the minimum useful signal (in dBm) at base station,

R_{MS} the minimum useful signal (in dBm) at mobile station.

The asymmetric link budgets can be calculated when the minimum useful signals under varying channel conditions are known. In worst the minimum of the down- and uplink budget must be used for signal coverage planning. Thus, coverage at a given position around the base station is achieved if

$$\min(B^{down}, B^{up}) > L \tag{31}$$

and the local mean power at the receiver for a downlink is

$$P_r = R_{MS} + B^{down} - L. \tag{32}$$

4.2 Empirical Models

A number of statistical models are available in the literature [AP77], [Hat80], [CCIR82], [Lee97], [Siw98] for the prediction and the calculation of transmission loss, but the main differences between them, as well as their usefulness in a particular situation, is not easy to assess. It is important, for mobile radio path loss modeling, that the models include the various parameters relevant to the particular environment, namely urban, suburban or open areas.

Unfortunately, most propagation models have not been developed specifically for application to the mobile radio channel, but rather in a more general perspective. As a result, there is no absolutely complete model, and each one requires the insertion of one or more parameters in order to be fully applicable to the mobile radio channel.

A comparative study in [DLLC85], based on propagation loss measurements taken in the Ottawa region at 910 MHz, shows general agreement between the predictions offered by the various methods under scrutiny, provided one compares them in the conditions where they are applicable or corrects them for specific conditions – such as diffraction losses over hilly terrain or due to buildings or vegetation. Still, in [DLLC85] the authors prefer two nearly complete and easy to use models, the one by Hata [Hat80] and the other by Allsebrook and Parsons [AP77]. They conclude that these two models appear to give the most satisfactory results.

Hata's prediction model and some variations are further explained in the following subsections. All of these models take only a few parameters about the base and mobile station into account. They usually do not consider the area in between. If we use for instance a triangulation with small triangles as a terrain model, then it is enough to compute the field strength prediction for terrain vertices only. Inside of the triangles we linearly interpolate the predicted values. This approach for empirical wave propagation models results in a general time complexity of $O(v{\cdot}N)$, where v is the number of antennas and N is the number of vertices in the triangulation.

4.2.1 The Okumura Model

Okumura et al. [OOKF68] measured signal strengths in the vicinity of Tokyo, over a wide range of frequencies, several fixed-site antenna heights, several mobile antenna heights, and over various irregular terrains and environmental clutter conditions. They then generated a set of curves relating field strength versus distance for a range of fixed-site heights at several frequencies. Further, they extracted various behaviors in several environments, including the distance dependence of field strength in open and suburban areas, the frequency dependence of median field strength in urban areas, and urban versus suburban differences. The completeness of the study has made the model a standard [CCIR82] in the field, but the data are available only as curves, so they are inconvenient to use and formulas have been devised by Hata to fit the Okumura curves.

The basic formulation for propagation losses is established for the urban environment and, for the other cases, several correction factors must be introduced. These factors make it possible to associate the results of the model to the type of environment, terrain irregularity and antenna heights corresponding to the actual mobile path being studied. We do not want to go into greater detail here about the underlying curves and all their correction factors. There is enough literature about the Okumura model [OOKF68], [Lee97], [HP99]. From our GIS inspired point of view we want to explain how Okumura's model can handle several terrain features and environment types.

The method was developed to predict received field strengths without knowing in detail the actual radio path between transmitter and receiver. Instead, certain easily estimated general terrain features are used as inputs to the model:

- *Base station effective antenna height:* This parameter is defined as the difference between the antenna height above sea level and the average height of the terrain between base and mobile station. Normally only the last ten to twelve kilometers of the path between base and mobile station are taken into account (or less, if the path length is shorter). Of course, the defined average height of the terrain represents the height of the terrain in a larger area only as good as the standard deviation of the terrain height distribution is small. In highly irregular terrain, e.g. in mountain areas like the Alpes, often the base station effective antenna height is meaningless.

- *Terrain undulation:* This parameter is defined as the ‘interdecile range’ of the terrain heights taken in a ten kilometers profile segment from the position of the mobile toward the base station. For an accumulated terrain height distribution, the undulation is the height difference between 10 and 90% of all terrain heights. Therefore, the terrain undulation is a measure of terrain irregularity. Most advanced models tend not to use this parameter to evaluate terrain irregularity losses. They consider terrain effects in detail by taking into account the losses caused by each obstacle along the radio path.
- *Isolated mountain and path parameter:* This parameter is used to account for the diffraction effects caused by an isolated ridge (the only blocking ridge) in the radio path. The loss introduced by this parameter is calculated using the knife-edge diffraction model (see also Subsection 4.4.2), which has been proved to be valid for frequencies in the VHF and UHF bands. The parameter is defined as the difference between the height of the ridge and the average height of the terrain. This parameter is as good as the average height of the terrain.
- *Mean slope of the terrain:* In sections of the terrain profile where a certain slope is observed for at least 5–10 km, a slope parameter may be defined as the positive or negative angle of a positive or negative slope, respectively.
- *Mixed sea-land path parameter:* This parameter is used to quantify the effects of propagation on paths that partially traverse water spans, such as lakes and bays. Several cases may be considered, depending on the order in which the stretches of land and water paths are arranged.

In order to take the shadowing effects of the immediate surroundings of the mobile station into account (assuming the base station is clear of obstructions in its neighborhood), the geomorphology in the vicinity of the mobile station must be considered. The Okumura model clearly distinguishes between three environment types:

- *Open area:* An area is considered to be open if there are no obstacles over 300–400 m in the direction of the base station and, in general, around the position of the mobile station.
- *Suburban area:* Areas with some obstacles in the vicinity of the mobile station, although low in density.
- *Urban area:* Urban areas are cities with tall buildings and houses with more than two stories. Further, two urban area correction factors are provided for average sized towns and large, dense cities.

One necessary geomorphologic category is not included in the model: the ‘wooden area’ class. In some works performed in Germany, it has been proven that attenuations in wooded areas are similar in magnitude to those observed for urban areas.

4.2.2 The CCIR Hata Model

Hata [Hat80] prepared a simple formula representation of Okumura's measurements in the form: $L = A + B \log(d)$ where A and B are functions of carrier frequency and antenna heights and d is the distance between transmitter and receiver. The basic formula for the medium path loss in urban areas was adopted by the CCIR [CCIR82] in the form

$$L_U = 69.55 + 26.16 \log_{10} f - 13.82 \log_{10} h_b - a(h_m) + (44.9 - 6.55 \log_{10} h_b) \log_{10} d, \quad (33)$$

where f is the frequency (in MHz) in the range of 450 to 1000 MHz, d is the distance (in km) between 1 and 20 kilometers, and h_b (in m) is the base station effective antenna height in the range of 30 to 200 meters. A mobile station height correction factor $a(h_m)$ is applied for mobile antenna heights h_m (1 to 10 meters) in function of the type of urban area:

$$\begin{array}{ll} \text{small and medium city:} & a(h_m) = (1.1 \log_{10} f - 0.7)\, h_m - 1.56 \log_{10} f + 0.8; \\ \text{large city, } f \le 200 \text{ MHz:} & a(h_m) = 8.29\, (\log_{10}(1.54\, h_m))^2 - 1.1; \\ \text{large city, } f \ge 400 \text{ MHz:} & a(h_m) = 3.2\, (\log_{10}(11.75\, h_m))^2 - 4.97. \end{array} \quad (34)$$

In suburban areas, Hata gives the path loss $L_{SU} = L_U + L_{su}$, where

$$L_{su} = -2\, (\log_{10} (f/28))^2 - 5.4 \quad (35)$$

and in open or rural areas as $L_O = L_U + L_o$, where

$$L_o = -4.78\, (\log_{10} f)^2 + 18.33 \log_{10} f - 40.94. \quad (36)$$

As already mentioned, the Hata prediction model does not take into account the losses over hilly terrain, but this can be added to the model. One way to add it, is to use a statistical estimation method of diffraction losses, such as the one outlined in [DLLC85]. An alternate approach to the computation of diffraction losses over hilly terrain using the effective antenna height as a parameter has been described in [Lee97].

4.2.3 The COST231 Hata Model

The COST231-Hata model has been developed to extend the Okumura/Hata model for use in the 1500–2000 MHz frequency range where it is known that the Okumura/Hata model underestimates the path loss. This model can also distinguish between three different environmental types.

The model is expressed in terms of the carrier frequency f (in MHz), the base station antenna height h_b (in m, between 30 and 200 meters), the mobile station antenna height h_m (in m, between 1 and 10 meters), and the distance d (in km, between 1 and 20 km) between transmitter and receiver. The path loss L_U is given as $A + B \log_{10} d$ for urban areas with some correction terms for suburban areas ($L_{SU} = L_U - 15.11$) and for open areas ($L_O = L_U - 30.23$). The terms A and B are expressed as follows:

$$\mathrm{A} = 46.3 + 33.9 \log_{10} f - 13.82 \log_{10} h_b - a(h_m), \quad (37)$$

$$\mathrm{B} = 44.9 - 6.55 \log_{10} h_b, \quad (38)$$

where $a(h_m)$ depends on the city type:

small and medium cities: $a(h_m) = (1.1 \log_{10} f - 0.7)\, h_m - 1.56 \log_{10} f + 0.8;$ (39)
large cities: $a(h_m) = 3.2\, (\log_{10} (11.75\, h_m))^2 - 7.97.$

4.2.4 Modified Hata Models

Several modifications of the CCIR Hata model are known. The reason for these modifications is either to improve accuracy relative to the Okumura curves or to extend the validity range of the Hata formulas. In the last subsection we have already seen an extension of the CCIR Hata model, the COST231 Hata model, where the validity range of the carrier frequency has been shifted to higher frequencies. Another interesting modification of the CCIR Hata model is published in [Siw98]. It also extends the validity range and additionally enhances the accuracy of the Hata formulas over the entire range of validity of the Okumura curves. Furthermore, it contains corrections for the earth curvature and for the percentage of buildings on the terrain.

The model is expressed in a single formula in terms of the carrier frequency f (between 100 and 3000 MHz), the base station antenna height h_b (between 30 and 300 meters), the mobile station antenna height h_m (between 1 and 10 meters), the distance d between transmitter and receiver (in the range of 1 to 100 km), the city type C (0 = small/medium, 1 = large city), the percentage of buildings on the terrain B (between 3 and 50, nominally 15.849), and the environment type E (0 = open, ½ = suburban, 1 = urban):

$$L = L_b + a(h_m) + S_0 + S_{ks} + B_0, \quad (40)$$

where

$$L_b = 69.55 + 26.16 \log_{10} f - 13.82 \log_{10} h_b + (44.9 - 6.55 \log_{10} h_b) \log_{10} d, \quad (41)$$

$$a(h_m) = (1 - C)\, a_m(h_m) + C\, (a_2(h_m)\, F + a_4(h_m)\, (1 - F)) \quad (42)$$

with

$a_m(h_m) = (1.1 \log_{10} f - 0.7)\, h_m - 1.56 \log_{10} f + 0.8,$
$a_2(h_m) = 8.29\, (\log_{10} (1.54\, h_m))^2 - 1.1,$
$a_4(h_m) = 3.2\, (\log_{10} (11.75\, h_m))^2 - 4.97,$
$F = 300^4 / (300^4 + f^4),$

$$S_0 = (1 - E)\,((1 - 2E)\,L_o + 4E\,L_{su}) \tag{43}$$

with

$L_o = -4.78\,(\log_{10} f)^2 + 18.33\,\log_{10} f - 40.94,$

$L_{su} = -2\,(\log_{10}(f/28))^2 - 5.4,$

$$S_{ks} = \left(27 + \frac{f}{230}\right)\log_{10}\left(\frac{17(h_b + 20)}{17(h_b + 20) + d^2}\right) + 1.3 - \frac{|f - 55|}{750}, \tag{44}$$

$$B_0 = 25\,\log_{10} B - 30. \tag{45}$$

4.3 Terrain Visibility

In physical-geometrical wave propagation models, questions about the visibility of base and mobile stations arise. Several models make direct use of the knowledge about a line-of-sight (LOS) connection between transmitter and receiver or about the location of a mobile station within the radio horizon. We therefore give a short overview about different visibility concepts and discuss some algorithms and data structures that can be used to efficiently answer questions about visibility on terrains. Based on these different terrain visibility concepts a huge number of optimization problems arise. Some of them are discussed in Chapter 5. In this section we are only interested in the computation of several terrain visibility concepts, their algorithms and their data structures.

There are several surveys about terrain visibility [Nag94, DeFM99]. Terrain visibility is also discussed as an application of computational geometry in GISs, e.g. in [DeFPM99].

Usually, when we talk about terrain visibility we assume a single viewpoint V (corresponding to the antenna of a base station) on or above a terrain T and a geometric definition of visibility. The geometric definition of visibility is based on a line-of-sight connection neglecting any refraction and diffraction effects: two points on or above a terrain are said to be *mutually visible* if the line segment that joins them does not pass below the terrain surface. Sometimes it is required that the line segment is strictly above T and at most its endpoints are allowed to touch. The intervisibility of a pair of points is a Boolean function. The basic visibility structure for a terrain is the viewshed. The *viewshed* of V is the set of points of T which are mutually visible from V.

Also relevant for visibility information is the *horizon* of V, which corresponds to the ‘distal boundary’ of the viewshed (see Fig. 10, p. 22). Such reduced information can replace the viewshed in some applications, with the advantage of lower storage costs. The horizon determines, for every radial direction around V, the farthest point on T which is visible from V. The size of the horizon on a TIN with N vertices is of order

$O(N\,\alpha(N))$, where α is the extremely slowly growing functional inverse of Ackermann's function [CS89]. The *radio horizon* of V is equal to the horizon of V on a curved (spherical/spheroidal) terrain surface, considering refraction effects. Refraction effects are usually treated by exaggerating the earth radius because of larger coverage ranges. This exaggerated earth radius is called *effective (earth) radius*, $k{\cdot}R$, where R is an approximation of the earth radius and k a factor depending on the atmosphere. The value of k for a standard atmosphere is 4/3 [Lee97].

Any visibility structure can be encoded either in a continuous or in a discrete way. For the viewshed, a continuous encoding subdivides each cell of a DEM into its visible and invisible parts; this form is called a *continuous visibility map*, and it is mainly used for TINs (see Fig. 10, p. 22). The continuous visibility map of a TIN with N vertices has a worst-case space complexity in $O(N^2)$. On dense, regular grids, the viewshed is usually represented in a discrete way, by marking each cell or each vertex as visible or invisible. The resulting array of Boolean values is called a *discrete visibility map*. The discrete visibility map has an $O(N)$ worst-case space complexity for a $O(\sqrt{N}) \times O(\sqrt{N})$ *Regular Square Grid* (RSG). Sometimes, a discrete visibility map may also be considered for TINs (see Fig. 10). In [Lee91], for instance, the discretization is achieved by considering a triangle as visible if all its three edges are completely visible.

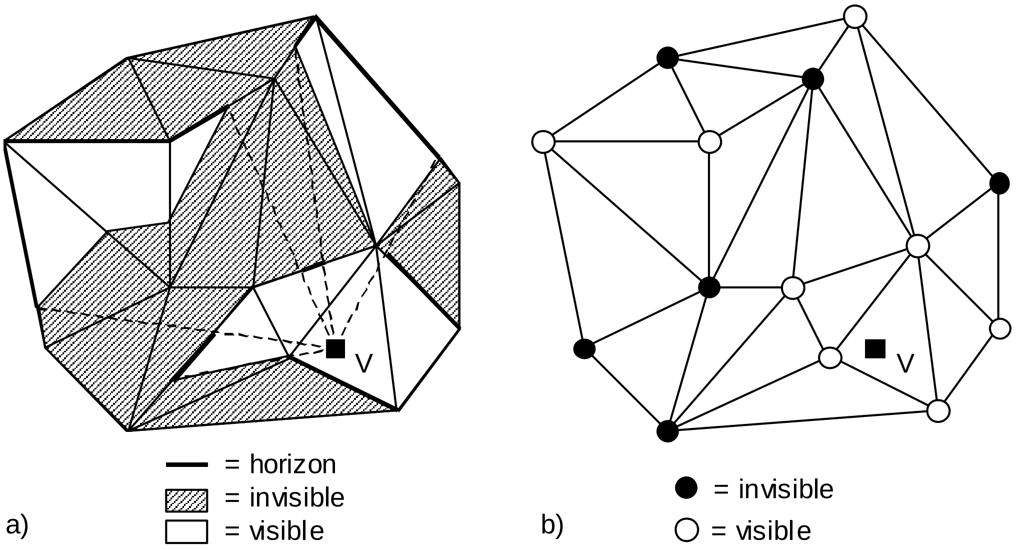


Fig. 10: a) The continuous visibility map and the horizon, b) the discrete visibility map of a TIN.

Both the single viewpoint and the geometric definition of visibility are optimistic assumptions which are sometimes invalid. Especially in the context of electromagnetic wave propagation neglecting refraction and diffraction can lead to unacceptable (visibility) results. Both refraction and diffraction effects depend on the carrier frequency of the electro-magnetic waves, so we could generalize intervisibility in terms of frequency and distance dependence. Before we define new frequency and distance dependent visibility terms, we need to introduce the first Fresnel zone. The *first Fresnel zone* is an ellipsoidal zone between transmitter and receiver that depends on the wave length and the distance between transmitter and receiver. The first Fresnel zone radius r_1 is defined as

$$r_1 = \sqrt{\frac{\lambda d_1 d_2}{d_1 + d_2}}\,, \tag{46}$$

where λ is the wave length and d_1 and d_2 are the projected distances from the endpoints to the point of interest (see Fig. 11, p. 80). If the first Fresnel zone is clear of any obstructions, then additional path losses to the free-space path loss between transmitter and receiver can be neglected.

Two points on or above a terrain T are said *radio visible* if the first Fresnel zone that joins them is clear, i.e., does not pass below the terrain surface. In case of visible light, the first Fresnel zone almost degenerates to a line-of-sight. The *radio viewshed* of V is the set of points of T which are radio visible from V. Obviously, the radio viewshed of V is a subset of the viewshed of V. Only a small number of terrain points are radio visible and usually at lot more have a sufficient field strength in the vicinity of V. Therefore, we need a more generous definition of radio visibility: two points on or above a terrain T are said *θ-radio visible* if the power loss between them is smaller than a predefined threshold θ. The *θ-radio viewshed* of V is the set of points of T which are θ-radio visible from V. Terrain points that are not mutually visible can still be θ-radio visible because of refraction and diffraction effects. In general the θ-radio viewshed of V is not a superset of the viewshed of V, because the power loss depends on distance, while the definition of the viewshed of V is distance independent. In the vicinity of V mutually visible points are also θ-radio visible. In computations and simulations the power loss is often approximated by an appropriate wave propagation model. Hence, the θ-radio viewshed is usually approximated concerning a wave propagation model. The computation of power loss for physical-geometrical wave propagation models is discussed in Section 4.4.

We may try to express θ-radio visibility in terms of the geometric definition of visibility. Because non-intervisible points can still be θ-radio visible because of diffraction effects, we could either replace each radio viewpoint by a set of LOS viewpoints with additional restrictions, or introduce another generalization of the geometric defined visibility: we call two points on or above a terrain *link-i visible* (L_i-visible) if they can be joined by a path of i or fewer line segments that do not pass below the terrain surface. L_1-visibility is the usual geometric defined visibility. A similar definition of L_i-visibility for polygons has been introduced in [She92].

If we ignore the assumption of a single viewpoint, then the intervisibility of the various types of entities is represented by the corresponding Boolean *visibility function* defined on a product space of the entities. In case of a terrain model on which three entities, surface-points, lines, and regions, can be specified, we get nine visibility functions. Among these nine, the most useful are the point-to-point, point-to-region, line-to-point, and line-to-region visibility functions. The *e*-to-point visibility functions can be seen as discrete versions of the corresponding *e*-to-region visibility functions, where *e* denotes one of the three terrain entities. Any visibility function can be represented by a *visibility graph* with arcs that link the nodes corresponding to intervisible entities. More specific data structures to store point-to-point and point-to-region visibility and algorithms to compute them are introduced in Subsections 4.3.1 and 4.3.3, respectively. Line-to-region visibility or its discrete version, line-to-point visibility, occurs, for instance in lighting computation, where the light source is not a point (e.g. a lightbulb) but a line (e.g. a long fluorescent lamp). In the context of electro-magnetic wave propagation we simply substitute the different light sources with corresponding antennas.

4.3.1 Point-to-Point Visibility

Let N be the total number of terrain data points considered. Then the point-to-point visibility among every pair of data points on or above the terrain can be represented by a Boolean array of size N^2, called the visibility matrix. The visibility matrix is symmetric. In case of equally distributed data points over the entire domain of the terrain, the visibility matrix provides useful and relatively compact information about the shape of the terrain. For instance in a bowl-shaped terrain all matrix entries are ‘true’; in a visibility matrix of a dome none are. Unfortunately, the visibility matrix gives less information than the visibility map about topography. We do not yet exactly understand, even in 1½D terrain, under what conditions a given visibility matrix corresponds to a realizable terrain. A related problem is recognizing and characterizing a visibility graph of a simple polygon [Gho97], [EC95].

A straight-forward approach to compute the discrete visibility map is based on ray shooting without preprocessing and requires $O(N^2)$ time on a TIN and $O(N\sqrt{N})$ time on a RSG with N vertices. It traces a ray from the viewpoint V to any other point P and starts walking along the ray from V to P. The walk terminates when either an intersection between the ray and a terrain edge is found before reaching P, or when P is reached. This approach is also able to handle earth curvature either by using spherical or spheroidal terrain models or by applying a distance dependent correction factor for each edge visited on the walk from V to P.

The correct radio visibility computation is based on intersection testing between terrain faces and the first Fresnel ellipsoid. This needs quite a lot of effort and should be avoided whenever possible. A somewhat simpler approximation of radio visibility, called vertical approximation of radio visibility, can be computed in a similar way as the point-to-point visibility. In the *vertical approximation of radio visibility* we only test for vertical clearance of the first Fresnel zone between the ray and a terrain face (see Fig. 11). At each edge on the walk along the ray from *V* to *P* we test for clearance not only between the ray and the edge but also between a vertical section of the Fresnel ellipsoid and the terrain face, since a flat terrain face can intersect with an ellipse even when the bounding terrain edges have enough clearance. This approach is, of course, only an approximation of radio visibility, because the horizontal extent of the first Fresnel ellipsoid is not considered. The horizontal extent of the Fresnel ellipsoid depends on the distance between *V* and *P*, as we easily can see from the maximum radius of the first Fresnel zone

$$r_{\max} = \max_{d_1} r_1 = \frac{1}{2}\sqrt{\lambda(d_1 + d_2)}\,, \tag{47}$$

where λ is the wave length and the sum $d_1 + d_2$ denotes the distance between *V* and *P*. Therefore, in each step of the walk, the number of additional faces taken into account is sublinear in the length of the walk. Usually, in practice we can claim that $r_{\max}$ is a small constant factor of the smallest extent over all faces in the terrain. Thus in practice we have to visit on each walk from *V* to a point *P* $O(N)$ triangles in a TIN or $O(\sqrt{N})$ in a RSG, respectively.

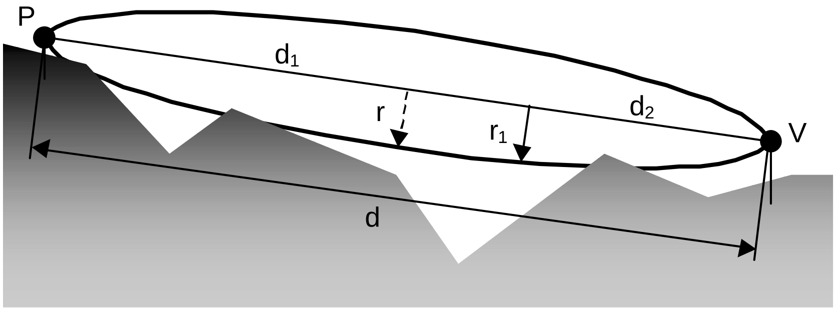


Fig. 11: Vertical approximation of radio visibility.

More sophisticated approaches for the classical point visibility problem are often based on the computation of the horizon of a viewpoint *V*. Existing horizon algorithms are of two types: divide-and-conquer or incremental. Divide-and-conquer methods run in $O(N\,\alpha(N)\log N)$ [Ata83] or with an additional segment sorting phase in $O(N\log N)$ [Her89] time. The latter is worst-case optimal since the inherent complexity of the problem is $\Theta(N\log N)$. Incremental approaches lead to a sub-optimal $O(N^2\,\alpha(N))$ time in a straight-forward implementation, while a randomized version has an expected time complexity of $O(N\,\alpha(N)\log N)$ [DeFM94]. Divide-and-conquer methods process the edges in a front-to-back order from the viewpoint *V*. Let $\pi(\cdot)$ be the parallel projection of a point, edge or ray onto the domain of the terrain. We say that an edge e_1 of a DEM is *in front* of another edge e_2 if the projection $\pi(r)$ of a ray r emanating from $\pi(V)$ intersects $\pi(e_1)$ before intersecting $\pi(e_2)$. 'In front' gives a partial order relation. A *front-to-back order* of a DEM is any total ordering consistent with the partial 'in-front' ordering (see Fig. 12). A DEM is called *acyclic* if a front-to-back order for every possible viewpoint V exists. Not all TINs are acyclic because of their irregular structure, but Delaunay triangulation have been shown to be [Ede90]. A cyclic TIN can always made acyclic by splitting some of its triangles [CS89].

In [CS89] a balanced binary tree, which stores a set of partial horizons, has been proposed. This *horizon tree* can be preprocessed for any acyclic (relative to V) polyhedral terrain and stored in $O(N\,\alpha(N)\log N)$ space. They reduce a point visibility query to a ray shooting query, i.e., to the problem of determining the first face of the terrain hit by a ray emanating from V and passing through the query point. Answering such queries can be done in $O(\log^2 N)$ time.

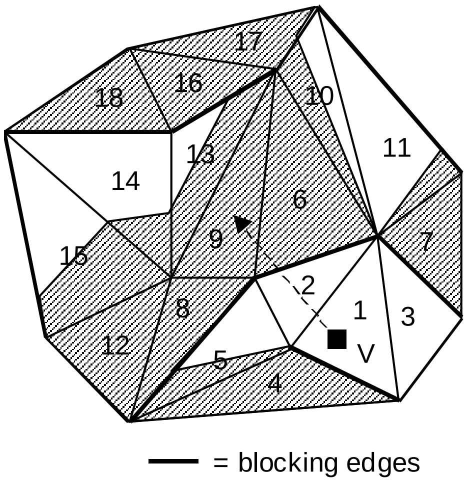


Fig. 12: Triangles of a TIN in a front-to-back order. Triangle 2 is in front of triangle 6 and both are in front of triangle 9.

If we restrict the query points to vertices of the terrain, then we can omit the additional ray shooting queries and directly compute the discrete viewshed while we build the horizon of V. Obviously, this direct approach has to take care of all edges of the terrain, but a horizon can be built with just a subset of the edges of the terrain, called blocking edges. From the perspective of a viewpoint V, a *blocking edge* represents the transition from a (partially) visible to an invisible face with respect to a front-to-back order from V (see Fig. 12). A *back-faced* face of a terrain T with respect to a viewpoint V is a face F of T with a normal n such that the angle between a ray from a point of F to V and n is larger than $\pi/2$. For example, triangle 9 in Fig. 12 could be a back-faced triangle. A *front-faced* face is a non-back-faced face. Back-faced faces are always mutually invisible, because any ray emanating from the viewpoint either hits the back of the face or does not hit the face at all. Therefore, a superset of the blocking edges can be simply built by taking all edges that represent a transition from a front-faced to a back-faced face with respect to a front-to-back order from V. It is obviously a superset of the blocking edges, because not all front-faced faces are partially visible.

4.3.2 A Divide-and-Conquer Approach for Vertex Visibility

To make clear how a horizon algorithm can be used to determine vertex visibility we present a divide-and-conquer method introduced in [Ata83] and a horizon algorithm based on this divide-and-conquer method introduced in [DeFM94]. First, the algorithm brings all terrain edges in a front-to-back order from a given viewpoint V and marks all vertices as visible. Then it recursively splits the set of edges into two halves, and merges in pairs the results to upper envelopes. An *upper envelope* of a set of segments is made of the portions of the segments visible from a given viewpoint. After the splitting phase the ordered edges are stored in the leaves of a binary tree. The edge in the leftmost leaf is the frontmost edge and the rightmost edge is the last in the order. So we know that in each merging step the left upper envelope is in front of the right. The resulting upper envelope is stored in the parent node of the two children. Merging two upper envelopes is performed through a sweep-line technique, where the only events are the vertices and the intersection points of the two upper envelopes. During the merging we mark each vertex of the right upper envelope as invisible if it is below the resulting new upper envelope. At the end of the recursion we get an upper envelope which represents the horizon and we know that every vertex has been visited at least once and maybe marked as invisible.

The merging works only with straight-line segments in a plane, so one has to transform the edges of the terrain into a plane. In [DeFM94] the authors suggest the following transformation: The edges are expressed in a spherical coordinate system centered at the viewpoint. Only the two angular coordinates λ (longitude) and φ (latitude) are considered. The Cartesian coordinates of some point (λ, φ, r) in spherical polar coordinates are given by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r\cos\varphi\cos\lambda \\ r\cos\varphi\sin\lambda \\ r\sin\varphi \end{pmatrix}. \tag{48}$$

Such a transformation produces a set of segments in the λ–φ plane. Unfortunately, this transformation does not work properly, because straight-line segments projected onto a sphere result in curved line segments in the λ–φ plane. Therefore, we cannot just connect the correct projected endpoints of the edges of the terrain with straight-line segments in the λ–φ plane. The following simple counterexample illustrates the wrong transformation. Let $s_1 = [(c, c, 0), (0, c, c)]$, $s_2 = [(½, ½, c), (0, 1, 0)]$ be two straight-line segments with $c = ½\sqrt{2}$ and let $V = (0, 0, 0)$ be the viewpoint. A transformation of the two line segments into the λ–φ plane results in $p_1 = [(d, 0), (2d, d)]$, $p_2 = [(d, d), (2d, 0)]$ respectively, where $d = \pi/4$. If we misinterpret p_1 and p_2 as straight-line segments, then their intersection point p has the two angular coordinates $\lambda = 3d/2$ and $\varphi = d/2$. In case of a correct perspective projection onto plane $y = 1$, we get the projected line segments $s_1' = [(1, 1, 0), (0, 1, 1)]$ and $s_2' = [(1, 1, \sqrt{2}), (0, 1, 0)]$, respectively. The perspective projection is computed by the following equation:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \frac{x}{y} \\ 1 \\ \frac{z}{y} \end{pmatrix}. \tag{49}$$

The intersection point p' of s_1' and s_2' is equal to $(1/(1+\sqrt{2}), 1, \sqrt{2} / (1+\sqrt{2}))$. Now, the perspective projection of p should result in p', but results in $(\cot p_\lambda, 1, \tan p_\varphi / \sin p_\lambda) = (1/(1+\sqrt{2}), 1, 0.448)$ which is not equal to p'.

Instead of the transformation suggested in [DeFM94] we propose projecting each pair of terrain edges onto one of the following four projection planes: $x = V \pm b$, $y = V \pm b$, where V is the viewpoint and b is a constant depending on the terrain extents. Each point except the vertical line through V can be projected at least onto one of these four planes, so these projection planes are in most cases sufficient. It may occur that a pair of edges with a common intersection point on one of these four projection planes cannot be projected onto the same plane because of projection degeneracies. In this case both edges are cut along one of the two planes $y - V_y = \pm (x - V_x)$ and the partial segments are projected onto two adjacent projection planes, respectively. This transformation has the same time complexity as the transformation suggested in [DeFM94], therefore, the overall time complexity of the algorithm does not change.

Now, we can ask if this divide-and-conquer algorithm is also able to handle earth curvature and radio visibility. Because a spheroidal terrain or triangulation is more general than a spherical one, we use in the remainder of this subsection the term spheroidal for both types. We start with the influence of the curvature of a spheroidal terrain.

The algorithm makes use of a front-to-back order. We therefore have to show that such an order exists on a spheroidal triangulation. In the original definition of the front-to-back order everything is projected onto the domain using a parallel projection. This works well, because the domain is a plane. In case of a spheroidal terrain all points must be projected onto the spheroidal domain in direction orthogonal to the domain. This is the analogue to the parallel projection in the classical case. Let $\pi(\cdot)$ be such a perspective projection of a point, edge or ray onto the spheroidal domain of the terrain. Then we say that an edge e_1 of a spheroidal triangulation is *in front* of another edge e_2 relative to a viewpoint V if the projection $\pi(r)$ of a ray r emanating from $\pi(V)$

intersects $\pi(e_1)$ before intersecting $\pi(e_2)$ and if the two straight-line segments from V to e_1 and e_2, respectively, do not intersect with the spheroidal domain. A *spheroidal front-to-back order* of a spheroidal triangulation is any total ordering consistent with the spheroidal 'in-front' relation. Based on this spheroidal front-to-back order the edges of the spheroidal triangulation are processed, i.e., perspectively projected from V onto an arbitrary plane and merged to projections of upper envelopes. The projection and the merging is the same as in the original algorithm. Therefore, the divide-and-conquer approach can also be used for spherical and spheroidal terrains.

In the last subsection we have seen that computing radio visibility with a ray-shooting algorithm has some difficulties and we proposed computing the vertical approximation of radio visibility instead. Now, we are investigating if the divide-and-conquer approach is able to compute this approximation. There are two problems we have to solve:

1. Show that we only have to consider the already computed upper envelopes.
2. Show that during the merging of two upper envelopes intersections and endpoints are the only relevant events.

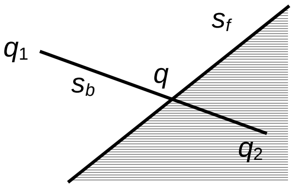


Fig. 13: Intersection of two projected terrain edges.

We start with the second problem and assume the case where we have two single edges $e_f = [p_1, p_2]$ and $e_b = [q_1, q_2]$, with the corresponding projected line segments, s_f and s_b respectively. e_f lies in front of e_b. Further, we assume that s_f and s_b intersect in point $\pi(q)$, where $\pi(q)$ is the perspective projection of q from viewpoint V. This situation is depicted in Fig. 13. Then we know that e_b is partially hidden by e_f. In case of classical visibility one part of e_b starting at q and ending at one of the endpoints of e_b, we say q_2, is hidden. In case of radio visibility this needs not be true, because e_f can still intersect the vertical section of the first Fresnel zone between V and a point $x \in [q_1, q]$ on the putative visible part of e_b.

Unfortunately, it gets worse. Now, we assume that s_f and s_b do not intersect. We know from (49) that the maximum radius of the first Fresnel zone $r_{\max}$ depends on the distance between the two endpoints of the zone. The larger the distance the larger the radius. Fig. 14 on page 85 shows the lower boundary surface of the first Fresnel zone between V and edge e_b (a) and its contour plot (b). Because of their concavity it is possible to place e_f such that both endpoints of e_b are radio invisible, but a part between them is still radio visible. Therefore, it is not allowed to reduce the sweep line events of the merging process to segment endpoints and intersection points.

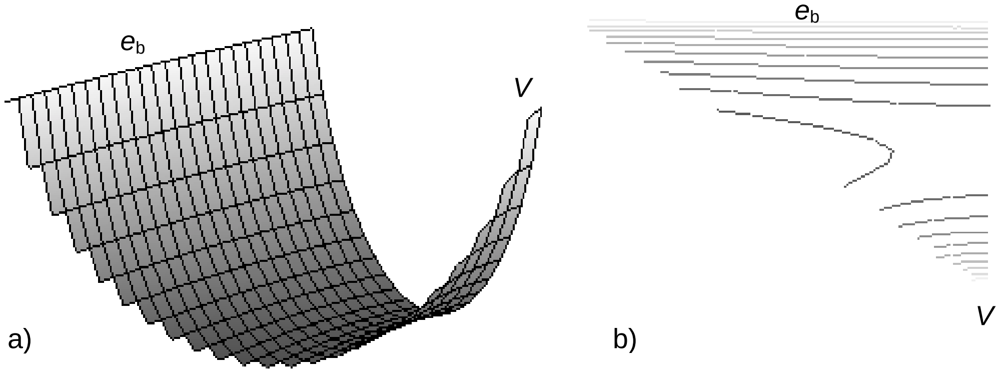
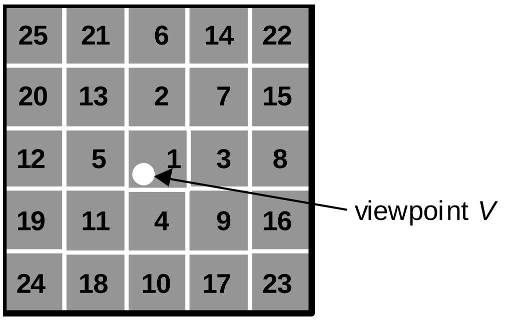


Fig. 14: (a) First Fresnel zone between viewpoint and edge; (b) Contour plot of (a).

Because we are unable to adapt the divide-and-conquer algorithm to solve the second problem, we omit discussion of the first problem. We state only here, that we can prove that it is enough to only consider the already computed upper envelopes. The problem cases above show the difficulties of combining the divide-and-conquer algorithm with radio visibility results.

In Section 2.2.4 we introduced the tiling concept for terrain data and three necessary conditions for algorithms using the tiling concept in an efficient way: locality, compatibility, and order. Now, we show the application of the divide-and-conquer approach to the tiling concept. First, we show in Fig. 15 an order of the tiles which is compatible with the front-to-back order of the triangles.



25	21	6	14	22
20	13	2	7	15
12	5	1	3	8
19	11	4	9	16
24	18	10	17	23

Fig. 15: Front-to-back order for tiles with viewpoint *V* on tile 1.

We see that any ray emanating from viewpoint *V* passes a sequence of tiles with an increasing order of tile indices. This means that the tiles are in a front-to-back order relative to *V*. Because no triangle of the terrain crosses a tile border, we find immediately a front-to-back order for all triangles of the terrain: we traverse the tiles in some front-to-back order relative to *V* and also order the triangles of each tile in some front-to-back order relative to V. If we assume that each tile contains the same number of triangles (which may only be true in the highest level of detail) and that the number of triangles in each tile is a power of two, then we see that some nodes of the binary tree built over all triangles of the terrain correspond to tiles. In this special case we can apply the divide-and-conquer approach first to the triangles of each tile and then to the resulting horizons of each tile. Let T be a terrain using the tile concept, let N be the number of triangles in T, and let n be the number of triangles in each tile. Then we can compute in $O(N\,\alpha(N)\log N/n) + N/n\cdot O(n\,\alpha(n)\log n)$ time the horizon of V and the intervisibility between V and any vertex in T. The first term represents the expense for merging the horizons of the tiles and the second term the expense for computing the local horizon of each tile. We call a horizon of a tile local if only the triangles of the tile have been used. Combining the two terms results in $O(N\,\alpha(N)\log N)$ time. It is the same complexity as for a terrain without the tiling concept. This result is still valid if n is not a power of two and if the number of triangles is equally distributed over all tiles. Therefore, computing the vertex visibility of a terrain based on the tiling concept need not result in a loss of efficiency if we compute the horizon for each tile separately (locality) and merge the tile horizons (compatibility) in a front-to-back order (order). Locality, compatibility, and order are three useful conditions for algorithms and data structures using the tiling concept presented in Subsection 2.2.4. In addition, parallelism on the tile level can easily be accomplished.

This result is more than a proof that efficiency does not suffer. It also implies a gain of efficiency during dynamic scene management. We assume a number of fixed viewpoints (antennas) on a terrain based on the tile concept. Further, we assume the matrix of tiles changes over time such that some rows or columns at the edge of the matrix are removed while others are loaded. However, some viewpoints are removed and others are loaded and inserted. It is obvious that removing of columns or rows at the edge of the matrix only influences horizons of removed tiles. The same is true for new loaded tiles, except for the viewpoints on these new tiles. For these new loaded viewpoints the local horizons of the new tiles have to be considered if the vertex visibility of all terrain vertices need to be computed. This means the visibility computation must be at least synchronized with the loading of the new tiles or started only after all new tiles have been loaded.

An efficient vertex visibility computation for each viewpoint and for each new loaded terrain vertex makes it necessary for each viewpoint to store the horizon of each tile. This needs $O(v\,N^2/n\,\alpha(N))$ space, because for each tile v horizons of length $O(N\,\alpha(N))$ are stored, where v is the number of actual viewpoints and N/n the number of tiles. Then the total visibility computation for the new vertices can be done in $O(v(N\,\alpha(N)\log k + k\,n\,\alpha(n)\log n))$ time, where k is the number of new loaded tiles. The first term represents the expense for merging $O(k)$ horizons and the second the expense for computing the local horizons of the k new loaded tiles. It is enough to merge only $O(k)$ horizons in order to compute the k new horizons, because only the horizons of the direct neighbors of a tile must be considered and each new loaded tile has not more than three direct neighbors toward a viewpoint.

In contrast, the application of a ray-shooting algorithm on a RSG takes $O(v\, k\, n\sqrt{N})$ time, because a ray is shot for each of the $k \cdot n$ new vertices and each ray is of length $O(\sqrt{N})$.

4.3.3 Point-to-Region Visibility

In the last two subsections we have presented algorithms and their implementations for vertex visibility. Unfortunately, the vertex visibility of a triangle provides hardly any information about the visibility of the interior of the triangle. Only if all three boundary edges are (in)visible, then the entire triangle is (in)visible. In TINs, where triangles may become large, we lose a lot of visibility information if we restrict the visibility computation to terrain vertices. Therefore, continuous encodings of visibility structures are more appropriate for TINs. Thus, the algorithms discussed in this subsection are designed for TINs: they exploit the fact that a TIN describes a polyhedral surface.

The problem of computing the continuous visibility map of a TIN is connected with the more general *hidden surface removal* (HSR) problem for a 3D scene. While some HSR algorithms compute the visibility information for an image plane only, by collecting the projections (images) of the visible portions from the viewpoint of each face of the scene, others compute the visible parts of the scene in the scene directly. The former group is called image-space HSR and the latter object-space HSR. This classification goes back to Sutherland in 1974. A well-known and often used image-space HSR algorithm is the Z-buffer algorithm of Catmull. In our case we are primarily interested in object-space HSR algorithms which are able to compute the continuous visibility map of a TIN. Since the divide-and-conquer version of the point-to-point visibility algorithm cannot handle radio visibility in a straight-forward way, we do not investigate continuous radio visibility maps for TINs.

[Dor94] gives a survey on object-space HSR. Some algorithms have a theoretical interest for their good asymptotic complexity, but are difficult to implement. Other algorithms have been successfully implemented and show a good practical performance, whereas they exhibit a poor worst-case complexity, or they even lack a precise theoretical analysis. Early object-space methods have a running time of $O(N^2)$ or $O(N^2\, \alpha(N))$ (N is the number of triangles in the terrain), independent of the complexity of the resulting visible portion. More sophisticated methods, like the method introduced by Katz, Overmars, and Sharir in [KOS91], have a time complexity of $O((N\, \alpha(N) + k) \log N)$ and use $O(N\, \alpha(N) \log N)$ working storage for terrains viewed from a fixed point. k is the complexity of the continuous visibility map measured in the number of edges or vertices.

Because we are interested in a concrete implementation of a point-to-region visibility algorithm for our signal coverage prediction framework we have two possibilities: either we implement such an algorithm or we look for an available implementation and integrate it into our framework. In the former case we are able to write source code which perfectly fits into our framework, while in the latter case either the implementation has to have the correct interface or our framework must provide the correct interface for the available implementation. The latter case is a problem only if we already have a fixed TIN data structure and this TIN data structure is not compatible with the available implementation of an algorithm. Unfortunately, TIN data structures and TIN algorithms of independent suppliers are rarely compatible, not even if the algorithms are based on another implementation of the same TIN data structure. That means that we have to use data structures and algorithms of the same supplier or we write our own implementations for algorithms which are not available or not compatible with the TIN data structure used in our framework. The point-to-region algorithm we present here is one of these algorithms which are not available in a compatible form. Therefore, we have to write our own implementation. That is why we discuss in the following paragraphs our implementation. Of course, these considerations are also valid for a huge number of other data structures and algorithms.

In our implementation of the object-space HSR algorithm by Katz, Overmars, and Sharir we assume that each triangle 'knows' its neighbors and that the TIN is acyclic. The neighborhood information may be stored in a undirected dual graph of the triangulation.

In a first step, we compute a front-to-back order from a viewpoint V over all triangles in the TIN. This can be done, for instance, by directing all arcs of the dual graph and applying a standard topologic sorting algorithm to the directed dual graph: each arc is directed from the triangle closer to the viewpoint toward the triangle further away. Such a topologic sorting algorithm takes $O(N)$ time and is found in good textbooks about algorithms and data structures, e.g. in [OW96]. A more TIN-specific approach, but also based on the ideas of topologic sorting, is described in [DeFM97]. They find a front-to-back order by incrementally visiting the triangles, while maintaining the property that no visited triangle is behind a non-visited triangle. This reduces to growing a star-shaped polygon around V, starting from the triangle containing V and adding one adjacent triangle at a time, until all triangles have been visited. This incremental approach also takes $O(N)$ time.

In the second step, the triangles in the TIN are partitioned into the four quadrants of a 2D Cartesian coordinate system, where the parallel projection of V lies on the origin. This is precisely the same partitioning as already described in the last subsection. Projected triangles lying in more than one quadrant are cut at the quadrant borders and the resulting polygons are computed in the corresponding quadrant. The subdivision into four quadrants is done to circumvent the problems of degeneracies usually occurring in perspective projections. It would be simpler projecting the triangles perspectively to an x–y plane, but vertices at the same height as the viewpoint would degenerate to infinity. We project all polygons of a quadrant to a vertical plane, e.g. the polygons of the first quadrant ($x \geq |y|$) are projected onto the plane $x = b$, where $b > 0$. We can simplify the third step of the algorithm if we project a collection of semi-unbounded vertical prisms, each of them consisting of all points lying below the corresponding polygon, instead of the polygons. The visibility map does not change by this transformation, but the complexity of it is reduced.

The third step is the HSR algorithm is applied on the perspectively projected polygons of each quadrant. First of all, the polygons are stored in a front-to-back order from V in the leaves of a balanced binary tree τ, the nearest polygon in the leftmost leaf. During a post-order traversal of τ we compute and store for each node δ the union U_δ of the visibility maps of its children. During a second, pre-order traversal of τ, we compute for each node the visibility map V_δ of U_δ. The visibility map of the root of τ, $root(\tau)$, is equal to $U_{root(\tau)}$. Let $left(\delta)$ and $right(\delta)$ be the left respectively right child of node δ. Then the visibility maps of them are defined as:

$$V_{\mathrm{left}(\delta)} = V_\delta \cap U_{\mathrm{left}(\delta)} \text{ and } V_{\mathrm{right}(\delta)} = V_\delta - U_{\mathrm{left}(\delta)}. \tag{50}$$

At the end of the second traversal the visible parts of the polygons are stored in V_δ of the leaves of τ. For each node δ both U_δ and V_δ are monotone polygons without holes. It is easily checked that each of the Boolean operations (union, intersection, difference) on monotone polygons without holes performed by the algorithm can be done in linear time, using standard line-sweeping methods. An efficient and correct implementation of Boolean operations for monotone polygons is quite intricate, because of the limited precision of standard number types in program languages. To correct for the limited precision problem, one could use a rational number type based on arbitrary long integers. Another way solving degeneracies in HSR is based on symbolic perturbation. The HSR implementation discussed in [Ket99] makes use of symbolic perturbation.

In the last step of this algorithm, the visibility maps of cut polygons are glued together. In theory this is just a union of monotone polygons. Now, we know all visibility maps of the triangles and we are able to transform them back to the visible parts of the triangles of the terrain. An adequate transformation back not only needs the visibility maps, but also the original triangles and their projections.

Using the tiling concept for dynamic scene management leads to the same visibility update operations as in the last subsection. The presented point-to-region visibility algorithm is also based on a front-to-back order and a divide-and-conquer strategy, so the same considerations we had for the divide-and-conquer point-to-point visibility algorithm still hold. Instead of a local horizon a local visibility map have to be stored for each tile. So, tiling does not influence efficiency negatively, but allows an efficient computation of the visibility maps of new loaded tiles.

The presented point-to-region algorithm not only works for TINs, but for acyclic polyhedral surfaces in general if we project the polygons instead of semi-unbounded vertical prisms in the second step. Unfortunately, the maps U_δ and V_δ are not monotone anymore, which results in poorer efficiency for the Boolean operations. However, this algorithm is also able to compute visibility maps of spherical and spheroidal triangulations, because spherical/spheroidal triangulations are special cases of polyhedral surfaces.

4.4 Physical-Geometrical Models

In this section we review some physical-geometrical wave propagation models which take visibility information into consideration. The visibility information is primarily used to estimate diffraction and reflection path losses and therefore, to increase the total accuracy of the path loss prediction. Physical-geometrical models require a more detailed representation of the terrain and a representation of buildings in urban areas. The required detailed description of the geometry of buildings, including the heights, makes it hard to use these models, because detailed 3D city models are not widely available and are very expensive. 3D city models are often built by stereoscopy based on aerial photos, for instance with the CyberCity Modeler[11].

Some of these physical-geometrical models are also part of our framework. A popular and widely accepted physical-geometrical model for urban area mobile system planning is the Walfish-Ikegami model. It makes use of terrain visibility and detailed information of the environment, such as the shape and distribution of buildings. Because of the wide acceptance of the Walfish-Ikegami model in the cellular radio network planning community, we are also interested in integrating this model into our framework.

In addition to the Walfish-Ikegami model, we further improve our empirical models of path losses in areas of the terrain not visible to the antenna. A separate group of geometric models performs this using the1½D profile of the terrain between the antenna and the receiver. A typical and widely accepted representative of this group is the multiple knife-edge diffraction model by Deygout which is also part of our framework and discussed in greater detail in Subsection 4.4.2.

We round-off the section with a short outlook on ray-tracing based models. Wave propagation prediction models based on ray-tracing techniques can drastically improve signal coverage prediction, because they are able to predict not only the slow but the overall (fast and slow) variations of the received signal.

4.4.1 The Walfish-Ikegami Model

The Walfish-Ikegami model is an evolution of the Ikegami model [IYTU84], [WB88], [COST91]. This model is applicable to urban areas and distinguishes between parts

---

[11] CyberCity Modeler is a trademark of CyberCity AG; Internet: www.cybercity.ethz.ch

which are visible from the base station transmitter and parts where no direct visibility of the transmitter is available due to obstruction buildings. This model, in addition to considering the influence of the street where the mobile station is located, includes the contribution to the total path loss due to the fact that the signal illuminating the street where the mobile station propagates above numerous buildings (multi-edge diffraction). The geometry of the model is shown in Fig. 16. The following street and path parameters intervene in the model:

- f: Carrier frequency (in MHz): $800 \le f \le 3000$ MHz;
- h_b: Base station antenna height (in meters) above ground: $4 \le h_b \le 50$ m;
- h_m: Mobile station antenna height (in meters) above ground: $1 \le h_m \le 3$ m;
- h_R: Mean building height (in meters, $h_R > h_m$);
- w: Width (in meters) of the street where the mobile station is located;
- b: Distance (in meters) between building centers;
- d: Distance (in kilometers) between base and mobile station: $0.02 \le d \le 5$ km;
- φ: Angle (in degrees) of the radio path with respect to the street axis;
- $\Delta h_b = h_b - h_R$;
- $\Delta h_R = h_R - h_m$.

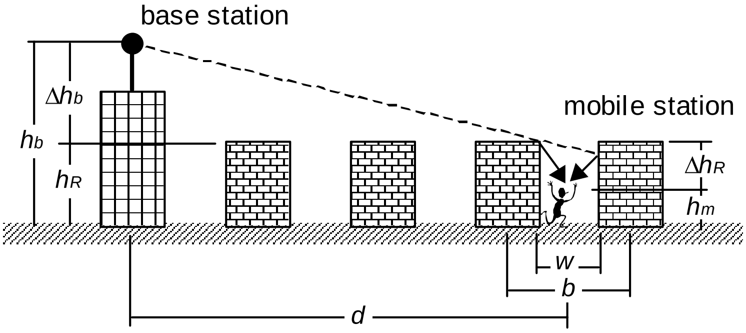


Fig. 16. Geometry of the Walfish-Ikegami model.

The Walfish-Ikegami model path loss L is expressed for LOS as

$$L = 42.6 + 20 \log_{10} f + 26 \log_{10} d \tag{51}$$

and for non-LOS as

$$L = L_{fs} + L_{rts} + L_{msd}, \tag{52}$$

where

L_{fs} is the free-space loss (see also 4.1.2);
L_{rts} is the loss due to the 'rooftop-to-street' diffraction;

L_{msd} is an estimate of the multi-obstacle diffraction effects the ray experiences between the transmitting antenna and the building closest to the receiver, due to intermediate buildings.

The ‘rooftop-to-street’ diffraction has the following expression:

$$L_{rts} = -16.9 - 10 \log_{10} w + 10 \log_{10} f + 20 \log_{10} \Delta h_R + L_{ori}. \tag{53}$$

If $L_{rts} \leq 0$ dB, a value of 0 dB is taken. L_{ori} considers the orientation of the street relative to the transmitter:

$$L_{ori} = \begin{array}{ll} -10 + 0.3574\,\varphi & (0 < \varphi < 35) \\ 2.5 + 0.075\,(\varphi - 35) & (35 \leq \varphi < 55) \\ 4 - 0.114\,(\varphi - 55) & (55 \leq \varphi \leq 90), \end{array} \tag{54}$$

where φ is the angle between the direct radio path and the axis of the street. The estimate of the *multi-obstacle diffraction* L_{msd} has the following expression:

$$L_{msd} = L_{bsh} + k_a + k_d \log_{10} d + k_f \log_{10} f - 9 \log_{10} b. \tag{55}$$

The parameters involved in this expression are calculated as follows:

$$\begin{array}{lll} L_{bsh} = & -18 \log_{10} (1 + \Delta h_b) & \text{if } h_b < 0,\ L_{bsh} = 0. \\[1ex] k_a = & \begin{cases} 54 \\ 54 - 0.8\,\Delta h_b \\ 54 - 1.6\,\Delta h_b \cdot d \end{cases} & \begin{array}{l} (\Delta h_b \geq 0) \\ (\Delta h_b < 0 \text{ and } d \geq 0.5) \\ (\Delta h_b < 0 \text{ and } d < 0.5) \end{array} \\ k_d = & 18 & (\Delta h_b \geq 0) \\ & 18 - 15\,\Delta h_b / h_R & (\Delta h_b < 0) \\ k_f = & -4 + 0.7\,(f / 925 - 1) & \text{for small/medium cities and suburban areas} \\ & -4 + 1.5\,(f / 925 - 1) & \text{for large metropolitan centers.} \end{array}$$

4.4.2 Path Profile Models

A path profile is the intersection of a vertical plane with a terrain. We also call a path profile a 1½-dimensional terrain. Path profiles are often used in wave propagation prediction to analyze *reflection* and *diffraction* effects between a transmitter and a receiver on or above the terrain. Propagation models using only path profile analysis neglect multi-path effects outside of the vertical plane between transmitter and receiver. Ray-tracing based models provide an expensive solution to trace all or the most important paths between transmitter and receiver. However, there are also some simpler and faster path profile models, which compute results reasonably well in $O(N \log N)$ time (N is the number of vertices in the terrain). These models usually classify the path profile in a number of classes (Table 4.1, p. 93) and choose the appropriate path loss model for each class.

Prior to carrying out this classification, the profile curvature must be corrected to take into account the earth curvature and *refraction* effects. We already discussed the problem of modeling the earth curvature in a more general context in Section 3.1. Normally, straight-line rays are used in propagation studies, and it is assumed that Earth, instead of having a radius of approximately $R = 6370$ km, has an effective radius $4R/3$.

a	Line-of-sight	sufficient path clearance
b		insufficient path clearance
c	Non-line-of-sight	1 obstacle
d		2 obstacles
e		3 obstacles
f		more than 3 obstacles

Table 4.1: Types of radio paths considered in the Joint Radio Committee (JRC) model

In case of a LOS path, the choice of the appropriate model depends on the sufficient path clearance. The *path clearance* is the vertical distance between the path and the terrain profile. It is the opposite of the *obstruction parameter h*, used in non-LOS paths (see Fig. 17, p. 94). The path clearance parameter denoted in h is a negative value. It shall be compared with the radius of the first Fresnel zone r_1 (Equation 46), to verify whether the direct ray is clear from the most outstanding terrain feature by at least 60% of r_1. If this happens, diffraction effects are negligible (so we are in case a).

In case b), where the path clearance is less than 60% of r_1, we look for the obstacles of influence. If there are several, regarding the one with the most influence is normally sufficient, though if more than three obstacles interfere, treating them as smooth earth may be more appropriate. To find the most one, we need a measure for the influence of obstacles on path loss. Usually, the *normalized obstruction parameter* $v = h\sqrt{2}/r_1$ is used for that. Since r_1 is a function of λ, d_1, and d_2 (see Equation 46), v is a function of λ, d_1, d_2, and h. Now, the *diffraction loss* L_d (in dB) of a single *knife-edge* obstacle is given as a function of v:

$$L_d = \begin{cases} 0 & (v < -0.8) \\ 6.02 + 9.0\,v + 1.65\,v^2 & (-0.8 \le v < 0) \\ 6.02 + 9.11\,v - 1.27\,v^2 & (0 \le v < 2.4) \\ 13 + 20\log_{10} v & (v > 2.4). \end{cases} \tag{56}$$

Equation 56 is only a good and simply computable approximation of the real function, because the original function contains Fresnel integrals. [Lee97] provides another approximation of the same function with slightly different intervals on v and therefore different partial functions. However, we see that $L_d(v)$ is a total function of v, which means that it covers the first three cases of Table 4.1.

The theory of propagation of radio waves over one single knife-edge has been investigated in theory and practice for many years. It is based on the work done by Fresnel in optics around 1815. There is a general agreement on the validity of that classical approach which expresses the diffraction loss over a sharp ridge as a function of the wave length λ, the distances d_1, d_2, and the obstruction parameter h (see Fig. 17). There are assumptions concerning the obstruction parameter

$$\lambda << h < \max\left(\frac{d_1}{10}, \frac{d_2}{10}\right) \tag{57}$$

and the shape of the obstacles (rounded hills show a different diffraction behavior and thus need some correction terms). A simple criteria suggested by Deygout to characterize a typical knife-edge obstacle path is to check the first Fresnel zones constructed around $\overline{SM}$ and $\overline{MR}$ are not obstructed (see Fig. 17).

The cases of two or more knife-edge obstacles (cases d, e, and f) were formally studied by Millington et al. [MHI62]. Because of the computation effort needed, this mathematical solution is unsuitable for quick estimates and impracticable whenever there are three or more intervening hills. For this reason, at least three different approximations have been introduced: [Bul47], [Bul77], [EP53], and [Dey66]. All three of them are well-known and often used in practice. We present here the model by J. Deygout, because of its elegant recursive definition. Deygout's model is also suggested in the textbook [Lee97], while the textbook [HP99] presents the modified JRC model [ED69], [IPD83], which is based on the two other diffraction path loss models.

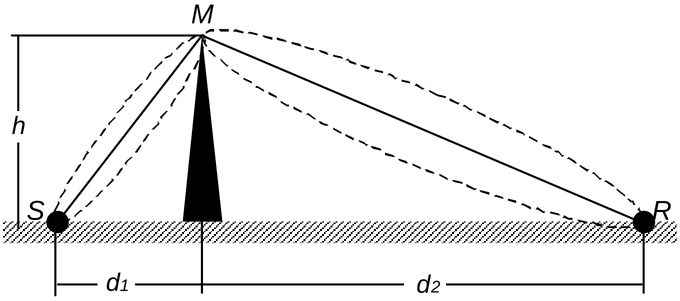


Fig. 17: Single knife-edge diffraction.

In order to better characterize the three distance arguments of the single knife-edge diffraction loss, we define a diffraction path loss function f

$$f(d_1, d_2, h) = L_d(v(d_1, d_2, h, \lambda)) \tag{58}$$

for a fixed wave length λ. L_d is the diffraction path loss function in and v is the normalized obstruction parameter.

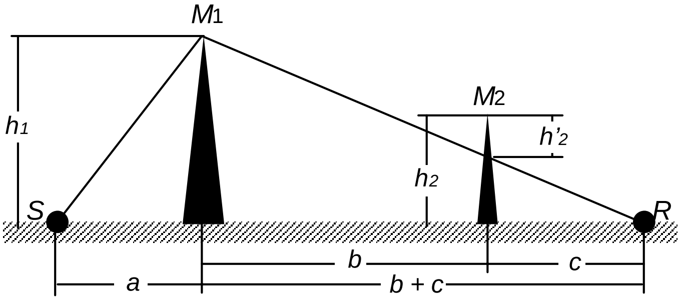


Fig. 18: Double knife-edge approximation by Deygout.

The case d) of two knife-edge obstacles is handled in Deygout's model as follows: The obstacle which has the greater value v is called *main hill* and its associated diffraction loss is computed first as if that hill were alone. The diffraction loss associated with the second obstacle is computed by considering the propagation around M_1R (see Fig. 18) with virtual sources above the main hill M_1. Assuming that $f(a, b{+}c, h_1) > f(a{+}b, c, h_2)$ the returned diffraction loss L is equal

$$L = f(a, b + c, h_1) + f(b, c, h_2'). \tag{59}$$

The shown approach for double knife-edge obstacles can easily be extended to three or more hills provided they are knife-edge obstacles as already described. The procedure is recursively defined.

W.l.o.g., let T be a 1½D terrain given by its N vertices in the u–v plane in ascending order by their u-coordinates. Further, let S and R be two given points on or above T, and let *MainHill* be the function returning the main hill M in the open interval (S, R) and the height h of M relative to the base line $\overline{SR}$. Then we define the recursive function *Loss* for a pair of given points A and B on or above T as follows:

```
Loss: $(A, B) \mapsto$
    $(M, h)$ := MainHill$(A, B)$;
    return $f(|\overline{AM}|, |\overline{MB}|, h)$ + Loss$(A, M)$ + Loss$(M, B)$.
```

The recursion stops when either *MainHill* does not find a main hill M or when a given recursion depth is reached.

The complexity of the function *Loss* depends on a concrete implementation of *MainHill*. At least, *MainHill* contains a search of a vertex M in the interval $I = (A, B)$ such that the normalized obstruction parameter v is maximized. This can be done in time linear in the number m of vertices of I. Without additional expense we can also check the assumptions concerning the height h of each main hill candidate in I (Equation 57). Unfortunately, if we want to check the suggestion by Deygout, to verify that the first Fresnel zones constructed around $\overline{AM}$ and $\overline{MB}$ are not obstructed, we need time $O(m)$ for each potential main hill M in I, and total time $O(m^2)$ to find a main hill in I. The total time complexity of the function *Loss* is

$$T(N) = X(N) + 2T\left(\frac{N}{2}\right), \tag{60}$$

where $X(N)$ is the complexity of *MainHill*. In case of linear time complexity of *MainHill* we get $T(N) = O(k \cdot N)$, where k is the maximum recursion depth. This can be easily seen, because in each recursion depth a total pass through the entire terrain is needed. k is usually set to a constant value. In case of quadratic complexity of *MainHill* we get $T(N) = O(N^2)$ if $k \in O(\log N)$ which is a realistic assumption.

Deygout compared his method with the method suggested by Epstein and Peterson [EP53] in a series of ten measurements done in France and in the U.S. Each terrain profile contained between two and five crests. Within a measurement inaccuracy of ±3 dB, the method by Epstein and Peterson appeared optimistic in eight cases out of ten, whereas Deygout's method appeared pessimistic only in one case with a maximum inaccuracy of 6 dB.

4.4.3 Ray-Tracing Based Models

To improve signal prediction even more, yet another group of geometric models can be considered. These models predict not only slow but also fast variations of the received signal, by using ray-tracing techniques with detailed urban data and high-resolution terrain models. [KCW93] gives an overview of a set of propagation models for rural and urban areas for 2D and 3D ray-tracing. Ray-tracing based models are only mentioned here to round off the section and to present an outlook for further research.

High-resolution terrain models for urban areas have resolutions better than two meters. Because of their resolution they describe not only the terrain surface but also buildings and big trees. Usually, urban data (e.g. 3D city models) is more preferred than high-resolution terrain models, because buildings are available in a vector instead of a raster format, and the vector format takes less memory. In addition, the number of vertices is drastically reduced which speeds the ray-tracing.

Ray-tracing techniques must be accompanied by electromagnetic models which allow computation of the magnitude, phase, and polarization of the different rays found in the tracing stage. A commonly used high-frequency electromagnetic technique is the *Geometrical Theory of Diffraction* (GTD), originally developed by [Kel62] with the extension to the *Uniform Theory of Diffraction* (UTD) as given in [KP74], [Mol87], [VDo94]. In [TL96] a somewhat simpler ray-tracing approach for regularly triangulated terrains is presented.

One fundamental feature of ray-tracing based propagation modeling is that it produces information in addition to field time series received along a given route (including fast and slow variations). It is also possible to obtain wideband parameter predictions, particularly power delay profiles, because when tracing rays, these may be classified by their times of arrival.

4.5 Conclusions

In this chapter we have presented several issues related to wave propagation and its integration into a VR-GIS. We have started with the problem of modeling a 3D antenna characteristic by its horizontal and vertical radiation patterns. Classical wave propagation prediction is based on 2D maps, so antenna suppliers only measure and provide 2D sections of the 3D antenna characteristic. Of course, in a 3D application the full 3D antenna characteristic is necessary. Therefore, we proposed three different approaches. Because of missing test data we could not quantize the quality of the 3D antenna models.

We have also presented a number of known empirical wave propagation models and their implementation. Most of them have been designed for flat terrains, so their accuracy is restricted to terrain parts visible from the antennas and not beyond the radio horizon. Therefore, we have introduced new definitions for radio visibility and discussed several algorithms to compute classical and radio visibility. Some of these algorithms are based on the computation of the horizon. This is helpful, because the radio horizon is related to the visible horizon of a curved surface. Furthermore, we have checked the applicability of these algorithms for spherical and spheroidal terrain models and their efficiency in combination with the tiling concept.

Empirical wave propagation models are often combined with diffraction loss models. Most of these diffraction loss models analyze the terrain profile between the transmitter and the receiver. We have presented one of them, the model by Deygout, in greater detail, and have shown a simple implementation.

„Mit dem Übergang von einer Kultur der Berechnung zu einer Kultur der Simulation verändert sich nicht nur das, was Computer für uns tun, sondern auch das, was sie mit uns tun – mit unseren Beziehungen und unseren Selbstkompetenzen.“

Sherry Turkle

5 Optimization

In this chapter we discuss two of several optimization possibilities during the four different network design phases. For instance in the radio network definition phase, where the network designer looks for the best antenna positions, network infrastructure costs are a natural optimization criteria. Because infrastructure costs mainly depend on the number of antennas, we investigate in Section 5.1 the following optimization problem: ‘Minimize the number of antennas while the entire terrain is covered by these antennas’. Of course, the coverage depends on a number of parameters, for instance antenna characteristic, wave propagation model, coverage threshold, visibility concept, and so on. Even for the simple case of geometric visibility this optimization problem is *NP*-hard [ESW99]. That is why we focus our work on approximation algorithms. Greedy heuristics are quite simple and sometimes surprisingly good. We introduce two of them and summarize their results during several tests.

Some of the other optimization possibilities depend on the network type: in cellular radio networks the co-channel interference should be minimized for a specified frequency reuse plan, while in simulcast networks the non-capture area with delay-spread should be minimized. In Section 5.2 we define a new optimization problem, called DELAY SPREAD problem, which addresses the delay-spread minimization in simulcast radio networks. The analysis of the problem leads to the conjecture that DELAY SPREAD is *NP*-hard. Therefore, we do not try to find a polynomial time algorithm to solve the problem. On the contrary, we present an algorithm based on Simulated Annealing which is able to quickly find a good approximation of an optimum solution. An additional benefit of Simulated Annealing is its incremental nature: at any time, it can be interrupted and queried for its current best solution, and then it can resume its search for a better solution. Both features make Simulated Annealing a scalable and user friendly approach.

5.1 Minimum Antenna Placement

During the last ten years the demand for radio services grew drastically. This increase went hand in hand with the growth of the number of suppliers and base stations. The new suppliers face the pressure of competition and the complexity of network infrastructure. Adequate models and tools for network design are key factors for successful competition. Various research groups have been motivated by the large number of open problems, focusing their work on automatic network design and optimization algorithms. A lot of this work has been done for the problem of automatic placement of base station transmitters (BSTs), called the BST LOCATION problem. The choice of transmitter locations depends primarily on RF design objectives (coverage, interference) and capacity engineering objectives (channels, teletraffic). This leads to a classification of the approaches for the BST location problem. The first class of approaches only considers RF design objectives, the second only capacity and teletraffic engineering objectives, while the third considers both.

An interesting approach of the first type was presented in [CJK+97]. They modeled the BST location problem as a MAXIMUM INDEPENDENT SET search problem. They used the covered area per base station as the objective function of the optimization. A similar approach, using a genetic algorithm, was investigated in [CGKW97].

In contrast to the first class, an algorithm which considers only the traffic distribution as a constraint for cell site locations was proposed in [IL97]. It is based on a computational geometry approach and constructs a tessellation of the planning region with a k-D-tree-like search algorithm.

The adaptive base station positioning algorithm presented in [FTL95] is of type three. It was one of the first methods which considered RF design objectives and the expected teletraffic at the same time. The algorithm is based on the idea of competing base stations which try to cover as much demand nodes as possible. The demand nodes describe the mobile subscriber density in a discrete way. These ideas have been extended in [Tut99]. A similar approach was presented in [FM97]. They reduced the BST LOCATION problem to a MINIMUM SET COVER problem instance and solved it with *integer programming* and *mixed integer programming*. The objective functions were the total covered area and the maximal spatial channel utilization, but they made no assumptions about the teletraffic distribution. The second objective was introduced in order to address the problem of co-channel interference.

The work we have done so far only focuses RF design objectives and is therefore of type one. A transition to type three should not be very difficult, because our independently investigated approach is based on techniques similar to those described in [Tut99]. Our BST LOCATION problem is the following: minimize the number of antennas while the entire terrain is covered by these antennas. We use a MINIMUM SET COVER instance to model the BST LOCATION problem, but our objective function is the number of transmitters for a fully covered terrain. Of course, this type of optimization is primarily of academic interest, because it neglects capacity, teletraffic, and interference issues. A valid solution of this problem requires coverage of the entire terrain, no matter how cost-effective the solution is. However, for an economic design of radio networks a trade-off between the cost of coverage and the benefit resulting from covering an area is desired. This network deployment objective leads to the definition of the transmitter location problem as a location problem that does not require the coverage of the entire terrain. In [Tut99] it has been defined as the MAXIMAL COVERAGE LOCATION (MCL) problem. The MCL problem assumes a limited budget and thereby restricts the number of base stations. The objective is to place a fixed number of base stations so that the coverage is maximized.

5.1.1 Minimum Set Cover Problem

Let $E = \{e_1, ..., e_n\}$ be a finite set (called universe) of elements. Further, let $S = \{s_1, ..., s_m\}$ be a collection of subsets of E, i.e., $s_i \subseteq E$ for $1 \leq i \leq m$, and let $c_1, ..., c_m$ be positive numbers which denote the cost of using the sets s_i. The problem of finding a subset $S' \subseteq S$ with minimal cost $\Sigma_{s_i \in S'} c_i$ such that every element of E belongs to at least one subset in S' is called MINIMUM SET COVER. For ease of discussion, let the elements in E and the subsets in S have an arbitrary, but fixed order, denoted by the index.

Let $\Gamma = \{\gamma_1, ..., \gamma_k\}$ be a finite set of base station transmitter characteristics, like location, height, power, etc., and let every $\gamma_i \in \Gamma$ be a finite set of values. We claim that every element of Γ is a finite set, so we have to discretize the feasible locations of any transmitter. A *base station transmitter configuration* b_l is a point in the k-dimensional transmitter parameter space $B = \gamma_1 \times ... \times \gamma_k$, where $1 \leq l \leq |B|$. The positive cost of a transmitter configuration b_l is denoted with $c(b_l)$. Further, let P be a finite set of terrain test points. A test point can be simply a terrain point, a terrain vertex, or even a triangle of the terrain. The *coverage* of every transmitter configuration $b_l \in B$ is a subset $P_l \subseteq P$.

The reduction from our BST LOCATION problem to set cover is the following: E is equal to the set of all test points P in the terrain. Usually, both P and the set of base station transmitter locations are a subset of the same set of terrain entities. Any transmitter configuration $b_l \in B$ can then be represented by their coverage P_l, so we set $s_l = P_l$. Hence, S represents the set of coverages of all feasible transmitter configurations B and therefore, $m = |B|$. Now, the problem is finding a subset of transmitter configurations $B' \subseteq B$ with minimal cost $\Sigma_{b_l \in B}\, c(b_l)$, which entirely covers P. Both the size of a problem instance and the time to create the instance are proportional to m and n, because for every transmitter configuration a fraction of n test points has to be checked. In real situations m is usually larger than n.

For a fixed terrain T we have two possibilities to govern the creation time of a BST location problem instance: first, we can vary n, and second, we can influence m. Of course, if we set n smaller than the number of corresponding entities in T, then we lose control over covering the entire terrain. The size m may be influenced by varying the range of each transmitter characteristic $\gamma_j \in \Gamma$. In the simplest form of our BST LOCATION problem we use unit transmitter cost, fix all base station transmitter characteristics to one value of their range, except the transmitter location, and define the coverage of each transmitter configuration as the terrain points visible from the antenna. In [ESW99] we have shown that this BST version is *NP*-hard.

Because of the popularity of MINIMUM SET COVER there are a huge variety of algorithms for it. Some of them search for an optimal solution while others try to find a good approximation. In case of an exact optimum solution the problem is formulated as a linear program and solved with integer programming [HB80], [Bea87], [JB92]. Unfortunately, this cannot be done in polynomial time in size of the set cover matrix. Therefore, a large number of approximation algorithms have been developed. Some of them are greedy, while others use randomized approaches like Simulated Annealing (e.g. [RROS96]) or genetic algorithms (e.g. [HHK94]).

5.1.2 Data Structures

A MINIMUM SET COVER instance may be stored in a Boolean matrix with m rows and $n + 1$ columns. The additional column is used to store the cost of each transmitter configuration. Such matrices can become very large (billions of entries), so we have to think about data structures which minimize the storage size and also efficiently support the most frequently used operations for solving MINIMUM SET COVER instances. For simplifying the discussion we assume that a Boolean value is zero if it is false and one if it is true.

Program languages often use the shortest integer type for a Boolean value, for instance an eight bit integer. Therefore, a matrix of type Boolean is not appropriate, because it takes eight times more memory space than needed. Bit strings based on packed integer arrays are a popular solution for this problem. By ‘packed’ we mean that every bit in a binary integer representation stores a single Boolean value. Another solution arises if we examine the structure of the matrix: the covered area of a transmitter is usually in the vicinity of, and spatially located around, the transmitter. If we order the test points along a space filling curve, then we usually get a matrix with a diagonal-like block structure. Such a matrix has on each column/row one or two long runs of zeros, so it can be compressed very well with a run length coder for runs of zeros. Although m is often larger than n, and hence compressing the columns instead of the rows is more efficient, it is usually done the other way around. The reason is that the matrix is built one row after the other.

Of course, there a more possibilities to store huge, sparse matrices, but for the moment two different data structures are enough to address some of the important issues. We investigate the efficiency of packed bit strings and run length coded matrix rows for commonly used operations in more general terms. The high-level operations depend primarily on the algorithm used to solve a MINIMUM SET COVER instance. In contrast, the low-level operations are usually the same for all algorithms. One of the low-level operations used most often is scanning a row or a column, counting, and reporting all ones. Scanning rows is in both representations easy. In case of compressed rows it is even more efficient. The rows are shorter and because we are only interested in all ones of a row a decompression is not needed. Unfortunately, scanning columns is a bit more complicate. We have to consult the data structures of each row. This means we need another low-level operation which checks the Boolean value of a row at a given position. Bit strings are more efficient when scanning columns, because of their direct access to the integer containing the searched Boolean value. In case of run length coding we have to scan until the given position, without any decompression necessary. Bit strings are more efficient for column scans and run length codes are more efficient for row scans, but run length codes are more space efficient. Therefore, one should consider transposing the matrix in the beginning if column scans are more important than row scans.

5.1.3 Approximation Algorithms

The BST LOCATION problem with a non-trivial coverage function as well as the MINIMUM SET COVER problem are *NP*-hard. This means that a polynomial time algorithm for solving them is very unlikely. If the optimal solution is unattainable, then it is reasonable to sacrifice optimality and settle for a 'good' feasible solution that can be computed efficiently. It is obvious that optimality should only be sacrificed as little as possible, while gaining as much efficiency as possible. Trading-off optimality in favor of tractability is the paradigm of approximation algorithms. The *goodness* of an approximation algorithm may be expressed in relation to an optimum solution measure over all possible instances of a problem.

Let Π be a combinatorial optimization problem, and let I be an instance of Π. Further, let $\Sigma(I)$ be the set of all feasible solutions of I, and let $\mathrm{OPT}(I)$ denote the size of an optimum solution of I, i.e., $OPT(I) = \max_{S\in\Sigma(I)} |S|$, if Π is a maximization problem, and $OPT(I) = \min_{S\in\Sigma(I)} |S|$, if Π is a minimization problem, where $|\mathrm{S}|$ denotes the cardinality of the set S.

Let A be an approximation algorithm for problem Π. By definition, an approximation algorithm runs in time polynomial in its input. The output of algorithm A on input of instance I is a feasible solution $S(A(I)) \in \Sigma(I)$. A is said to achieve an *approximation ratio* of $R_A(I)$ if for all instances I of a problem Π:

$$R_A(I) \geq \max\left(\frac{OPT(I)}{|S(A(I))|}, \frac{|S(A(I))|}{OPT(I)}\right). \tag{61}$$

Note, that the approximation ratio is thus defined to always be a number greater or equal to one, no matter whether the optimization problem is a maximization or minimization problem. We say that an optimization problem Π cannot be approximated with an approximation ratio of $R(I)$ if for every (polynomial time) approximation algorithm A for Π, there exists an instance I of Π with:

$$R(I) < R_A(I). \tag{62}$$

In this case we speak of an inapproximability result. Of course, such inapproximability results are always under an assumption such as $NP \neq P$.

MINIMUM SET COVER can be approximated with an approximation ratio that is logarithmic in the number of elements of the problem instance. The original Greedy algorithm that achieves this ratio consists of recursively adding to the solution S' a set s_i with maximum ratio $|s_i \backslash S'|/c_i$, where $s_i \backslash S'$ are the elements not yet contained in S' obtained so far [Joh74].

In case of our BST LOCATION problem we call the original Greedy algorithm *Best Transmitter First*. Another greedy approximation algorithm, called *Worst Test Point First*, is the following: recursively add to the solution S' a set s_i containing a 'worst covered' test point not yet contained in S' obtained so far. Among all possible sets s_i takes the one with a maximum ratio of not yet covered elements per cost. With 'worst covered' we mean a test point which is covered by the smallest number of transmitters.

For some special cases of our BST LOCATION problem we have presented some new inapproximability results in [ESW98], [ESW99]. In these papers we have assumed an uniform wave propagation model (geometric visibility) and uniform cost for all base stations. In case of uniform base station costs it makes no sense to provide several base station transmitter configurations at the same location, because a solver would use for every location a configuration with maximum coverage. Due to geometric mutual visibility we speak about guarding instead of wave propagation. Each antenna is equivalent to a guard on or above a terrain covering (seeing) a subset of the terrain. The set of test points in a guarding problem is equal to the set of triangles of the terrain. In the special case VERTEX GUARD ON TERRAIN (VGT) we further restrict the base station locations to terrain vertices and set the antenna height above ground to zero. In another special case, called GUARDS AT FIXED HEIGHT OVER TERRAIN (FHT), we restrict the antenna height in such a way that for all base stations the antennas are exactly on the same altitude. Despite the changes of the terrain height, all antennas are placed higher than the highest mountain and on the same height above sea level. Both problems, VGT and FHT, cannot be approximated by a polynomial time algorithm with an approximation ratio of $(1 - \varepsilon)/12 \ln N$ for any $0 < \varepsilon < 1$, unless $NP \subseteq TIME(N^{O(\log \log N)})$, where N is the number of terrain vertices. Further inapproximability results for polygons with and without holes, as well as approximation algorithms for VGT and FHT have been presented in [Eid00]. Both VGT and FHT can be approximated by a polynomial time algorithm with a ratio of $O(\log N)$.

5.1.4 Tests and Results

In the previous subsections of this chapter we have primarily presented theoretical results. Here, we present some practical measurements for our version of the BST Location problem. We compared the two greedy heuristics *Best Transmitter First* (BTF) and *Worst Test Point First* (WTF) with a third heuristic, called *Alternative Greedy* (AG) [GW97], and with a more sophisticated solver, called *Cover* (COV) [Res96]. Cover is based on a Lagrangian [Bea90] and a surrogate [LL94] heuristic. The three greedy heuristics, as well as a Simulated Annealing solver, are part of the module 'Optimization' of RA$_3$DIO (see also Chapter 6). *Cover* is a standalone Set Cover and Set Partitioning problem solver running under Unix. The optimization module of RA$_3$DIO contains a text based import and export interface for problem instances and solution vectors. We used this interface to transfer problem instances from RA$_3$DIO to *Cover* and solution vectors back to RA$_3$DIO.

All the RA$_3$DIO tests in this chapter have been done on a PC system with an Intel Pentium$^{\text{TM}}$ III microprocessor, 866 MHz, 256 KByte L2 cache, 384 MByte RAM, and Windows NT 4.0. The following three test sets have been used for our BST Location problem tests.

1st Test Set

Terrain:	70×70 km, 250 m resolution, Switzerland (Lucerne, Zurich)	
Test points:	terrain vertices: 78'400 points	
BST config.:	locations:	every fifth test point in both dimensions (= 3136)
	power:	40 dBm, 45 dBm, 50 dBm
	frequency:	900 MHz
	antenna height:	40 m
	antenna type:	isotropic
	wave model:	CCIR Hata, threshold = –80 dBm
Set Cover:	cost function:	BST power + 100
	matrix size:	9408×78'400 = 737'587'200

2nd Test Set

Terrain:	70×70 km, 250 m resolution, Switzerland (Lucerne, Zurich)	
Test points:	every fifth terrain vertex in both dimensions: 3136 points	
BST config.:	locations:	every fifth test point in both dimensions (= 3136)
	power:	40 dBm, 45 dBm, 50 dBm
	frequency:	900 MHz
	antenna height:	40 m
	antenna type:	isotropic
	wave model:	CCIR Hata within visible region, threshold = –80 dBm
Set Cover:	cost function:	BST power + 100
	matrix size:	9408×3136 = 29'503'488

3rd Test Set (equal to 2nd test set, except threshold = −100 dBm)

In Table 5.1 we summarize a subset of our time and cost measurements for the BST Location problem. In the header line of the table are the test sets and their construction time for the problem instances denoted. The greedy heuristic *Worst Test Point First* shows usually slightly better results than *Best Transmitter First*. Both are extremely fast compared to the much more sophisticated set cover solver *Cover*. Unfortunately, Cover was not able to solve our biggest test set. In the other two test sets it clearly outperforms our greedy heuristics: in test set two by 25% and in test set three by 15%. The solving times are also significant different. The greedy heuristics are more than ten times faster. The Alternative Greedy algorithm shows its best performance for small problem instances.

	1st Test Set: 23′			2nd Test Set: 1h50′			3rd Test Set: ~1h30′		
Solver	#BST	Cost	Time	#BST	Cost	Time	#BST	Cost	Time
BTF	751	111'001	20″	419	60'100	4″	125	18'100	1″
WTF	736	109'200	30″	393	56'820	4″	118	17'035	1″
AG	?	?	?	399	57'285	2′	116	16'755	10″
COV	out of memory			300	43'635	42″	100	14'525	8′

Table 5.1: BST Location problem results

5.2 Interference Minimization

In asynchronous cellular radio networks, interference may appear if antennas use the same carrier frequency and the distance between the antennas is not large enough. Interference should be minimized whenever possible, because it drastically reduces radio signal reception.

BST Location problems can be combined with interference minimization in cellular radio networks. In [GRV00] two different approaches are presented: in the first approach they try to locate a fixed number of base stations in such a way that the coverage is maximized while the interference must not exceed a given threshold. In the second approach they specify a minimal coverage and try to minimize the number of base stations. Again the interference must not exceed a given threshold, but now, the coverage must reach the given lower bound.

In simulcast or quasi-synchronous radio networks a special type of interference, called delay-spread, may reduce the captured area. We have already introduced in Subsection 2.3.6 the principle of delay-spread. In this section we present an exact definition of the DELAY SPREAD minimization problem. Furthermore, we discuss two possible solution approaches, one a greedy heuristic and the other one an algorithm based on Simulated Annealing. The latter has been implemented, used, and tested in realistic scenarios.

5.2.1 Delay-Spread Minimization

Let P be a set of test points on a terrain T, and let B be a set of base station transmitters on T. Each $b \in B$ contains the usual transmitter parameters (location, power, etc.) and in addition an initial delay $\delta(b)$. The initial delay is a sum of two delays: the delay between a signal source (e.g. a satellite) and b, and a second, controllable value. Further, let $\pi(b, p)$ be the power of transmitter b at point p, and let $\delta(b, p)$ be the total delay of $\delta(b)$ plus the signal delay between the position of b and point p. The coverage of each b is a subset of P. A $p \in P$ can be covered by several transmitters, so we define for each p the number of transmitters $n(p)$ which cover p, a maximum power $\pi_{\max}(p) = \max_{b \in B} \pi(b, p)$, and a transmitter $b_{\max}(p)$ with $\pi_{\max}(p)$ power at p. Now, we can define a subset $P' \subseteq P$ containing all points which are covered by at least two transmitters and which have a power difference between $\pi_{\max}(p)$ and any other power at p of less than a given power threshold $\Delta\pi$:

$$P' = \{p \mid \forall p \in P\text{: } n(p) > 1 \wedge \forall b \in B \backslash b_{\max}(p)\text{: } \pi_{\max}(p) - \pi(b, p) \le \Delta\pi\}. \qquad (63)$$

The set P' contains the points of the terrain which fulfill the first condition for delay-spread. Let $B_{\max}(p)$ be the set of all transmitters with maximum power $\pi_{\max}(p)$ at point p. Now, for a given delay threshold $\Delta\delta$ we can define a subset $P'' \subseteq P'$ which also fulfills the second condition of delay-spread:

$$P'' = \{p \mid \forall p \in P' \, \forall b \in B \, \exists b_{\max} \in B_{\max}(p) \, \|\delta(b_{\max}, p) - \delta(b, p)\| > \Delta\delta\}. \qquad (64)$$

The problem DELAY SPREAD consists of finding a set of initial delays $\{\delta(b) \mid \forall b \in B\}$ such that the cardinality of P'' is minimal.

We already mentioned that the radio signal delay consists of three parts: first, a delay from a radio source to a base station, second, an (artificial) delay in the base station, and third, a delay between the base station and a mobile station. The optimization algorithm modifies the second part of the delay chain in order to minimize the non-capture area. The optimization also depends on the first delay, so we need the ability to measure the distance between a base station and a point source. Normally, the point source is in line-of-sight with the base stations. As long as the location coordinates of both endpoints (point source and base station) are in the same coordinate system, measuring the distance between two points in line-of-sight is quite easy. In case different coordinate systems (based on different projections) and different reference systems are used, more complex computations are needed. Unfortunately, this scenario occurs. For example, we want to compute the distance between a stationary satellite and a point in Switzerland. Location in Switzerland are usually referenced to the Swiss coordinate system. The Swiss coordinate system is based on a conformal oblique conical projection and uses the Bessel reference system. It is primarily used in Switzerland, so we have difficulties if a point source is abroad or even in space. In case of a stationary satellite we only know the height above the earth and its longitude and latitude. Position of satellites are mainly referenced to the WGS84 reference system. This means, we have two different coordinate systems, a conical projection, and two different reference systems.

All necessary parts to measure distances between points in several different coordinate systems, projections, and reference systems have been implemented and integrated into RA$_3$DIO (see also Chapter 6).

5.2.2 Analysis of the Delay-Spread Problem

In this subsection we analyze a simplified version of the DELAY SPREAD problem. We assume that the transmitters are represented by points on a plane in the Euclidean space and their coverages are circles with the transmitters in the center. The radii of the circles depend on several transmitter parameters (propagation model, power, antenna height, etc.) and on the coverage threshold. The power loss of a transmitter along a ray emanating from the center of the circle is a monotone decreasing function inside the coverage circle. A typical simple power loss function would be $L(d) = -40 \log(d)$, where d is the distance to the transmitter point.

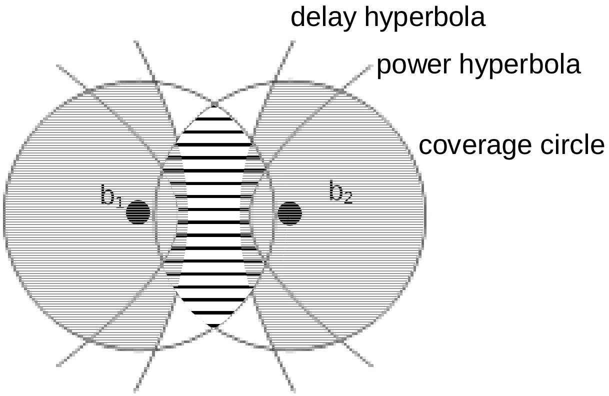


Fig. 19: Simplified delay-spread situation with two transmitters.

In the most simple case, depicted in Fig. 19, we only have two antennas with the same circle radius, the same initial delay, and the same wave propagation function. We have a totally symmetrical configuration. The coverage circle intersection is the area where delay-spread can occur. Part of it, horizontally hatched, is the area with a power difference smaller or equal a given threshold $\Delta\pi$. The geometric definition of its boundary depends on the path loss function. In case of a linear path loss function the boundary is described by two coverage circles and two power hyperbolas. The two gray shaded areas are the area where the delay difference is greater than a given threshold $\Delta\delta$. The delay is a linear function of the distance, so both areas are bounded by a hyperbola and a circle. The two delay hyperbolas are defined by all points p with $|\delta(b_1, p) - \delta(b_2, p)| = \Delta\delta$. The intersection of the gray shaded and the hatched area is the searched delay-spread area.

If we modify the initial delay of one of these two transmitters, we say b_1, then both delay hyperbolas change. The left delay hyperbola defines all positions p where $\delta(b_2, p) - \delta(b_1, p) = \Delta\delta + \delta(b_1)$ and the right delay hyperbola all positions p where $\delta(b_1, p) - \delta(b_2, p) = \Delta\delta - \delta(b_1)$. Graphically, if $\delta(b_1)$ is positive, then the left delay hyperbola moves left and becomes rounder, and the left delay hyperbola also moves left, but becomes flatter. Hence, the delay-spread area on the left decreases and the delay-spread area on the right increases. On the right, the hatched area becomes wider if we move the right delay hyperbola to the left, but on the left narrower if we move the right delay hyperbola to the left. Hence, the delay-spread area increases in total. Therefore, the symmetric initial delay is the optimum.

Now, we consider a more realistic and also more complicated situation with more transmitters. The transmitters have different coverage radii but still the same linear path loss function. The situation is shown in Fig. 20.

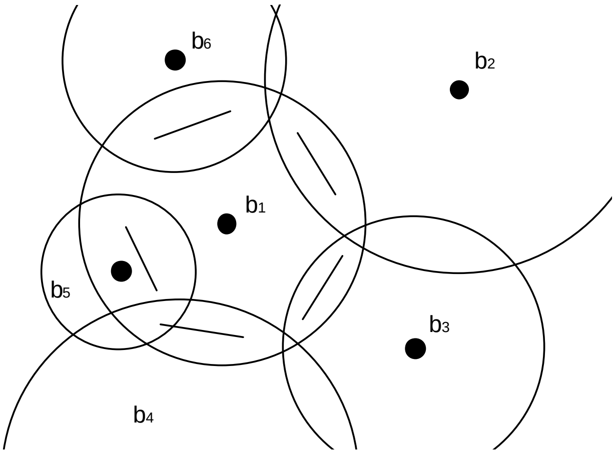


Fig. 20: Delay-Spread situation with six transmitters.

Because the circles around the transmitters are defined by a coverage threshold, we know that for each transmitter the power on the circle boundary is identical to this threshold. We also know that the intersections of the coverage circles are the areas with potential delay-spread. Let I_{ij} be the intersection of the two coverage circles of transmitter b_i and b_j. Under the assumption of an identical linear path loss function for all transmitters we know that the power is equal for each pair of transmitters b_i and b_j on the direct line between b_i and b_j in the middle of I_{ij}. For transmitter b_i, the straight-line segments in Fig. 20 pass these points perpendicular to the direct lines between b_i and the other transmitters. Each straight-line segment approximates a region of almost equal power between b_i and some other transmitters, and delay-spread will only occur in these regions. It occurs if the delay difference in these regions is large. A large delay difference is equivalent to a large distance difference. The situation between transmitter b_1 and b_5 is a typical situation for delay-spread, because the transmitters are at very different distances to the straight-line between them. If we set an initial positive delay for transmitter b_5, then the delay difference moves to the left in direction of the straight-line. In a situation like the one shown in Fig. 20 we can simply move all delay differences by modifying the values $\delta(b_i)$, $2 \leq i \leq 6$, toward the straight-line segments and thus prevent delay-spread. Of course, this is only a local optimization and needs not yield to a global improvement if there a more transmitters around the six depicted transmitters, or even between other pairs such as b_2 and b_5.

Unfortunately, in a global optimization it is not clear which transmitter need to be modified and by what value. Such situations are preferred for using Simulated Annealing more than greedy heuristics. An approach based on Simulated Annealing is presented in the next subsection. A greedy heuristic could be the following: in preprocessing, we compute the set P' of points with a power difference smaller or equal to $\Delta\pi$, and for each base station b the subset of $P'(b) \subseteq P'$ covered by b. At the same time we construct a graph $G = (B, E)$, where B is the set of base station transmitters and $E = \{(b_i, b_j) \mid \exists\, p \in P' \colon p \in P'(b_i) \wedge p \in P'(b_j)\}$. This takes $O(N + vN')$ time, where v is the number of transmitters, N the number of terrain points and N' the cardinality of P'. We call two transmitters b_i, b_j *neighbors* if there is an edge $(b_i, b_j) \in E$. Let $n(b)$ be the cardinality of $P'(b)$ for each $b \in B$. Now, we iterate through all transmitters in decreasing order of $n(b)$ and optimize as follows. For transmitter b we monotonously increase or decrease $\delta(b)$ in discrete steps as long as $n(b)$ decreases and no $n(b_i)$ of an already processed neighbor b_i increases. Now, b is marked as processed and we pick a new transmitter b with a largest $n(b)$ of all unmarked transmitters. The number of unmarked transmitters decreases in each step, while $n(b)$ does not, so the algorithm stops with a number of points with delay-spread not larger than in the beginning. In each iteration we change the delay $\delta(b)$ c times, where c is an arbitrary constant bounded by a maximum $|\delta(b)|$. After each change we consult the points in $P'(b)$ and compute the delay-spread at each point. The computation of delay-spread takes $O(v^2)$ time, because there are several possible transmitters with maximum power. Picking the next transmitter needs only $O(v)$ time. Therefore, in each step we need $O(c \cdot v^2 N')$ time for $O(c \cdot v^3 N')$ total. That means this algorithm runs in time polynomial in its input size.

5.2.3 Simulated Annealing

A competing heuristic for large scale combinatorial optimization problems is Simulated Annealing. In contrast to greedy algorithms, which often require experience and intuition to give good performance for complex problems, Simulated Annealing only requires an understanding of what the form of solutions are, rather than deeper understanding of the entire solution space structure. Simulated Annealing belongs to the class of 'local search' algorithms. It is a randomized approach and it can be asymptotically viewed as an optimization algorithm. We only outline the major components of Simulated Annealing. A detailed presentation of the theory and applications can be found in [RROS96].

Algorithm Simulated Annealing

```
1   $t := t_{start}$;
2   $z$ := 'number of points on terrain with delay-spread';
3   $z_{min} := z$;
4   while $t \geq t_{end}$ loop
5       for $i$ in 0..$i_{max}$ loop
6           $b$ := 'chose randomly a transmitter';
7           $\delta_{old} := \delta(b)$;
8           if $Random[0,1] = 0$ then $\delta_{new} := \delta_{old} + 1$; else $\delta_{new} := \delta_{old} - 1$;
9           $\Delta z = Objective\ (b, \delta_{old}, \delta_{new})$;
10          if $\Delta z \leq 0$ or $Random[0, 1) < e^{-\Delta z/t}$ then
11              $z := z + \Delta z$;
12              $\delta(b) := \delta_{new}$;
13          end if;
14          if $z < z_{min}$ then
15              $z_{min} = z$;
16              'save $\delta(b)\ \forall b \in B$';
17          end if;
18      end loop;
19      $t := \alpha \cdot t$;
20  end loop;
```

Simulated Annealing is a random walk through the solution space. The algorithm starts at high temperature t_{start} (line 1). It computes in every step a new solution (lines 6 to 8), compares the new with the old one (line 9), and decreases the temperature based on a (exponential) 'cooling schedule' (line 19). If the new solution is better than the old one, then the algorithm takes it (line 10). If not, then the algorithm accepts the worse solution with a probability depending on the temperature (line 10). The algorithm stops if a given low temperature t_{end} is reached (line 4).

Before we start with our Simulated Annealing we reduce the set of test points to P' according to Equation 63 in a pre-processing step. Let N' be again the cardinality of P'. The two core components of our Simulated Annealing algorithm are on lines 8 and 9. On line 8 we modify the initial delay $\delta(b)$ of a base station transmitter b: we either increment or decrement $\delta(b)$. On line 9 we compute the change Δz in the number of terrain points with delay-spread depending on the modified transmitter b. If we can estimate the mean value of Δz, then experience shows that we should choose t_{start} such that $e^{-\Delta z/t_{\text{start}}} = 0.9$. The new value $\delta(b)$ is always accepted (see line 12) if the delay-spread is reduced ($\Delta z < 0$). In the beginning, when the temperature is high, the algorithm sometimes accepts large deteriorations in the current quality of its solution. As the temperature t decreases, only small deteriorations in solution quality are accepted. This is implemented by comparing the value $e^{-\Delta z/t}$ with a random number generated from a uniform distribution on the interval [0, 1) (see line 10). Due to this feature, Simulated Annealing is able to escape from local minima while it still exhibits the favorable features of local search algorithms, i.e., simplicity and general applicability.

The efficiency of this algorithm primarily depends on the function *Objective*. Let $v = |B|$ be the number of base stations on the terrain. A simple but less efficient implementation of the function *Objective* computes for all terrain points the delay-spread, changes the initial delay of b, recomputes the delay-spread, and returns the number of changed points with delay-spread Δz. In most cases Δz is greater than 0, so the changes must be undone. This can be achieved by simply resetting the initial delay of the last transmitter changed.

```
Function Objective ($b$, $\delta_{\text{old}}$, $\delta_{\text{new}}$)
    $\Delta z := 0$;
    forall $p \in P(b)$ loop
        if $DiffDelay1(p) > noDelay$ then
            $\delta(b) := \delta_{\text{new}}$;
            if $DiffDelay1(p) = noDelay$ then $\Delta z := \Delta z - 1$;
            $\delta(b) := \delta_{\text{old}}$;
        else
            $\delta(b) := \delta_{\text{new}}$;
            if $DiffDelay2(p, b) > noDelay$ then $\Delta z := \Delta z + 1$;
            $\delta(b) := \delta_{\text{old}}$;
        end if;
    end loop;
    return $\Delta z$;
```

A more sophisticated approach is presented in the code example of the function *Objective*. It is based on the idea that a delay modification for every transmitter $b \in B$ can only influence the delay-spread of points covered by b. The set $P'(b) \subseteq P'$ of terrain points with a power difference smaller or equal to $\Delta\pi$ covered by b is computed and stored in a pre-processing step before entering in the main loop of the Simulated Annealing algorithm. This can be done in $O(vN')$ time and with $O(vN')$ space. In a reasonable base station distribution the number of multiple covered terrain points should be small and therefore the factor of N' is much smaller than the cardinality of B.

In the function *Objective* we use two helper functions: the first, *DiffDelay1*(p), computes Equation 64 at position p. In case of delay-spread it returns the differential delay and otherwise the constant *noDelay*. *DiffDelay1* runs in $O(v^2)$ time, because every pair of transmitters is checked. The second helper function, *DiffDelay2*(p, b), looks for a positive delay difference between transmitter b and every other transmitter at position p. It returns either the constant *noDelay* or the delay difference if greater than $\Delta\delta$ and if at least one of the two involved transmitters has a power equal to $\pi_{\max}(p)$. The function *DiffDelay2* runs in $O(v)$ time.

Now, we discuss the function *Objective*. We iterate through every point p covered by transmitter b. On line 3 we decide between two cases: in the first case there is delay-spread at position p between a pair of transmitters. If we modify the initial delay of b, then we recheck all pairs of transmitters to find if there is delay-spread. In the second case we currently have no delay-spread at p, and we only get delay-spread if the delay between b and another transmitter at p exceeds the threshold $\Delta\delta$.

The analysis of our first approach of the function *Objective* shows that we have to scan the set P' twice. For each point $p \in P'$ we compute the delay-spread in $O(v^2)$ time. Together we get a worst time complexity of $O(v^2N')$. The analysis of the second approach primarily depends on the extent and the overlapping of the transmitter coverages. In worst case they cover the entire terrain and overlap completely and therefore, we also get a worst case time complexity of $O(v^2N')$. In practical situations the overlapping and also the covering is much smaller because of the limited cell radius of base stations. Therefore, the transmitters partition the terrain into radio cells, which means that each transmitter covers N'/v test points on average. In addition, almost every point p is only covered by one transmitter with maximum power. In practice, we have found that the function *DiffDelay1* is almost linear in v for a total practical complexity of approximately $O(N')$.

5.2.4 Tests and Results

In the last subsection we presented our implementation of a Simulated Annealing algorithm. Now, we evaluate the algorithm with a number of tests and test sets. One of the test sets has been artificially created with the BST LOCATION problem solver from the last section. We call this artificial test set A. Another test set, called B, is based on a real paging network. It is best suited for the DELAY SPREAD problem, because delay-spread minimization is primarily used for simulcast or quasi-synchronous radio networks.

Test set *A* is similar to test set two from the previous section. In test set *A* we use a slightly different coverage threshold (−82 dBm) and the CCIR Hata wave propagation model without visibility. In addition, we use a power difference threshold $\Delta\pi$ = 10 dBm and a delay difference threshold $\Delta\delta$ = 30 µs. *A* contains 210 base station transmitters in the first test.

The results of these tests are summarized in Fig. 21. In all three simulations we got a solution of just over half the size of points with delay-spread in fewer than ten minutes. For example, in the first simulation the algorithm reduced the number of points with delay spread from 5618 to 3181.

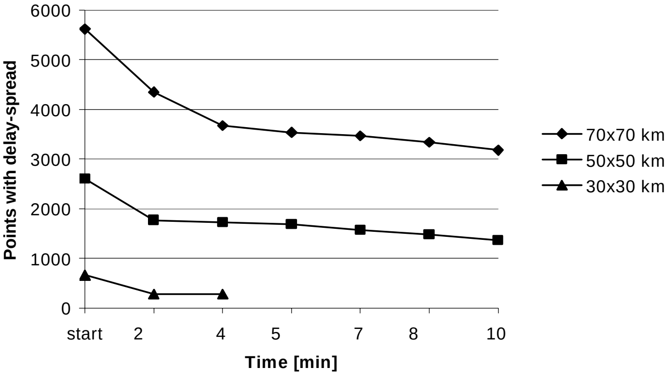


Fig. 21: Simulated Annealing algorithm performed on test set *A* with decreasing terrain size.

Test set *B* is a terrain of size 50×50 km around Zurich. It contains 40'000 regularly distributed test points and 94 real existing paging transmitters. All antennas are modeled as isotropic and almost all transmitters have a nominal power of 47 dBm (~50 W). The heights above ground vary between ten and one hundred meters. We use the same wave propagation model for all antennas, namely CCIR Hata with visibility and with coverage threshold set to −82 dBm. For the delay-spread computation we set the two threshold $\Delta\pi$ = 10 dBm and $\Delta\delta$ = 30 µs.

A visualization of the situation before and after the optimization of test set B is shown in Fig. 22 (p. 115). This visualization and optimization have been done with RA$_3$DIO (see also Chapter 6). Before the optimization we count 8561 points (21.4%) with delay-spread. After a runtime of 45 seconds the algorithm reduced the points with delay-spread to 6735 (16.8%), and after another minute to 5369 (13.4%). After 15 minutes runtime we had 2370 (5.9%) points with delay-spread. This is a reduction of more than 72%.

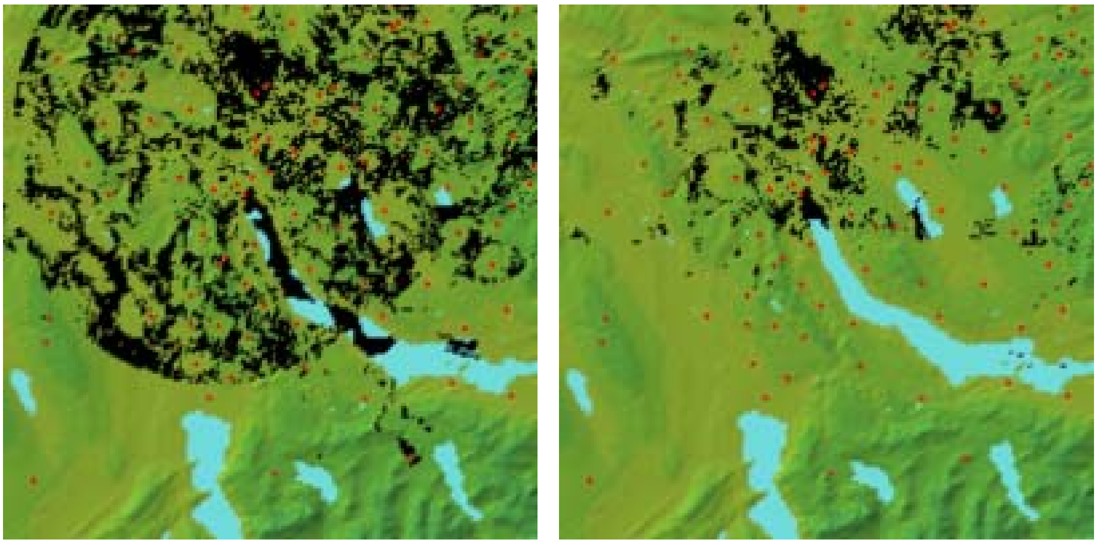


Fig. 22: Delay-Spread (black points) around Zurich before and after minimization.

5.3 Future Extensions

Our new BST LOCATION problem only considers RF design objectives, but an integration of capacity and teletraffic objectives seems possible. To do that the test points have to be replaced by demand nodes. A demand node denotes a position on the terrain with a certain teletraffic demand. This teletraffic demand can be static or variable over different time scales (daily, weekly, yearly). For instance, the teletraffic changes during the day: in the morning and in the evening are people at home and their mobile phones are used at their sleeping places, while in between they are phoning from their working places. Optimal network planning needs to be aware of such variable teletraffic distributions. Furthermore, a base station transmitter has only a fixed number of teletraffic channels, so it cannot cover more demand nodes than its available capacity. In areas with large teletraffic demand, like in dense cities, coverage is rarely the problem, rather capacity is the real limitation. This means, there are small micro- and picocells with reduced power necessary to cover this large teletraffic demand. Unfortunately, the number of micro- and picocells cannot be arbitrarily increased, because the total number of carrier channels is fixed and because cells using the same channels must be far apart from each other. This distance between cells using the same frequencies must be considered, too. The way this is usually done is to introduce an interference measure, e.g. co-channel interference. A somewhat more sophisticated radio network optimization should take all these considerations into account.

The radio signal delay from a point source, e.g. a satellite, to a base station, and from there to a mobile station, may induce an interference problem in simulcast or quasi-synchronous radio networks. The problem occurs if radio signals from different transmitters with almost equal power but large different delays reach a mobile station. Based on this, we have defined a new optimization problem, called the DELAY SPREAD problem. While we have not yet proven it, it seems that DELAY SPREAD is *NP*-hard. Therefore, we have presented an approximation algorithm based on a greedy heuristic and a Simulated Annealing algorithm. The greedy algorithm has not been implemented, since we have focused our interests on Simulated Annealing because it is scalable.

„Wenn man einen Computer programmiert, entäussert man sich eines kleinen Stückes seiner selbst, das so zu einem kleinen Stück des Computerselbst wird. Auf diese Weise wird es für einen sichtbar.“

Sherry Turkle

6 RA$_3$DIO

Classical GISs usually act as a repository of static or long time shifting geographic digital data like land registers, cartographic maps, transport networks, and so on. Whenever simulations are observed, changes in the simulation state are watched over time. Often, these changes occur in a quick consecution, since the goal of the simulation might be to find out some properties of a long time process. Such fast simulations are getting more and more important in geographic contexts, too. For instance the simulation of snowslides not only needs a 2½D terrain model but also a system capable of handling and visualizing quick simulation state changes. This means, a modern GIS with real-time simulation capabilities needs to be able to reflect quick changes in a virtual reality (VR) manner.

A number of visualization systems have been implemented which integrate 3D visualization techniques with large spatial geographic information and terrain data. Some systems stress accurate rendering of global images, or accurate modeling of environmental processes, often sacrificing interactivity of the system [NSTN93], [RDH+93]. Other systems emphasize tight integration of the 3D visualization with the powerful spatial analysis capabilities of GISs [Erv93]. Systems such as VGIS [LKR+97] and ViRGIS [POS+98] place a high priority on real-time, highly interactive 3D visualizations of the spatial data. Maintaining truly real-time update rates in the face of large, complex datasets requires special techniques and time-critical visualization system designs. Some issues particular to such time-critical computations in visualization environments are discussed in [BJ96].

In our context of radio wave propagation prediction, simulation is a powerful tool for planning base station locations and varying the huge number of parameters needed to operate a radio network in an almost optimal way. Fast changes in coverage and interference prediction due to altered simulation parameters are very desirable, because they allow a greater number of constellations to be checked and therefore a greater chance to find a better set of parameters in the same time. Real-time simulation and consequently the use of a GIS with VR and real-time simulation capabilities instead a classical GIS only makes sense if the prediction is fast. Unfortunately, faster prediction methods often suffer from less accurate results. Thus a coverage prediction system built as an application of a VR-GIS acts mainly as an interactive (first step) planning system and eventually as a presentation system. In case of prediction methods which are scalable in accuracy and therefore in time, a VR coverage prediction system can cover fast interactive planning as well as time consuming high accurate planning.

The system we present in this section, RA_3DIO (Radio Antenna placement with 3D Interactive Optimization)[12], is such a coverage prediction system based on a VR-GIS. It supports the design, analysis, and optimization of mobile radio network systems. RA_3DIO is the development of our research prototype WorldView [BES+98], [Bec99]. WorldView has been implemented to visualize and explore virtual terrains and terrain related themes based on the ideas in [POS+98]. WorldView and thus RA_3DIO maintain hybrid raster-vector data (such as surface triangulations), raster image data (such as those from satellite images, aerial ortho-photos, and digital topographic maps), and non-geometric data (such as population counts of communities). It allows a user to explore a geographic scene in real-time by means of a standard input device such as a mouse or a specific 3D input sensor, e.g. geometry/space ball, glove, 3D/space mouse, etc. It also supports a wide range of output devices such as standard monitor display, stereoscopic shutter glasses, head-mounted display, and so on.

Multiple software tools are available, handling real-time walk-through in 3D scenes. Usually, these tools limit their exploration capabilities to a scene that is loaded a single time in main memory. It is not possible to replace any parts of the scene. Only the whole scene can be removed and new one can be loaded. While this is good enough for several applications, it is far from satisfactory for terrain exploration, where gigabytes or even terabytes of data need to be explored. That is, while the walk-through progresses, data must be loaded dynamically from disk. Our VR component implements such a dynamic maintenance of a part of a very large scene in main memory, to be visualized using the *WorldToolKit* (WTK)[13].

RA_3DIO is currently based on the analytical design approach to cellular network planning (see also Subsection 2.4.2). This approach focuses on radio planning aspects, i.e., selection of cell sites, frequency planning and antenna design. A shift to the integrated design approach (see also Subsection 2.4.3) should be easy after all planning modules have been realized. In the current version of RA_3DIO only radio network definition and propagation analysis phases are supported for asynchronous cellular radio networks.

During the radio network definition phase, you choose transmitter locations which are based on your experience, the geographical terrain of the area to be supplied, and the output of the optimization tool of RA_3DIO. Using these transmitter locations, the propagation analysis phase evaluates the radio coverage by field strength prediction

---

[12] RA_3DIO is a trademark of xeraina GmbH, http://www.xeraina.ethz.ch.

[13] WorldToolKit is a registered trademark of Sense8 Corporation.

methods. Here, stochastic channel models as well as more sophisticated approaches, discussed in Chapter 4, are applied. RA_3DIO also provides co-channel interference analysis. This is the first step toward the frequency allocation phase.

In quasi-synchronous or simulcast network planning, like paging, RA_3DIO also supports the fourth phase of the analytical design approach, the radio network analysis. In this phase the delay-spread or non-capture areas are computed and visualized, but RA_3DIO goes further. It uses optimization techniques to minimize the non-capture areas by adding an additional individual signal delay to each base station.

RA_3DIO is able to handle a huge number of real antenna types as well as the isotropic antenna, which is mainly of theoretical interest. The characteristic of a real antenna type is described by a dozen of parameters as well as a vertical and horizontal radiation pattern (see also Subsection 4.1.1). These two radiation patterns show the antenna gain as a function of the vertical and horizontal angle, respectively. The patterns can be visualized in an additional window. Major antenna manufacturers (e.g. Kathrein, Huber & Suhner) make these parameters and patterns available in a digital form. We implemented a parser for the widely used Planet-MSI[14] data format. Because of the simplicity of the Planet-MSI data format, users and suppliers are able to write their own antenna pattern files.

In radio relay link planning the visualization of the terrain profile between transmitter and receiver is quite important. RA_3DIO offers the visualization of such a terrain profile in a separate window. Either the static profile between two fixed points or the changing profile between a fixed point and the current pointer position can be investigated in real-time. A radio relay link is called almost perfect if no obstacle intersects the first Fresnel zone between transmitter and receiver. In that case free-space path loss can be considered. Therefore, the first Fresnel zone is an important parameter and is computed and drawn in the terrain profile window, too.

The remainder of this chapter is organized as follows. The first section presents some system requirements and underlying design principles, which we followed during the entire design and development process of WorldView and RA_3DIO. Section 6.2 describes the overall system architecture of RA_3DIO. RA_3DIO consists of a pre-processing unit, a run-time system, and a database system. The first component, the pre-processing unit, is used to gather the different types of data and transform them into a format that is readable by and quickly accessible to the latter. It is described in Section 6.3. The run-time system of RA_3DIO is the interactive part of RA_3DIO. The user interacts with it using a (3D) input sensor and gets visual feedback on some (3D) output device. To update the visualized scene in real-time the real-time system sends range queries to and retrieves terrain and object data from the database system. The run-time system is presented in Section 6.4 and the database system in Section 6.5. In

---

[14] Planet is a registered trademark of Mobile Systems International (MSI);
Internet: http://www.msi-us.com/planet.htm

Section 6.6 we describe a valuable feature of RA$_3$DIO, the bidirectional link to a statistical graphics package, XGobi. Section 6.7 concludes the chapter.

6.1 System Design and Requirements

The rationale behind many of the design decisions made in the development of RA$_3$DIO was driven by both hardware constraints and user requirements. Typical hardware constraints include platform, small size of main and texture memory in contrast to external storage, available precision in geometric calculations, rendering speed, etc., while the user requirements include interactivity, intuitive user interface, display accuracy, off-line processing time, soundness and scalability of algorithms, as well as flexibility and extensibility from both the end user's and developer's perspectives.

Due to the large datasets that RA$_3$DIO typically works with, most of the data must be put in external storage, and dynamic scene management is used to bring in and cache the appropriate data for display. Level of detail (LOD) techniques are applied to both geometry and texture to further limit the amount of data stored in main and texture memory. To obtain a high degree of interactivity, the system is broken down into a number of asynchronous threads and any task which tends to monopolize the CPU is run in a separate thread.

RA$_3$DIO has been designed to run on different Windows platforms, but is primarily targeted toward high end *Windows NT*[15] graphics workstations. The system is written in *Visual C++*[16] and makes use of several off-the-shelf software packages. We use *Microsoft Foundation Classes* (MFC)[17] for the graphical user interface (GUI) and simple data structures, the OpenGL[18] based WTK for the rendering pipeline, and LEDA (Library of Efficient Data types and Algorithms)[19] for efficient and robust data structures.

6.1.1 Precision and Correctness

Precision in geographic and geometric calculations is a very important issue. In RA$_3$DIO, to gain maximum accuracy the earth is represented as a spheroid. Using a

---

[15] Windows NT is a registered trademark of Microsoft Corporation

[16] Visual C++ is a registered trademark of Microsoft Corporation

[17] MFC is a class library part of the Visual C++ development studio

[18] OpenGL is a registered trademark of Silicon Graphics, Inc.

[19] LEDA is available for academic customers at the Max-Planck-Institut für Informatik, Saarbrücken; Internet: http://www.mpi-sb.mpg.de

single global coordinate system for the entire globe and assuming 32-bit single precision floating point is used to describe geometrical objects (this is typically the highest precision available in current graphics hardware), the highest attainable accuracy on the surface of the Earth is half a meter. Clearly, this is not sufficient to distinguish features with details as small as a few centimeters. To overcome this problem, a number of local coordinate systems, which have their origins displaced on the spheroid surface that defines the Earth sea-level, should be defined. In the current implementation of RA_3DIO a maximum accuracy on the surface of only one meter is provided.

Geometric algorithms are typically derived under two simplifying assumptions: the underlying machine model is the *real RAM model* which can compute with real numbers, and inputs are in general position. However, the number types offered by program languages are only crude approximations (limited in range and precision) of real numbers and practical input is frequently degenerate. In case of the limited range and precision of floating point arithmetic the simple line intersection problem may result in a bizarre situation, where the intersection point of both lines is not part of neither. This can easily lead to incorrect geometric implementations. The two approaches to overcome this problem suggested in [MN99] are inexact arithmetic [Mil88] and exact arithmetic [FvW96]. The former replaces member and equality tests by adequate tests with limited precision and the latter introduces new data types with arbitrary precision, e.g. rational based on arbitrary long integers.

Aside from precision and degenerate input issues, the intricacy of geometric algorithms makes them difficult to implement correctly. The implementation of published and correct geometric algorithms is often a very time-consuming, intellectual challenge. So, how can we ensure that the implemented algorithms are correct? Careful, readable programming, extensive testing, and the concept of program checkers [BK89] are very helpful. Programming and testing issues are not discussed here, but we briefly present program checkers. Consider a program P that computes a function f. We call P checkable if for any input x it returns y, the alleged value of $f(x)$, and maybe additional information I that makes it 'easy to verify' that indeed $y = f(x)$. By 'easy to verify' one means two things. Firstly, there must be a simple program C (a checking program) that, given x, y, and I, checks whether indeed $y = f(x)$. The program C needs to be so simple that its correctness is 'obvious' or it needs to itself be checkable by a simpler checker, where simplicity refers to provable correctness. This guarantees a chain of increasingly simple checkers. Secondly, the running time of C on inputs x, y, and I should be no larger than the running time of P on x. This guarantees that the checking program C can be used without severe penalty in running time.

In LEDA many algorithms come with checkers [MN99]. A simple example is the following: Consider a program that takes a matrix A and a vector b and is supposed to check whether the linear system $A \cdot x = b$ has a solution. The program returns a Boolean value indicating whether the system is solvable or not. This program is not checkable. In order to make it checkable, the interface is extended. It either returns a vector x if the system is solvable or it returns a vector c such that $c^{T}A = 0$ and $c^{T}b \neq 0$. The extended program is easy to check. If it answers true, we check that indeed $A \cdot x = b$ and if it answers false, we check that $c^{T}A = 0$ and $c^{T}b \neq 0$. Thus the check amounts to a matrix-vector and a vector-vector product which are fast and easy to program.

In RA_3DIO, one of the major algorithms is the divide-and-conquer point-visibility algorithm. The algorithm returns an array of Boolean values indicating whether a point is visible from a given viewpoint or not (see also Subsection 4.3.1). This geometric algorithm is quite intricate and difficult to implement, so a checker program would be very helpful to verify the answer of the algorithm for a single problem instance. Of course, this is much less than program verification which gives a guarantee for all problem instances, but it is assuring. However, it is by no means obvious how to find a checker program for point-visibility in terrains. The output of the algorithm, the discrete visibility map, is a row of the point-to-point visibility matrix of the terrain. Thus any simple matrix or graph property which applies and is able to characterize point-to-point visibility matrices or graphs would be a candidate for a checker program. Unfortunately, such properties are still not known. We do not yet exactly understand, even in 1½D terrain, under what conditions a given visibility matrix corresponds to a realizable terrain. Another approach for a point-to-point visibility checker program could be the following: program a simpler ray-shooting point-visibility algorithm and compare the outputs of both algorithms. Unfortunately, this is not a good checker program since its running time could be larger, but we currently do not know any better possibilities. At least this approach has the advantage that we can choose between different visibility algorithms depending on the problem instance. An extension of the algorithm interface does not help. The only additional information we can return for each testing point p is the vector of segment chains which have been tested against p if p is visible from V, or a single blocking chain c if p is not visible from V because of c. In both cases it takes a lot of time to extract the relevant edges of the terrain and to check the visibility or invisibility.

6.1.2 Efficiency

As already stated in the beginning of this section, efficiency is a major design goal in the RA_3DIO project. Our algorithms are mainly based on LEDA data structures and algorithms. The latter itself are usually implementations of the asymptotically most efficient algorithms known for a particular problem.

Sometimes there is a trade-off between efficiency and other software engineering principles like code readability and flexibility. To avoid the most dangerous pitfalls in a project where efficiency is a major design goal we provide the following three rules:

- *efficiency in the beginning:* You should start with the most efficient data structures and algorithms. This could be difficult because the knowledge about the complete design is missing, the estimates of the production load on the system are not very realistic, and the production system might not be available. Therefore, a popular motto in the community is: ‘Make it run, make it right, make it fast.’ The code complexity (size and dependency) of a project is small in the beginning and increases during the project. Because efficient algorithms are often more difficult to implement than less efficient ones, it is a big advantage to provide freedom to introduce new data structures and other information, without regard to many dependencies. Additionally, the earlier you introduce a specific and efficient data structure the more algorithms can use it from the beginning. This helps to avoid re-implementing multiple algorithms for efficiency reasons. Probably it results in less transparent dependencies as we can see in the following example. Assume we have a directed graph as a data structure in our program and the outgoing edges of each node are stored in a incidence list. In some algorithms circular ordered incidence lists can improve the efficiency of the algorithm, so we want to assert this graph representation property, but we do not check this precondition in the concerned algorithms, because it would kill the efficiency gain. It follows that the dependency on this specific graph representation can neither be seen in the parameter list nor in the precondition. In a later state of the project an algorithm which used the specific representation of the incidence list is replaced by another which does not need a circular ordered incidence list. Then we might think about to simplify the graph construction and to omit the property of circular ordered incidence lists. But what impacts does it carry? Which part of the code depends on circular ordered incidence lists? This type of dependency is not transparent since usual program languages[20] as well as data structure libraries do not provide specific (language) constructs to formulate properties concerning the data structure representation. Therefore, we introduce a property attribute for each instance and predefined set of properties for every type of a data structure. This property attribute may be tested in pre- and postconditions and thus increase the transparency of dependencies.

- *off-the-shelf components with big O:* The use of off-the-shelf components can help to drastically reduce the development expenses. This insight is not very new and also not very original. Unfortunately, if efficiency is a major design goal, then many developers get cold feet if the exact implementation or algorithm of an off-the-shelf component is not known and they implement their own solution. This leads to more transparency and to more insight but also to additional and maybe unnecessary work. Therefore, our proposal is addressed more to library and component developers than to library and component users. Whenever a function or procedure is hidden in a library, the user of the library should get as much information about the implementation of the function as possible. This usually stands in contrast to the habits of library sellers that only want to provide as little information as necessary to hide important details from competitors. A fair compromise

---

[20] The property and attribute concept in C# may be used to solve this problem.

between the two extremes would be the declaration of the asymptotic worst case time complexity in big O notation. With the big O notation the library user gains some knowledge about the efficiency and allows to guess about the algorithm used, but it does not provide much information for competitors. The value of the big O declaration can be illustrated by a simple example. We assume a library data type String and a function *Length*(*s*: String) which returns the number of characters in the string *s*. The user neither knows the implementation of the data type String nor the implementation and efficiency of the function *Length*, so she may store the result of *Length* in her own data structure. Now, she can be sure not to scan the string each time she needs the length of the string. Maybe the type String already contains an attribute *length* and the function *Length* only returns the value of this attribute. Then it is a waste of space to store the length again in a outer data structure. A simple declaration of the time complexity of *Length* would solve the problem.

- *readability on line-level:* For many programmers code efficiency takes place on line-level. They replace $x + 1$ by x++ or increment the loop iterator during conditioning testing or do something similar. We call that *optimization on line-level* because in general it only increases the code efficiency of a couple of lines but not the overall algorithm efficiency in terms of asymptotic time and space complexity. Often this type of code optimization results in difficult to read code. Additionally, most of the available compilers can do that type of optimization better than a programmer. Therefore, we propose to write 'well readable and understandable' code and to enable compiler optimization instead. What 'well readable and understandable' code precisely means is covered in several textbooks about software engineering and programming techniques [McC93], [Mey97].

6.1.3 Extensibility and Reducibility

Extensibility and reducibility are additional major design goals in the RA$_3$DIO project. Usually, extensibility comes alone without reducibility since most often added functionality and hence added code is seen as the only added value. This may either lead to fat, clumsy, and slow programs or to text editors which are able to navigate a space lab. We are aware of the great value of extensibility, but we propose extensibility and reducibility together, called *modularity*. Modularity also allows removal of unneeded functionality.

Modularity in RA$_3$DIO is achieved on the level of user panels (see also 6.4.3). Each user panel with the according data structures and algorithms can be enabled or disabled at compile time. This makes it possible to provide for each user or each different system an optimal combination of modules. To achieve this, a high degree of modularity with strong cohesion is needed, but with few dependencies over module boundaries. To design a system with a strong cohesion on the level of modules but with very small cohesion on a higher level is a hard task which demands a lot of experience in software engineering. Since it rarely succeeds to eliminate all dependencies between modules (without reducing the number of modules), careful programming is necessary to make hidden dependency visible. A language construct like the compiler directives in C and C++ are very helpful, not only to exclude some code from compilation, but also to mark dependencies over module boundaries.

6.1.4 Graphical User Interface (GUI)

Past experience has shown that users of virtual environments are vulnerable to getting 'lost' and disoriented in the virtual world. A number of navigation techniques have been proposed to alleviate this effect and aid understanding of the virtual environment [DS93]. To prevent the user from getting lost is only one design goal for a good VR GUI. Three more general design goals for any type of GUI are the following:

- The user should see at a glance all important data but not much more.
- The user interface should avoid hidden status and should make status clear.
- The user interface should be bidirectional. This means, all changes in the 'document' should be immediately mirrored in the user panel and all inputs to the user panel should be immediately transformed to the 'document'. Here, the term 'document' refers to the view of a document. A document need not only be a text or a spreadsheet, it can also be a more complex 3D world.

In RA$_3$DIO all four design goals have been taken into account. To prevent the user from getting lost four different techniques have been implemented: First, the user can watch her viewpoint position on a survey map. Second, the current coordinates of the viewpoint and the pointer position on the terrain are always visible. Third, the user can turn on hypsometric color and texture information (e.g. aerial ortho-photos, remote sensing satellite data, or topographic pixel maps). And fourth, the user can toggle abstract community objects which provide additional information about communities that can be hard to display on map level (e.g. name, population, language, etc.).

6.2 System Overview

RA$_3$DIO has been designed to run on different Windows platforms. It runs on both small Notebook computers and high-end *Windows NT* graphics workstations. The Windows platform has been chosen because of its wide distribution, its huge number of standard components, and its comfortable developer tools. The same reasons have been decisive for the choice of the programming language. LEDA and WTK are only available in C++ and in C/C++, respectively, so we decided to write our program code in C++. Today, Java or even C# could be an alternative, but five years ago, when we started with WorldView, the development tools of Java had teething troubles and C# had not been available.

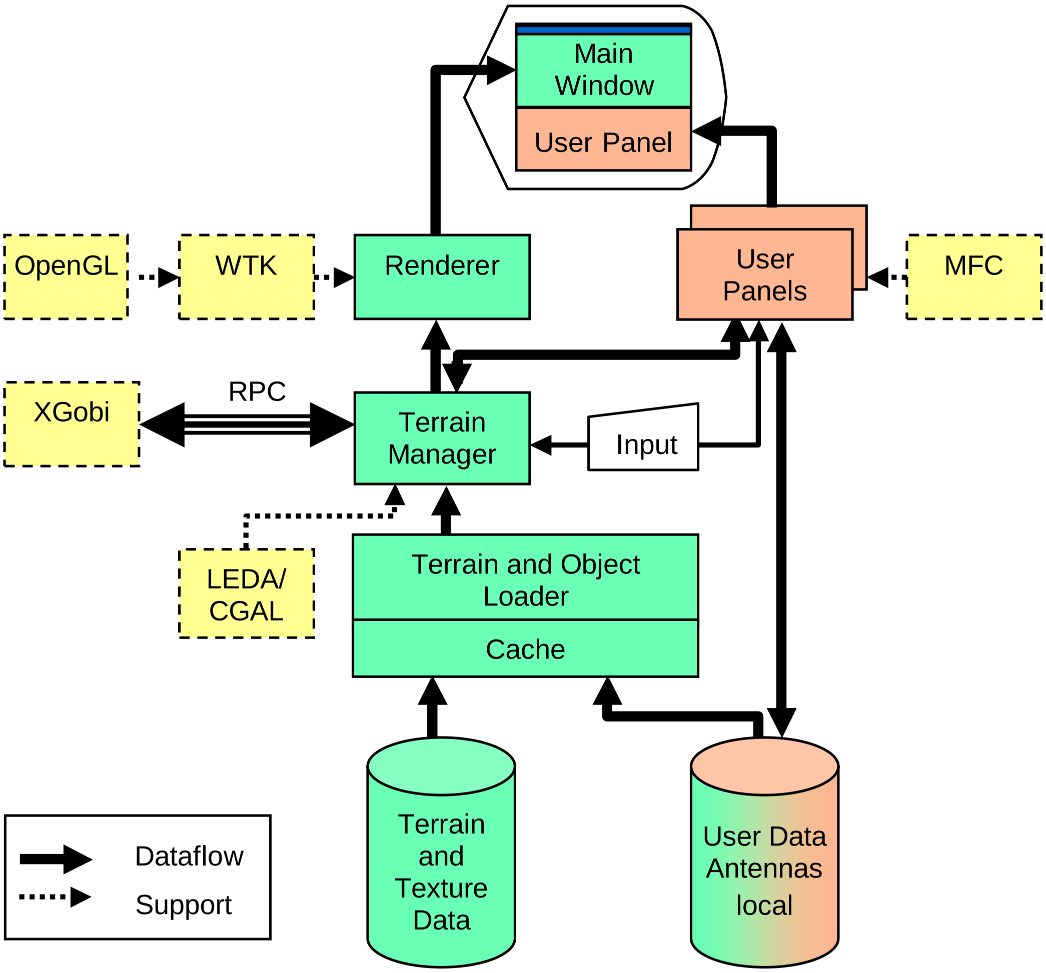


Fig. 23: Architecture of the run-time system of RA$_3$DIO.

RA$_3$DIO consists of a dataset pre-processing unit, a run-time visualization system, and a database component. Due to the vast volumes of data that make up the terrain database, the data is prepared off-line and structured in a form that makes it simple to read into main memory by the terrain manager. This form of pre-processing includes data format conversion and projection. The run-time system sends range queries to the database, retrieves data and visualizes it using the rendering capabilities of WTK. Fig. 23 schematizes the architecture of the run-time system. At the top, you see the screen with the application window consisting of three parts: *window frame*, *main window* and *user panel*. The main window contains the rendered 3D scene and the user panel is built by a number of property pages. Both together are framed by a standard Windows frame window.

At the bottom of Fig. 23 (p. 126), there are two database systems. One of them contains the (read-only) terrain and texture data and the other contains user specific terrain objects, e.g. antennas. The reason for the splitting the dataset into two database systems and the exact database architecture is introduced in Section 6.5. Terrain data flows from the terrain database via the terrain loader into the heart of the run-time system, the terrain manager. The terrain manager manages the currently visible terrain, the terrain objects, and also the user input to the main window. During the terrain load phase, special data structures are created to handle the various kinds of terrain operations (e.g. point localization, terrain vertex access, triangle neighborhood, etc.). From the terrain manager the data flows further through the rendering pipeline, where it is prepared for the visualization.

6.3 Pre-Processing

The purpose of the pre-processing component, or dataset builder, of RA$_3$DIO is to gather the different types of data and transform them into a format that is readable by, and quickly accessible to, the run-time system. For terrain data we use the binary file format (BFF) of WTK, and for imagery (texture data) either the PGF (see also Subsection 3.3.2) or the JPEG file format. A nice visualization of terrains often makes use of shading which is based on vertex normals (see also Subsection 3.4.2). Vertex normals have to be computed by inspecting the normals of the surrounding faces. All these tasks together are inherently compute intensive and cannot be performed by the run-time system quickly enough for proper frame rates. Additionally, the pre-processing only has to be done once for a given dataset and stored in an appropriate way, so it makes sense to pay the penalty of assembling a dataset up front.

Our dataset builder handles a large variety of different DEM, geomorphologic data, and imagery data types. The source data may consist of multiple and/or variable resolutions which have to composite into a single regular triangulation if so desired. The regular triangulation represents the original DEM without any errors. Based on this start triangulation several new triangulations with decreasing numbers of triangles and increasing errors are generated. We call these new triangulations the different level of details (LODs). The methods to compute hierarchies of triangulations have been discussed in Section 3.2. Because the run-time system handles terrain data in form of a dynamic scene management, the dataset builder is also able to cut the dataset into a matrix of terrain tiles. Each terrain tile is stored in a separate file which can be directly addressed by its coordinates. Finding the best size of a terrain tile is very crucial. If the size is too large, then it takes too much time to load the new tiles into main memory during the real-time terrain exploration. This may result in terrain rendering interruption and appears as a disturbing stagnation during the smooth flight. On the other hand, if the tile size is too small, then the total size of all tile borders is too big. A large tile border size is a drawback, because all vertices on borders are stored in main memory at least twice. Therefore, small tile sizes increase the storage requirement.

In addition, each source data may refer to another coordinate system. Therefore, the dataset builder needs to be able to convert between different coordinate systems. In the current implementation, the system handles geodetic coordinates in almost every geodetic datum and coordinate systems based on a (Oblique) Mercator projection, like the Swiss coordinate system. To be able to handle these coordinate systems it has to compute both forward and inverse Mercator and Oblique Mercator projections. The dataset builder and also the run-time system of RA_3DIO can each handle both geodetic coordinates in a user specified geodetic datum and projected coordinates. This is very helpful for the user, because the use of unusual coordinate systems can handicap system usage. For instance in Switzerland, the users are very familiar with the Swiss coordinate system but only few people know the corresponding geodetic coordinates.

The dataset builder is a stand-alone application that can also be started from inside of the run-time system. It has been designed to be very flexible and modular in terms of incorporating many data types.

6.4 Run-Time System

The run-time system of RA_3DIO is the main component. It is a very interactive program. A user interacts with it using an (3D) input sensor and gets visual feedback on some (3D) output device, for instance a standard color display. Depending on the output device and the rendering capabilities of the graphics hardware either a normal or a stereo view is produced. The virtual scene rendered to the output device is based on a scene graph that contains all necessary information about the scene, such as light nodes, geometry nodes, and transformation nodes. All geometry nodes of the scene graph are coupled with the terrain data structure in the terrain manger.

The terrain data structure is basically a matrix of terrain tiles which easily supports dynamic scene management. The size of the tile matrix determines the maximum visible part of a much larger terrain on disk. It can be specified by the user if so desired. This visible part of the terrain is shifted through the entire terrain by moving the user viewpoint. Normally, the user viewpoint lies above the central tile of the matrix. Whenever it is moved across the boundary of the central tile, then the matrix is shifted and a terrain update request is sent to the terrain and object loader. This loader then sends an appropriate range query to the database system and retrieves the new terrain tiles and/or objects from there. Tiles no longer needed are moved to the *terrain cache* and the new loaded terrain tiles are integrated at the correct position in the terrain matrix. Moving a tile to the terrain cache entails disabling the corresponding geometry node in the scene graph.

To achieve real-time scene updating and rendering at interactive frame rates a multithread kernel is very helpful. Time consuming tasks such as loading new terrain data or optimization jobs are performed in separate parallel threads. The main thread polls the Windows message queue for user interface messages and initiates the rendering loop each time the scene graph of the main window has changed. Viewpoint position changes are also observed by the main thread and reported to the terrain/object loader by inserting an update job.

The terrain/object loader uses a priority queue data structure for its job queue. This allows prioritization of the update jobs depending on the data type and tile position. Terrain tiles in the center of the matrix are updated earlier than tiles on the edge because the user usually looks at the center of the terrain matrix. Also, terrain update jobs get a higher priority than object loading jobs, because the user wants to see the terrain before the objects placed on the terrain. The job with the highest priority is dequeued by the loader. In the special case of a terrain tile load request the data is only loaded from disk if it is not already in the terrain cache. After transferring the tile data from disk a tile data structure is allocated and the geometry of the tile is inserted into the scene graph. The functionality of building and updating a scene graph, and the rendering of the scene graph, is completely under the control of WTK.

6.4.1 Scene Graph

A virtual *scene* is a collection of geometries and lights, along with the positional information that places these elements at particular locations. In WTK, the only other element that is considered to be directly part of the scene is fog. A geometry includes a list of vertices and a list of polygons (triangles). WTK maintains the current elements of the scene in a directed acyclic graph known as a *scene graph*.

The exact structure of RA$_3$DIO's scene graph is shown in Fig. 24 (p. 130). It consists of a number of different node types, represented by different symbols. The exact behavior of each type is discussed in the reference manual of WTK. We only give a short overview about the structure of the scene graph used in RA$_3$DIO. Every scene graph starts with a root node. The first child of the root node of our scene graph is a transformation node. Transformation nodes transform all geometries of the following subgraphs. We use this transformation node to scale the height coordinates of the terrain data online. An increased height may improve the visual impression. The next node in the graph is a moveable geometry. It represents our sky hemisphere which follows the viewpoint, therefore it must be moveable. The sky moves with the viewpoint, so the user has the impression of constant distance to the sky. The last children of our root node are *n* group nodes including the terrain tiles of the scene map and the objects placed on these tiles. Such a tile group consists of a terrain geometry, a group of transmitters, and a group of city objects. Each transmitter as well as each city object is a separator node. Separator nodes are similar to group nodes but include by definition a transformation node as is first child. Additionally, separator nodes have the nice property of limiting the influence of their transformation node to all geometries in the subgraph of the separator node. This allows to define the geometries in a local coordinate system and to use the transformation node for their transformation into a world coordinate system. Cities are simply represented by boxes in RA$_3$DIO, while transmitters have a bit more sophisticated visual appearance. Each transmitter includes five different geometries for five different levels of detail. The switching between the different geometries is controlled by a LOD group node in such a way that the appropriate geometry is chosen relative to the distance to the viewpoint. This is a very important feature if the user has to see the transmitter from varying distances.

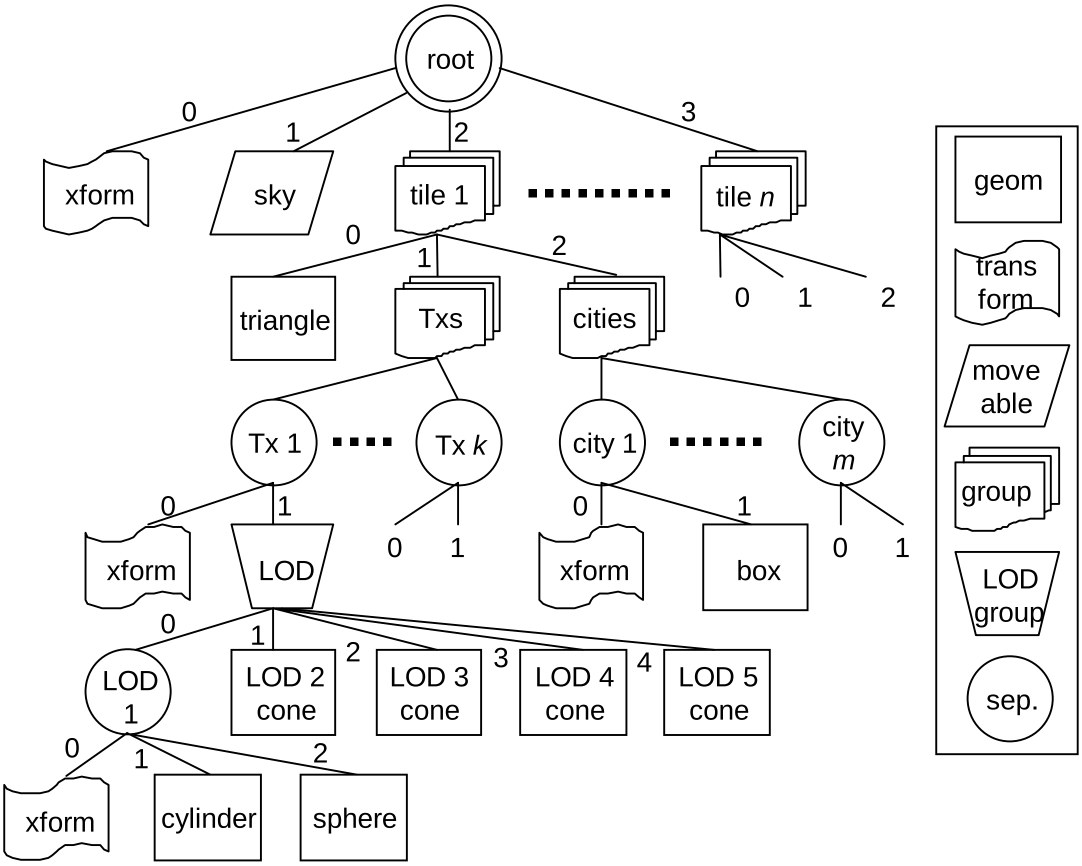


Fig. 24: Scene graph of RA$_3$DIO.

In RA$_3$DIO the diameter of the transmitter symbols varies between 10 and 1600 meters. The smallest transmitter symbol is used for distances up to five kilometers, while the largest is used for distances greater than 250 kilometers. This allows the user identifying each transmitter from distances up to 500 kilometers provided that the distance between neighboring transmitters is large enough.

6.4.2 Terrain Manager

The main part of the terrain manager is its terrain data structure. This data structure needs to allow simple dynamic scene management and of course contain all necessary information to visualize the terrain. In [Paj98] they used a double connected mesh of rectangular tiles while we use a simple matrix of rectangular tiles. In both cases the tiles contain the actual terrain information. Because the main purpose of RA_3DIO is the placing of antennas on terrain and the computation of their wave propagation, the tile data structure needs to efficiently support point location and terrain traversal in any front-to-back order. There are several different data structures known for polyhedral terrains, e.g. DCEL, Halfedge, Winged-Edge. All of them need at least four references per edge. A survey and a comparison of edge-based representations is found in [Ket99]. In our case a more space efficient data structure is an undirected dual graph of a triangulation together with the WTK geometry data structure. The undirected dual graph needs only two references per edge. Both the point location and the traversal are well supported by the dual graph.

As already mentioned, the terrain data structure is a simple matrix of tiles. Each tile consists of a matrix M of vertex information, a dual graph G of all triangles inside the tile, and a border data structure B. M represents the regular grid at the highest terrain resolution and is limited to a maximum of 256×256 entries. This means the matrix can be sparse at levels of less detail.

A vertex information *VInfo* is a record containing three parts: 1) a reference to a vertex of a WTK geometry of the scene graph, 2) a list of intensities, and 3) a number of properties. Each polygon includes a counterclockwise ordered vector of vertices and each vertex includes a normal, a color, and a 3D position. The intensity lists are used to store the relevant antennas. For each antenna in the list, we store trivalent visibility information (visible, invisible, unknown) and the amount of propagation power at the specified vertex position. Additionally, the properties of each *VInfo* instance include information about the geomorphology at this particular terrain location.

The undirected dual graph G of a tile t defines the neighborhood between the triangles in t. Each triangle is represented by a node of G and each arc of G connects two triangles sharing a common triangle edge. Triangles at the four borders of t have fewer than three neighbors inside t, so we introduce four additional nodes, each representing a neighboring tile. It follows that each triangle always has three incident arcs which simplifies several traversing algorithms. Furthermore, the nodes and arcs contain information. Each node contains a pointer to the corresponding WTK polygon and the arcs contain information about both endpoints of the common edge of the two neigh-

neighboring triangles. The arcs store the corresponding matrix indices of the two *VInfos* such that the direction of the terrain edge (represented by the order of the two *VInfos*) is parallel to the spin of the source of the arc. Because of the limited number of entries of M the arc information is a simple 32-bit value. Such a dual graph allows both a traversing of all triangles in a front-to-back order and access to the underlying WTK data structures and vertex information.

We have already seen that the dual graph contains an additional node for each of the four neighboring tiles. These nodes allow direct access to a neighbor tile, but usually this not enough since we want to know the exact neighboring triangle and not only the neighbor tile. Therefore, we introduce a border data structure B. B is a dictionary with a linear ordered key type. We use the matrix index of the starting vertex of a border edge as the key and the corresponding graph arc as the information. Thus, the searched triangle is found by looking up the correct vertex index of the given triangle in B and by following the returned arc.

Let N be the number of all vertices in the internal terrain data structure and let p be the number of terrain tiles in the matrix. In the worst case the number of vertices n in any tile is equal N/p. Further, let $v \in O(N)$ be the number of antennas in the terrain. Then the overall space complexity is $O(p \cdot n \cdot v) = O(N^2)$. The coverage of each antenna is limited by a maximum distance, so the number of covered points is normally a small fraction of N. In another way, in a good distribution of antennas each point of the terrain is only covered by a small number of antennas, so v can usually be treated as constant.

6.4.3 User Interface and Navigation

We designed the user interface of the run-time system with the four design goals of Subsection 6.1.4 in mind. As already stated the graphical user interface (GUI) consists of three main parts:

- *window frame*: The window frame contains a pull-down menu at the top and an integrated status line at the bottom. Navigation through large pull-down menu trees is often time consuming and not very transparent, so the menu mainly contains rarely used items, such as options and properties. The status line provides free sight to data which should be always visible. Thus it contains all necessary flags to avoid hidden status and data about the current position of the pointer on the terrain, e.g. coordinate, antenna name, distance to selected antenna, field strength, delay-spread, etc.
- *main window:* The main window shows the rendered view of the terrain and the objects placed on the terrain. It is the only window which visualizes a part of the 3D scene. On hardware systems with stereo rendering capabilities the main window can be toggled to stereo viewing. The user can interact with the terrain and the terrain objects by using a pointing device, e.g. a standard three-button mouse. An-

tennas can be placed on terrain, selected, directed, and dragged by pointer device interaction.

- *user panel:* The user panel is placed below the main view. It consists of a number of property pages arranged in a tab dialog box. This tab dialog box is a good way to save screen space without the drawback of increased loss of transparency. Additionally, the tab dialog box is easy to extend and to adapt for various kinds of program use. All information set and displayed in the user panel can be saved for the next program session.

We further equipped the user's interaction possibilities with a huge bundle of features. Only a few of them are listed here:

- Five terrain *rendering modes* are provided in the user panel 'Display': wireframe, flat shading, Gouraud shading, texture mapping, and transparent texture mapping. In the Gouraud shading mode the terrain is shaded according to the angle between a light source and the normal of the terrain vertices. The base color of the terrain vertices depends on altitude (hypsometric colors) and geomorphologic features. In texture mode, imagery is mapped onto the terrain. This improves the realistic impression, especially if the imagery has good quality. The transparent texture mode is helpful, whenever the user needs to see both geomorphologic and texture data.
- The user viewpoint is always located above the central tile of the terrain matrix. It has six degrees of freedom. Its position and direction can be changed either by pointing into the main window (free flight mode) or by provided user interface objects in the user panel. A common viewpoint direction points from top vertical down to the terrain. For this top view we switch to a parallel projection instead of a perspective one.
- During free flight mode the user can toggle automatic flight height adjustment. When enabled the flight height is adapted by a control loop, which takes the current flight speed, direction and the terrain profile into account. This allows to fly or drive in great speed over the terrain. In addition, there is collision detection which prevents the user from crashing into the terrain.
- The user panel 'Navigation' contains a small overview map. Inside of this map the current viewpoint position is marked by a crosshair cursor and the current visible part of the terrain (matrix of tiles) is marked by a rectangle. The size of the terrain matrix can be specified by the user during the session.
- Sometimes there are only small changes in terrain data, where the height differences between the vertices are too small to influence an expressive shading. In such a situation the user has the possibility to scale the height coordinate of the terrain data online. This increased height both helps to detect topographic details and improves the visual impression.
- Although the run-time system is very intuitive, it also provides a good help system. It is based on standard Windows help and contains an index, an overview, and context sensitive help.

- The last but not least item in this enumeration is the printing feature. The user is able to print the visible scene on any standard printer with an additional title, grid, scale, and legend.

6.5 Database

The database components maintain terrain data, imagery, and other geographic and geometric objects in different spatial access structures within a database management system. Depending on the available hardware and network capabilities we provide two different database concepts.

The first database concept is based on two databases. One of them operates as a (remote) terrain server and the other stores user specific data locally on the client machine. The terrain server database handles the existing terrain objects of the virtual scene and offers a corresponding network-wide retrieval service for interactive and dynamic 3D applications. This concept allows the use of the same terrain data concurrently on several machines. The exact data structures of such a terrain server are described in [Paj98] in great detail. The efficiency of this concept primarily depends on the network transfer rate and the retrieval efficiency of the terrain server, which has to be independent of the visualization task. The main drawback of this concept is the permanent need of fast network access.

The second database concept is based on a single local database system, circumventing the need of permanent network access. We predominantly use this second database concept in combination with a portable client machine. To get the full efficiency of the first database concept, the client would have to retrieve the new terrain data, build the data structures, and visualize the scene with one single processor at the same time. This is far beyond processor speed, capacity, and data throughput of today's portable computers. A possible way to overcome this efficiency lack is the use of a two stage strategy (see Fig. 25, p. 135).

The first stage, the database system, contains the DEM, the compressed texture images, point objects with non-geometric data (e.g. antenna objects), and a spatial data structure (e.g. LSD tree or R-tree) for answering range queries in an efficient way. The second stage can be seen as a *database cache* with a fixed maximum storage size. This database cache must not be confused with the terrain cache mentioned in Section 6.4. In contrast to the terrain cache which is in main memory this database cache is on disk. The database cache is only used for terrain data but not for terrain objects. The triangulated terrain tiles, which have been computed during earlier database range queries, are stored for further access in this database cache. Therefore, a new range query for the same terrain tiles can be answered much quicker and without re-triangulating. Because of the fixed database cache size, we update a least-recently-used list whenever the cache is accessed. If the database cache reaches its maximum size, then we delete the least recently used terrain tile.

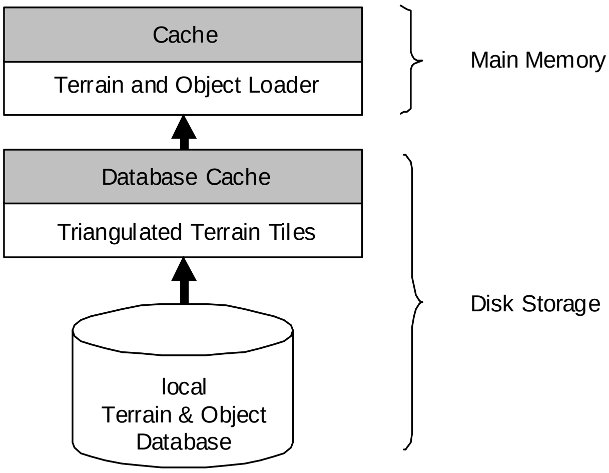


Fig. 25: Database concept.

6.6 The XGobi Link

The information that a system like RA_3DIO provides is in many cases incomplete without the proper statistical evaluations. For the influence of a mobile phone antenna, for instance, it is important to know its activity pattern: When is the antenna in use, and to what degree? What does this imply on the actual signal strength distribution over time and space, as compared to the maximum signal strength? These questions and many others might arise in the antenna placement decision process, and it is certainly wise to have a powerful off-the-shelf statistics package available to answer them. Such a package must interact properly with the VR/database system. This is actually a fairly complex task: The user might want to have several windows on her screen, one with the virtual reality, another with a histogram of antenna usage, and as she moves through the scene, she might want the histogram to dynamically change according to what she sees. If there is an interesting item in the histogram, she might want to have the system highlight it in the virtual reality window, and vice versa. This latter functionality is called *linked brushing*, because the selection (by brushing) of an item in one window also selects the same item in all other windows. It is easy to imagine that with such a system, far more interesting conclusions can be drawn than with a system that lacks any one of the components.

Unfortunately, it is by no means obvious how to connect the components in the best possible way. Certainly, it seems possible, just not desirable to change the source code of two systems in order to make them communicate. In this section, we report on the issues involved in a system interaction mechanism that we designed and implemented for the application above: RA_3DIO interacts with a statistical graphics package, XGobi.

XGobi [SCB98] can be best described as a high-interaction statistical graphics package. These packages have been developed over the last 15 years to enhance the graphic facilities developed for exploratory data analysis since the early 1960's. XGobi is mainly available for Unix platforms.

Our link connects RA_3DIO and XGobi with *Remote Procedure Calls* (RPCs). There are several possibilities to link or couple different programs. We call a coupling loose if the communication is realized by reading and writing shared data files. In contrast, a close coupling is based on built-in software components, which can be invoked by another program [GHW92]. RPC is a typical method of close coupling. For a fast working link that has to send information in real-time, close coupling is the method of choice.

RPCs are often integrated among other services in middleware. We considered several middleware standards like CORBA [OPR96], DCE [Sch99], DCOM [EE98], and Java RMI [JRMI]. We decided to use DCE (Distributed Computing Environment), because of its efficiency and its independence of operating systems and networks.

In the next two subsection we briefly introduce XGobi and DCE, the underlying communication method used for our link, before we present the ideas of the link between RA_3DIO and XGobi. We also provide insights into how to construct similar linked software environments and look at possible applications of our link.

6.6.1 XGobi

XGobi is a dynamic statistical graphics program that can be used for visual data mining. Some examples where XGobi has been successfully used are the display and clustering of shopping-frequency data [KS96] and the visualization of experimental measurements of laser performance and high-dimensional geometric objects [BCS96]. XGobi can be used in applications as varied as the classification of archaeological sites and for human motion data.

One of the main features provided by XGobi is the grand tour [Asi85]. In addition to the standard grand tour, XGobi also supports the projection pursuit guided tour, a combination of two complementary methods into an interactive and dynamic framework. Another main feature is linked brushing of multiple XGobi windows that are displaying different projections of the same data. Linked brushing is helpful in the exploration of clusters in high dimensions. The term *linked brushing* refers to the concept of marking points in one window using different symbols and colors and automatically updating all 'linked' views.

6.6.2 Distributed Computing Environment (DCE)

Today, there are several competing middleware standards. One of them is the OSF (Open Software Foundation, now The Open Group) DCE. This is a key technology in three of today's most important areas of computing: security, the World Wide Web, and distributed objects. It is the only suite of integrated services from a vendor-neutral source that enables organizations to develop, use, and maintain distributed programs across heterogeneous networks [DCE]. DCE provides a client/server model. A service exporter registers services at a server and a service importer can bind to those services. All DCE services are based on the RPC mechanism which is typical for middleware. Although the connected machines can have different data formats, there is no explicit data conversion necessary in the program's source code. DCE automatically converts data in the correct way during the marshalling/unmarshalling process.

In DCE, a program can offer a set of RPCs. It has to declare them in an interface specified in the *Interface Definition Language* (IDL). The IDL compiler generates a client and a server stub, which are linked to the client and to the server, respectively. The stubs perform the 'real work', the platform dependant communication, invisible to the programmer. IDL also generates some procedure declarations that let RPCs look like simple local procedure calls. Only the initialization and the finalization contain DCE-specific code. Each interface has a *universal unique identifier* (uuid) built from the network address and the system time at the time of the creation. This ID is important for the interface registration at a DCE server machine. After the registration, the server program runs in a status of waiting for incoming calls, where it is possible to answer multiple calls at once in multiple threads. A client can connect to the server and bind to an interface at run-time.

6.6.3 The RA$_3$DIO/XGobi Link

A link between RA$_3$DIO and XGobi is very useful and interesting. It offers the possibility to analyze the various kinds of data provided by RA$_3$DIO; an actual example is radiation emissions. RA$_3$DIO can provide a large amount of data for visual exploration in XGobi, e.g., the terrain data (polyhedral triangulated data) on its own or the spatial data objects handled in RA$_3$DIO, such as cities with their properties (city name, spoken language, number of habitants, city area, etc.) or the antenna data set (position, direction, antenna height, power, antenna type, carrier frequency, etc.). Even combinations of the previously mentioned multivariate, spatially referenced data objects can serve as a source for an exploratory data analysis in XGobi.

Our implementation of the link is based on the ideas of the previously developed link between ArcView, a GIS, and XGobi [CMSC96], [CSMC97], [SMCM97]. This link provides several different features to display spatial data in XGobi. One feature, the 'basic link', simply transfers multivariate data from ArcView into XGobi where it is displayed and can be interactively manipulated. For the other features, XGobi performs a pre-computation after the data has been transferred from ArcView. For instance, XGobi computes the variogram-cloud plot or spatially lagged scatterplots [SMMC97]. The pre-computation in XGobi is a little bit archaic because ArcView's script language Avenue was slow and not very well prepared for complex computations. For our application the pre-computation means that XGobi provides extended possibilities compared to the standard XGobi. All features of XGobi that are available in the ArcView/XGobi link can also be accessed in the RA$_3$DIO/XGobi link.

The main interactive function of our new link is linked brushing. This function is characteristic for the bidirectional link: On one side, user actions in RA$_3$DIO may change the data and the brushing information in XGobi. On the other side, interactive brushing that occurs in XGobi may result in data changes in RA$_3$DIO. Both XGobi and RA$_3$DIO are working as DCE clients and DCE servers at the same time. Therefore, they can communicate in both directions (see Fig. 26). XGobi provides 35 remote procedures as a server and calls one remote procedure as a client (update brushing information). Below, we describe some of the remote procedures in greater detail.

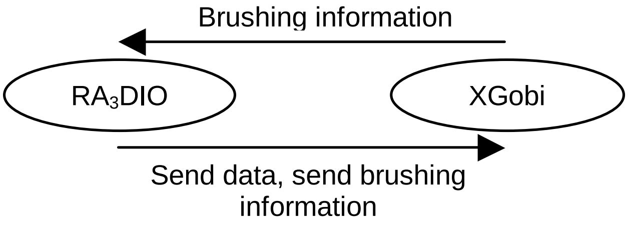


Fig. 26: Schematic RPC communication.

The 'basic link' between XGobi and an external program (e.g. RA$_3$DIO) consists of nine remote procedures that are provided by XGobi. First, XGobi has to be started in a remote mode. Then, by calling RPC_Init_Data, the client transfers the name, the number of data points, and the number of dimensions of the initial data set to XGobi. RPC_Send_Init_Data is the procedure to transfer the data to XGobi. With RPC_Send_Init_Symbols, XGobi can be informed about the initial color and/or symbols that are used to display the data points. After the RPC_Make_XGobi call, XGobi opens an X-window and gets ready for user interactions. Fig. 27 (p. 139) shows a simplified chart of the interaction between RA$_3$DIO and XGobi.

The biggest problem for XGobi was the synchronization with the RPCs, because XGobi is a single-threaded application. That means only one thread can access internal data structures at a time. In DCE, RPCs can occur at any time, so there has to be a synchronization between each RPC thread and the main XGobi thread. This is the classical mutual exclusion problem that can be solved using semaphores. In XGobi, every RPC first runs into a lock statement. The thread stops until the semaphore variable is unlocked by the XGobi main thread. XGobi calls the unlock statement many times per second. Then a waiting RPC can start its execution. The XGobi thread becomes suspended at the following lock statement until the RPC finishes its computation and unlocks the semaphore again. This method works in the same way if multiple RPCs (from different clients) are allowed at the same time.

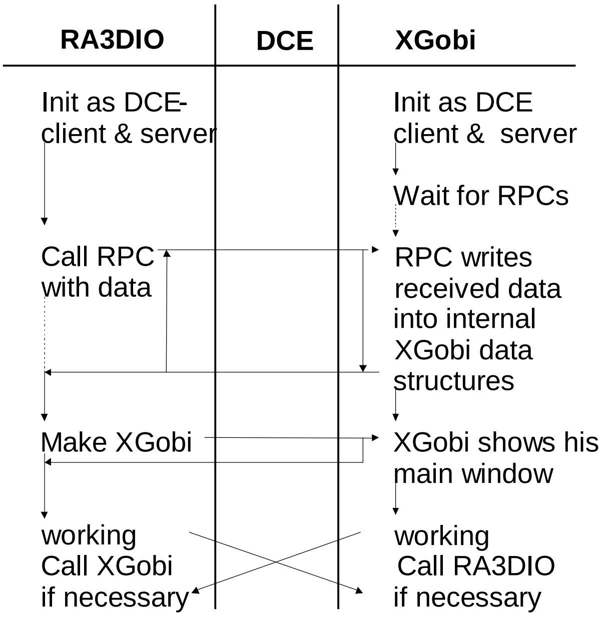

RA3DIO
DCE
XGobi
Init as DCE-client & server
Init as DCE client & server
Wait for RPCs
Call RPC with data
RPC writes received data into internal XGobi data structures
Make XGobi
XGobi shows his main window
working
Call XGobi if necessary
working
Call RA3DIO if necessary

Fig. 27: Interaction between RA_3DIO and XGobi.

The RA_3DIO side of the link is substantially simpler than the XGobi side because RA_3DIO is a thread-safe multithreaded program. Thus, the thread-based DCE-RPCs cooperate very well with the behavior of RA_3DIO. After a short initialization of the link, RA_3DIO is immediately ready to send the initial data set to XGobi and to open XGobi either on the PC with a X-window emulator or on a Unix workstation. During the initialization of the link, a thread handler must be started. This handler invokes a number of concurrent threads to serve all the incoming remote procedure calls. In combination with XGobi, Xfer_Brushinfo is the only remote procedure used on the RA_3DIO server side.

Once XGobi has been initialized, the user is able to interact with the multivariate data set sent by RA_3DIO. While brushing data points of the data set, XGobi calls Xfer_Brushinfo to inform RA_3DIO about the updated brushing information. The updated data points are numbered according to the order initially used and exactly these ordinal numbers together with the brushing information (symbol, size and color) are sent to RA$_3$DIO. RA$_3$DIO maps these numbers to the objects using an array and updates the specified appearance of the objects. XGobi only sends the ordinal number of the data point, so RA$_3$DIO has to store the array for the entire time while communicating with XGobi.

We have already mentioned that RA$_3$DIO visualizes only a small part of the whole terrain data at the same time. Thus, it makes sense only to transfer the visible data into XGobi. While exploring the terrain, new terrain tiles are loaded and tiles that are no longer used are removed. Hence, the data set in XGobi no longer represents the visualized data in RA$_3$DIO. In this situation, it would be desirable that XGobi can handle dynamically changing data, but it cannot. Therefore, we have to reinitialize the whole data set stored in XGobi and reinitialize the array to map the new ordinal numbers to the new objects.

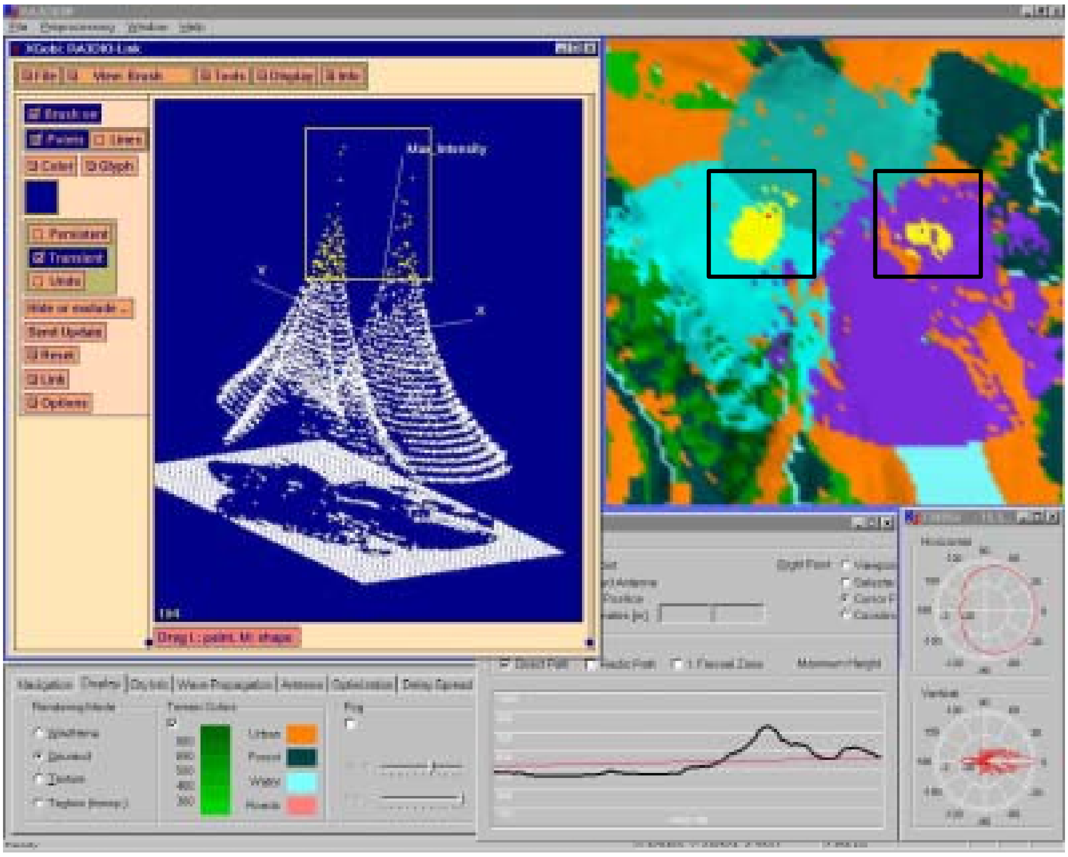


Fig. 28: Bidirectional link between RA$_3$DIO and XGobi at work.

Fig. 28 (p. 140) shows the link at work. The XGobi view (smaller window on the left) shows the 3D-coordinates of the antennas visible in RA_3DIO. Points in XGobi are brushed using different symbols and colors and the corresponding spatial locations are marked with the same symbols and colors in RA_3DIO. In addition, the link between RA_3DIO and XGobi allows the user to explore the terrain and view one or multiple additional variables related to the spatial location of the antennas in the XGobi window. This extra data can be the power of each antenna or the radiation it emits. XGobi allows the user to brush points that relate e.g. to antennas with high radiation emissions in her view. Through linked brushing, the corresponding spatial locations become highlighted in the RA_3DIO view.

6.6.4 Future Work

There is a need to extend XGobi's functionality of the link, because the 'send data' protocol of the link was not originally designed for dynamic data updates. This means that in the current implementation RA_3DIO has to resend all data of the actual matrix of tiles to XGobi and not only the data of the new loaded tiles since the last visible scene update. This results in a superfluous and excessive network load. An interim solution to handle this problem could be a data mirror residing on the same machine as the XGobi client.

Other future work will focus on the CORBA compliant standards. So far, there is no common standard for linking statistical software packages. CORBA can integrate some of these different viewpoints and should be the future basis for all links from application programs to statistical software packages.

6.7 Conclusions

We developed a VR-GIS prototype system, RA_3DIO, that simulates electromagnetic wave propagation for mobile phones and pagers, based on a 2½D digital terrain model. The system supports exploratory interaction, placement of antennas, computation of essential characteristics such as delay-spread in paging networks or co-channel interference in cellular radio networks, and interactive optimization of delay-spread and antenna positions. It has interfaces to antenna description formats of major manufacturers, to databases, and to other valuable tools.

RA_3DIO is based on a virtual reality 3D GIS kernel. This GIS kernel is able to handle spherical and spheroidal terrain data as well as classical terrain models defined on a plane. A virtual reality framework in the background provides three-dimensional scene management with rendering capabilities. Such a 3D renderer is a key component for visualizing spherical and spheroidal terrain models without any map projection.

However, classical map projections are still used because a lot of terrain models are based on projected coordinate systems.

We also studied and solved the problem of connecting two different software packages [SSSW00]. We linked RA_3DIO and the dynamical statistical graphics package XGobi. The users of RA_3DIO can now enjoy seeing additional statistical data in XGobi and analyze it with a standard package. At the present time, it is not necessary that every software package provides every useful function by itself. Instead, it is more economical to provide an open interface to off-the-shelf software packages so that other programs can use their special features. If the specification of the interface is public, a program can conquer new applications years after the programmer wrote the last line of its source code – even if the source code is not available to others. To link the two software packages, RA_3DIO and XGobi, we used a method of close coupling because the data throughput is a key component in a VR system. Remote Procedure Calls are our chosen method of close coupling. Among different middleware standards providing RPCs, we chose DCE-RPCs because of their independence from the operating system and network and because of their communication speed.

Our work has shown that it is an advantage to think about middleware already at the design stage. Many programmers still build huge monolithic systems that do not provide connections to the outside world. It should be of a greater interest to design interfaces of open systems [Gue98] and to connect them to program specific functionality. Furthermore, we have seen that a research prototype of a large computer program needs a very high level of completion to get a chance to be commercialized. The expense to reach this level is high and often underestimated.

„Ich will mein Bestes tun, um Ihre Fragen zu beantworten ...“

Glen Gould

7 Conclusions and Outlook

We developed a prototype of a virtual reality geographic information system (VR-GIS) that simulates electromagnetic wave propagation for mobile phones and pagers. The system is called RA$_3$DIO, which stands for *Radio Antenna placement with 3-Dimensional Interactive Optimization*. It supports exploratory interaction, placement of base station transmitters, computation of essential radio network characteristics such as delay-spread in paging networks or co-channel interference in cellular radio networks, and interactive optimization of delay-spread and antenna positions. It has interfaces to antenna description formats of major manufacturers, to databases, and to other valuable tools.

Nowadays, VR-GISs are able to visualize the earth in a large range of different scales. This range of scales passes all three scale groups of classical cartography. Therefore, the different assumptions about the shape of the earth for each group can only be conserved with additional effort. A more general approach is to use the most accurate assumption, thus the spheroidal assumption. Because the classical definition of a terrain is based on the Euclidean plane, we have introduced new definitions for spherical and spheroidal terrain models based on spherical and spheroidal domains, respectively.

The GIS kernel of RA$_3$DIO is able to handle spherical and spheroidal terrain data as well as classical terrain models. A VR framework in the background provides three-dimensional scene management with rendering capabilities. Such a 3-dimensional scene renderer is a key component for visualizing spherical and spheroidal terrain models without any map projection. However, classical map projections are still used because a lot of terrain models are based on projected coordinate systems.

In a global GIS different levels of detail are needed to show entire countries on one hand and details of cities on the other hand. Sophisticated concepts, such as multiresolution triangulations, help to retrieve the right terrain data needed in an efficient way. In this context we have presented a new adaptive hierarchical multiresolution triangulation based on the longest side bisection triangulation. This triangulation has the following nice properties:

- It only produces triangles whose smallest angles are always greater or equal to $2\alpha/3$, where α is the smallest angle of the initial triangle.
- All triangles produced belong to a finite number of similarity classes of triangles.

- The refinement always terminates in a finite number of steps with the construction of a matching triangulation.
- It satisfies the following smoothness condition: for any pair of adjacent triangles t_1, $t_2 \in \tau$ (τ is a matching triangulation) with diameters h_1 respectively h_2 it holds that $\min(h_1, h_2)/\max(h_1, h_2) \geq \delta > 0$, where δ only depends on the smallest angle of the initial triangulation.

Our new modified longest side bisection refinement rule has improved the lower bound of the smallest occurring angle from $\alpha/2$ to $2\alpha/3$. For interactive visualization of terrain surfaces it is important not to have small angles and thin triangles because of rendering artifacts.

Different levels of detail are important for all geometric objects in a global VR-GIS, not only because of performance reasons, but also because of visualization limitations. If we have to see very small objects far away, then these objects must be increased in size, but if we are only a few meters in front of them, then we want to see the real dimensions. Normally, objects and details which only improve the visual appearance are not modeled in three dimensions, because they need too much computing power. A much simpler way to provide visual details is the use of texture data based on imagery, e.g. aerial ortho-photos, digital pixel-maps, etc. The considerations about different levels of detail are also valid for these images. This means we need storage concepts supporting very fast progressive extraction of different levels of detail. With this goal in mind we have implemented a new graphics image file format, called PGF. PGF is based on fast integer discrete wavelet transform and makes use of either lossless or lossy compression. During its very fast decoding, it progressively provides images in higher resolutions. This is exactly what we need for texture mapping in RA_3DIO. The quality of a natural image compressed with PGF lies between JPEG and the newer and better JPEG 2000 for a fixed compression ratio. This advantage over JPEG is without the cost of additional encoding or decoding time in contrast to JPEG 2000. Fortunately, the compression ratios in PGF's lossless mode are also good for natural true-color images which makes PGF valuable for other areas as well.

Imagery helps the user to better recognize known terrain parts, but it does not influence the wave propagation. Modeling wave propagation in a fast and accurate way for ultra high frequencies is major goal of RA_3DIO. Therefore, we have presented several issues related to wave propagation and its integration into a VR-GIS. Classical wave propagation prediction is based on two-dimensional maps, therefore antenna suppliers only measure and provide two dimensional sections of the three dimensional antenna characteristic. Of course, in a 3D VR-GIS the full 3D antenna characteristics are necessary. Therefore, we proposed three different approaches to model a 3D antenna characteristic based on the horizontal and the vertical radiation patterns.

We have also presented a number of known empirical wave propagation models and their implementation. Most of them have been designed for flat terrains, so their accuracy is restricted to terrain parts visible from the antennas and not beyond the radio horizon. We have introduced new definitions for radio visibility and discussed several algorithms to compute classical and radio visibility. Some of these algorithms are based on the computation of the horizon. This is helpful, because the radio horizon is related to the visible horizon of a curved surface. Furthermore, we have checked the applicability of these algorithms for spherical and spheroidal terrain models and their efficiency in combination with a tiling concept used in the dynamic scene management of RA_3DIO. Empirical wave propagation models are often combined with diffraction loss models. Most of these diffraction loss models analyze the terrain profile between the transmitter and the receiver. We have presented one of them in greater detail and have discussed a simple implementation.

In RA_3DIO the user can interactively place base station transmitters and watch the wave propagation prediction in real-time. Unfortunately, finding a good distribution of the base stations is a hard task. Sometimes the user wants to cover a part of a terrain with a minimum number of antennas or a set of base stations with minimum cost. Based on these objectives we formulated a new version of the base station transmitter (BST) location problem. Although our version only concerns visibility and RF design objectives, it is not solely interesting from a theoretical point of view. More practical results as well as implementations of greedy heuristics and Simulated Annealing are part of this work, while several interesting theoretical results, for instance the *NP*-hardness of a version of our BST location problem, have been presented in [Eid00].

An integration of capacity and teletraffic objectives into RA_3DIO is possible. To do that all test points need to be replaced by demand nodes. A demand node denotes a position on the terrain with a certain teletraffic demand. This teletraffic demand can be static or variable over different time scales (daily, weekly, yearly). Optimal network planning needs to be aware of such variable teletraffic distributions. Furthermore, a base station transmitter has only a fixed number of teletraffic channels, so it cannot cover more demand nodes then its available capacity. In areas with large teletraffic demand, like in dense cities, coverage is rarely the problem, rather capacity is the real limitation. This means, there are small micro- and picocells with reduced power necessary to cover this large teletraffic demand. Unfortunately, the number of micro- and picocells cannot be arbitrarily increased because the total number of carrier channels is fixed and because cells using the same channels must be far apart from each another. This distance between cells using the same frequencies must be considered, too. The way this is usually done is to introduce an interference measure, e.g. co-channel interference. A somewhat more sophisticated radio network optimization needs to take all these considerations into account.

In simulcast or quasi-synchronous radio networks another interference problem is of interest: the problem occurs if radio signals from different transmitters with almost equal power but large different delays reach a mobile station. The radio signal delay consists of three parts. First, a delay from a point source, e.g. a satellite, to a base station, second a (artificial) delay in the base station, and last a delay from the base station to a mobile station. Based on this, we have defined a new optimization problem, called the DELAY SPREAD problem. It seems that DELAY SPREAD is *NP*-hard, but we have not yet proven it. We have also presented a greedy heuristic and a more promising implementation of a Simulated Annealing algorithm.

RA$_3$DIO has been useful in improving the BST location problem and in decreasing the DELAY SPREAD problem. For exact solutions an external solver and appropriate interfaces are necessary. Such interfaces have been implemented, tested and used for large problem instances. We also studied and solved the problem of connecting two different software packages. We linked RA$_3$DIO and the dynamical statistical graphics package XGobi. The users of RA$_3$DIO can now enjoy seeing additional statistical data in XGobi and analyze it with a standard package. At the present time, it is not necessary that every software package provides every useful function by itself. Instead, it is more economical to provide an open interface to off-the-shelf software packages so that other programs can use their special features. If the specification of the interface is public, a program can conquer new applications years after the programmer wrote the last line of its source code – even if the source code is not available to others. To link the two software packages, RA$_3$DIO and XGobi, we used a method of close coupling because the data throughput is a key component in a virtual reality system. Remote Procedure Calls are our chosen method of close coupling. Our work has shown that it is an advantage to think about middleware already at the design stage. Many programmers still build huge monolithic systems that do not provide connections to the outside world. It should be of a greater interest to design interfaces of open systems and to connect them to program specific functionality.

Color Plates

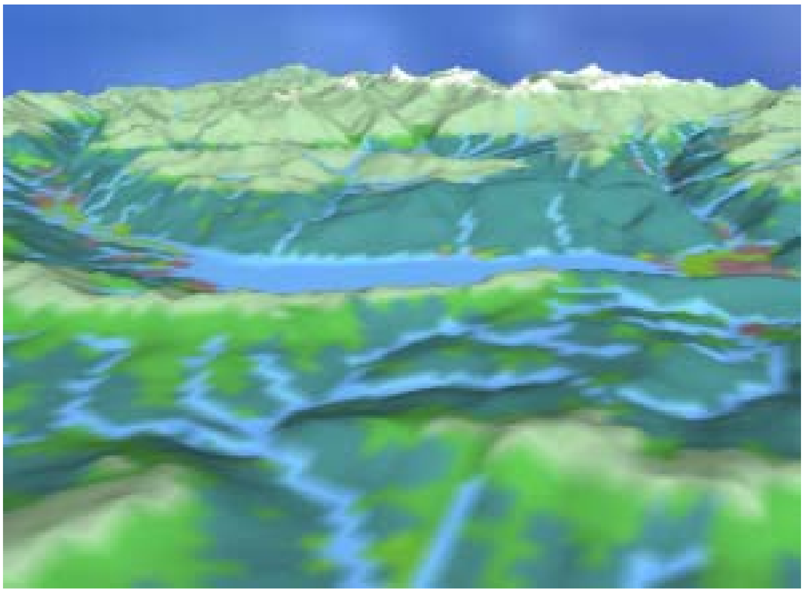


Fig. 29: DEM with hypsometric and geomorphologic colors near Interlaken [DHM©], [VEC©].

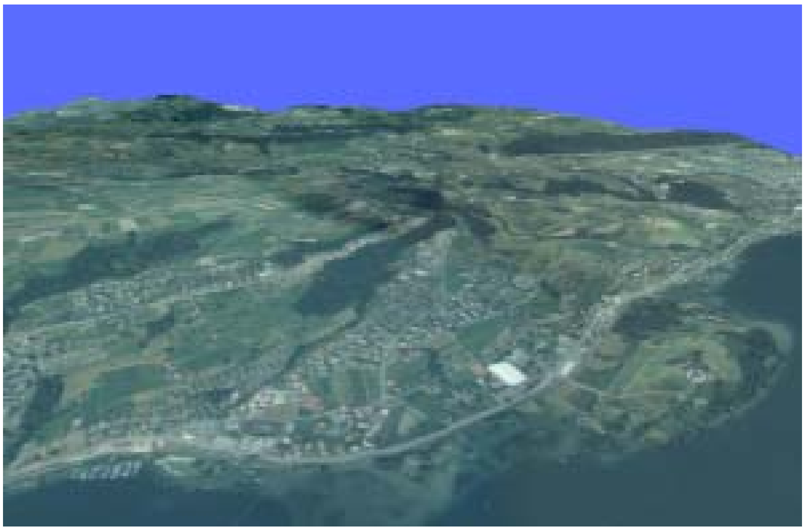


Fig. 30: DEM with aerial ortho-photos near Au [DEM©], [IMAGE©].

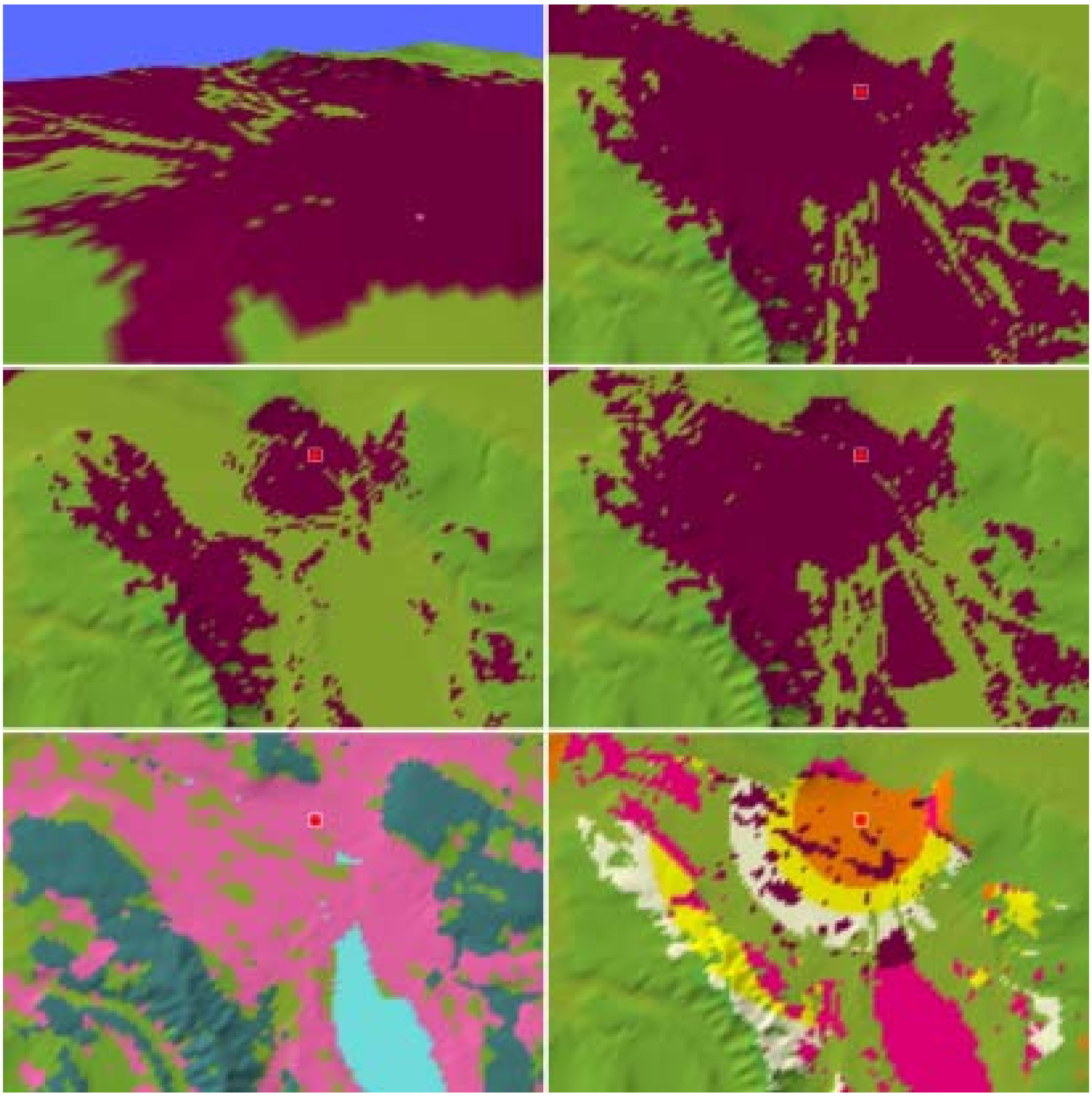


Fig. 31: Different visibility concepts [DHM©], [STAT©]: (a) perspective view of region near Zurich, (a, b) dark red regions are in line-of-sight of a viewpoint 30 meters above the terrain (earth curvature considered), (c) vertical approximation of radio visibility at 400 MHz (path clearance 60% of the first Fresnel zone, mobile station height = 2 m), (d) vertical approximation of radio visibility at 1800 MHz, (e) geomorphologic data (pink: urban, light blue: water, dark green blue: forest), (f) approximation of θ-radio visibility using a modified Hata wave propagation model within the line-of-sight region with θ equal to –99 dBm (antenna power = 50 dBm, carrier frequency = 1800 MHz).

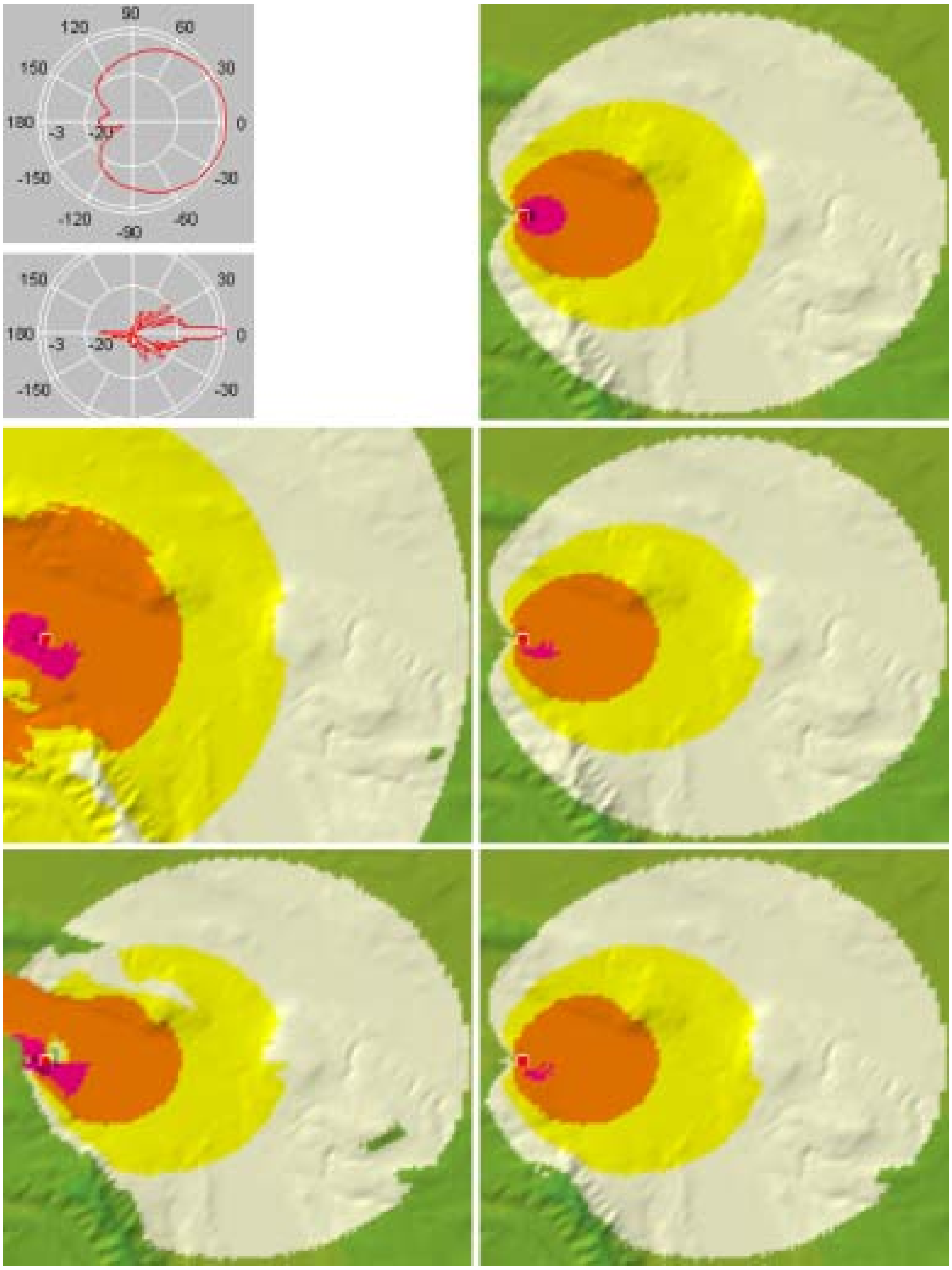


Fig. 32: Different approximations of 3D radiation patterns [DHM©]: (a) horizontal and vertical radiation pattern, (b) solely horizontal pattern, (c) solely vertical pattern, (d) minimum of both gains, (e) vertical pattern interpolation, (f) horizontal pattern scaled with the appropriate value of the vertical pattern.

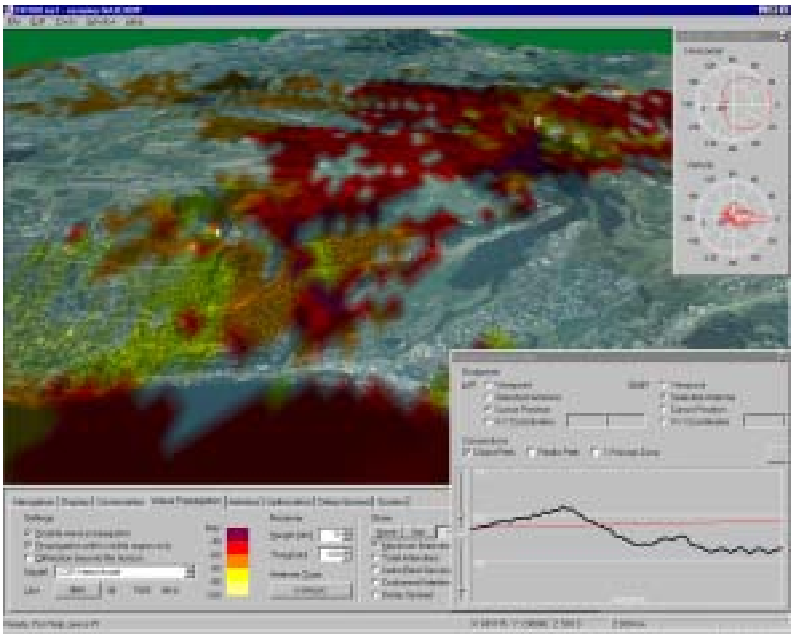


Fig. 33: Wave propagation prediction of two base stations in the 1800 MHz band [DEM©], [IMAGE©], [STAT©].

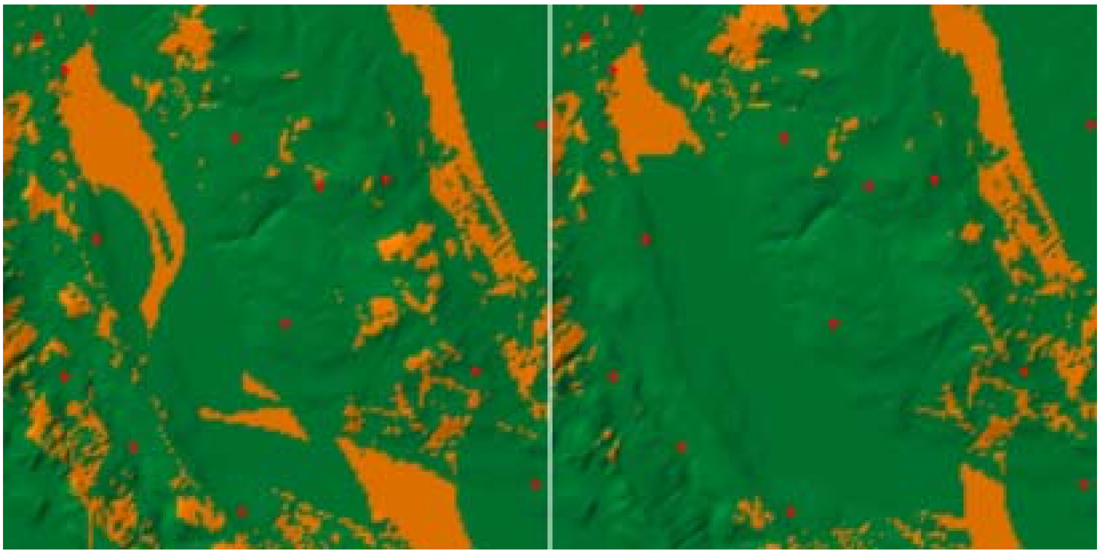


Fig. 34: Delay-spread reduction of 36% near Zurich [DEM©], [STAT©].

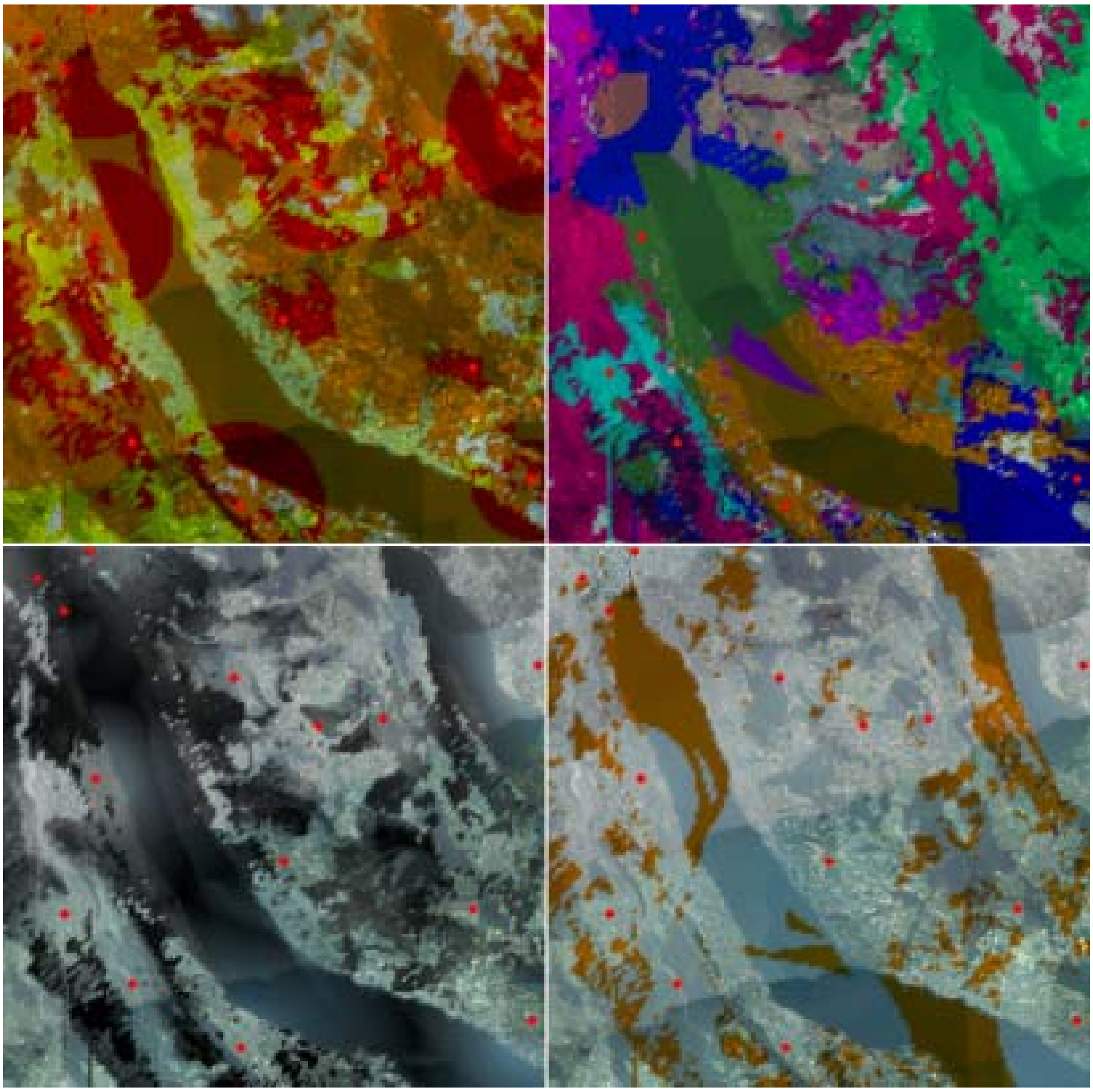


Fig. 35: Wave propagation prediction near Zurich in the 466 MHz band [DEM©], [IMAGE©], [STAT©]: (a) maximum intensity, (b) cells with best server, (c) co-channel interference, (d) delay-spread without optimization.

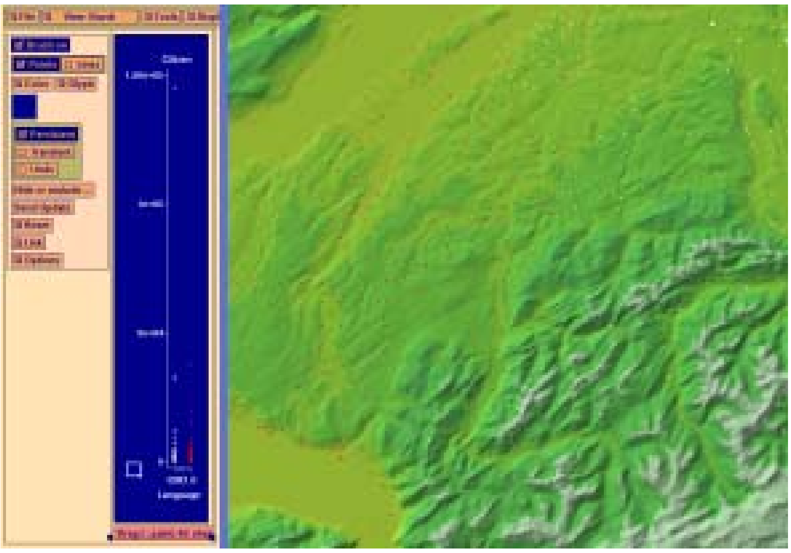


Fig. 36: Bidirectional link between RA$_3$DIO and XGobi [DHM©], [STAT©]. Communities with a majority of French or German speaking people are visualized with red or white dots, respectively.

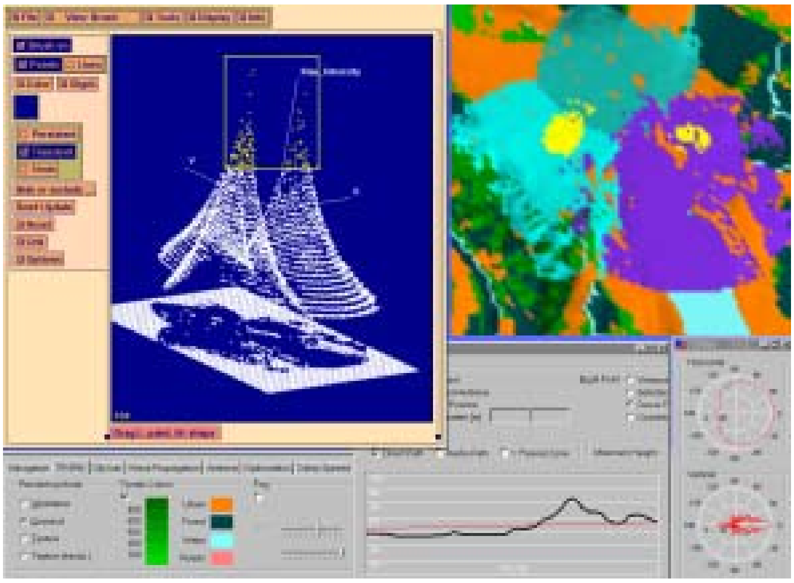


Fig. 37: Bidirectional link between RA3DIO and XGobi at work [DHM©], [STAT©].

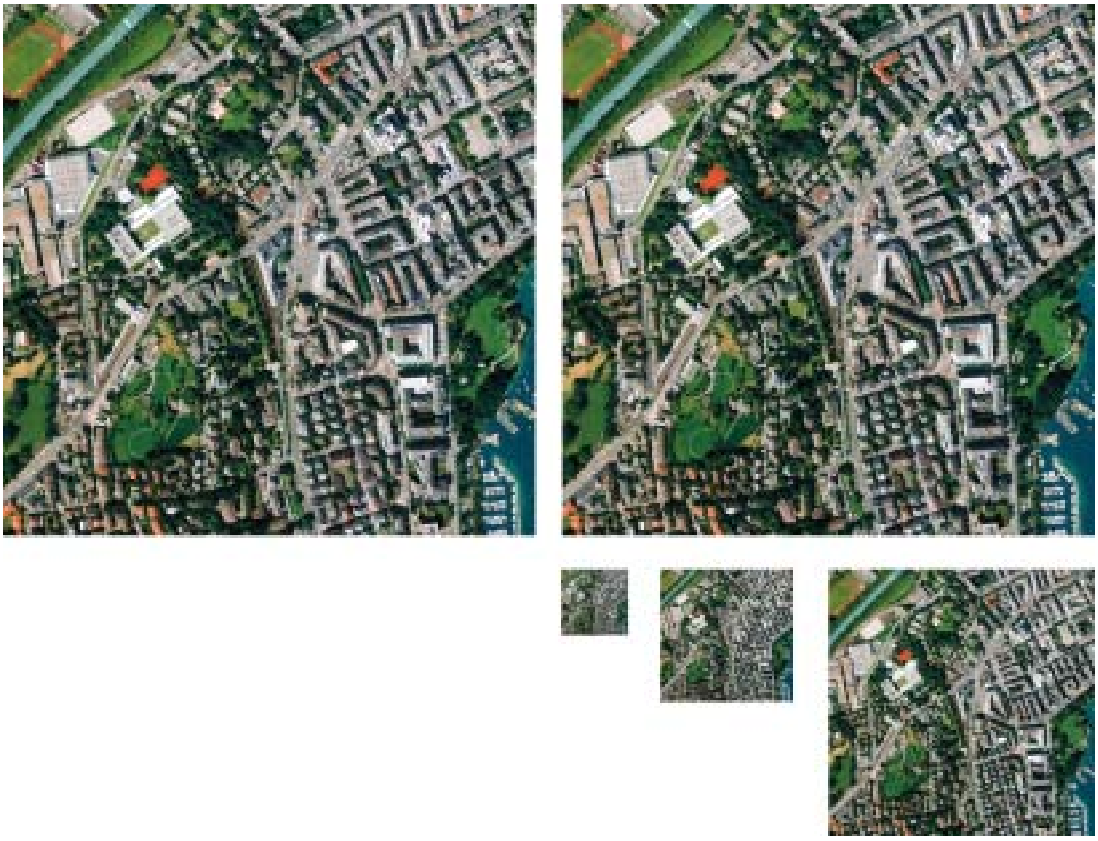


Fig. 38: Progressively loaded PGF ortho-photo (the four images in the right half) and its original bitmap (top left image). The compression ratio is 1:7.

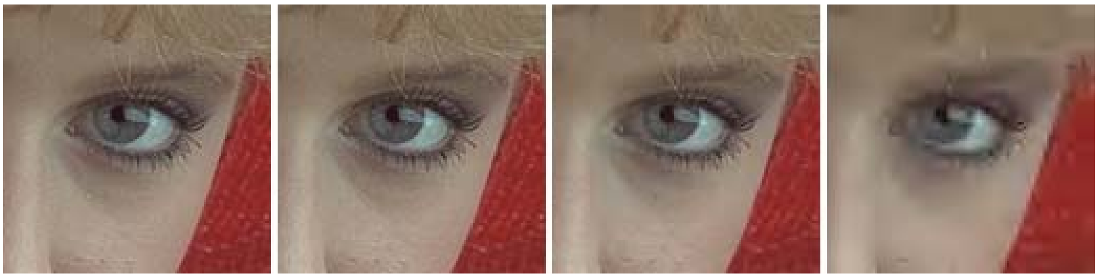


Fig. 39: PGF images with different compression ratios: (a) original, (b) 1:7, (c) 1:24, (d) 1:99.

Copyrights

[DHM©]	Digitales Höhenmodell DHM25 © 1999 Bundesamt für Landestopographie, Wabern, Reproduktionsbewilligung DV1455, 2.11.1999.
[VEC©]	Vektordaten VECTOR200 Version 85 © 1999 Bundesamt für Landestopographie, Wabern, Reproduktionsbewilligung DV1455, 2.11.1999.
[IMAGE©]	Luftbilder des Kantons Zürich © 1997 Endoxon AG, Swissphoto AG, Regensdorf, Reproduktionsbewilligung 12.4.2001.
[STAT©]	Arealstatistik der Schweiz 79/85, Volkszählung 1990, Kirchspitzkoordinaten 1998 © 2000 Bundesamt für Statistik, GEOSTAT, Neuenburg, Lizenznummer BFS01G00580, 20.7.2000.

References

[ABMD92] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies. Image Coding using the Wavelet Transform. In *IEEE Transactions on Image Processing*, 1(2):205–220, 1992.

[AP77] K. Allsebrook, J. D. Parsons. Mobile radio propagation in British cities at frequencies in the VHF and UHF bands. IEEE Transactions on Vehicular Technology, 26(4): 313–323, 1977.

[Asi85] D. Asimov. The Grand Tour: A Tool for Viewing Multidimensional Data. *SIAM Journal on Scientific and Statistical Computing*, 6(1):128–143, 1985.

[Ata83] M. J. Atallah. Dynamic Computational Geometry. *IEEE Symposium on Foundations of Computer Science*, 24:92–99, 1983.

[BCS96] A. Buja, D. Cook, D. F. Swayne. Interactive High-Dimensional Data Visualization. *Journal of Computational and Graphical Statistics*, 5(1):78–99, 1996.

[Bea87] J. E. Beasley. An algorithm for set covering problem. *European Journal of Operational Research*, 31:85–93, 1987.

[Bea90] J. E. Beasley. A Lagrangian heuristic for the set covering problems. *Naval Research Logistics*, 37:151–164, 1990.

[Bec99] M. Beck. WorldView – Ein generisches Virtual Reality Framework für die interaktive Visualisierung grosser geographischer Datenmengen. Dissertation, wirtschaftswissenschaftliche Fakultät, Universität Zürich, 1999.

[BES+98] M. Beck, S. Eidenbenz, C. Stamm, P. Stucki, P. Widmayer. A Prototype System for Light Propagation in Terrains. In F. E. Wolter and N. M. Patrikalakis, editors, *Proceedings of Computer Graphics International*, 103–106. IEEE Computer Society, 1998.

[BFS] GEOSTAT, a subgroup of the Federal Office of Statistics. Internet: http://www.statistik.admin.ch/dienstle/elektron/dgeostat01.htm.

[BJ96] S. T. Bryson, S. Johan. Time Management, Simultaneity and Time-Critical Computation in Interactive Visualization Environments. In *Proceedings of Visualization '96*, 255–261, 1996.

[BK89] M. Blum, S. Kannan. Designing programs that check their work. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '89)*, 86–97, 1989.

[BS95] L. M. Bugayevskiy, J. P. Snyder. *Map Projections – A Reference Manual*. Taylor & Francis Ltd., 1995.

[Bul47] K. Bullington. Radio Propagation at Frequencies above 30 Megacycles. In *Proc. IRE*, 35(10):1122–1136, 1947.

[Bul77] K. Bullington. Radio Propagation for Vehicular Communications. In *IEEE Trans. on Vehicular Tech.*, VT-26(4):295–308, 1977.

[CCIR82] International Radio Consultative Committee. Propagation in non-ionized media. *Recommendations and Reports of the CCIR*, volume V, 253–268. ITU, Geneva, 1982.

[CF97] H. Chao, P. Fisher. An Approach to Fast Integer Reversible Wavelet Transforms for Image Compression. CompSci, 1996.

[CGKW97] P. Calégari, F. Guidec, P. Kuonen, D. Wagner. A Genetic Approach to Radio Network Optimization for Mobile Systems. In *Proceedings of the IEEE/VTS 47th Vehicular Technology Conference*, vol. 2:755–759, 1997.

[CJK+97] B. Chamaret, S. Josselin, P. Kuonen, M. Pizarroso, N. Salas-Manzanedo, D. Wagner. Radio network optimization with maximum independent set search. In *Proceedings of the IEEE/VTS 47th Vehicular Technology Conference*, vol. 2:770–774, 1997.

[CMSC96] D. Cook, J. J. Majure, J. Symanzik, N. Cressie. Dynamic Graphics in a GIS: Exploring and Analyzing Multivariate Spatial Data Using Linked Software. *Computational Statistics: Special Issue on Computeraided Analysis of Spatial Data*, 11(4):467–480, 1996.

[Col] Department of geography at University of Colorado. Internet: http://www.colorado.edu/geography/.

[COST91] COST 231. Urban transmission loss models for mobile radio in the 900 and 1800 MHz bands (revision 2). *COST 231 TD(90)119 Rev. 2*, The Hague, 1991.

[CS89] R. Cole, M. Sharir. Visibility Problems for Polyhedral Terrains. In *Journal of Symbolic Computation*, 7:11–30, 1989.

[CSMC97] D. Cook, J. Symanzik, J. J. Majure, N. Cressie. Dynamic Graphics in a GIS: More Examples Using Linked Software. *Computers and Geosciences: Special Issue on Exploratory Cartographic Visualization*, 23(4):371–385, 1997.

[Dau88] I. Daubechies. Orthonormal Bases of Compactly Supported Wavelets. *Comm. Pure Appl. Math.*, 41:909–996, 1988.

[dBer97] M. de Berg. Visualization of TINs. In M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, editors, *Algorithmic Foundations of Geographic Information Systems*, Summerschool, Udine, volume 1340 of Lecture Notes in Computer Science, 79–97. Springer-Verlag, 1997.

[dBD95] M. de Berg, K. Dobrindt. On levels of detail in terrains. In *11th Symposium on Computational Geometry*, C26–C27. ACM, 1995.

[DCE] The Open Group. The Open Group Portal to the World of DCE. Internet: http://www.opengroup.org/dce/.

[DeFM94] L. De Floriani, P. Magillo. Visibility Algorithms on Triangulated Digital Terrain Models. *International Journal of Geographic Information Systems*, 8(1):13–41, 1994.

[DeFM97] L. De Floriani, P. Magillo. Visibility Computations on Hierarchical Triangulated Terrain Models. Geoinformatica, 1(3):219–250, Kluwer Academic Publishers, Norwell, 1997.

[DeFM99] L. De Floriani, P. Magillo. Intervisibility on Terrains. Chapter 38 in *Geographic Information Systems: Principles, Techniques, Management and Applications*, Maguire, Goodchild, Longley, Rhind (Editors), John Wiley & sons, 543–556, 1999.

[DeFMP96] L. De Floriani, P. Marzano, E. Puppo. Multiresolution Models for Topographic Surface Description. *The Visual Computer*, 12(7):317–345, 1996.

[DeFMP99] L. De Floriani, P. Magillo, E. Puppo. Applications of Computational Geometry to Geographic Information Systems. Chapter 7 in *Handbook of Computational Geometry*, J.R. Sack, J. Urrutia (Editors), Elsevier Science, 333–388, 1999.

[DeFP95] L. De Floriani, E. Puppo. Hierarchical Triangulation for Multiresolution Surface Description. *ACM Transactions on Graphics*, 14(4):363–411, 1995.

[Del34] B. Delaunay. Sur la sphere vide. A la memoire de Georges Voronoi. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskih i Estestvennyh Nauk*, 7:793–800, 1934. Translated as Bull. Acad. Sci. USSR: Class. Sci. Math. Nat.

[Dey66] J. Deygout. Multiple Knife-Edge Diffraction of Microwaves. In *IEEE Transactions on Antennas and Propagation*, AP-14(4):480–489, 1966.

[DLLC85] G. Y. Delisle, J.-P. Levèvre, M. Lecours, J.-Y. Chouinard. Propagation Loss Prediction: A Comparative Study with Application to the Mobile Radio Channel. *IEEE Transactions on Vehicular Technology*, 34(2):86–96, 1985.

[DS93] R. P. Darken, J. L. Silbert. A Toolset for Navigation in Virtual Environments. In *Proceedings of the ACM Symposium of User Interface Software and Technology*, 157–165, 1993.

[Dor94] S. E. Dorward. A survey of object-space hidden surface removal. Int. Journal of Computational Geometry & Applications, 4(3):325–362, 1994.

[Dou86] D. H. Douglas. Experiments to Locate Ridges and Channels to Create a New Type of Digital Elevation Model. *Cartographica*, 23(4):29–61, 1986.

[EC95] H. Everett, D. Corneil. Negative Results on Characterizing Visibility Graphs. *Computational Geometry*, 5:51–63, 1995.

[ED69] R. Edwards, J. Durkin. Computer Prediction of Service Areas for VHF Mobile Radio Networks. In *IEE Proc.*, 116(9):1483–1500, 1969.

[Ede90] H. Edelsbrunner. An acyclicity theorem for cell complexes in d-dimensions. *Combinatorica*, 10:251–260, 1990.

[EE98] G. Eddon, H. Eddon. *Inside Distributed COM.* Microsoft Press, Redmond, WA, 1998.

[Eid00] S. Eidenbenz. (In-)Approximability of Visibility Problems on Polygons and Terrains. Ph. D. thesis, Institute of Computer Science, ETH Zurich, 2000.

[EP53] J. Epstein, D. W. Peterson. An experimental study of wave propagation at 850 Mc. In Proc. IRE, 41:595–611, 1953.

[Erv93] S. M. Ervin. Landscape Visualization with Emaps. *IEEE Computer Graphics and Applications*, 13(2):28–33, 1993.

[ESW98] S. Eidenbenz, C. Stamm, P. Widmayer. Positioning Guards at Fixed Height above a Terrain – An Optimum Inapproximability Result. *Lecture Notes in Computer Science*, 1461:187–198, 1998.

[ESW99] S. Eidenbenz, C. Stamm, P. Widmayer. Inapproximability Results for Guarding Polygons and Terrains. Accepted for publication in *Algorithmica*, 1999.

[Fab92] F. Fabris. Variable-length to variable-length source coding: a greedy step-by-step algorithm. In *IEEE Trans. Inform. Theory*, 38:1609–1617, 1992.

[Far96] S. Faruque. *Cellular Mobile Systems Engineering*. Artech House, Boston, 1996.

[FCD+95] A. Fournier, M. F. Cohen, T. D. DeRose, M. Lounsbery, L.-M. Reissell, P. Schröder, W. Sweldons. Wavelets and their Applications in Computer Graphics. *Course Notes*, SIGGRAPH '95, 1995.

[Fek90] G. Fekete. Rendering and Managing Spherical Data with Sphere Quadtrees. In *Proceedings of Visualization '90*, 1990.

[FM97] L. C. P. Floriani, G. R. Mateus. An Optimization Model for The BST Location Problem in Outdoor Cellular and PCS Systems. In *Proceedings of the 15th International Teletraffic Congress IT*C, 1997.

[FTL95] T. Fritsch, K. Tutschku, K. Leibnitz. Field Strength Prediction by Ray-Tracing for Adaptive Base Station Positioning in Mobile Communication Networks. In *Proceedings of the 2nd ITG Conference on Mobile Communication*, 1995.

[FvW96] S. Fortune, C. van Wyk. Static analysis yields efficient exact integer arithmetic for computational geometry. *ACM Transactions on Graphics*, 15:223–248, 1996.

[GG79] D. Gomez, A. Guzman. Digital model for three-dimensional surface representation. *Geoprocessing*, 1:53–70, 1979.

[GG97] V. Gaede, O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 1997.

[GGS95] M. H. Gross, R. Gatti, O. Staadt. Fast Multiresolution Surface Meshing. In *Proceedings of Visualization '95*, 135–142, 1995.

[GHW92] M. F. Goodchild, R. P. Haining, S.Wise. Integrating GIS and Spatial Data Analysis: Problems and Possibilities. *International Journal of Geographical Information Systems*, 6(5):407–423, 1992.

[Gra95] A. Graps. An Introduction to Wavelets. In *IEEE Computational Science and Engineering*, 2(2), Los Alamitos, 1995.

[GRV00] C. Glasser, S. Reith, H. Vollmer. The Complexity of Base Station Positioning in Cellular Networks. *Workshop on Approximation and Randomized Algorithms in Communication Networks (ARACNE)*, Geneva, Switzerland, 2000.

[GS92] M. F. Goodchild, Y. Shiren. A Hierarchical Spatial Data Structure for Global Geographic Information Systems. *CVGIP: Graphical Models and Image Processing*, 54(1):31–44, 1992.

[Gue98] O. Günther. *Environmental Information Systems*. Springer, Berlin, Heidelberg, 1998.

[GW97] T. Grossman, A. Wool. Computational experience with approximation algorithms for the set covering problem. *Euro. J. Operational Research*, 101(1):81–92, August 1997.

[Hat80] M. Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE Transactions on Vehicular Technology*, 29(3):317–325, 1980.

[HB80] A. Ho, E. Balas. Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study. *Mathematical Programming*, 12:37–60, 1980.

[Her89] J. Hershberger. Finding the Upper Envelope of n Line Segments in O(n log n) Time. *Information Processing Letters*, 33:169–174, 1989.

[HG97] P. S. Heckbert, M. Garland. Survey of polygonal surface simplification algorithms. In *SIGGRAPH 97 Course Notes 25*. ACM SIGGRAPH, 1997.

[HHK94] J.-T. Horng, W.-C. Huang, C.-Y. Kao. A genetic algorithm approach for set covering problems. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, 569–574, 1994.

[HP99] J. M. Hernando, F. Pérez-Fontán. *Introduction to Mobile Communications Engineering*, Artech House Publishers, Boston/London, 1999.

[Hop96] H. Hoppe. Progressive meshes. In *Proceedings SIGGRAPH 96*, 99–108. ACM SIGGRAPH, 1996.

[HSW88] A. Hutflesz, H.-W. Six, P. Widmayer. Globally order preserving multidimensional linear hashing. *Proceedings 4th International Conference on Data Engineering*, Los Angeles, 572–579, 1988.

[HU94] H. M. Hearnshaw, D. J. Unwin, editors. *Visualization in Geographical Information Systems*. John Wiley & Sons, Chichester, 1994.

[IL97] L. J. Ibbetson, L. B. Lopes. An automatic base site placement algorithm. In *Proceedings of the IEEE/VTS 47th Vehicular Technology Conference*, vol. 2:760–764, 1997.

[IPD83] M. F. A. Ibrahim, J. D. Parsons, D. E. Dadson. Signal Strength Prediction in Urban Areas Using Topographical and Environmental Databases. IEEE Conference ICC 83, 1983.

[IYTU84] F. Ikegami, S. Yoshida, T. Takeuchi, M. Umehira. Propagation Factors Controlling Mean Field Strength on Urban Streets. *IEEE Transactions on Antennas and Propagation*, AP-32(8):822–829, 1984.

[JB92] K. Jornsten, J. E. Beasley. Enhancing an algorithm for set covering problems. *European Journal of Operational Research*, 58:293–300, 1992.

[Joh74] D. Johnson. Approximation Algorithms for Combinatorial Problems. *Journal Comput. System Sci.*, 9:256–278, 1974.

[JRMI] Java RMI Tutorial. Internet: http://java.sun.com/docs/books/tutorial/rmi/.

[KCW93] T. Kürner, D. J. Cichon, W. Wiesbeck. Concepts and Results for 3D Digital Terrain-Based Wave Propagation Models: An Overview. In *IEEE J. on Selected Areas in Communications*, 11(7):1002–1012, 1993.

[Kel62] J. B. Keller. Geometrical Theory of Diffraction. In *J. Opt. Soc. America*, 52(2):116–131, 1962.

[Ket99] L. Kettner. Software Design in Computational Geometry and Contour-Edge Based Polyhedron Visualization. PhD thesis, Institute of Computer Science, ETH Zurich, 1999.

[KLR+95] D. Koller, P. Lindstrom, W. Ribarsky, L. F. Hodges, N. Faust, G. A. Turner. Virtual GIS: A real-time 3D geographic information system. In *Proceedings Visualization 95*, 94–100, IEEE Computer Society Press, Los Alamitos, Calirfornia, 1995.

[KOS91] M. J. Katz, M. H. Overmars, M. Sharir. Efficient hidden surface removal for objects with small union size. In *Proceedings 7th ACM Symposium on Computational Geometry*, 31–40, ACM Press, New York, 1991.

[KP74] R. G. Kouyoumjian, P. H. Pathak. A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. In *Proc. IEE-E*, 62(11):1448–1461, 1974.

[KS96] M. A. Koschat, D. F. Swayne. Interactive Graphical Methods in the Analysis of Customer Panel Data (with Discussion). *Journal of Business and Economic Statistics*, 14(1):113–132, 1996.

[Lan83] G. G. Langdon, Jr. An adaptive run-length encoding algorithm. In *IBM Tech. Discl. Bull.*, 26:3783–3785, 1983.

[Lee91] J. Lee. Comparison of existing methods for building triangular irregular network models of terrain from grid digital elevation models. In *International Journal of Geographical Information Systems*, 5:267–285, 1991.

[Lee97] W. C. Lee. *Mobile Communications Engineering*, 2nd edition, McGraw-Hill, New York, 1997.

[LKR+96] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, G. A. Turner. Real-Time, Continuous Level of Detail Rendering of Height Fields. In *Proceedings SIGGRAPH 96*, 109–118, 1996.

[LKR+97] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, A. Op den Bosch, N. Faust. An Integrated Global GIS and Visual Simulation System. *Technical report GIT-GVU-97-07*, 1997.

[LL94] L. A. Lorena, F. B. Lopes. A surrogate heuristic for set covering problems. EJOR, 79:138–150, 1994.

[LT92] R. Laurini, D. Thompson. *Fundamentals of Spatial Information Systems*, Academic Press, London, 1992.

[Mac96] R. C. V. Macario, editor. *Modern Personal Radio Systems*. IEEE, London, 1996.

[Mal92] D. H. Maling. *Coordinate Systems and Map Projections*, 2nd Edition. Pergamon Press, Oxford, 1992.

[Mal99] H. S. Malvar. Fast Progressive Wavelet Coding. *Proceedings IEEE DCC'99*, 1999.

[McC93] S. C. McConnell. *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press International, 1993.

[McD79] V. H. MacDonald. The cellular concept. *Bell System Technical Journal*, 58:15–49, 1979.

[Mey97] S. Meyers. *Effective C++: 50 Specific Ways to Improve Your Programs and Design*. Addison Wesley Publishing Company, 1997.

[MHI62] G. Millington, R. Hewitt, F. S. Immirzi. Double knife-edge diffraction in field-strength predictions. In *IEE Monograph*, 507E:419–429, 1962.

[Mil88] V. J. Milenkovic. *Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic*. PhD thesis, Carnegie Mellon University, 1988.

[MN99] K. Mehlhorn, S. Näher. The Dangers of Floating Point Arithmetic. In *LEDA – A platform for combinatorial and geometric computing*, 609–612, Cambridge University Press, 1999.

[Mol87] F. Molinet. Geometrical Theory of Diffraction (GTD). In *IEEE AP Newsletter*, 6–17, Aug. 1987.

[MS95] N. D. Memon, K. Sayood. Lossless image compression: A comparative study. In *Proceedings Conference on Electronic Imaging*, 2418:8–20. SPIE, 1995.

[Nag94] G. Nagy. Terrain Visibility. *Computer & Graphics*, 18(6):763–773, 1994.

[NH95] A. N. Netravali, B. G. Haskell. *Digital Pictures: Representation, Compression and Standards*. Plenum Press, New York and London, second edition, 1995.

[NHS84] J. Nievergelt, H. Hinterberger, K. C. Sevcik. The grid file: an adaptable, symmetric multikey file structure. *ACM Transactions on Database Systems*, 9(1):38–71, 1984.

[NSTN93] T. Nishita, T. Sirai, K. Tadamura, E. Nakamae. Display of the Earth Taking into Account Atmospheric Scattering. In *Computer Graphics Proceedings*, 175–182, ACM SIGGRAPH, 1993.

[NW97] J. Nievergelt, P. Widmayer. Spatial data structures: Concepts and design choices. In M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, editors, *Algorithmic Foundations of Geographic Information Systems*, Summerschool, Udine, volume 1340 of Lecture Notes in Computer Science, 153–197. Springer-Verlag, 1997.

[OOKF68] Y. Okumura, E. Ohmori, T. Kawano, K. Fukuda. Field strength and its variability in VHF and UHF land-mobile radio-service. *Rev. Elec. Commun. Lab*. 16(9–10):825–873, 1968.

[OPR96] R. Otte, P. Patrick, M. Roy. *Understanding CORBA: The Common Object Request Broker Architecture*. Prentice Hall, Upper Saddle River, NJ, 1996.

[OW96] P. Ottmann, P. Widmayer. *Algorithmen und Datenstrukturen*. 3. überarbeitete Auflage, Spektrum Akademischer Verlag, 1996.

[Paj98] R. Pajarola. Access to large scale Terrain and Image Databases in Geoinformation Systems. Ph. D. thesis, Institute of Computer Science, ETH Zurich, 1998.

[Paj98a] R. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. In *Proceedings Visualization 98*, Los Alamitos, California. IEEE Computer Society Press, 1998.

[PM92] W. B. Pennebaker, J. L. Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1992.

[POS+98] R. Pajarola, T. Ohler, P. Stucki, K. Szabo, P. Widmayer. The Alps at your Fingertips: Virtual Reality and Geoinformation Systems. In *Proceedings 14th International Conference on Data Engineering, ICDE*, 550–557, 1998.

[PS97] E. Puppo, R. Scopigno. Simplification, LOD and Multiresolution Principles and Applications. *Eurographics '97 Tutorial*, 16(3), 1997.

[RDH+93] R. Rost, J. Dozier, B. Hibbard, P. Kochevar, L. Treinish, T. von Sant. Visualizing Planet Earth. *Course Notes 71*, ACM SIGGRAPH '93, 1993.

[Res96] J. Rese. Lagrange- und Surrogate Heuristiken zur Lösung großer Set-Covering Probleme. Diplomarbeit, Fachbereich Mathematik/Informatik, Universität Paderborn, 1996.

[Riv93] M.-C. Rivara. A discussion on mixed (longest-side midpoint insertion) Delaunay techniques for the triangulation refinement problem. In *Proceedings 5th Canadian Conf. Comp. Geometry*, 42–47, 1993.

[RROS96] C. R. Reeves, V. J. Rayward-Smith, I. H. Osman, G. D. Smith. *Modern Heuristic Search Methods*. John Wiley and Sons, 1996.

[RS75] I. G. Rosenberg, F. Stenger. A lower bound on the angles of triangles constructed by bisection the longest side. *Mathematics of Computation*, 29(130):390–395, 1975.

[Sal98] D. Salomon. *Data compression: the complete reference*. Springer-Verlag, New York, 1998.

[Sam84] H. Samet. The quadtree and related hierarchical data structures. *Computing Surveys*, 16(2):187–260, 1984.

[Sam89] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison Wesley Publ., Reading, 1989.

[SB66] J. W. Shwartz, R. C. Baker. Bit-plane encoding: a technique for source encoding. In *IEEE Trans. Aerospace Electron. Syst.*, 2:385–392, 1966.

[SCB98] D. F. Swayne, D. Cook, A. Buja. XGobi: Interactive Dynamic Graphics in the X Window System. *Journal of Computational and Graphical Statistics*, 7(1):113–130, 1998.

[Sch99] A. Schill. Distributed Platforms. *Encyclopedia of Microcomputers*, 22:97–116, 1999.

[SEA+00] S. Santa Cruz, T. Ebrahimi, J. Askelof, M. Larsson, C. Christopoulos. An analytical study of JPEG 2000 functionalities. In *Proceedings of SPIE of the 45th annual SPIE meeting, Applications of Digital Image Processing XXIII*, vol. 4115, 2000.

[SEP98] C. Stamm, S. Eidenbenz, R. Pajarola. A Modified Longest Side Bisection Triangulation. In *Proceedings 10th Canadian Conf. Comp. Geometry*, 1998.

[Sha93] J. M. Shapiro. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. In *IEEE Transactions on Signal Processing*, 41(12): 3445–3462, 1993.

[She92] T. Shermer. Recent Results in Art Galleries. In *Proceedings of the IEEE*, 1992.

[Siw98] K. Siwiak. *Radiowave Propagation and Antennas for Personal Communications*. Artech House, Boston/London, 2nd edition, 1998.

[SMCM97] J. Symanzik, J. J. Majure, D. Cook, I. Megretskaia. Linking ArcView 3.0 and XGobi: Insight Behind the Front End. *Technical Report 97–10*, Department of Statistics, Iowa State University, Ames, Iowa, 1997.

[SMMC97] J. Symanzik, I. Megretskaia, J. J. Majure, D. Cook. Implementation Issues of Variogram Cloud Plots and Spatially Lagged Scatterplots in the Linked ArcView 2.1 and XGobi Environment. *Computing Science and Statistics*, 28:369–374, 1997.

[Sou91] D. A. Southard. Piecewise Planar Surface Models from Sampled Data. *Scientific Visualization of Physical Phenomena*, 667–680, 1991.

[SP96] A. Said, W. A. Pearlman. A new, fast, and different image codec based on set partitioning in hierarchical trees. In *IEEE Transactions on Circuits and Systems for Video Tech.*, 6:243–250, 1996.

[SS92] R. Sivan, H. Samet. Algorithms for constructing quadtree surface maps. In *Proceedings 5th Int. Symposium on Spatial Data Handling*, 361–370, 1992.

[SSSW00] M. Schneider, C. Stamm, J. Symanzik, P. Widmayer. Virtual reality and dynamic statistical graphics: A bidirectional link in a heterogenous, distributed computing environment. In *Proceedings International Conference on Parallel and Distributed Processing Techniques and Applications*, PDPTA, 4:2345–2351, 2000.

[Stü96] G. L. Stüber. Principles of Mobile Communication. Kluwer Academic Publishers, Boston, 1996.

[Tom90] C. D. Tomlin. *Geographic Information Systems and Cartographic Modeling*. Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

[TL96] K. Tutschku, K. Leibnitz. Fast Ray-Tracing for Field Strength Prediction in Cellular Mobile Network Planning. *IEEE Vehicular Technology Conference*, 1996.

[Tut99] K. Tutschku. Models and Algorithms for Demand-oriented Planning of Telecommunication Systems. Ph. D. thesis, Institute of Computer Science, University of Würzburg, 1999.

[VDo94] Van Dooren, G. A. J. A Desterministic Approach to the Modeling of Electromagnetic Wave Propagation in Urban Environments. University of Eindhofen, Ph. D. Thesis, 1994.

[vHB87] B. von Herzen, A. H. Barr. Accurate triangulations of deformed, intersecting surfaces. *Computer Graphics*, 21(4):103–110, 1987.

[vKre97] M. van Kreveld. Digital Elevation Models and TIN Algorithms. In M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, editors, *Algorithmic Foundations of Geographic Information Systems*, Summerschool, Udine, volume 1340 of Lecture Notes in Computer Science, 37–78. Springer-Verlag, 1997.

[VBL95] J. D. Villasenor, B. Belzer, J. Liao. Wavelet Filter Evaluation for Image Compression. *IEEE Transactions on Image Processing*, 1995.

[WB88] J. Walfish, H. L. Bertoni. A theoretical Model of UHF Propagation in Urban Environments. *IEEE Transactions on Antennas and Propagation*, AP-36(12):1788–1796, 1988.

[WGS] WGS 84. Internet: http://www.wgs84.com/.

[WGZ97] S. G. Wolf, R. Ginosar, Y. Y. Zeevi. Spatio-chromatic image enhancement based on a model of human visual information processing, 1997.

[Xia93] H. H. Xia, et. al. Radio Propagation Characteristics for Line of Sight Microcellular and Personal Communications. *IEEE Trans. on Antennas and Propagation*, 41(10):1439–1447, 1993.

[YST00] Q. Yang, J. P. Snyder, W. R. Tobler. *Map Projection Transformation; Principles and Applications.* Taylor & Francis Ltd., 2000.

Curriculum Vitae

Christoph Stamm

born on December 17, 1968 in Schleitheim, Switzerland

Education

1975–1985	Primary and secondary school in Schleitheim and Schaffhausen
1985–1989	Apprenticeship in cartography at Orell Füssli, Graphische Betriebe, Zurich
1989–1992	Student in Technical Computer Science at HTL Brugg/Windisch (FH Aargau) dipl. Informatik-Ing. HTL (B. Sc. in Technical Computer Science)
1992–1996	Student in Computer Science at ETH Zurich dipl. Informatik-Ing. ETH (M. Sc. in Computer Science)
1995–1997	Student in Teaching Computer Science at ETH Zurich Didaktischer Ausweis in Informatik (certificate for teaching computer sciences in upper secondary schools and colleges)
1996–2001	PhD student at Institute of Theoretical Computer Science, ETH Zurich Examiner: Prof. Dr. Peter Widmayer Co-Examiner: Prof. Dr. Werner Bächtold

Employment

1989	Cartographer at Orell Füssli, Zurich
1992–1995	System Software Engineer at Union Bank of Switzerland, Zurich
1995–2000	Teaching in computer science at Technikerschule der Grafischen Industrie (TGZ), Zurich
1996	Teaching in digital technology at HTL Brugg/Windisch (FH Aargau)
1996–2001	Research and Teaching Assistant at Computer Science Department, ETH Zurich
2000–present	CEO xeraina GmbH

Projects

- working in the CoSy project (Control System for DEC computer systems) at UBS Zurich
- leading the WorldView project (real-time 3D visualization of very large scale terrain data) at ETH Zurich
- leading the RA$_3$DIO project (radio signal coverage prediction in terrains) at ETH Zurich
- supervising several master thesis in computer science at ETH Zurich

Publications

- S. Eidenbenz, C. Stamm, P. Widmayer. Inapproximability Results for Guarding Polygons and Terrains. *Algorithmica*, 31, 79–113, 2001.
- S. Eidenbenz, C. Stamm. Maximum Clique and Minimum Clique Partition in Visibility Graphs. *Lecture Notes in Computer Science* (IFIP TCS 2000), 2000.
- M. Schneider, C. Stamm, J. Symanzik, P. Widmayer. Virtual reality and dynamic statistical graphics: A bidirectional link in a heterogeneous, distributed computing environment. *Proceedings PDPTA*, 4, 2345–2351, 2000.
- M. Beck, S. Eidenbenz, C. Stamm, P. Stucki, P. Widmayer. Worldview: A Virtual Reality Framework for the Design, Optimization and Management of Mobile Telematics Infrastructure. *SI Informatik/Informatique*, Nr. 3, Juni 1999.
- S. Eidenbenz, C. Stamm, P. Widmayer. RA$_3$DIO – Wellenausbreitung in 3D. *Computerworld Schweiz*, 1999.
- S. Eidenbenz, C. Stamm, P. Widmayer. Inapproximability of Some Art Gallery Problems. *Proceedings 10th CCCG*, 1998.
- S. Eidenbenz, R. Pajarola, C. Stamm. A Modified Longest Side Bisection Triangulation. *Proceedings 10th CCCG*, 1998.
- S. Eidenbenz, C. Stamm, P. Widmayer. Positioning Guards at Fixed Height above a Terrain - an Optimum Inapproximability Result. *Lecture Notes in Computer Science 1461* (ESA'98), 187–198, 1998.
- M. Beck, S. Eidenbenz, C. Stamm, P. Stucki, P. Widmayer. A Prototype System for Light Propagation in Terrains. *Proceedings CGI*, 1998.

Index

A

achromatic 54
Alternative Greedy 105
analytical design approach 33
antenna
  characteristics 66
  radiation pattern 67
approximation algorithm 103
  goodness 103
  ratio 103
area type
  open 73
  rural *See open area type*
  suburban 73
  urban 73

B

back-faced 81
base distance 42
Bessel ellipsoid 17
Best Transmitter First 104
blocking edge 81
BST 100
  configuration 101

C

C/I *See co-channel interference*
cellular network design 32
cellular radio concept 27
chromatic 54
cluster 27
co-channel interference 28
compression
  lossless 52
  lossy 52
  quality driven 53
  speed driven 53
correctness 120
Cover 105
coverage 101

D

data
  geomorphologic 62
  line 18
  point 18
  polygon 18
  raster 62
  texture 51
  urban 96
  vector 62
database
  cache 134
  concept 134
  geographic 63
  management system 134
  range query 134
  spatial 62
  system 119
  terrain 21
Daubechies filter 55
DCE 137
DCS 29
DCT 53
Delaunay
  pyramid 43
  TIN 43
  triangulation 20
delay-spread 31
  area 109
  computation 114
  minimization 107

- problem 108
DEM 20
demand node 35
Deygout 94
  - knife-edge 95
diffraction 70, 92
  - effects 73
  - knife-edge model 73
  - loss 71, 93
  - multi-edge 91
  - multi-obstacle 92
  - rooftop-to-street 92
  - theory of 96
digital elevation model *See DEM*
distributed computing environment *See DCE*
DWT 53
dynamic scene management 21

E

earth curvature 39
effective
  - antenna height 72
  - power 70
  - radius 77
efficiency
  - of compression 55
  - of GISs 17
  - of personal communications networks 27
  - of program code 124
  - of software 122
  - of terrain servers 134
extensibility 124

F

field strength 70
first Fresnel zone 78
frequency
  - allocation 34
  - group 28
  - reuse 27
front-faced 82
front-to-back order 81
  - spheroidal 84

G

geodetic reference system *See* GRS
geographic information system *See* GIS
geographical coordinates 40
geometric algorithm 121
geomorphologic data 62
GIS 15
  - classical 16
GPRS 29
graphical user interface *See GUI*
Greedy 104
GRS 16
GSM 29
GUI 125

H

Hata model
  - CCIR 74
  - COST231 74
  - modified 75
height field 20
hidden surface removal *See HSR*
horizon 76
HSCSD 29
HSR 87

I

IDL 137
in front 81, 83
integer programming 100
integrated approach 34

interface definition language *See IDL*
interference minimization 106
ISDN 29
isolated mountain 73

J

JPEG 2000 53

K

knife-edge 93

L

land-usage type *See area type*
last mile 30
latitude 40
  auxiliary 41
  geocentric 41
  geodetic 41
level of detail *See LOD*
line-of-sight 30 *See also LOS*
link budget 70
linked brushing 135, 136
locations variability 66
LOD 22
longest side bisection 43, 44
  modified 45
  modified rule 50
longitude 40
LOS 36

M

main hill 95
main window 132
map data 16
Mercator projection 17
mixed sea-land path parameter 73
mobile communications 27
  network 26
modularity 124
morphographic data *See geomorphologic data*
mutually visible 76

N

network
  cellular 27
  cellular design 32
  design objectives 32
  irregular triangulated *See TIN*
  mobile communications 26
  planning tools 30
  radio 27
  radio link 27
  wired 27
NLOS 36
normalized radiation pattern 67

O

object carrier surface 19
objectives
  capacity and teletraffic 33
  network deployment 33
  network design 32
  RF design 33
obstruction parameter 93
  normalized 93
off-the-shelf components 123
optimization 99
  algorithm 100
  criteria 99
  on line-level 124
  problem *See problem*

P

paging 31
path clearance 93
path loss

  - diffraction 71
  - free space 69
- peak signal to noise ratio *See PSNR*
- personal communications 27
- PGF 53
- plane assumption 39
- precision 120
- problem
  - BST location 100
  - delay-spread minimization 107
  - maximal coverage location 101
  - minimum set cover 101
- progressive
  - image refinement 52
  - mesh refinement 50
- Progressive Graphics File *See PGF*
- propagation
  - analysis 33, 34
  - channel 35
  - loss *See path loss*
  - prediction models *See propagation models*
- propagation models
  - CCIR Hata 74
  - empirical 71
  - Okumura 72
  - path profile 92
  - physical-geometrical 90
  - ray-tracing 96
  - Walfish-Ikegami 90
- PSNR 55

R

- RA$_3$DIO 117
  - database 134
  - navigation 132
  - pre-processing 127
  - run-time system 128
  - system overview 125
  - terrain manager 131
  - user interface 132
- radiated power 70
- radiation pattern *See antenna*
- radio
  - frequency *See RF*
  - horizon 77
  - network analysis 34
  - network definition 33
  - visibility *See visibility*
- readability 124
- real RAM model 121
- reducibility 124
- refinement method 21
- reflection 70, 92
- refraction 93
- regular square grid *See RSG*
- remote procedure call *See RPC*
- rendering mode 133
- resolution
  - fixed 52
  - of grids 63
  - of height fields 20
  - scalable 52
- RF 32
- RGB 53
- Rivara refinement 45
- RMSE 55
- root mean square error *See RMSE*
- RPC 136
- RSG 77

S

- scene 129
  - graph 129
  - manager 21
  - map 22
- simplification method 21
- Simulated Annealing 111
- space distance 42
- spherical assumption 39

spheroid 38
spheroidal assumption 39
split propagation 50
strict hierarchical 43

T

terrain 20
  2½D 16
  cache 128
  data 16
  high-resolution 30
  mean slope 73
  model 20
  spherical 41
  spheroidal 41
  undulation 73
  visibility 76
  visualization 25
texture data 51
tile 22
tiling concept 23
  compatibility 25
  locality 24
  order 25
TIN 20
  acyclic 81
triangulated irregular network *See TIN*
triangulation 20
  adaptive hierarchical 43
  Delaunay 20
  hierarchical 43
  hierarchical Delaunay 43
  multiresolution 21
  restricted quadtree 23
  spherical 41
  spheroidal 41
triangulations
  adjacent 44
  matching 44

U

UMTS 29
universal unique identifier *137*
upper envelope 82
user panel 133

V

vertical approximation 80
viewshed 76
  radio 78
ViRGIS 117
visibility
  function 79
  graph 79
  link-i 78
  point-to-point 79
  point-to-region 87
  radio 78
  vertex 82
visibility map
  continuous 77
  discrete 77
VR-GIS 19

W

WGS84 17
window frame 132
WLL 30
WorldView 118
Worst Test Point First 104

X

XGobi 136
  link 135

Y

YUV 54