Acknowledgements

This thesis is submitted for the degree of Cand. Scient. at the Department of Informatics, University of Oslo.

The work has been carried out at the Division of Information Technology at SINTEF SI[1], Department of Industrial Mathematics. I would like to thank all the staff of the division for the stimulating environment they have supplied, both professionally and socially. Without mentioning your names, you should all know how thankful I am for your friendship and the answers you have provided to all my questions.

I would like to thank my supervisors Erlend Arge and Morten Dæhlen for their inspiring guidance in general and the advices given on mathematical issues in particular. I am especially grateful to them for giving me sufficient freedom to ensure the creativity of the work.

A special thank to Bjørn Skjellaug at the Department for Cooperative Systems, who has been an informal advisor on the data modeling aspects of my work. He has also guided me in the art of writing scientific reports, and generously encouraged me when I needed it at most.

To David Skogan and Olav Mork Bjørnås: I have very much appreciated our vivid discussions, and your comments to selected parts of the thesis.

At last, but not least, I want to express my sincere gratitude for the personal support given by my wife, Hilde Brodahl, and the patience she has showed during the work. She has also been of great help in proof reading of the thesis, and in giving a helping hand in other editorial processes.

Oslo, November 1993

Gunnar Misund

[1]SI, Center for Industrial Research, Oslo, is part of the SINTEF Group, which employs about 1500 research scientists.

Contents

List of Figures

Introduction

The Ptolemaic paradox

In 1472, the first printed version of 'Geography' by the Alexandrian multi-scientist Claudius Ptolemy, was published (see [Bro49][2] for details of Ptolemy[3] and 'Geography').

The book was written in the second century A.D., and is a compilation of the contemporary knowledge about the Earth, also including a treatise on cartography. Ptolemy describes how to design maps, for example how to make projections from the spherical surface. 'Geography' also describes about 8000 places in the then known world. 'Geography' is recognized as the first *atlas*, and this special form of presenting geographic information has changed very little since the days of Ptolemy. The book, and the cartographic traditions it was based on, was forgotten in the Western civilization during the Middle Ages. In this period the Earth was considered to be a circular (or even sometimes rectangular) disc, and most maps were merely presenting legends, phantasy and religious views of the world. Fortunately, Arab and Byzantine scholars and copyists kept the Ptolemiac tradition alive during the period from 200 A.D to 1400 A.D.

Before the turn of the fifteenth century, not less than seven folio editions of 'Geography', also called 'Cosmography', was published, expensively illustrated and in most cases supplemented with maps. After the rediscovery of these important writings, cartography experienced a revival after the 1200 years of standstill. The demand for better and more comprehensive maps, mostly due to the discovery and the beginning exploitation of new territories[4], was matched by efficient supply made possible by Gutenberg[5] and the rise of mass media. The combination of new application areas for the map and novel technology lead to quite a revolution in map making.

Now, about half a millennium after the rediscovery of Ptolemy and the introduction

[2]Regretfully, only references to Anglo-American and Norwegian literature are made throughout the thesis. This is indeed not indicating that relevant literature in other languages do not exist, but reflects the fact that the author do not master other languages well enough to include such references.

[3]Claudius Ptolemy is perhaps more known as the father of the astronomical system where the planets circles around the fixed Earth. This is described in his 'Megiste Syntaxis' ('The Great System'), also called 'Amalgest'. He also wrote 3 volumes of music theory which represent our main knowledge of ancient western music theory.

[4]One of the important events in this context was the discovery of America in 1492. However, Cristopher Columbus' expedition represented only the culmination of a series of remarkable discoveries made within the last part of the fifteenth century.

[5]Johann Gutenberg, Germany, 1397–1468, invented at about 1440 A.D. the art of printing books with movable types.

of the printing technology, we experience a similar revolution. As society has grown more complex and our exploitation of the Earth is reaching the limits where fatal and nonreversible damage threatens, new demands have emerged calling for wider and more intensive and advanced use of maps. The demands are, as during the map revival in the Renaissance, met by the introduction of a new technology. Now we encounter computer aided cartography, or *CAC* for short, in naval navigation, in production of limited editions of customized maps, in land resources assessment, in military missile guidance systems and in global environmental surveillance. The term 'Geographical Information Systems', GIS for short, covers many of the applications utilizing a digital map.

Still, as indicated by Burrough in [Bur92], and to some degree shown in section 3.3.3, it is the same map model that is the foundation in computer aided cartography as it was in manual cartography in the Renaissance, which again was based on the ancient model described by Ptolemy.

One of the main problems in traditional cartography is how to project the spherical surface of Earth onto a planar medium[6]. All maps are essentially distorted representations, and one of the consequences of this is that it is difficult to compare information given in two different projections. For this reason, many cartographers have promoted large globes as the most ideal maps. Still, the majority of CAC applications operates in planar coordinates. Cartographers have finally got access to a tool making it trivial to store, compute and analyze geographic information directly related to the spherical surface. For some reasons this opportunity has not yet been fully taken advantage of.

This is an example on what will be called the *Ptolemaic paradox* in this thesis, that the contemporary and quite sophisticated information technology is not being fully exploited in computer aided cartography. Even though computers in several cases are able to handle *new* problems or to offer *new* solutions to existing problems, the main objective for the use of information technology in contemporary cartography, has been to make *existing* cartographic methods more *efficient* and *accurate*.

This mismatch between advanced available tools and the limited and quite simple model of the world they are applied on, constitutes the main motivation for the thesis. As shown in *Prologue*, the Ptolemaic paradox is not only restricting the possibilities in CAC, but it also gives rise to anomalies that may cause malfunctioning. This is due to the fact that the traditional map is designed for manual treatment, and is certainly not prepared for the semi-automatic procedures introduced in computer aided cartography.

Scope

The overall scope[7] of the thesis is to identify and to some degree solve selected problems due to the Ptolemaic paradox. Details of the scope are outlined as follows:

- The thesis should provide enhancements and additions to the traditional map concept. This *augmented map concept* is to be understood as a framework within which more realistic models of spatiotemporal information may be developed and implemented.

[6]The matter is even worse, the surface is as known not a sphere, but a complicated geometric object close to an ellipsoid.

[7]The scope of the thesis has gradually evolved during the work.

- The augmented map concept should provide efficient support for *cartographic generalization*, which is a central issue in cartography.
- In the augmented map model, spatial and non-spatial information should be treated in an integrated manner and as equally important aspects of the geographic entities.
- The emphasis will be on the spatial aspects of cartography. Still, selected issues concerning non-spatial information have to be discussed to ensure a certain degree of comprehensiveness.
- A limited object-oriented implementation should be carried out to illustrate some of the main features of the concept.

The thesis should, according to this scope, develop an augmented map concept that is conceptually comprehensive but fragmented with respect to detail, especially in the non-spatial domain. The scope also implies the accomplishment of a complete process from a complex problem to a computer program.

Outline

The thesis is organized as follows.

- *Prologue* is a brief presentation of a few problems caused by the Ptolemaic paradox. Some questions are asked that initiate the quest for an augmented map concept.

- *Part I, CARTOGRAPHY AND GIS*, gives a short introduction to some basic issues in cartography, both traditional and computer aided. Cartographic generalization is introduced as perhaps the most central aspect of cartography, and is briefly discussed. The traditional map model is characterized and termed the *Paper Map Model.* The idea of *augmenting the map concept* is introduced to facilitate the development of a new generation of systems designed for management of spatiotemporal information.

- In *Part II, MULTIMODELS*, we introduce and develop the *Multimodel* concept as a flexible mechanism designed to structure and to a certain degree solve some fundamental aspects of the generalization problems identified in *Part I.* The problems are associated with the management of the multitude of different *scales*, *moments or intervals in time* and *editions.* The Multimodel offers an homogeneous way to handle model variants, if possible, in a consistent and compact manner.

- *Part III, METAMAP*, is devoted to the elaboration of *Metamap*, our contribution to the augmentation of the traditional map concept. Metamap is an object-oriented high level framework, offering a flexible method for structuring spatiotemporal information. The Multimodel principle introduced in *Part II* is a key notion in Metamap. Metamap is initially presented at a conceptual level as a metamodel. A limited version of this model is refined into an object model called *MINIMAP.*

- *Part IV, CONCLUSIONS AND IMPLEMENTATIONS*, summarizes the thesis. We give some conclusions and make suggestions on further research on Multimodels and Metamap. A generic Multimodel customized for geographic information, called MUL-

TIMOD, is implemented, giving examples of text, records and piecewise linear curves and surfaces as Multimodels. A modest implementation of the Metamap model, called MINIMAP, is carried through to demonstrate key aspects of an augmented map concept.

- We close the thesis with the *Epilogue*, by recalling the questions given in *Prologue*, and suggesting some answers.

Geographic information science

The increasing interest and activity around the use and development of geographic information systems converges, according to some researchers, for example Rhind, Goodchild and Maguire [RGM91], page 317, into a new discipline of its own. The new discipline will in the thesis be referred to as Geographic Information Science. Recently, Canada has suggested that ISO (International Standardisation Organization) should include this field, termed 'geomatics', in their standardization efforts. GI[8] science is now being studied at several universities and colleges around the world as an independent subject, and not only as part of courses in geography, geodesy, land resources assessment or computer science.

One of the characteristics of GI science, is the interdisciplinary nature of the subject. This is reflected in the thesis, as the different parts are depending on different disciplines:

- *Part I, CARTOGRAPHY AND GIS*, is dominated by traditional cartography and applications of GI systems, but requires no special knowledge from the reader.
- However, *Part II, MULTIMODELS*, which is founded on computer aided geometric design and mathematical decomposition theory, and to a certain degree object-oriented analysis and design, assumes that the reader has some basic understanding of the main principles in these fields.
- In *Part III, METAMAP*, knowledge from traditional cartography, object-oriented analysis and design and GIS modeling are the main building blocks. Readers not trained in these disciplines will hopefully still gain some insight in the area by reading the text.
- In *Part IV, CONCLUSIONS AND IMPLEMENTATIONS*, examples are given on the craft of programming a computer in an object-oriented fashion. However, readers not possessing this special vocational skill, will hopefully still enjoy the examples given on the use of the application, especially after reading the previous sections, or parts of them.

According to the interdisciplinary nature of GI science, the emphasis in the thesis is rather on the framework, the augmented map concept, than on the specific problems solved within it.

[8]Since 'GIS' may be interpreted both as Geographic Information Systems and Geographic Information Science, the terms 'GI systems' and 'GI science' will be used when the interpretation is not clear from the context. In the thesis GI systems is restricted to the physical computer based programs and systems of programs used in the management of spatiotemporal information.

GIS activities at SINTEF SI

The work on the thesis has been carried out during 1992 and 1993 at SINTEF SI, Division of Information Technology. The thesis is founded on the knowledge and traditions represented by this research environment. Department of Industrial Mathematics has a long-standing international reputation in geometric modeling in general and spline technology in particular. The Department of Cooperative Systems has been involved in many international research programmes concerning object-oriented modeling and integration of system architectures. Both departments are part of the Division of Information Technology, and have been involved in GI activities since the early 1980's.

The two departments have since the middle of 1992 taken part in preparations of launching a five year national research programme devoted to geographical information technology.

During the preparations, some of the foundational issues treated in this thesis, have been proposed as basis for parts of the technological developments in the programme. However, the thesis is to be considered as an independent contribution to the development of more sophisticated real world models for the use in GI systems. The first publication of the Multimodel and Metamap concepts, which are the authors terms, is found as a high level conceptual description in a preliminary technical report [MS93], published in June 1993.

Chapter 1

Prologue

In the autumn of 1991, dr. Erlend Arge and dr. Morten Dæhlen at SINTEF-SI developed an enhanced version of the well known Douglas Peucker [DP73] algorithm, designed for simplification of piecewise linear curves. In short, an approximation, or 'caricature' of the curve is generated, being represented with less points than the original. The new curve deviates from the original within a given tolerance. In this way, both a smoothing, or reduction of noise, and a data reduction are achieved. Arge and Dæhlen also contributed with a new algorithm, called the 'Intersecting Cones Algorithm' [AD91]. The development of the algorithms were motivated by a project where some of the goals were to develop, evaluate and implement routines for use in an ECDIS[1].

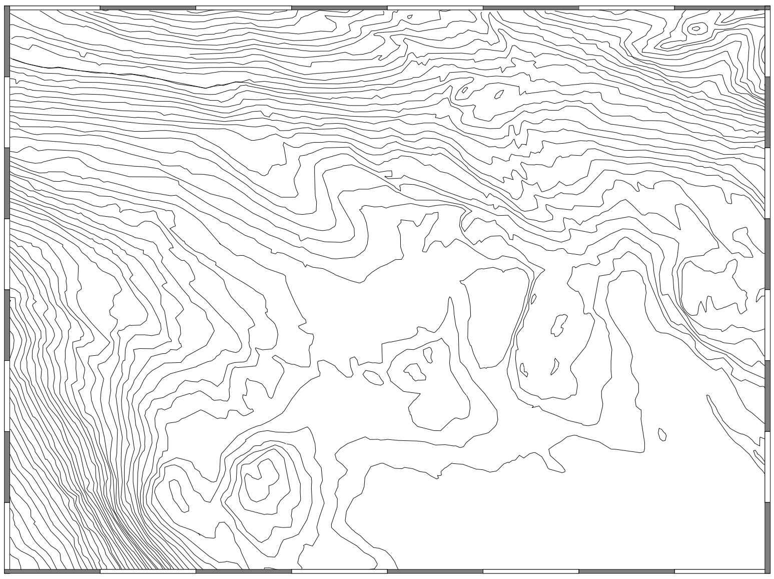


Figure 1.1: Contour map

In nautical navigation it is typical that it is necessary to view the same area in different

[1]ECDIS: Electronic Chart Display Information System, used for on-board route planning and navigation at sea.

scales, ranging from small scale ocean-crossing charts to large scale harbor charts. In an ECDIS, there are constraints on how long it should take to refresh a digital display of a chart [Int90]. Much of the information in nautical charts, especially of coastal waters, is represented as curves, either coastal contours or depth contours.

In this context, the importance of efficient algorithms for reducing the amount of data needed to represent these curves becomes obvious. Even with extremely fast hardware, the ECDIS will not meet the performance requirements if the contours are to be generated from too large data sets.

I had the pleasure of working with the evaluation of the new algorithms compared to some of the traditional algorithms [ADWM92]. The testing included reduction of height contours in topographic maps, such as the one in figure 1.1.

During the testing, some side effects occurred due to the fact that each curve in the maps where treated separately, independent of the other curves in the data sets. The anomalies could be classified as violation of the *topology* of the maps. In this context topology is referring to the geometric relations between the curves in the maps. A discussion of the notion of topology is found in section 8.2 and 10.2.4.

The mishaps were of two main categories, and motivated the formulation of the two questions below:

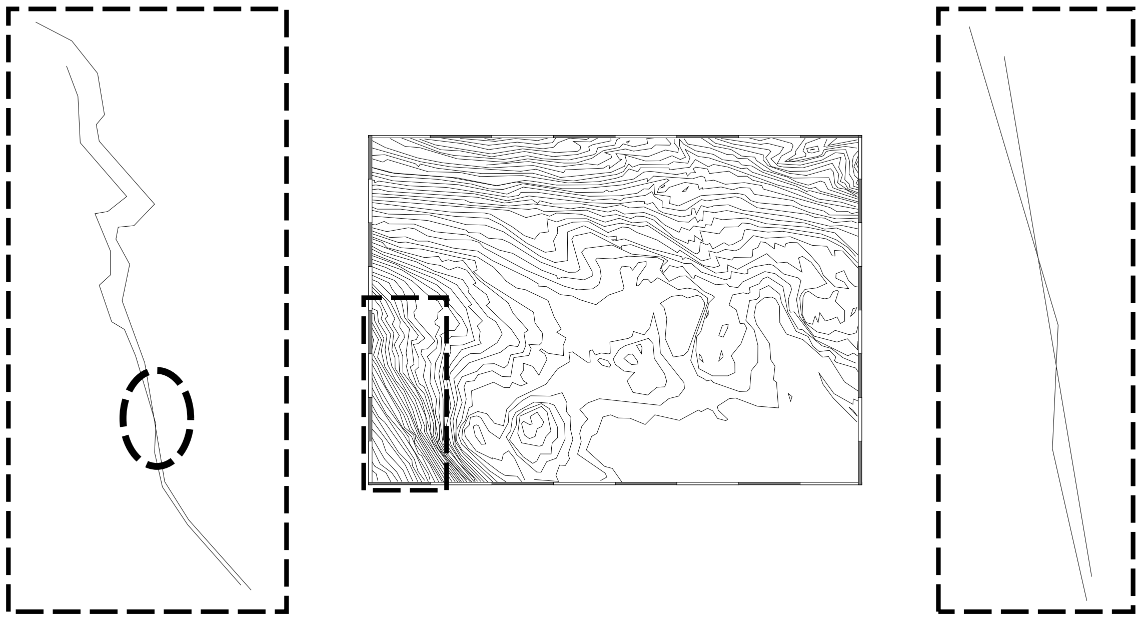


Figure 1.2: Crossing contours, 10 meter tolerance, corresponding to ca. 1 : 50.000

Question 1 Crossing contours.

Figure 1.2 shows the contour map in in figure 1.1 after simplification with the Douglas Peucker algorithm. The tolerance is set to 10 meters, which corresponds to the maximal error allowed at a scale of approximately 1 : 50.000. *Two of the contours are chosen and slightly enlarged. In the circle we see that the contours are intersecting. This same phenomenon,*

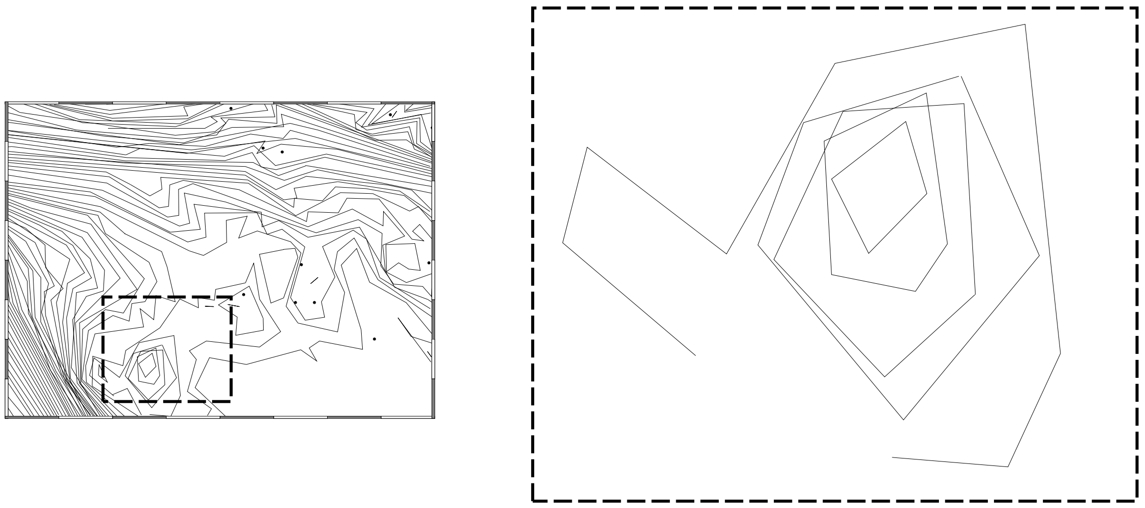


Figure 1.3: Crossing contours, 50 meter tolerance, corresponding to ca. 1 : 250.000

but more exaggerated, may be observed in figure 1.3. This is the same map reduced with a tolerance of 50 meters, corresponding to a scale of around 1 : 250.000.

In many GIS[2] *applications, height contours are stored as non-intersecting closed polygons (see for example [Bur90] chapter 2). The system will interpret the triangle formed by the intersection as a closed polygon, but will encounter serious trouble in deciding which height to assign to the new contour. Such inconsistencies may lead to system crash or other serious malfunction.*

Is there any way to structure[3] *the map information that guarantees that contour topology will be maintained during a simplification process?*

Question 2 Dislocation.

Figure 1.4 illustrates another violation of topology due to line simplification. The map represents the shoreline of an island, in addition to a road. After data reduction of the map, the geometry of the island has degenerated to a linear feature, and the road is reduced to a line segment located off *the island*[4]*. Clearly, this is not a desirable result, and may lead to strange effects when the system finds an offshore road.*

What kind of representation of geographic objects, such as roads and shorelines, could help preventing dislocation during simplification?

A third problem was addressed in the ECDIS project mentioned earlier. The simplification procedures resulted in several variants of logically the same chart, each characterized by a

[2]GIS: Geographic Information System, see chapter 3.

[3]Indeed, not only structure is important when solving problems like this, the design and usage of algorithms is equally important. Still, this thesis is based on the assumption that the structuring of a problem is the first step to take in problem solving processes.

[4]This may be regarded as a pathological example, using extremely large tolerances. However, similar but less pronounced, anomalies are frequently encountered in simplification of cartographic curves.

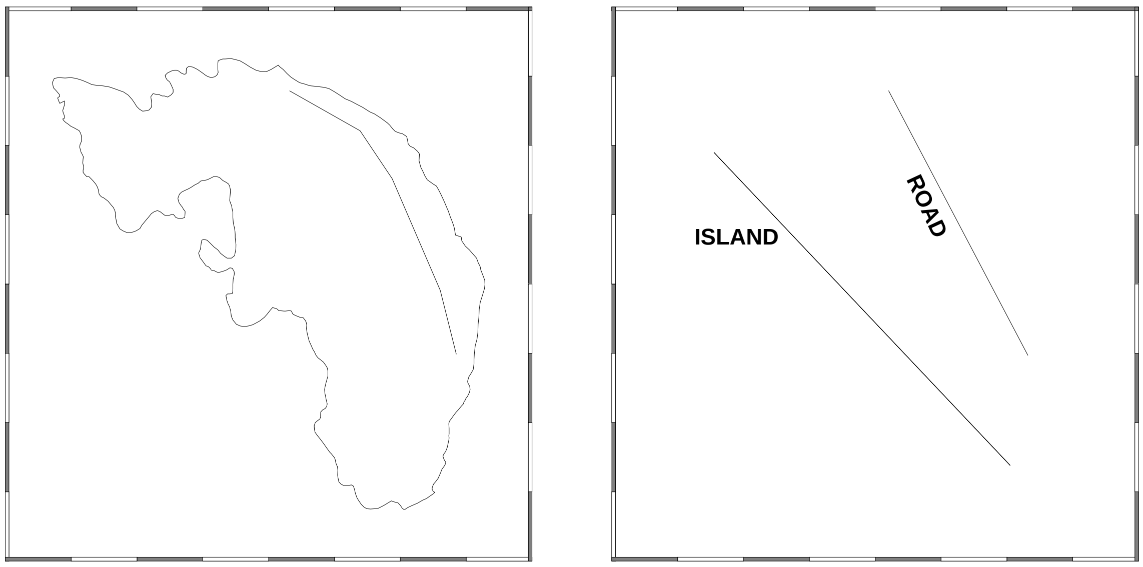


Figure 1.4: Dislocation causing offshore road

specific tolerance corresponding to a given scale of the map. In an ECDIS, one of the functions is the ability to zoom in and out of the current map. This is achieved by 'jumping' from one scale to another according to a set of threshold tolerances, and gives rise to the following problem:

Question 3 Multi scale structures.
Given a set of cartographic contours, performing line simplification according to a set of given tolerances yields a collection of variants of logically the same map, differing only with respect to scale. We may call it a multi scale map*.*
Is there any efficient way to represent this (and related) multi scale structures?

The questions suggest that the representation of the digital maps is too primitive and inadequate for this special purpose, that is to produce variants of different scales from one original map by the means of traditional line simplification algorithms.

With question 1, 2 and 3 in mind, some questions of more fundamental character arise, which are the main sources for the inspiration behind the results obtained in the thesis.

Question 4 Augmentation of the map concept

- *Is the traditional map concept fully capable to be a foundation for digital systems dedicated to the management of geographic information?*
- *If not, in what ways are the traditional maps inadequate?*
- *Is it possible to augment the map concept, such that it could withstand the impact of the wave of information technology?*

The questions 1, 2, 3 and 4 span a vast area of knowledge, experience and research. They have been asked before, and answers have been given. P. A. Burrough states it this way [Bur92]:

> Now it is time that GIS-users (...) should ask if the data structures that current commercial GIS offer are really what is needed, and if not, then please would someone pick up this interesting and complex challenge to provide something better.

It is far beyond the scope of the thesis to meet such a challenge in its full extent. Still, this is an attempt to take a few first steps on a path through the interdisciplinary wilderness of GI science. Hopefully, the path will lead to an *augmented map concept.*

The first thing to do, is to take a closer look at the traditional map.

Part I

CARTOGRAPHY AND GIS

Outline

The overall purpose of the thesis is to identify and to a certain degree solve problems due to the Ptolemaic paradox, i.e. that computer based geographic systems widely utilize a literally medieval (and in fact, ancient) map concept (see the *Introduction*). *Part I* is therefore devoted to the science of maps, cartography, and to computer aided systems which rely heavily on the map model.

In chapter 2, we take a closer look at traditional cartography. A general definition of maps is given. The traditional map concept is analyzed and termed the Paper Map Model. It is shown to be, in many ways, limited compared to the general definition of the map.

We then investigate the notion of cartographic generalization, which is claimed to be a fundamental aspect in the management of geographic information. Generalization is basically referring to the process of abstracting and representing real world phenomena in a cartographic setting.

Some details are given on generalization of both spatial and non-spatial information, and three main classes of generalization are proposed, scale generalization, time generalization and edition generalization. A formal definition of generalization is given, based on considerations of information theoretical nature.

Chapter 3 gives a brief survey of the use of computers in handling geographic information. A few definitions of Geographic Information Systems are given, and some of the major research trends in computer aided cartography over the past thirty years are mentioned. A distinction is made on generalization in traditional cartography, termed visual generalization, and in CAC, called analytic generalization.

We then discuss the map models utilized by GI systems, both for representing topographic and thematic information, and comment the extensive vector/raster debate. We claim that the ancient Paper Map Model is the core in most GI systems. To support this assertion, we give some examples from a widely accepted and used GIS standard, the Vector Product Format (VPF).

We close the part by listing some of the many challenges in GI science, and propose an augmentation of the Paper Map Model as a step towards a better foundation for computer aided management of geographic information.

Chapter 2

Cartography

In 'The Multilingual Dictionary of Technical Terms in Cartography' [McoCI73], we find the following definition of *cartography*:

Definition 1 (Cartography) *The art, science and technology of making maps, together with their study as scientific documents and works of art. In this context maps may be regarded as including all types of maps, plans, charts, and sections, three-dimensional models and globes representing the Earth or any celestial body at any scale.*

This is a very flexible and broad definition and extend the field of cartography beyond the common interpretation of the subject. In the development of the Metamap (*Part III*), the flexibility will be of great advantage.

The statement is also an implicit definition of *map*, which in many senses is an invitation to *augment* the traditional map concept. By adding the aspect of time, we get the following definition of a map, which this thesis can be said to be based upon:

Definition 2 (Map) *A map is a model of the Earth or any celestial body (or part of it). It is any representation in 3 dimensions or any planar projection of a such at any scale, represented over a span of time.*

In the thesis only topics related to the scientific and technologically aspects will be treated. The artistic perspective of map making will also, hopefully, benefit from the improvements suggested in the sections to come. Before analyzing the traditional map concept, we will give some examples in order to broaden the view of the map.

2.1 Representing the Earth

The need to make representations of our physical surroundings, according to definition 2, one can expect to be as old as mankind itself.

There exist a wide variety of suchs representations, but they all share the following characteristics:

- They are supported by a physical medium, such as paper and magnetic tape.

- The representation is realized by the means of some sort of coding.
- The users must be able to decode the representation.

The figures 2.1, 2.2 and 2.3 illustrate the range of different types of maps. They show that maps are dependent of what kind of world view the designers and users represent. Cultural background, physical needs, religious ideas and the purpose of the maps are all important factors contributing to the world view.

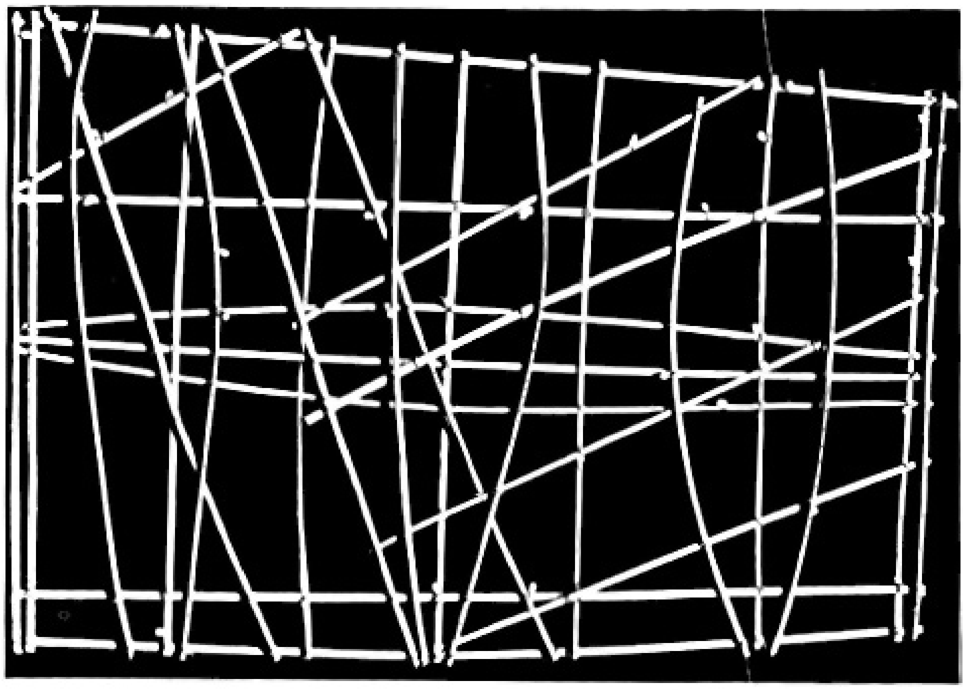


Figure 2.1: Primitive sea chart

The 'chart' in figure 2.1 was made by seagoing natives of the Marshall Islands some time in the 19th century. The chart consists of a framework of palm leaf fibers tied together with leather ropes. Tiny shells are scattered over the framework, representing islands and coral reefs. The branches have a function beyond supporting the shells, they indicate major wavefronts and phenomena significant to nautical navigation.

This representation is far from our common understanding of a sea chart. First of all, the medium is quite different from paper. Secondly it emphasizes topology, the relations between elements of the map, rather than topography, the geometric description.

The Roman road map in fig 2.2, the so called Peutinger Table, originates from the 12th or 13th century. It is a copy of a map made in the first century AD. It is another example in which topology is the main objective. The Roman Empire is squeezed into a 21 by 7 inches paper roll, totally ignoring the topographic distortions. All roads lead to Rome, and this map is undoubtly well suited for a division of Roman soldiers returning from a mission in the outskirts of the Empire. See [Rai38], Part One, for more information on the Peutinger Table and the Marshall Island sea chart.

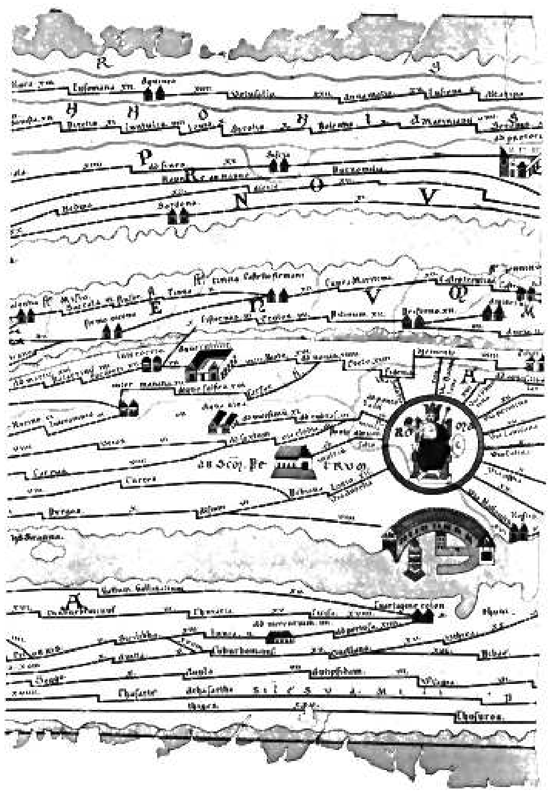


Figure 2.2: Roman road map

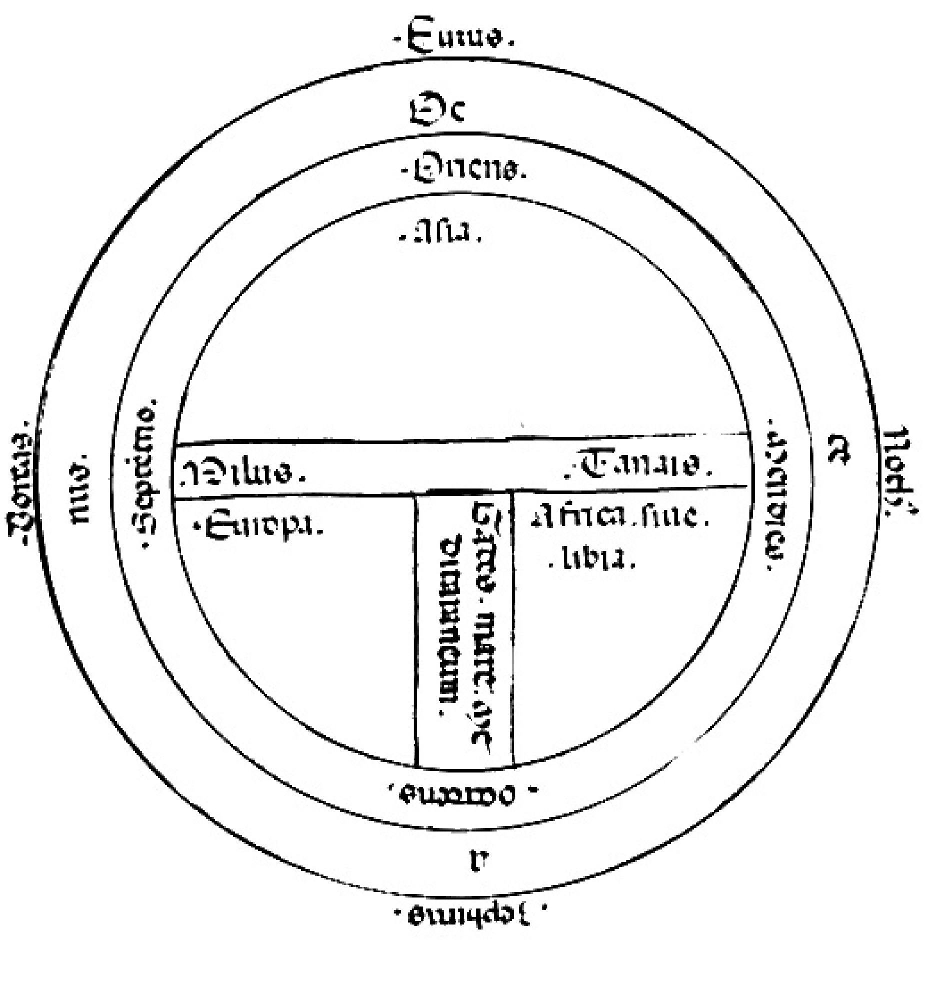


Figure 2.3: TO-map

The only familiar feature of the TO-map in figure in 2.3 is the circular disc reflecting the spherical nature of Earth. The name *TO* stems from the T-like shape inscribed in an O. The vertical bar in the T represents the Mediterranian, the left part of the top bar is the water systems originating from the river of Danube, and the right part is the Nile. The landmasses are divided into three segments, Asia on top, Africa to the right and Europe to the left. The double outer ring is the river Ocean, and in the inner ring there is information on the direction of prevailing winds and observations of celestial bodies. In [RSM78], chapter 'The History of Mapmaking', p. 18ff, a thorough description is given of the TO-maps.

The TO-map is an extreme abstraction of the then known world, and is a striking example of the process called *cartographic generalization*, which roughly speaking is how to simplify and reduce the scale of a representation of a part of or the whole Earth. The topic is discussed in some detail in section 2.3.

In spite of its infantile and almost ridiculous simplicity, the TO-map provided the Mediterranian sailor with significant, and in some cases sufficient, information on how to navigate in these waters. The TO-maps and related representations were commonly used as illustrations in manuscripts from a few hundred years BC up to the Middle Ages. In fact, the first known printed map was such a map. It appears in a book dated 1472. The text is a copy of an 'explanation of the world' by St. Isidoor of Seville, written in the 6th century [TB89].

2.2 The Paper Map Model

The general map concept covers a wide range of representations of the Earth, as illustrated by the examples given in section 2.1 and stated in definition 2. Still, traditional cartography is mainly concerned with what will be called the *Paper Map Model* in the thesis[1]. This is essentially the real world (or part of it), represented on a planar medium (usually paper), with the help of graphic attributes such as lines, dots, text, color, pattern, etc.

Burrough, in his 'Principles of Geographic Information Systems for Land Resource Assessment', [Bur90], chapter 2, takes on this limiting approach when he defines a map:

> A map is a set of points, lines and areas that are defined both by their location in space with reference to a coordinate system and by their non-spatial attributes.

Burroughs view of the map is clearly focused on a planar projection, even if he also adds:

> A map is usually represented in two dimensions but there is no reason to exclude higher dimensions except through the difficulty of portraying them on a flat piece of paper.

To obtain a definition of the Paper Map Model, it is useful to start with some basic ideas from traditional cartography. It is common to classify maps into two main categories, *topographic maps* and *thematic maps*, see [RSM78], p.8-12, [AS81], p.19-20, [Ass84], p. 17.

- *Topographic maps* represent the terrain and a limited number of visible, topographic features. Height contours and curves drawing for example shorelines are the most common modeling tools.

[1]The Greek word for map (or chart) is '$\chi\alpha\rho\tau\eta\varsigma$', originally meaning 'leaf of papyrus'.

- *Thematic maps* focus on one or a few selected themes. The topography is classified according to the selected themes, typically resulting in a partitioning of the topography into curves or areas. Each theme is coded according to a legend[2], for instance that areas with the highest soil fertility is colored dark green. Maps of sewage and drainage systems, soil fertility overviews, aerial and nautical navigation charts and visualization of demographic variables in an urban region are examples of thematic maps.

There is however no sharp distinction between the two classes. The classification motivates a corresponding structuring of the information in a map, claiming that the information may be classified either as topographic or thematic. This dichotomy is characteristic when handling geographic phenomena. The logically same 'thing', or entity, may be described geometrically according to its shape and appearance, or interpreted or classified according to some predefined criteria. This *geographic duality* will be further stressed during the development of Metamap in *Part III.*

The topographic information describes essentially the shape or geometry of the world and thereby models the spherelike surface of the Earth. It is common to achieve this in one of three ways (or in a combination):

- Horizontal cross sections[3] with constant height related to the sea level (height contours). Each contour is an open or closed curve, and has to be associated with the corresponding height value in some way.

- Profiles, or vertical sections, as generated by multibeam echo sounders in hydrographic surveys.

- Points associated with height value. Single soundings in sea charts are examples of points representing surfaces.

The topography in traditional cartography is thereby modeled by points, and open or closed curves in the plane. Height contours are often not closed, due to intersection with the borders of the map, or missing data, commonly encountered in sea charts. All the methods mentioned above are kinds of sampling of a continuous surface, and the user generates the surface by 'mentally interpolation and extrapolation'. There may be additional information, such as different color coding of the various height intervals, but they are essentially derived from height contours and point samples. Such additional information may also be classified as thematic information.

The thematic information is associated to parts (or the whole) of the planar projection of the topographic surface, and is located by

- points,

[2]The legend is in traditional cartography an explanation of what the various graphical attributes, such as patterns, symbols and color, are representing (in the thematic domain). The term arouse during the Middle Ages, where perhaps the most important part of a map was the elaborated and colorful illustrations of stories, myths and legends associated with the places on the map.

[3]Strictly speaking, the height contours are generated by sectioning the terrain with spherical offsets according to the *geoid*, the complex geometric description of the earth at constant zero height.

- open or closed curves or
- regions (planar surfaces) represented as the interior of closed curves. The regions may be complex, e.g. with 'holes'.

Burrough [Bur90] emphasizes the common and limited attitude towards geographic by stating:

> All geographic data can be reduced to three basic topological concepts - the point, the line, and the area.

We observe that Ptolemy, in his 'Geography' (see the *Introduction*), in fact demonstrated a broader approach, by at least including textual descriptions as an important part of geographic data. 'Geography', which has been the prototype of the atlas for 2000 years, contains for example a description of about 8000 places in the then known habitable world.

The information itself is with few exceptions represented with one of (or a combination of) graphic attributes such as text, color and pattern, according to some given legend or standard. How to choose the appropriate visual variables to represent the information graphically, is an important issue in traditional cartography. Interested readers should consult the rich literature in this field, such as [Ans88], [RSM78] and [Cur88].

The themes may be regarded as classifications or interpretations of distinct, physical parts of the world. An area bounded by a closed polygon may be classified as a national park. It is obvious that the same part of the world may have several different thematic interpretations. A certain part of the national park may also be described as a primeval forest. This fact is the main motivation for the *overlay concept*. It is common practice among mapping authorities to supply maps separated into a number of overlays (or foils). The overlays are then combined in the most convenient way for the different users and customers.

Usually there is one overlay (perhaps several) representing the topography, most often supplying coastlines and height contours. The thematic foils represent waters systems, roads, county borders etc.

The map is an attempt to model the reality. Still, it is indeed not a complete model, but an abstraction. The most obvious abstraction is that the map is a scale reduced version of the reality. In addition, it is a selection of the huge number of possible topographic elements and their still larger number of thematic interpretations. At last, the selected and scale reduced features are presented in a customized version dependent of the purpose of the map. The same geographic area may look quit different in a road map compared to a map designed for land resources assessment. Even if they are based upon the same selection of topographic and thematic features, the use of colors, texture and graphical symbols can make the appearance of the maps quite different.

All these three processes, scale reduction, selection of topographic and thematic elements, and customization, are encompassed by the concept of cartographic generalization. In section 2.3, some generalization methods will be briefly discussed.

Based on the discussion above, and the fact that paper maps are abstractions representing limited parts of reality bounded by rectangles (not in reality, but a rectangle in the given planar projection), we propose the following characterization of the traditional paper map model:

Definition 3 (Paper Map Model, PMM) *The Paper Map Model is a planar representation of the real world (or part of it) with the following characteristics:*

- *Decomposition of the reality into cartographic elements, which represent topographic or thematic information.*
- *Utilization of points, curves and areas combined with graphic attributes to represent the reality.*
- *Each thematic element is located to a distinct part of the topography. A part of the topography may have several thematic interpretations.*
- *Representation of the reality at a given scale or resolution, and at a given moment*[4].
- *Representation of the reality in a specific edition, or generalized version.*
- *Tessellation of the reality into rectangles according to the projection used.*
- *Coding of the graphic attributes according to a legend, either provided by the map, or given as some sort of common understanding or agreement.*

The Paper Map Model will in this thesis some times be referred to as PMM. Note that the model complies with definition 2 of a map, limited to a certain moment or interval in time and being a 2D projection of the real world or part of it.

In chapter 3.3.3, it is claimed that the traditional Paper Map Model is the core of most GI systems today. It will also be noted that the PMM imposes severe limitations on the systems, especially with respect to the new generation GI systems. Still, we will show that the model is well suited for additions and enhancements that may lead to an *augmented* map concept.

In the next sections, some details will be given on a central topic in cartography, the generalization process.

2.3 Generalization

In section 2.1 a map was defined essentially to be a representation of the Earth. Such a representation has inevitably to be proceeded by some sort of abstraction. It is this abstraction process that is commonly referred to as cartographic generalization[5]. This is a sophisticated discipline, relying both on theoretical insight, vocational skills and sound understanding of the various uses of maps.

A review of five different generalization models is given in [McM91], indicating the multitude of approaches developed to describe the process formally. The following sections focus on the underlying structures of the generalization problem, and not on the process itself. Some selected issues in cartographic generalization is discussed, helping to sort out tractable problems and possible methods for structuring geographic information in a way suitable for generalization.

Before giving details on various generalization procedures, we present some formalism related to generalization as a tool in controlling and manipulating of cartographic information.

[4]Some maps model variation over time, for instance a historical map showing the rise and fall of the Roman Empire, as a phenomena represented at several distinct moments in time in the same map.

[5]The term 'generalization' in this context must not be mixed up with the same term used in data modeling methods and programming languages.

2.3.1 Cartographic information

Cartography is essentially concerned with compilation, organization, storage and distribution of any types of locational information, i.e. information possible to associate to a distinct spatial description of the reality. The following decomposition of cartographic information is, slightly modified, taken from [AS89].

Denote information as I and using the subscripts tot(al), exp(licit), imp(licit), $topo$(graphic) and $thema$(tic) we have that

$$I_{tot} = I_{exp} + I_{imp}.$$

The explicit information, I_{exp}, is further decomposed as

$$I_{exp} = I_{topo} + I_{thema}.$$

I_{topo} is essentially derived from topographic descriptions, such as the shape of a coastal contour or the area of a lake.

I_{thema} is supplied by the coding of the graphic symbols used in the map, such as color, pattern and text fonts. I_{thema} can be considered as the part of the information that would become meaningless without a legend or some prerequisite knowledge of the graphic language.

I_{imp} is a result of a synergy process between the various elements of the explicit information. If separate cartographic objects, all carrying distinct explicit information, together by synthesis generate new information, not initially present, this kind of information is classified as implicit information.

Needless to say, the concept of implicit information is closely related to the skills and experience of the map reader. It is not a trivial task to analyze such information e.g. by the means of computers.

As stated above, one of the main purposes of maps is to transfer information. The receiver is traditionally the human user, but now (see chapter 3.2), the use of computers in transferring information is rapidly growing.

In many contexts, it is important to perform the transfer as efficient as possible, i. e. to transfer maximum information during a minimal span of time.

A naive solution to the problem could be to represent as much information technically possible, limited by such factors as the resolution of the display medium. The amount of transferred information will, however, in most cases not be proportional to the density of the displayed data. Figure 2.4 illustrates a possible scenario of the correlation between the information density of the map and the amount of absorbed information by the receiver (or user). The essence of the illustration is that if we increase the information density, the transferred amount of information (that is, the fraction of the total information displayed that the user will absorb) will reach a maximum after a S-shaped development. In rare cases the graph will converge to the maximum where absorbed data equals the displayed data. Usually the amount of absorbed information will start to decrease when increasing density beyond the critical point. Too much information confuses the reader and obscures information at more basic levels. Ultimately, at the point where the display or paper is completely filled with elements, there is not any transfer at all.

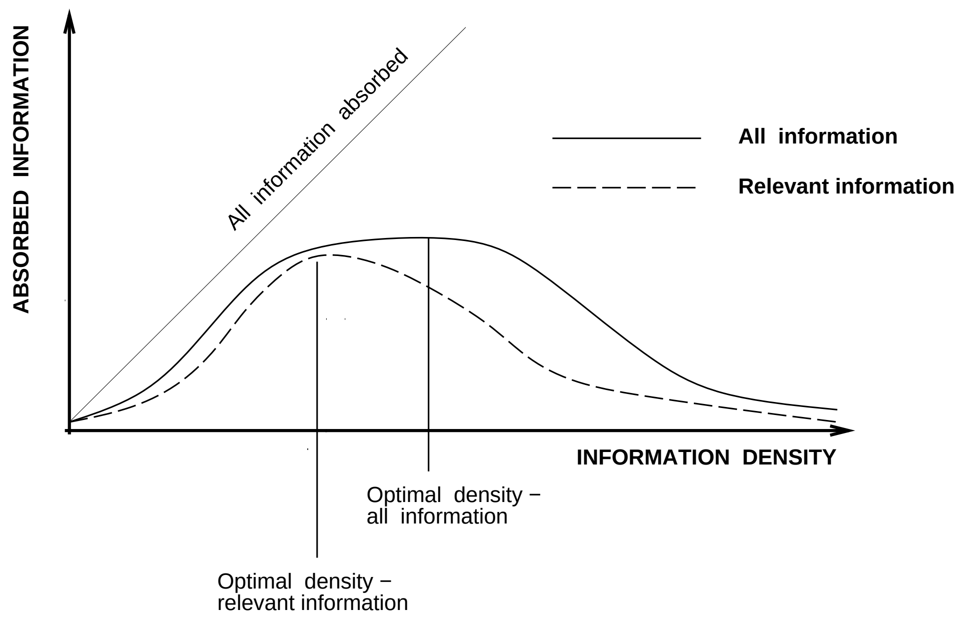


Figure 2.4: Transfer of information

Another aspect to consider, is that not all the information may have the same relevance to the user. Figure 2.4 illustrates that the amount of *relevant* absorbed information may behave different from the transfer of all information.

In cartography, one of the fundamental goals is to optimize the information density, so that a maximum amount of relevant information will be absorbed by the user. This optimization procedure is hard to formalize, since the measurements of both information density and absorbed information will always be of a heuristic nature. As a rule, it will be difficult, if not impossible, to find *the* optimal solution. Still, with the process of cartographic generalization, cartographers are constantly trying to solve this optimization problem.

2.3.2 Generalizing topographic information

In the thesis, a distinction between topographic and thematic generalization will be made. Topographic generalization deals with the geometric descriptions of physically recognizable elements of the map. Thematic generalization, on the other hand, is concerned with how the thematic information associated with the topography can be represented at different scales, in various editions and at several moments or intervals in time.

We will now give three examples of topographic generalization.

- *Simplification* is a reduction of the complexity of linear features, also referred to as smoothing. Figure 2.5 shows a coastline in a given scale $1 : X$. The two smaller maps are of scale $1 : 4X$. The map at left is generated by simply reducing the size of the original, $1 : X$. The coastline is unnecessary detailed. The map at right is produced by simplification of the curve representing the coastline, yielding a presentation that

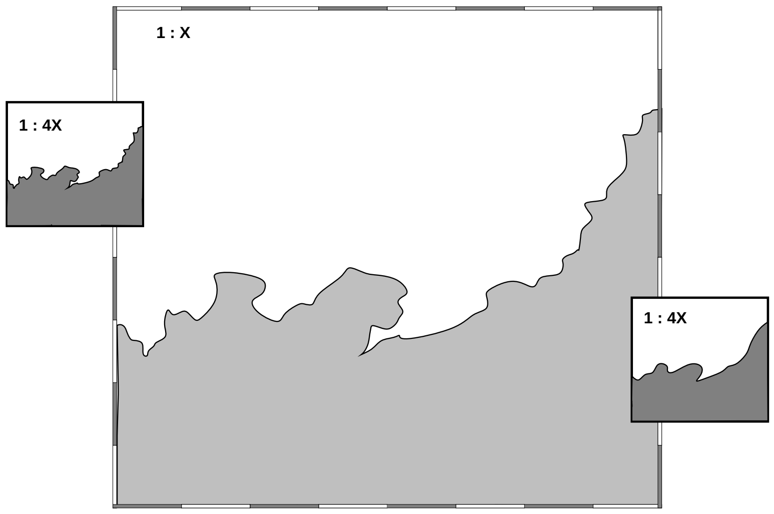


Figure 2.5: Simplification

is 'better' than the one to the left. Better in this context means aesthetically more pleasing and more efficiently performed information transfer. As noted earlier, this kind of measurements are of typical heuristic nature.

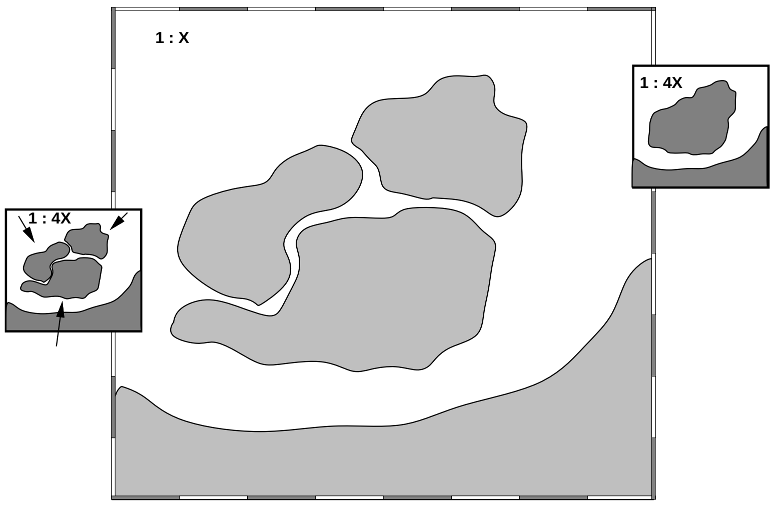


Figure 2.6: Combination

- *Combination* is the merging of two or more objects into a single one.
  Figure 2.6 presents three islands and a coastline, and two editions of a scaled down version. An alternative could be to merge the three islands into a single new one, as it is done at the map to the right.

- *Deformation* is a arbitrary change of the original geometry of an object. Figure 2.7 shows

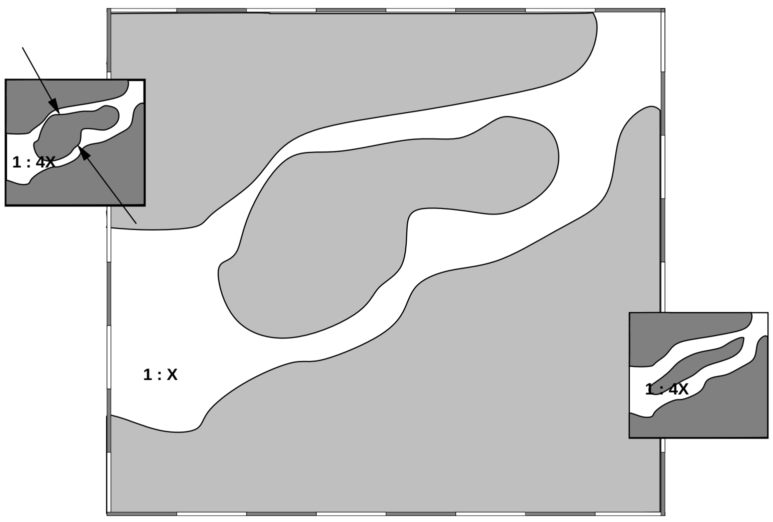


Figure 2.7: Deformation

yet another variation over the sea chart theme. Here, to emphasize sailing channels on both sides of the island, it is squeezed from two sides, yielding a relatively thinner presentation of the island.

In these examples, the three *generalization operators* are associated with scaling down existing maps. Still, the operators, especially combination and deformation, are also used when different *editions* are produced from the same map. The scale is then unchanged, but the maps are manipulated, or edited, to fit the purpose of the map. A navigation chart and a map designed according to recreational activities may emphasize quite different aspects of the same area. In addition, we might think that different versions were produced in order to model variations over time, for example how the contours of the islands changed according to the tide.

Other topographic generalization operators do indeed exist, such as *selection* which selects a subset of the cartographic objects in question, and *displacement*, which translates or (and) rotates an object. See [AS89], [Ans88], [RSM78] and [BM91] for further details on generalization operators.

2.3.3 Generalizing thematic information

As with the topographic information, thematic information is also subject to cartographic generalization. Throughout the thesis, the emphasize is on the topographic information rather than on thematic issues. Still, it is necessary for the development of both the Multimodel (*Part II*) and the Metamap (*Part III*) concepts to briefly touch relevant aspects of thematic information.

Thematic information may be of near say any kind. The only condition is that the information has to be associated to a topographic element in some way. In traditional maps there is a limited number of possible representations for this kind of information. The most

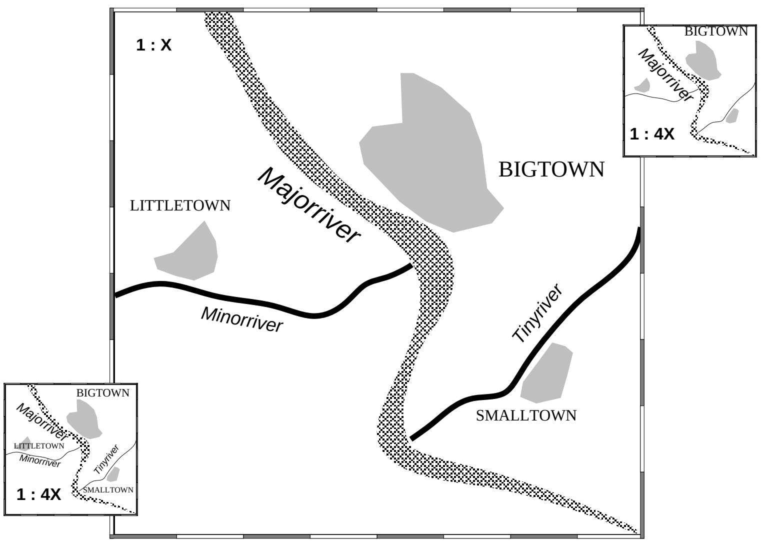


Figure 2.8: Generalization of text

common is plain textual information, such as names on cities, rivers and lakes. Other possibilities is the use of graphic attributes such as color and pattern, that are to seen as coding according to a legend or a given standard. In the traditional atlas, additional information is given in tables, illustrations and textual descriptions.

Figure 2.8 shows an example of generalization of textual information. In the generalized $1 : 4X$ edition at right, names of minor features are simply omitted to make a more readable map. This procedure corresponds to the selection operator in the previous section.

In figure 2.9, a geographic area is classified in an original map according to 4 levels of soil fertility. Plain scale reduction yields a confusing picture, at left, obscuring main trends[6]. In the right map, the information is aggregated to two levels, and this results in a map easier to comprehend. This is essentially the same process that the combination operator performs on topographic information.

It is not hard to realize that the generalization operators for thematic information may differ substantially from those used in topographic generalization. Still, they share the common purpose to optimize transfer of information. Thus, all later references to generalization in the thesis include both topographic and thematic information, unless something else is explicitly stated.

2.3.4 Cartographic generalization

As stated earlier, generalization is needed both as the *scale* of the map is changed, when the map is customized into a specific *edition* and in modeling *temporal* changes. Thus, we may introduce the following classification of the various generalization processes. Given a map,

[6] In fact, there is a topographic aspect in this kind of thematic generalization, since it includes merging of areas into larger ones.

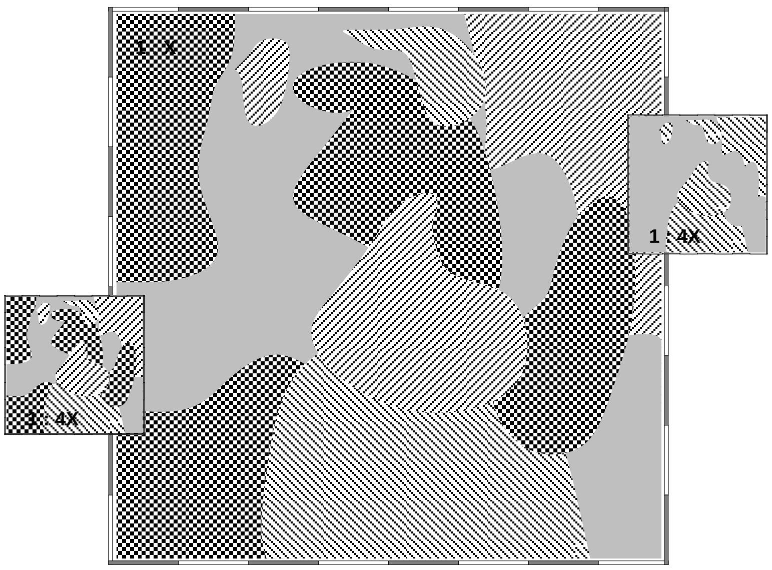


Figure 2.9: Generalization of pattern

there are essentially three main categories of generalization we may want to perform (and of course combinations of them):

- Change the scale. This may also be considered as a change of resolution, or accuracy, of the map information.
- Customize it (within the same scale). The customized versions is to be considered as different editions or variants of essentially the same piece of information.
- Adjust it to represent a certain moment or interval in time.

In fact, we will se later, in chapter 6, that the edition and time aspects of generalization involve basically identical operations, even if the motivation and the process as such are quite different.

International Cartographic Association (ICA), has made the following explanation of generalization [**?**]:

> ...the selection and simplified representation of detail appropriate to scale and/or purpose of the map.

We observe that temporal changes is not considered part of the generalization process. Nevertheless, we take the liberty of claiming that temporal changes should be encompassed by the generalization concept.

With this description and the observations made during the last sections in mind, we make a more precise statement on the nature of cartographic generalization:

Definition 4 (Generalization) *Cartographic generalization is the process of optimizing the information density of maps (according to definition 2) under the constraints provided by*

- *scale (or resolution),*

- *edition (defined by map purpose and skill/experience of the user) and aesthetic guidelines*[7] *and*
- *moment (or interval) in time.*

Generalization may be performed on both topographic and thematic information.

According to the three main aspects of generalization, the terms scale generalization, edition generalization *and* time generalization *will occasionally be used in the thesis when referring specifically to one of the three aspects of the generalization process.*

The definition motivates key aspects of the development of the Multimodel structure in chapter 6. This structure aims to support certain stages in the generalization process, as it will offer a compact and consistent representation of a set of generalized maps.

[7]Occasionally in this thesis, in certain contexts, the term 'generalization' will be used in a limited setting, referring only to the edition aspect and not scale or time.

Chapter 3

Geographic Information Systems

Computerized handling of spatial data by the use of *Geographic Information Systems*, has become an important decision support tool in a wide range of areas, such as environmental surveillance, route planning, land resources assessment and aerial and nautical navigation.

As hardware and software technology during the last two decades has grown more mature, the demand for additional functionality, higher capacity and advanced user interfaces in GIS has grown accordingly.

During the past few years, much attention has been paid both from advanced users and leading vendors to design and develop the new generation geographic information systems. In many ways, this thesis may be regarded as a contribution to this ongoing effort.

In this chapter, we give some details on GIS in general and computer aided cartography in particular.

3.1 Managing Spatiotemporal Information

Geographic Information Systems, is the common term covering software capable of various degrees of managing spatiotemporal information. There are many explanations of the concept, and Maguire in [Mag91], page 10-11, lists some of them:

- A system for capturing, storing, checking, manipulating, analyzing and displaying data which are spatially referenced to the Earth.
- Any manual or computer based set of procedures used to store and manipulate geographicly referenced data.
- An information technology which stores, analyzes and displays both spatial and non-spatial data.
- A powerful set of tools for collecting, storing, retrieving at will, transforming and displaying spatial data from the real world.
- A decision support system involving the integration of spatially referenced data in a problem-solving environment.
- A system with advanced geo-modeling capabilities.

The explanations illustrates the great variety of uses of GI systems and the different expectations to how the systems should perform. Even though none of the definitions explicitly mentions the aspect of time, it is obvious that much of the spatial referenced information will vary over time. Thus, no mistake will be made by suggesting a more general definition:

Definition 5 (GIS) *A Geographical Information System handles spatiotemporal information in a structured manner, utilizing a model of the real world or of a part of it.*

Note that a GI system not necessarily is a digital system, a traditional map is clearly a GIS according to this definition. In particular one might characterize the traditional atlas as probably the most common GI system today. However, in this thesis, the use of the term 'GIS' will refer to a computer implemented system, unless something else is explicitly stated.

GI systems differ from other information systems by the fact that the information to be handled is of spatiotemporal character, i.e. that it is related to both space and time.

The information system part of a GIS is concerned with actually storing (in a database), retrieving and analyzing the information. This is a research field of it's own, and discussions on such aspects (e.g. whether object-oriented databases are more suited for implementations of GI systems than traditional relational databases) is beyond the scope of the thesis. The focus is rather on the structure of the real world model, than on how to store and retrieve the data in such a model. These two aspects of a GI system are of course not disjoint, but are closely interrelated and are indeed mutually dependent. Still, one may view a GIS as a core consisting of the real world model, surrounded by an information system. The latter acts as an interface between storage sources, users, applications and the real world model, as illustrated in figure 3.1.

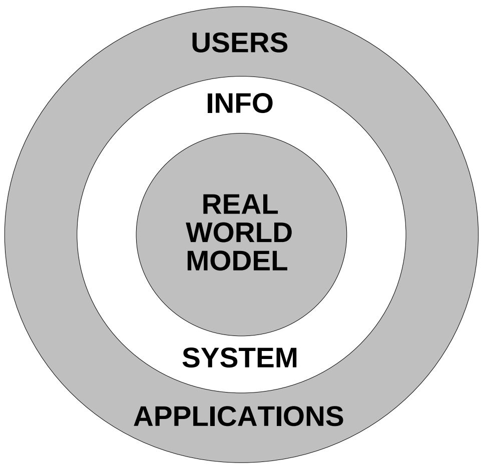


Figure 3.1: A Geographic Information System

Recalling definition 2 of the map, we may very well characterize the core in a GI system as a *map*. It then becomes clear that cartography plays a fundamental role in GI science. In computer based GI systems, computer aided cartography is thereby of vital importance. The topic is briefly discussed in the next section.

3.2 Computer Aided Cartography

One of the characteristics of cartography is the huge amount of data involved. The advantages of computerizing some of the processes were early recognized. From the late 1950's, the use of computers in cartography has increased steadily. Still, the complexity of both data and the use of them until now have limited computerization to certain areas like storage and simple analysis.

Advances in computer technology in the late 80's, like large volume storage devices, fast processors, high resolution graphic displays, sophisticated printing devices and object-oriented modeling and programming tools, have now made it possible to address unsolved problems.

It is convenient to distinguish between certain tasks in the area of CAC:

- *Data capture*, the process of gathering primary cartographic information from sources such as
  - geodetic surveys using digital instruments and recording equipment,
  - digital soundings from multi beam echo sounders,
  - digital analysis of aerial and satellite images and
  - scanning of existing printed maps.
- *Compiling* and ordering of spatial data according to some kind of model, such as a topographic map with height contours.
- *Storage*, using digital storage technology like magnetic tape or optical discs.
- *Production* of traditional printed maps. Advanced drawing programs and standard CAD/CAM applications are frequently used to compile and edit maps digitally prior to printing them by traditional techniques.
- *Analysis*, such as computation of areas and distances and finding the shortest route in a network of roads are a typical tasks well suited for computerization.

The operations above are not necessarily exclusively associated with the map domain of a GIS, but may partly be associated to the information system, which facilitate the retrieval procedures and other management operations.

Automated generalization has been one of the goals in CAC. In [BM91] three epochs of research in this field are identified:

- *Period I*, 1960 - 1975:
  - Focus on algorithm development with emphasis on line simplification.
  - Early in the period, experiments using raster images.
  - Later, focus on vector representation and topological data structures.
- *Period II*, 1975 - late 1980's:
  - Algorithmic efficiency.
  - Investigation of methods to deal with the scale dependent nature of geographic phenomena.
- *Period III*, rescent:
  - Formalization of cartographic knowledge.
  - Comprehensive models.
  - Knowledge based systems.

The Metamap development in *Part III* is based on trends emerging in *Period III*, advanced computer aided design (CAGD), object-oriented modeling, and last but not least, traditional cartographic knowledge.

3.2.1 Automated Generalization

Generalization is a slow and labour consuming process, and it has been put much effort in enabling computer systems to automate cartographic generalization. In limited and fairly simple applications, mainly concerned with topographic information (see [AS89] for an example), this task has been accomplished to a certain degree. A collection of articles on contemporary research in the field is found in [BM91]. Still, as Freeman simply puts it in the preface in this collection:

> This (automated map generalization) has been difficult to achieve.

In other areas, such as computer aided geometric design (CAGD), there is a common census on the need for human interaction in complex computerized processes. This should undoubtly also apply to computer aided mapping.

In CAC, an additional dimension is encountered in the generalization concept. In definition 4, section 2.3.4, generalization is formulated as an optimization problem. In traditional cartography, it is a human user that is the target for the information transfer from the map. This may be the case in CAC, but it may be as well a computer that is retrieving information from the map representation in the GI system. From this motivation it is natural to make a distinction between *visual* and *analytic* generalization. Visual generalization becomes equivalent to traditional generalization, and analytic generalization may be defined as follows, slightly different from definition 4.

Definition 6 (Analytic generalization) *Analytic generalization is the process of optimizing the information density in digital represented maps (according to definition 2) under the constraints provided by*

- *scale*[1]*, or resolution, according system specific variables such as processor speed, numerical accuracy and algorithmic constraints,*
- *edition, defined by map purpose and type of application accessing the map, and*
- *moment (or interval) in time.*

Analytic generalization may be performed on both topographic and thematic information.

[1]In CAC, the term 'scale' becomes sort of meaningless. Scale is defined as the ratio between the the size of a geographic object as it is represented in the display medium, traditionally paper, and the the size in reality. In CAC, there is a characteristic independence of the digital representation and the displayed representation. Thus, the scale of the same digital map would vary according to if it was printed by a plotter or edited on a computer screen.

Instead, the term *resolution* may be used as corresponding to scale. The digital map is always a discretization of some real phenomena, and the resolution is proportional to how dense or detailed the real world is sampled in the map. The scale concept may be generalized to cover this aspect, such as the scale express the level of accuracy in the map. There would then be possible to define some bidirectional mapping between scale and resolution.

For this reason, in this thesis the term 'scale' will also be applied to digital maps and GIS, even if it would be more correct to use 'resolution'. This is done to be consistent to the basic idea in the thesis, to augment the traditional map concept.

In a GIS, generalization will often be a combination of both the visual and the analytic considerations, since accessing the real world model in most cases involve both computer processing and visual inspection of the result.

3.3 Map Models in GIS

There exist a variety of map models used in GI systems, and a multitude of ways to describe them. Frank proposes to differentiate between three levels of description of real world phenomena [Fra92]:

- *Concepts:* Ideas, notions and relations between them that are used by humans to organize and structure their perception of reality.
- *Data models:* A comprehensive set of conceptual tools to be used to structure data.
- *Data structures:* Detailed and low level descriptions of storage structures (...).

This approach is quite common. Papers discussing related issues in GIS are often making these distinctions, but usually at an implicit level.

In the early days of GIS modeling, the emphasis was rather on the more low level data structures than conceptual frameworks. This implied that notions associated to implementation issues became valid as characterizations of GI systems at a conceptual level. Burrogh states that this has been a major constraint in many different application areas using GIS, especially natural resources study ([Bur92], page 395). Recently, however, there has been a significant shift towards the conceptual aspects in GI science.

As indicated in section 2.2 and 2.3.1, a map handles both topographic and thematic information. Still, many papers concerning GIS modeling focus on the topographic, or more frequently called geometric or spatial, information. The treatment of thematic issues is most often secondary treated at lower levels, typically reduced to discussing how to implement attributes in databases.

In the next sections, some aspects of the conceptual models characterizing the GIS scene, will be discussed. At this point, there is no need to give details on the implementation level, but selected issues relating to data models are touched briefly.

3.3.1 Topographic models

There seems to emerge a fundamental, main classification of how to describe the topography of the real world. Frank et. Mark use the terms *Kantian*[2] and *Descartian*[3] to describe the two world views [FM91], page 148:

- ⋆ A *Kantian*, also called feature based, point of view implies an emphasis on the *objects* that fill the geographic space. In this way, space, or location, becomes an attribute of an object.
- ⋆ From a *Descartian* viewpoint, also called location based, each point in space is described by which objects that are encountered. The objects are in this manner properties of

[2]Immanuel Kant, 1724 – 1804, German philosopher, known as the father of *Criticism*.

[3]René Descarte, 1596 – 1650, French philosopher and mathematician. Emphasized the *duality* in the existence, distinguishing between spiritual and physical aspects of the world. Father of the 'mechanistic' world view.

the location.

Frank et. Mark point out that both views are used interchangeably in real life, depending on what is more suitable for the task at hand.

Burrough [Bur92] is probably having the Kantian/Descartian distinction in mind when describing the top layer in a six level schematic overview of stages on the way from the real world to a graphical implementation model. He claims that reality may consist of 'fully defined and fully definable objects/entities', or 'incompletely defined or incompletely definable spatial entities'.

Goodchild [Goo92] defines the 'fundamental element of geographic information' as the tuple $T = \{x, y, h, t, z_1, \ldots, z_n\}$, giving the values of n spatial variables where z_i at the geographic location (x, y) at height h at the given moment t. This defines a field over the entire spatiotemporal (4D) space, and he terms it the *field model.*

He juxtapositions this model to what he call the *object model.* Here, a set of discrete objects is represented by a set of tuples $\{i, a_1, \ldots, a_m\}$, where i is an object and a_1 through a_m are m attributes of the object. Location is described by a set of tuples $\{x, y, o_1, \ldots, o_i, \ldots\}$ where o_i is a binary variable indication the presense or absence of object i at location (x, y).

Goodchild notes that this object/field dichotomy is a long standing issue in cartography. The dichotomy has also motivated a classification of GIS that is perhaps the most widely used today, into *vector* based and *raster* based GI systems.

Vector representation is the term for modeling topographic objects by the means of points, piecewise linear curves and polygons, all structures which may be expressed as vectors of geographic coordinates. This corresponds to the object view of the world.

Raster representation originates conceptually from a field view of the world[4]. A raster represents a limited version of a field, in the sense that in a raster each cell is commonly associated with only one value. The raster is in its nature a regular tessellation of the topographic surface, most common rectangular or quadratic, but also triangular tessellations, so called trixels, are used together with more exotic versions such as hexagonal tessellation. See [HB92] for a discussion of models raster encoding, and [Mag92] for an example of a GI system based on raster representation.

As data models, the raster and the vector concepts represent truly different approaches to the world, following the Descartian respectively Kantian approach. However, as data structures, one might argue as Burrough in [Bur90], page 33, that to a certain degree the two representations are equivalent. Raster representations may be transformed into a vector representation and vice versa, allowing for a loss of information, especially in conversions from vector to raster.

Following this approach, we claim that the object view in geographic modeling encompasses the field model, and support the assertion with the following argument.

Assume we have a tessellated model, or in other words, a piecewise constant field model, e.g. a raster representation. This model may be transformed to an object model without loss of information in one of two ways:

[4]A more prosaic reason to employ the raster structure, is that one of the main input sources in GIS is digitally scanned paper maps. Scanning yields directly a raster image, from where contour lines an other objects may be extracted and classified.

- We may define the object spatially as the boundary defined by the tessellated area and interior, trivially without loss of information. Further, let the tessellation become the *thematic* description of this object, thus we have *encapsulated* the information carried in the tessellation. The spatial description in addition to this thematic information will together represent the information in the original field model.

- One and each of the tessellated cells may define spatially a tiny object, and the value of that cell may trivially be assigned to the object as thematic information. The set of all cell-objects together with their thematic values will together carry exactly the same information as the field.

3.3.2 Thematic models

Thematic information is secondary treated in contemporary works on GI science. The emphasis is clearly towards spatial issues. This is reflected in many GI systems, that provide thematic information as attributes to spatial features. Such attributes are usually of quite primitive character, e.g. alphanumeric codes and fixed-length text strings. The opposite approach is rarely seen in GIS, that location is an attribute in the thematic objects.

The two characterization of a GI system could be termed spatial orientation respectively thematic orientation. An example of such a thematic oriented information system could be a property management system where the main purpose was to handle information such as technical data on buildings, the size of the estates and tenants and their rent paying status. In such a system, the spatial information could be reduced to a single attribute, in fact it could very well only be an implicit reference such as the street address.

The degree of spatial respectively thematic orientation makes a continuous range of information systems, from clean cut spatial systems, for example topographic maps, through main stream GI systems to pure general information systems.

The Paper Map Model (PMM) is also clearly spatial oriented. Thematic information is typically represented by text and numbers, color and pattern codes and special graphical symbols, constrained by e.g. the size of the map and printing techniques.

The concept of *information integration* is often used in discussion of how to integrate topographic and thematic information. Shepherd reviews different ways of connecting thematic and topographic information, both traditional and alternative [She91]. Issues concerning information integration will be further discussed under the development of Metamap, *Part III.*

3.3.3 GIS standards

Most standards relating to GI are aimed at low levels of GIS modeling. Some are limited to data structure definitions, others includes issues concerning the data model level. Very few, if any at all, discuss conceptual relations.

VPF, *Vector Product Format*, is an example of a widely used GIS standard [VPF92]. The standard is developed by an ad hoc organization in NATO, Digital Geographic Information Working Group (DGIWG), and is also known as DIGEST-C (Digital Geographic Exchange Standard, Annex C). The standard is freely distributed, and has been accepted in many application areas, not only within the military communities, and also outside NATO. The well known DCW (Digital Chart of the World, distributed as public domain data included necessary software), is an example of a product based on VPF.

VPF is a low level standard in the sense that it is possible to implement the standard more or less directly in an arbitrary relational data base. This is perhaps one of the reasons behind its popularity.

A brief review of VPF reveals the connection between the standard and the Paper Map Model.

- VPF is a representation of a planar projection of the reality, as indicated by the use of projection codes, [VPF92], Appendix G, table 69, page 153.
- On page 24, [VPF92], the statement 'Real-world objects are referred to as entities or features,...' shows that VPF follows the object approach to GI modeling, just as the Paper Map Model.
- The geometric primitives of VPF are nodes, edges and faces, [VPF92], page 26, and this corresponds to the points, curves and areas of the PMM.
- In [Kot92], page 31, it is focused on the 'levels of reality that are the foundation of the VPF view, thematic coverages and primitive geometry components within the themes'. This corresponds nicely with PMM, where each thematic element is located to the topography, and where a single topographic element can have several thematic interpretations.
- Regarding scale or resolution, VPF does not associate any such information at all to a certain map, other than optionally, and implicit, in 'data quality' tables, [VPF92], page 69. In cases where data quality information is not added, VPF is inferior to PMM. VPF do not handle phenomena varying over time, except from modeling them explicitly as different data sets.
- Different editions due to generalization has also to be represented in separate data sets, as in the PMM.
- As PMM, VPF is based on tiling the geographic space into a set of smaller, rectangular areas, [VPF92], page 38-39. Some mechanisms are provided to make the connections between consecutive tiles as seemless as possible.

As a conclusion, VPF complies closely to the Paper Map Model defined in section 2.2, except for minor, insignificant deviations.

In section 2.3, the attention was drawn towards the concept of generalization, one of the most basic characterizations of cartography. In a nutshell, generalization is the process of generating different variants of logically the same geographic reality, differing in scale, edition and time.

Still, VPF, and most of the other commonly accepted standards[5] do not consider this important issue explicitly. It offers only one solution to the problem, to produce the different generalizations and to describe and store them separately, even if they only differ in some minor details.

[5]One exception might be the SDM (Schlumberger Data Model), used by the petroleum industry in activities associated with E&P (exploration and production). This standard handles different versions of certain sets of data, like deviating interpretations of the same collection of seismic data. The mechanisms are simple and rudimentary, but still they try to encounter the problem explicitly, see [sch92], chapter 2, page 10.

The characterization of one single standard will of course not apply to all other existing standards[6], but a thorough analysis of the main standards would probably yield the same result, that they more or less describe different implementations of the Paper Map Model. Kottman's compact survey on GI standards support this assumption [Kot92].

As a conclusion, one might say that GIS today is dominated by the Paper Map Model. In the next chapter, it is claimed that this imposes severe limitations on GI systems, and that such systems should benefit from adopting a map model closer related to the real world, instead of relying on an efficient implementation of the traditional map concept.

In the next chapter, we will make the first general description of an augmented map concept.

[6]There are trends within the standardization bodies to develop the basis of GI systems into more comprehensive models. An example of this is the work done by CEN (Comité Européen de Normalisation), TC 287 (Working Group 287). The goal is to supply the European countries with a high level GI standard by the end of the century. Their revised 'philosophy paper' of June 1993 constitutes a promising foundation for a standard closer to reality [Com93].

Chapter 4

An Augmented Map Concept

In this chapter, we consider some of the challenges the core map model in a GIS is facing. We then make a sketch of an augmentation of the map concept.

4.1 Challenges

GIS is emerging as an important tool in a wide variety of application areas. Articles on the use of GIS in the following areas are found in [MGR91]:

- Socio-economic applications:
  - Land information systems
  - Car navigation systems
  - Market analysis
  - Population counting
- Environmental applications:
  - Soil information
  - Integration of geoscientific data
  - Multinational environmental GIS
  - Global GIS databases
- Management applications:
  - Land resources information systems
  - GIS in urban planning
  - Integrated planning information systems

These and other applications represent indeed new challenges for the core of GI systems, the map model. Traditional cartography has been used for similar tasks before, but not at the scale and complexity as the above application areas represent.

To meet the new challenges, a GIS has to offer a wide spectrum of functionality. Raper and Maguire make the following classification of GIS functionality [RM92]:

- Data capture

- Transfer
- Validation and editing
- Structuring
- Restructuring
- Generalization
- Transformation
- Query
- Integration
- Analysis
- Presentation

Some of these functions are already found in traditional cartography, but together they represent an integrated basis of functionality that would be difficult, if not impossible to fully realize within the Paper Map Model.

Due to the new challenges, Rhind, Goodchild and Maguire [RGM91] foresee that the recent research will supply:

- data models to handle 3-D and time dependence, and complex interactions between objects;
- support for complex analytical applications, including tracking of data lineage, tools for visual interactions with the stages in the analysis process, propagation of uncertainty;
- support for quality assurance and quality control(QA/QC) especially in GIS applications where litigation is a constant problem;
- support for multiple media - unstructured images, both digital and NTSC, text and sound;
- integration of GIS with the capabilities of GPS for data collection and compilers;
- tools for visualizing 3-D and time-dependent data;
- tools for data compilation, particularly in 3-D;
- improved techniques for conducting functional requirements studies, evaluating costs and benefits, benchmarking and other aspects of the GIS acquisition and project management process.

On the background of section 2.3 one might add:

- support for generalization by the means of structures for consistent and compact management of different scales and editions.

Indeed, such results imply the utilization of a reality representation far more sophisticated than that provided by the Paper Map Model.

The Ptolemiac paradox, presented in the *Introduction*, together with the recognition of generalization as on of the fundamental mechanisms in cartography and the challenges outlined above, constitutes the main motivation for the introduction of an augmented map concept.

4.2 Augmenting the Paper Map Model

Considering the discussion in the last sections, it becomes clear that GI systems need something more sophisticated than the Paper Map Model as a core map model. Today, the majority of GI systems are based on the PMM, and this map model is stretched to it's limits (and often beyond) in many applications. As briefly touched in the *Prologue*, this may cause inconsistencies and anomalies that leads to errors, malfunctions and other unpredictly misbehavior of the system.

In this thesis, the problem with the inadequate core model is encountered by *augmenting* the traditional map concept, the PMM, rather than starting totally from scratch. This approach takes advantage of the rich literature, experience and technology associated with traditional cartography, and merges it together with advanced data modeling techniques and state of the art technology in CAGD into an enhanced map model. One of the goals will be to maintain, where it is possible, the terminology and concepts used by the cartographic community. However, the concepts and terminology will be enhanced and supplied with the additions needed to meet the challenges and requirements provided by GI science.

The various addition and enhancements are merged with the PMM as defined in section 2.2, into the *Augmented Map Model* (AMM), which is defined at conceptual level below:

Definition 7 (Augmented Map Model, AMM) *An Augmented Map Model is a realistic representation of the real world (or part of it) with the following characteristics:*

- *Decomposition of the reality into cartographic objects, which represent both topographic and thematic information.*
- *Utilization of advanced and flexible structures, both spatial and thematic, which in combination represent the reality.*
- *Each thematic element is located to a distinct part of the topography. A part of the topography may have several thematic interpretations.*
- *Representation of the reality integrating a range of different scales or resolutions, at several moments or intervals in time, and in a multitude of editions, or generalized versions.*
- *Seamless representation independent of tessellations.*
- *Facilitates the usage of a wide range of presentations, independent of the internal representation of the reality model.*

In *Part III*, we will design the augmented map concept Metamap, founded on definition 7. An implementation of a limited Metamap case will be carried through in appendix B.

Summary

We opened this part by a general and indeed broad definition of the map. By defining and studying the Paper Map Model, we found that this was a narrow and limited map concept, mainly due to technological constraints.

Contemporary information technology has removed some of the barriers encountered in cartography since the Middle Ages. A geographic information system may be considered as a computer based atlas, adding some new functionality and capacity. Still, the Paper Map Model was shown to dominate the GI scene.

The GI research during the past thirty years has been dominated by problems concerning databases and how to store and index large amounts of information, efficient implementation of low level data structures, advanced visualization techniques and a never-ending discussion of the low level data structures raster vs. vector, both representations of primitive geometric objects. We find that this orientation somewhat diverts the attention from more fundamental issues concerning the core map model, for example how to handle the various aspects of cartographic generalization in a computer based environment.

The expectations to the performance of future GI systems, regarding capacity, application areas and functionality, implies that such systems have to manage huge amounts of inhomogeneous information, both topographic and thematic, in a compact and consistent manner.

We claim that the Paper Map Model is not capable of meeting these challenges, it was designed for use in a completely different technological setting. Still, we see the advantages of making additions and enhancements to the traditional map concept, rather than discard a rich and sophisticated tradition, evolved during a couple of millenniums.

This motivates the notion of augmenting the traditional map concept, in order to take full advantage of the information technology and thereby meet the somewhat overwhelming new challenges in geographic information management, within the framework of traditional cartography.

The rest of the thesis is devoted to investigate a possible augmented map concept, according to definition 7. The investigation is divided in two parts.

In *Part II* we study the consequences of cartographic generalization. We develop a method for homogeneous management of sets of variants, which we call the Multimodel concept. We pay special attention to structural issues associated to consistent and compact representation of variants.

The problem of integrating spatial and non-spatial information is treated in *Part III*, where we introduce the Metamap as an augmented map concept. Metamap is designed to fully exploit the flexibility provided by the Multimodel concept.

Part II

MULTIMODELS

Outline

As a first step towards an *augmented map concept*, according to definition 7, this part investigates methods and concepts for the integration of multiple representations of geographic information.

In order to increase the sparse supply of techniques dedicated to managing a set of variants of an initial model, we will introduce and develop, in some detail, the concept of the *Multimodel*, as a general mechanism for structuring a set of multiple representations of an initial model.

To motivate the elaboration of the Multimodel, we will give some examples of model variants encountered in a GIS setting. The variations in the models are due to the inherently multiple nature of geographic information, and to the processes of cartographic generalization, as described in *Part I*.

Before presenting our own approach to the problem of multiple modeling, we briefly outline a few existing methods developed in this research area. It will be focused on what categories of geographic information the different approaches encompass, an what kind of variations they cover. The notions of compactness and consistency are introduced in order to obtain an additional classification of methods in multiple modeling.

The Multimodel development is initiated by formulating the concept at a general level. Then a definition of the digital model is made in order to facilitate the exploration of the Multimodel in a computer aided setting. Certain properties differentiating various digital models are highlighted. On this basis, some fundamentally different categories of Multimodels are outlined. We concentrate on describing basic operations on the Multimodel, and discuss what degree of compactness and consistency the different Multimodels offer.

The Multimodel concept is then presented as an object model of a generic class library, which will be the basis for a limited Multimodel implementation in appendix A.

We close the part with the description of an informal methodology for Multimodeling. The method is applied to piecewise linear curves and surfaces, both examples of geometric objects highly relevant to geographic information systems.

Chapter 5

The Multiple Nature of Geographic Information

Geographic information may be characterized by its *multiple nature*. In section 2, it was illustrated that maps over the same part of the reality may differ substantially. The map is depending on a wide range of factors, such as map purpose, cultural background of the map compiler and available technology.

As indicated by the Ptolemiac Paradox stated in the *Introduction*, map makers have until recently been restricted to various methods of planar representations of the geographic reality. Definition 3 of the Paper Map Model characterizes this traditional map concept, which has been developed into highly sophisticated tools and products which indeed have served their purposes well.

However, the Paper Map Model is a single representation, in the sense that it is modeling the given geographic information in a single scale, as a single generalized edition and at a single moment or interval in time[1]. Thus, to achieve a more complete picture of the reality, one has to relate to a set of different maps. These maps will together yield a *multiple representation.*

The Ptolemiac Paradox implies that in spite of the information technology revolution experienced in the last part of our century, it may be claimed that no significant improvements have been introduced in computer aided cartography to handle the multiple nature of geographic information in a more structured and integrated manner. This is not entirely true, some achievements in this direction have indeed been made, and a few of these will be outlined in section 5.2.

5.1 Examples of Multiple Geographic Information

To motivate the further treatment of modeling multiple geographic entities, three examples on geographic objects will be investigated in order to illustrate variations over time, according to scale and due to edition generalization.

[1]However, some maps do indeed model time, e.g. a certain historic map describing the rise and fall of the Roman Empire.

5.1.1 Text

Assume we have a setting where a GIS is used to illuminate certain aspects of the historical development in Europe. One of the natural entities in such a context would be a *nation*. In addition to spatial descriptions of for example borders and coastlines, we may want to supply textual descriptions of the nations. Let us assume that the texts are essays written by different historians. The texts would certainly not have any inherent common structure beyond that they are collections of groups of characters.

Since our purpose is to model the development of the nations of Europe through a certain time span, it is natural to operate with a number of variants representing significant moments or periods. Each of the variants may differ more or less in the spatial description, but the differences in the textual description would probably be substantial, both by length and contents.

This set of different pieces of text constitutes together a multiple thematic description of the nation.

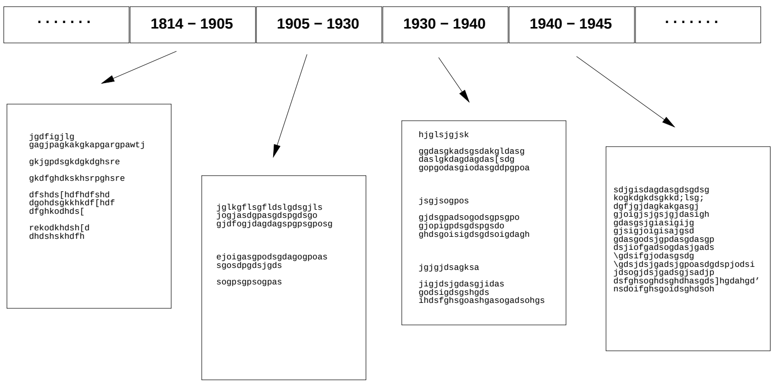


Figure 5.1: Textual description as thematic time-varying information

There are no obvious 'smart' way to structure this inhomogeneous set of models, other than arranging them in an array where the indexes correspond to the time span they represent, as illustrated in figure 5.1.

5.1.2 Parametric curves

Suppose we were modeling a railway network with the help of a GIS. The railway in our case is extremely simple, consisting of only one branch, from 'Beginville' to 'Stoptown'. In between, the railroad passes just outside 'Halfway Village'. A map is produced as shown in figure 5.2.

However, in this particular scale, the map gives the impression that the railroad passes *through* Halfway Village, and not *just outside*. To highlight the distinction, a generalized variant is produced by altering a little part of the parametric curve[2] representing the railroad,

[2] A parametric curve in the plane is defined such that for every value of a parameter t in a given definition interval $[a, b]$, there exists a point $(x(t), y(t))$ in the plane, where x and y are functions $x, y : [a, b] \to \mathbb{R}$. In contrast to a explicit curve defined by a real function, a parametric curve may describe loops, circles,

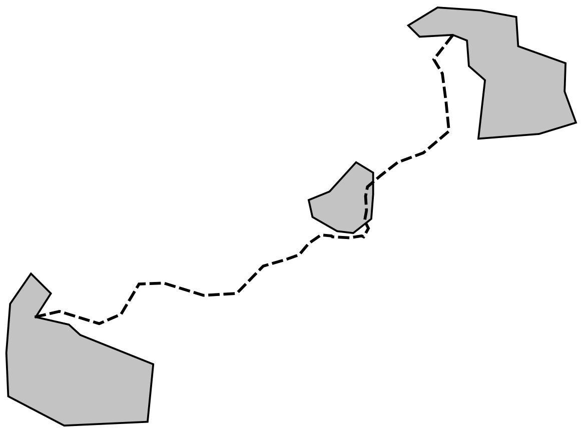


Figure 5.2: Initial map

as shown in figure 5.3.

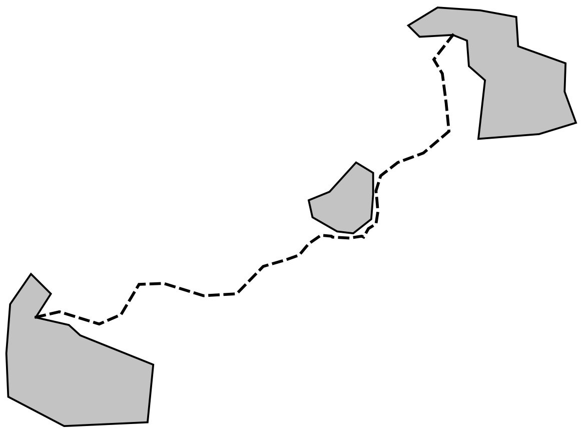


Figure 5.3: Generalized edition

If we wanted both editions of the map, we might store each map separately, as commonly practiced in most GI systems today. This is, however, not efficient, since the maps are identically except for the minor changes introduced by the generalization. Assume that the curves represent the variants of the railroad as vectors of equal length, where the elements are points in the plane. Indeed it is straightforward to subtract the original from the generalized vector, thus obtaining a difference or delta vector. The interesting part of the original vector is as follows:

Original	x	...	471	476	472	474	473	475	489	507	...
vector:	y	...	541	550	558	569	584	594	606	620	...

The corresponding generalized vector is as follows:

Generalized	x	...	471	476	484	486	485	484	489	507	...
vector:	y	...	541	550	555	568	580	590	606	620	...

self-intersections and other complex geometries.

The delta vector is characterized by the large amount of zero elements:

Delta x	$\cdots$	0	0	12	12	12	9	0	0	$\cdots$
vector: y	$\cdots$	0	0	-3	-1	-4	-4	0	0	$\cdots$

It is straightforward to realize that the generalized vector may be generated by adding the delta vector to the original vector.

Two advantages is achieved by this representation:

- *Compactness:* If we find a way to store a sequence of zero-points that takes less space than a corresponding sequence of arbitrary points, we obtain representation that is more *compact* than the explicit representation. See for example [Nel91] for an overview of such *data compression* techniques.
  We observe that the non-zero numbers in the delta curve above are 'smaller' than the corresponding explicit values. In fact, it would be possible to represent the delta values with fewer bits than the explicit representation. The integer -4 would require 3 bits including the sign, in contrast to 11 bits needed to represent 580. Similar effects may occur in delta representation of other objects than curves, and may be taken advantage of in order to store the differences in a compact manner.

- *Consistency:* If there is need for an updating of all editions of a map, let say the railroad station in Beginville was moved to another part of the town, we may say that the models are dependent. In our case, the changes need only to be applied to the initial model. Due to the construction of the other editions by summation of the initial model and the corresponding differences, the changes will automatically propagate to the various editions. Such representations will be referred to as *consistent*, if they 'automatically' maintain the dependencies between the various models. This is illustrated in figure 5.4. The delta models are visualized as differences added to the originals, where the nonzero sequences in the delta vectors are highlighted.

Section 6.3.1 will supply more details on compact and consistent representation.

5.1.3 Functions

In many applications it would be interesting to have information on the tidal variations of the sea. As known, these variations vary across the globe and are dependent of many factors, ranging from the constellation of the moon and the sun to meteorological conditions. For simplicity, let us assume that the tide is a diurnal variation, i.e. that it varies periodically within a cycle of exactly 24 hour, with only one ebb and one high water ocurring during the period. If the hour of the day is mapped to a real number in the range $[0, 24]$, the tidal variations may be described by the function $T : [0, 24] \rightarrow \mathbb{R}$, giving the sea level in meters relative to some reference level.

There are several ways to model this variation in a GIS. The tide could be embedded in the spatial description of the sea, as different time variants (see section 5.2.4 for a brief discussion on temporal modeling). Another possibility, which we will choose, is to let the

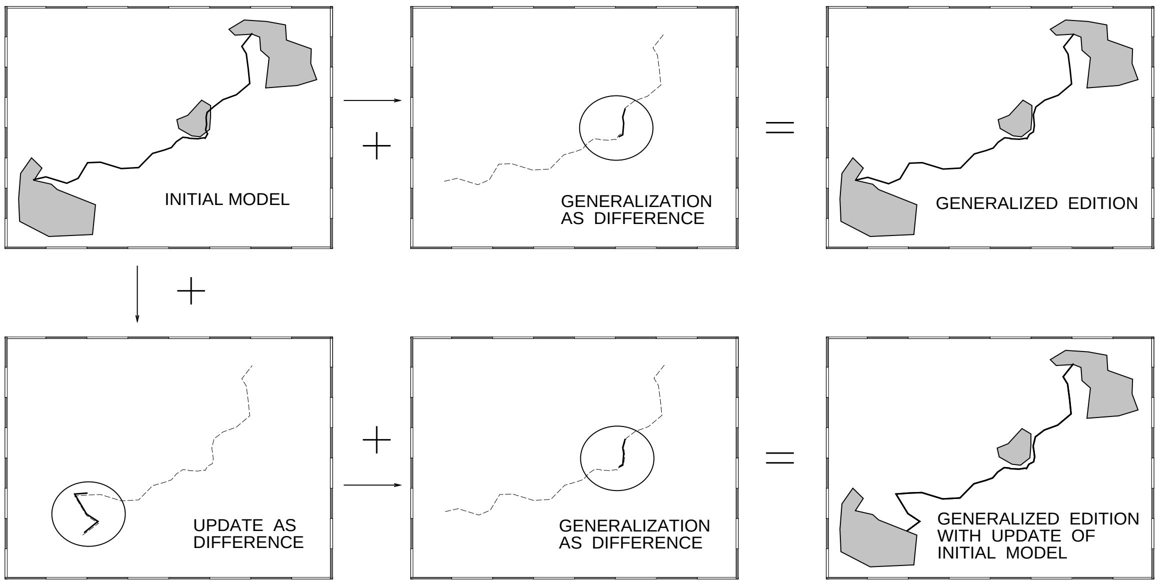

INITIAL MODEL
+
GENERALIZATION AS DIFFERENCE
=
GENERALIZED EDITION
+
UPDATE AS DIFFERENCE
+
GENERALIZATION AS DIFFERENCE
=
GENERALIZED EDITION WITH UPDATE OF INITIAL MODEL

Figure 5.4: Propagation of change in initial model

tidal function T be part of the thematic description of the ocean[3].

Assume in addition that our application should be used to produce digital sea charts in three different scales for the use in an ECDIS[4], one small scale version for deep sea navigation, let say 1 : 1.000.000, one scale for coastal navigation, 1 : 50.000, and a large scale harbor chart in 1 : 5000. Let us further assume that the system, by using the tidal function T, is able to produce versions of the three scales according to time of the day. To accomplish this task in an efficient manner, we may assume that the ECDIS system needs to access the tidal function in three degrees of accuracy corresponding to the different scales.

Figure 5.5 shows three variants of a tidal function T, described as piecewise linear functions defined by a vector of nine samples of $T(t)$, $t \in \{0, 3, 6, 9, 12, 15, 18, 21, 24\}$. The vector v_0 is the coarsest description corresponding to the smallest scale, describing a constant sea level of 0.7 meters above a given zero sea level.

The vector v_1 models T in medium resolution, showing that the tidal variation yields a high water at $t = 06.00$ corresponding to a level of 2.5 meters, and an ebb of -1.1 meters occurring at $t = 18.00$.

The most detailed variant of T, corresponding to the largest scale, is given by v_2. This vector models a sine-like function, giving an even more detailed picture of the tide.

The obvious way to represent the set of the three variants of T, is to store three arrays of function values explicitly. In this manner, no relations between the vectors are obtained.

[3]This is an example of the consequences of the duality of geographic information, see section 9.2. In many cases, a certain aspect of a geographic phenomenon may be considered both from the thematic and spatial point of view, resulting in equally 'good' descriptions.

[4]ECDIS: Electronic Chart Display Information System.

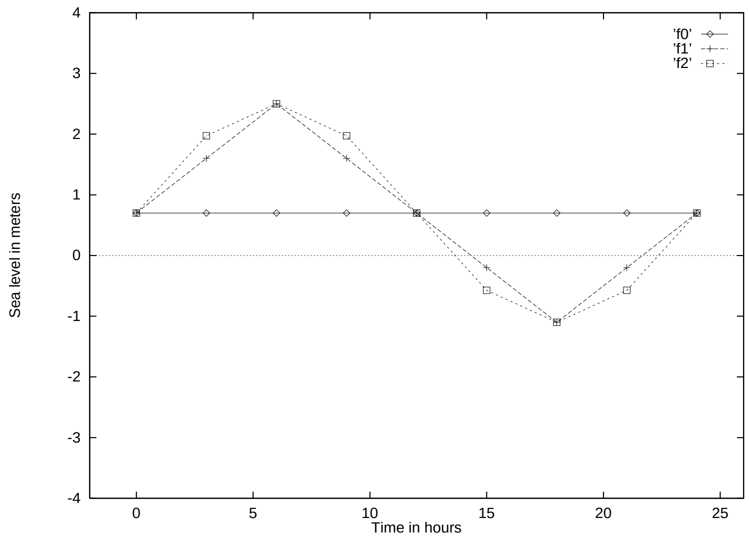


Figure 5.5: Description of tidal variation in three resolution

Delta representation

An alternative approach to the explicit representation illustrated in figure 5.5, is to *decompose* the vector-represented functions. Since the vectors are all of the same length, it is possible to define the delta-vectors (or difference-vectors) $\delta_1 = v_1 - v_0$ and $\delta_2 = v_2 - v_1$, as shown in figure 5.6.

If the delta-vectors δ_1 and δ_2 are stored together with the initial vector v_0, we have an alternative *implicit* representation of the three vectors as

$$\begin{aligned} &v_0 \\ v_1 &= v_0 + \delta_1 \\ v_2 &= v_0 + \delta_1 + \delta_2 \end{aligned}$$

The construction of v_2 as $v_0 + \delta_1 + \delta_2$ is illustrated in figure 5.7.

Consistency

Apparently, at first glance, the delta representation seems a little unmotivated and useless. However, suppose that reference zero level was changed[5] with a constant value $\hat{\delta_0}$, thus affecting all the representations of T. If we had an explicit representation, both v_0, v_1 and v_2 had to be updated to yield the new vectors $\hat{v_0} = v_0 + \hat{\delta_0}$, $\hat{v_1} = v_1 + \hat{\delta_0}$ and $\hat{v_2} = v_2 + \hat{\delta_0}$.

In contrast, with an implicit representation, only v_0 has to be updated, because both v_1 and v_2 is defined relative to the initial vector. In other words, v_1 and v_2 are dependent of v_0. The following shows that the structure is consistent, since the change $\hat{\delta_0}$ v_1 and v_2: 'automatically' propagates to

$$\begin{aligned} \hat{v_0} + \delta_1 &= v_0 + \hat{\delta_0} + \delta_1 &&= v_1 + \hat{\delta_0} &&= \hat{v_1} \\ \hat{v_1} + \delta_1 + \delta_2 &= v_0 + \hat{\delta_0} + \delta_1 + \delta_2 &&= v_2 + \hat{\delta_0} &&= \hat{v_2} \end{aligned}$$

[5]This is not unrealistic, the various hydrographic offices do in fact use a variety of such zero levels.

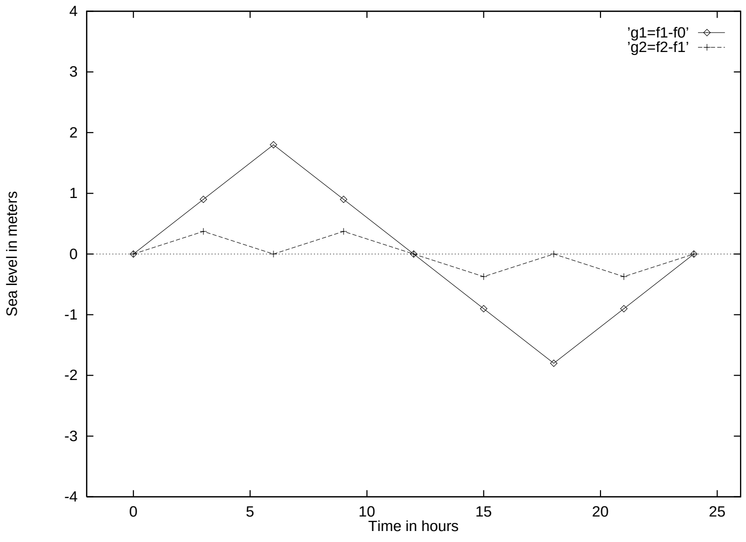


Figure 5.6: Delta vectors

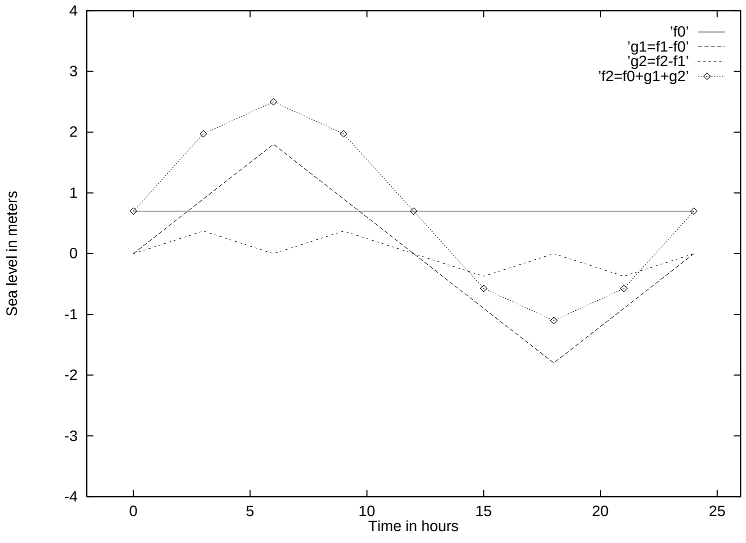


Figure 5.7: Adding initial vector and differences

Further, assume that the decomposition was motivated from the fact that the tidal function T varied slightly across the globe, and that this variation was due only to variations in the component δ_1, such that we for example had two variants of this component, δ_1 and $\hat{\delta}_1$. In an explicit representation, two main variants of the tidal function T, each represented in three scales, had to be stored. In a delta representation, however, only the $\hat{\delta}_1$ had to be stored[6] in addition to v_0, δ_1 and δ_2.

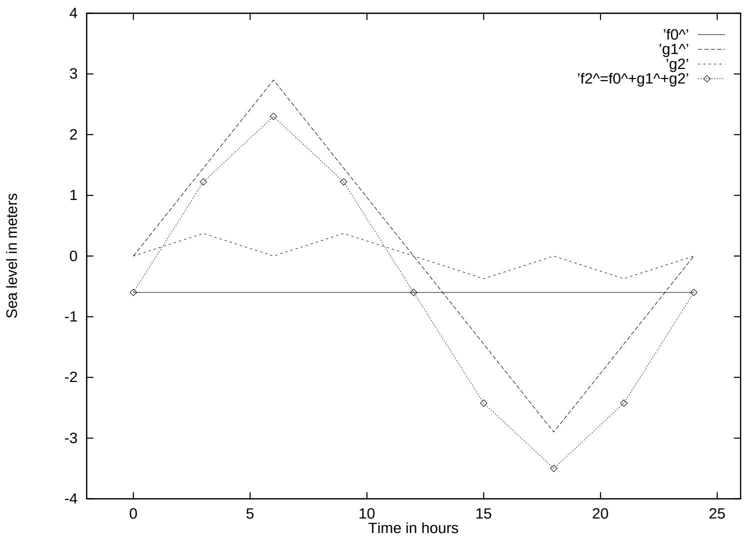


Figure 5.8: Change in initial vector and the δ_1 component

Figure 5.8 illustrates how changes of both the initial vector v_0 and of the component δ_1 affect the most detailed vector v_2, which is obtained by the addition $v_0 + \hat{\delta}_0 + \hat{\delta}_1 + \delta_2$. This example shows the consistency in the delta-representation, i.e. that changes and updates automatically propagate to depending models.

Compactness

There is another property associated with the delta-representation, concerning the amount of storage needed for the representations.

An explicit storage scheme uses in our example a storage equivalent to $3 \times 9 = 27$ function values. Without any modification, this is exactly the same that is needed with the delta-representation. However, due to the correlation of the different curves, we observe in figure 5.6 a fairly large amount of zeros in the delta-curves δ_1 and δ_2. In the difference δ_1, every second value is a zero, and δ_1 displays 3 zero values. In fact, the total number of non-zeros in the complete delta-representation of the three curves is 19, in contrast to the 27 values required by the explicit representation.

[6] Indeed, the new initial vector $\hat{v}_0$ and the new component $\hat{\delta}_1$ may also be expressed in delta representations, but in order to avoid confusion, we do not propose more complicated storage schemes at this point of the thesis.

If we were able to store the zeros and sequences of zeros using less space than required for the storage of corresponding arbitrary values, we would achieve a more compact representation. As mentioned earlier in the section, such methods exist, but it is beyond the scope of the thesis to investigate them more closely.

Data reduction aspects

There are still more redundant information in the delta-representation. The vector v_0 is essentially representing a straight line segment, needing just two values to be uniquely determined, leaving seven values redundant. Accordingly, δ_1 needs only four values, and δ_2 eight. Figure 5.9 illustrates these three *data reduced* representations of the initial vector and the differences. The points used in the reduced variants are marked.

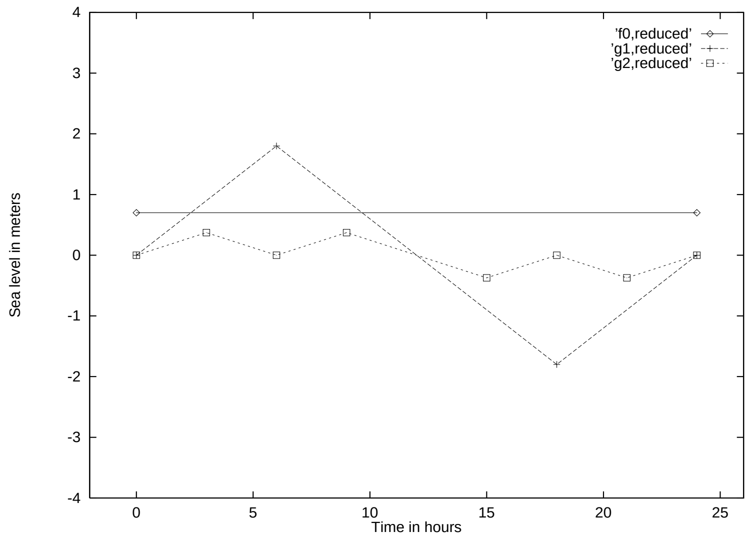


Figure 5.9:

The data reduction may seem a little out of place since it is no longer possible to subtract or add the new vectors, as they are of different length. Thus, we must introduce some extra machinery to overcome this minor obstacle.

Our goal is to bring the reduced variants over to the space spanned by all vectors of length 9 representing function values for all $t \in \{0, 3, 6, 9, 12, 15, 18, 21, 24\}$. In other words, we want to *refine* the coarse models into more detailed ones, such that the construction by summation may be applied. We want to design a *refinement operator* $\mathcal{R}$, such that

$$\begin{aligned} v_1 &= \mathcal{R}v_0 + \mathcal{R}\delta_1 \\ v_2 &= \mathcal{R}v_0 + \mathcal{R}\delta_1 + \mathcal{R}\delta_2 \end{aligned}$$

The operator has to 'lift' v_0, δ_1 and δ_2 into the space to which the original vectors belong. In this case the operator may be trivially defined by inserting 'missing' values computed by linear interpolation (see section 6.2.3 for more information on such refinement). This *decomposition* approach, included the notation, is adapted from mathematical decomposition theory, described in e.g. Dæhlen and Lyche [DL92], and applications of such techniques in computer aided cartography as outlined by Arge and Dæhlen [AD92]. In section 6.4.3 the decomposition concept is examined more closely.

This tidal function and its different representations provides an example of thematic information varying according to scale. In addition, the information was decomposed into a representation consisting of the initial representation and a set of delta-models. The components were reduced such that they were represented using fewer function values than initially, and a method for performing arithmetic operations on these reduced models was introduced.

In the next section, more general approaches to modeling multiple geographic information will be discussed.

5.2 Multiple Modeling

In the article 'Generalization of Spatial Databases', Muller [Mul91] discusses *scaleless* and *scale-dependent* databases as different ways of structuring a set of variants with different scales representing the same geographic area.

The *scaleless*, or *scale independent* database is a single representation of the most detailed variant of the map. From this representation it should, ideally, be possible to generate arbitrary views at any desired scale (or variant according to a given resolution) in real time. The real time scaling functionality is often referred to as 'zooming'. The complexity associated to the process of geographic generalization, as discussed in section 2.3.4, is most likely the main motivation for the following statement from Muller:

> This futuristic notion of scale-independent spatial databases has yet to be realized[7].

Questions concerning such real time generalization will not be addressed in the thesis.

A more realistic approach, according to Muller, is the *pseudo-scaleless database*. This is a pyramidal structure of different levels corresponding to certain scales. He stresses that each level should be accessible without duplication of data. In many ways this can be viewed as a discretization of an ideally continuous range of scales, see figure 5.10.

The third category of multiple representation, is the *scale-dependent* database, where the different scales are stored explicitly, resulting in an overhead of redundant information.

Muller highlights the following advantages of scaleless databases:

- Avoids duplication in storage.
- Enables production of flexible scale-dependent outputs ('any' scale is available, say 1:41067).
- Ensures consistency and integrity between the various scale outputs.

The principle of avoidance of duplication will in this thesis be referred to as compactness, and is considered as one of the main goals in modeling multiple structures.

Another major objective in our treatment of representation of multiple information, is the the notion of consistency. Basically, in a consistent multiple representation, an update in any part of the model will automatically propagate to those parts of the model that is depending

[7]Still, some believe that this concept is not too futuristic. Aasgaard [Aas92], as an example, has studied some selected real time generalization procedures of spatial objects.

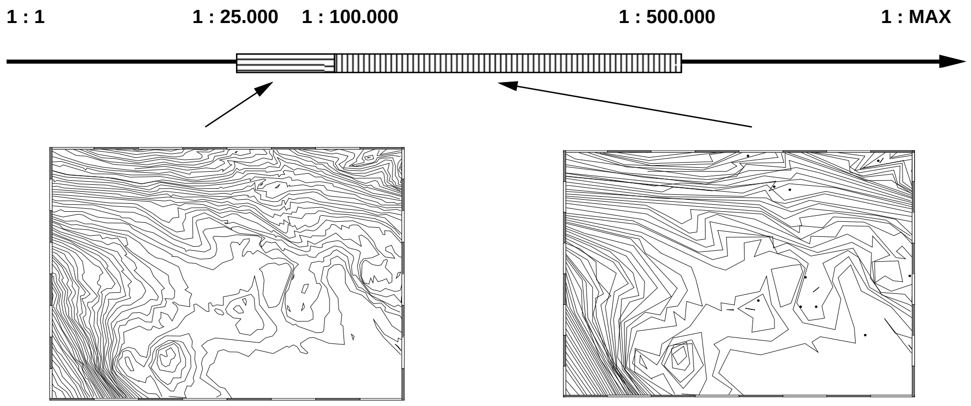


Figure 5.10: Pseudo-scaleless structure

on the updated part. The scale-dependent structure, where each level is represented explicitly and independently, is certainly not to be considered as consistent.

Muller restricts his characterizations to concern spatial objects varying according to scale. However, in this thesis Mullers classifications will be generalized to apply to non-spatial information, not only multiply represented regarding to scale, but also varying due to differences in time and editions. In other words, we are interested in representations of any kind of geographic information, that may give rise to a multiple structure when subjected to any form of cartographic generalization according to definition 4 in *Part I*.

Thus, in this thesis, the classification of the various multiple structures as scaleless, pseudo-scaleless and scale dependent, and the concept of consistency and compactness, will be applied to both spatial and non-spatial information, varying according to scale, time and edition.

In the next section, we will briefly outline a few existing approaches to multiple modeling of geographic information. Except for Kuhn and Bruegger [BK91], all the methods and structures are limited to handle spatial information only, and most often restricted to the most primitive objects such as polygonal curves.

5.2.1 Vector approaches

A vector based data model of geographic information is essentially using points, arcs and areas as the fundamental geometric entities of which more complex objects are constructed (see section 3.3.1).

Several methods for representing a vector based object in a range of scales corresponding to a set of given tolerances have been proposed. Two such methods are outlined below, one concerning piecewise linear plane curves, the other focusing on TINs (Triangulated Irregular Networks). Both methods handle scale generalizations only, temporal or edition based variants are out of the scope.

Multi-scale line trees

Several approaches have been made to structure multiple cartographic curves, or piecewise linear curves, as produced by successively performing scale generalization of an initial curve.

Jones and Abraham [JA86] suggest a 'multi-scale line tree' for this purpose. The well known method of Douglas and Peucker [DP73] is used to generate a set of curves according to a growing tolerance, such that each curve is represented by a subset of the points representing the successor.

These curves are organize by building a straightforward *tree structure* where the nodes contains pointers to nodes in the successing level. The bottom level, representing the original curve, or the finest resolution, contains pointers to the data representing the curve.

The method clearly yields a more compact structure, storing only the original curve and a set of pointers to the data, and consistent in the sense that a change in the original points will affects approximated versions, but the tolerance requirements may no longer be met after such a change. In addition, the structure provides fast access to the different layers of the three, and thereby efficient reconstruction of a certain approximant, since only nodes corresponding to those in the final curve will be visited (however, they may be visited more than once, see for example [PS85], Introduction, which includes a discussion of the tree and related data structures).

Hierarchical TINs

In computer aided cartography, the triangulated irregular network (TIN) has been a popular structure for representing digital elevation models. A TIN is a complete tessellation of a part of the plane into a number of mutually disjoint triangles of different size and shape. Each vertex in the tessellation is associated with a value corresponding to the elevation of the terrain. Since a plane in space is uniquely determined by three points, the TIN easily yields a piecewise linear surface that may represent the topographic surface of the Earth. Since the triangles may differ in size and shape, great flexibility is offered in modeling the multitude of different classes of terrain, ranging from undulating deserts to ragged mountain areas.

Since the late 1970's, much effort has been given to the design of structures capable of efficient handling of multiscale TINs. Floriani gives an overview over hierarchical surface models and presents a pyramidal TIN structure, which to a certain degree may be considered both compact and consistent [Flo89].

More details on TINs in general and their multiple representations in particular will be given in section 7.3.

5.2.2 Tessellation approaches

In a raster based model, the world is tessellated into mutually disjoint cells of equal size, exhausting the space to be described. The most simple (and most common) approach is to divide the plane into a set of squares. Each cell is then assigned a set of attributes, which are to be understood as constant within the cell (see section 3.3.1).

By successively grouping four cells into a new cell, a so called *quadtree* is constructed. Under the assumption that the most detailed tessellation, the basis model, contains 4^n quadratic cells, the quadtree will have $n+1$ levels. If the coarsest level, consisting of one cell, is defined to be level 0, level i corresponds to a tessellation consisting of 4^i cells, see figure 5.11. The quadtree structures are also widely used in digital image processing.

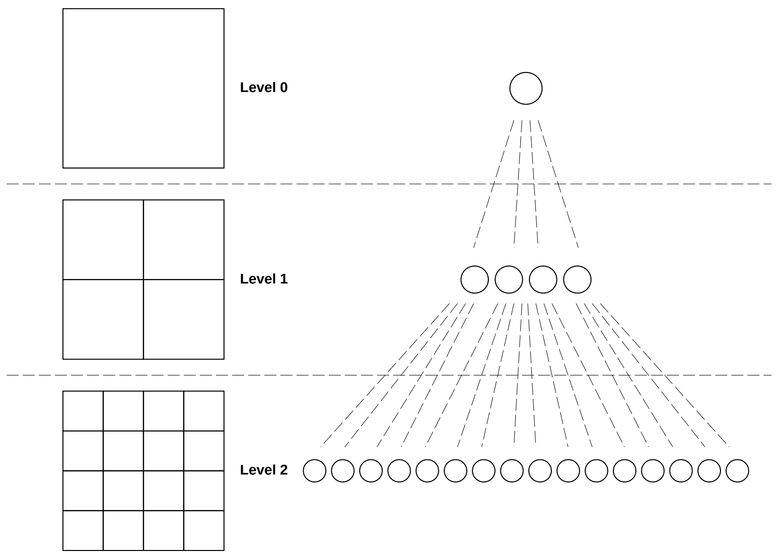


Figure 5.11: Quadtree

Each level of the quadtree may be defined to correspond to a certain resolution, which again may be associated to a specific scale. The quadtree then represents a discretization of a continuous range of resolutions, thus yielding a *multiresolution structure*.

By generalizing the two-dimensional quadtree to three dimensions, the octree is obtained. A volume partioned by 8^n cubes yields a $n+1$ level octree.

Other types of tessellations have also been proposed and implemented. Considering the spherical nature of the Earth, hierarchical tessellations by regular triangles have been claimed as more appropriate than quad- and octrees in global GI systems. For a closer look at the various tessellation approaches, see [GS92] and references therein.

The tessellation approaches only concerns spatial information at multiple scales. The temporal and generalization aspects are not covered by these methods. Tessellated structures are neither compact nor consistent, since each level is represented explicitly and independent of the other levels.

5.2.3 Topological approaches

The multiscale line tree and the hierarchical TIN focused on single geometric primitives, and do not take into account the relations between the various objects.

Topological models (see section 3.3.1) may be considered as enhanced vector models. The various geometric objects are interconnected by topological relations, thereby yielding a more coherent and consistent structure.

Kuhn and Bruegger [BK91] describes the *multiple topological representation (MTR)*, as a hierarchical structure of a multiscale topological model. The concept is introduced to overcome the problems of accessing large objects aggregated by many smaller entities, which in a single scale system may take unacceptable long time.

In the MTR approach, several layers of different topological structures of the same geographic area are constructed by the means of hierarchical relations. Kuhn and Bruegger emphasize that the extraction of data can be efficient at any level of abstraction, but that insertion of new objects is time consuming due to the redundancy in the structure, forcing new data to be inserted in several representations instead of just one. The MTR approach is then neither compact nor consistent.

5.2.4 Temporal reasoning

During the past 20 years, there has been a growing activity in the field of modeling time in general, often called *temporal reasoning*, and in particular how spatial objects are varying over time, frequently referred to as *spatio-temporal modeling*. For a general bibliography on this large research area, see [ATSS92]. Spatio-temporal problem in GIS has been addressed in various papers, see e.g. Al-Taha and Barrera [ATB90] and references therein.

The discussions in the field of temporal reasoning reveal a vast area of research, were only some few initial steps have been made towards a thorough understanding of the subject.

In this thesis, a very simple approach is made towards time modeling. Even if it may be possible to model time as a continuous process, this thesis is restricted to model time either as a discrete point or an interval in the one-dimensional time space[8]. The various geographic objects are considered to exist in different variants associated to a point or interval in time, see figure 5.12. Thus, investigating more sophisticated aspects of temporal reasoning is beyond the scope.

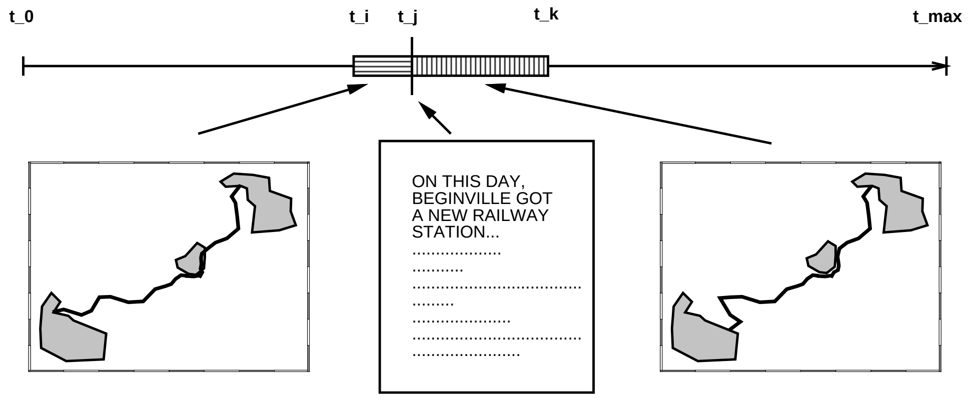


Figure 5.12: Simpel temporal modeling

This time slice approach has the advantage that each time variant may be thought of as special kind of generalization, according to definition 4, of the initial model representing the first moment in time. The development of the Multimodel concept in the sections to come will take advantage of this approach.

[8]In fact, we have already made an exception in modeling the tidal variation (section 5.1.3) as a *continuous* periodic function.

5.3 Integrated Modeling

In this context, integrated modeling refers to methods and structures able to represent multiple models varying not only according to one parameter, say a tolerance corresponding to a scale, but rather a set of variables. In geographic modeling, there is need for methods capable of structuring spatial and non-spatial information that vary according to both time, space and edition.

In the previous sections, examples have been given on contemporary approaches to multiple representations of spatial objects, varying according to scale or time. In spite of the importance of *edition generalization*, as pointed out in section 2.3.4, in both manual and computer aided cartography, there have been few attempts to design multiple representations *integrating* encompassing multiple editions. Still fewer works present *integrated* views of the three main aspects of generalization, i.e. scale, edition and time. In addition, little has been done to investigate multiple representations of the thematic, or non-spatial, information within GI systems.

However, an approach to integrated modeling is given by Guptill [Gup92]. Both spatial and thematic information is considered, varying over time and represented in several scales. The approach is somewhat limited, in the sense that it is investigated in detail how a specific geographical data model, the *Digital Enhanced Line Graph (DLG-E)*[9], can be implemented by two different relational database management systems. The model provides an integrated representation of a set of time and scale variations of both spatial and thematic entities. The method is neither consistent nor compact, since the variants are represented quite explicitly without any dependencies.

Arge and Dæhlen [AD92] proposes another approach to integrated modeling, outlining a method for structuring several variants of spatial information differing both according to scale and limited edition generalization. The high level structures and algorithms presented are motivated from mathematical decomposition, and offers indeed both a more compact and consistent representation. The Multimodel concept to be developed in chapter 6, may be viewed as a generalization and extension of the concepts introduced by Arge and Dæhlen [AD92], for example by incorporating time generalization and including non-spatial objects.

[9]Developed and used by U.S. Geological Survey.

Chapter 6

The Multimodel

In chapter 5 several examples of the multiple nature of geographic information were given. Various approaches towards the integration of versions based on the same model, referred to as multiple modeling, were briefly outlined. In this chapter, the concept of the *Multimodel* is introduced and developed as a mechanism for structuring multiple representations of geographic information, varying both according to time, edition and scale. With the help of the Multimodel technique, both spatial and non-spatial information can be handled, as far as possible, in a compact and consistent manner.

The use formalism and notation in this chapter is adapted from mathematical decomposition theory, as described in e.g. Dæhlen and Lyche [DL92], and application of such techniques in computer aided cartography as outlined by Arge and Dæhlen [AD92].

The Multimodel is firstly introduced and defined at a general, conceptual level. Then, in order to investigate the use of Multimodels in computer based systems[1], a definition of the *digital model* is given. A discussion is made on the possibility of applying the binary operations addition and subtraction, and the unary operations approximation and refinement, on a set of digital models.

Based on this discussion, a few categories of Multimodels are proposed as follows:

- Explicit Multimodel
- Implicit Multimodel:
    - Multi-edition models:
        - Pseudo-delta model
        - Delta model
    - Multi-scale models:
        - Selection model
        - Decomposed model
        - Multiresolution model

The chapter closes with a description of the *composite Multimodels*, exemplified by an object model of a composite Multimodel customized for use in geographic modeling, called *GeoModel*.

[1]The general Multimodel is not restricted to the digital domain.

6.1 The Multimodel Concept

As a first step towards the Multimodel, let us consider the scale problem in GIS. If we have a geographic object V, we can define the scale $1 : 1$ as the 'exact' representation of the phenomenon. Further, we have potentially an infinite set of variants of the object which corresponds to any scale $1 : x$, where $x \in [1, \infty\rangle$.

Consider a subset of $n + 1$ of all these possible scale variants of V as the indexed set $W = \{V_0, V_1, \cdots, V_n\}$, we may define a mapping $I : X \rightarrow W$, that for any value of the parameter k, and thus for any scale $1 : x$, associates one an only one model from W. We may typically define I such that every V_i corresponds to an interval in $[1, \infty\rangle$. In the example of the tidal function in section 5.1.3, the variants of the tidal function could be defined to correspond to the intervals $[1, 50.000\rangle$, $[50.000, 1.000.000\rangle$ and $[1.000.000, \infty\rangle$.

In this manner, we can structure a set of variants of the initial model V such that for every scale we may access a certain version that corresponds to the scale. This approach is consistent with the notion of the pseudo-scaleless structure described by Muller [Mul91] (see 5.2). We will call this structure a *Multimodel*, and define it as a set of parameters, a set of $n + 1$ models and the mapping that for every parameter picks a unique variant,

$$W = \langle\{V_0, V_1, \cdots, V_n\}, X, I\rangle.$$

The Multimodel is essentially a function that according to a given parameter (or set of parameters, see section 6.4.3), generates one and only one of many possible variants of a certain model. It is important to note that a Multimodel is self-contained, in the sense that it contains all information needed to generate the various variants. The Multimodel concept is formally defined as follows:

Definition 8 (Multimodel) *Assume we have a set of model variants Υ, and a set of parameters X*[2]*.*

The Multimodel is formally defined as

$$W = \langle\Upsilon, X, I\rangle,$$

where I is the mapping $I : X \rightarrow \Upsilon$, such that

$$I(x) = V \in \Upsilon \textit{ for all } x \in X.$$

A Multimodel is said to be dependent *if a change in one of the models in W is assumed to affect any of the other models, according to some defined relation among the models*[3]*. A dependent Multimodel is said to be* consistent *if such a change is 'automatically' followed up by the needed change(s) in the depending model(s).*

Note that definition 8 is not restricted to the digital domain. In order to enable us to study more closely the use of the Multimodel concept in computer systems, the next section gives a description and a definition of the *digital model.*

[2]In the scale problem outlined above, the set X was the interval $[1, \infty\rangle$, but in other cases the parameter set may be quite different.

[3]See the railway case in section 5.1.2 for an example of dependency.

6.2 Digital Models

The text, the parametric curves and the functions investigated in 5.1, are all examples on *digital models*. This section formalizes the notion of the digital model. The term 'digital' is used to emphasize that we are concerned with models that are possible to implement in a computer based system. We will investigate some possible binary and unary operations on sets of digital models.

6.2.1 Definitions

A digital model is essentially a finite, ordered set of attributes or data, and a transformation that maps the attributes to a set of 'real world models'. The transformation may typically consist of a computer program or subroutine that decodes and transforms the attributes into a comprehensible presentation.

Let $\boldsymbol{a}$ denote an ordered vector of n attributes $[a_1, a_2, \ldots, a_n]$. Further, define A_i to be the set of all possible instances, or values, of the specific attribute a_i, and let Λ be the set of all possible attribute vectors with the same number and types of attributes, $\Lambda = \{A_1 \times A_2 \times \cdots \times A_n\}$. The set Λ will occasionally be referred to as the *attribute space*. Note that some or all of the attribute sets A_i's may be identical. In the example of the parametric curve in section 5.1.2, all the attributes were elements in $\mathbb{R}^2$, and thus the A_i's were equal.

Further, let V_{world} be the 'real world model', e.g. the text (section 5.1.1) as printed on the map, or the function (section 5.1.3) as it is plotted on the screen. Let Υ_{world} be the set of all possible 'real world model' models.

A transformation T is then defined as a mapping

$$T : \Lambda \to \Upsilon_{world}$$

that takes the attribute vector as an argument and transforms the data to a 'real world' model: $T(\boldsymbol{a}) = V_{world}$.

A digital model V is formally defined as follows:

Definition 9 (Digital Model) *An attribute vector $\boldsymbol{a} = [a_1, a_2, \ldots, a_n] \in \Lambda$ of digitally represented features a_i's together with a transformation $T : \Lambda \to \Upsilon_{world}$, is called a Digital Model. Formally it is defined as the pair*

$$V = \langle \boldsymbol{a}, T \rangle.$$

The definition is recursive in the sense that the attributes a_i's may be fully defined digital models:

$$a_i = \{\boldsymbol{\alpha}, \tau\}, \quad \text{where } \boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_\nu]\,.$$

The attribute vector $\boldsymbol{\alpha}$, and the transformation τ is defined correspondingly to T and $\boldsymbol{a}$.

A digital model by this definition is a discretization of an analog or 'real world' model. We will some times use the terms V and $\boldsymbol{a}$ as they were equivalent, and let the context decide which to use. The transformation T is in many cases quite trivial, as in the example of textual models in section 5.1.1.

Before we start investigating different operations on digital models, some definitions concerning the 'likeness' of digital models will be stated.

Definition 10 (Likeness of digital models) *Assume we have two models*[4],

$$V = \langle \boldsymbol{a}, T \rangle \quad and \quad V' = \langle \boldsymbol{a}', T' \rangle.$$

The attributes are elements in the attribute spaces Λ and Λ'. We characterize the models as outlined below:

- **Non-compatible:**
  The two models V and V' are said to be non-compatible *if $T \neq T'$*[5].
  Two digital models, one representing an image, the other a piece of text, are clearly non-compatible since their transformations indeed are different. The two transformation are defined over different sets of attribute spaces Λ and Λ'. However, note that models with attribute vectors from the same space, and even with identical vectors, may still be non-compatible. An example of this is an array of real values representing a function sampled according to a given set of argument values. We may have that T interprets the attributes as a piecewise linear function, while T' constructs a cubic interpolant to the data. These two digital models are considered non-compatible.

- **Compatible:**
  Accordingly, the models are compatible *if their transformations are identical, $T = T'$. Note that the attribute vectors may be of different attribute spaces, $\Lambda \neq \Lambda'$, assuming that the transformation accept both the attribute vectors in question as arguments. Two vectors of points in the plane, but of different length, supplied with a transformation that interprets the vectors as piecewise linear curves, are considered to be compatible.*

- **Equivalent:**
  If the models share a common transformation, and have attribute vectors from the same attribute space, $T = T'$ and $\Lambda = \Lambda'$, then they are defined to be equivalent.
  Two vectors of points in the plane, with the same number of elements, together with a transformation that generates piecewise linear curves, are equivalent. Note that the values of the different attributes may vary from one model to the other.

We have now established the framework needed for the investigation of operations on digital models.

6.2.2 Model arithmetics

The parametric curve (section 5.1.2) and the function (section 5.1.3) may both be considered as digital models. In order to construct difference models and generate variants as sums of an initial model and corresponding differences, we applied the operations *subtraction* and *addition*.

We were able to perform these arithmetic operations of two reasons:

[4]The term 'model' will be used instead of the more precise 'digital model', if the context do not require a distinction.

[5]Two transformations T and T' are said to be equal if $T(\mathbf{a}) = T'(\mathbf{a})$ for all $\mathbf{a} \in \Lambda$.

- For each of the attributes, i.e. points in plane and reals, we could define addition and subtraction in a meaningful manner, trivially in the case of the real values defining the functions, and a bit more advanced as pointwise operations in the parametric curve example.
- The number and type of the attributes of the models in question were identical, thus the models were equivalent according to definition 10. It was then straightforward to define arithmetic operations on the attribute*vectors* as element-wise addition and subtraction. In the case of the parametric curve the addition of two attribute-vectors were defined as
$$\boldsymbol{a} + \boldsymbol{a}' = \{a_1 + a'_1, a_2 + a'_2, \cdots, a_n + a'_n\},$$
where again the addition of the elements $a_i = (x_i \in \mathbb{R}, y_i \in \mathbb{R})$ was defined as
$$a_i + a'_i = (x_i + x'_i, y_i + y'_i).$$

Based on these examples, we define addition and subtraction of digital models as follows:

Definition 11 (Model arithmetics) *Assume that we have two equivalent models, according to definition 10,*
$$V = \langle \boldsymbol{a}, T \rangle \quad \text{and} \quad V' = \langle \boldsymbol{a}', T \rangle,$$
Assume further that the set Λ, in which the attribute vectors $\boldsymbol{a}$ and $\boldsymbol{a}'$ are members, is a group[6] *under the addition operator '+'.*

We trivially define the addition operator '+' as:
$$V + V' = \langle \boldsymbol{a} + \boldsymbol{a}', T \rangle.$$
The subtraction operator '−' is defined, as common, as addition of the inverse element:
$$V - V' = V + (-V'),$$
where the inverse model $-V'$ has an attribute vector of inverse attributes $-\boldsymbol{a}' = \{-a_1, -a_2, \cdots, -a_n\}$.

The new model $V + V'$ (or $V + V'$) is then equivalent *to both V and V'.*

[6]The notion of the group is a fundamental mathematical structure, simple but far from uninteresting. The formal definition of a group is given here, and readers interested in details of group theory is referred to e.g. Herstein [Her75].

Definition 12 (The Group) *The nonempty set Υ is forming a group, if there is a binary operator '+' such that*

1. $V, V' \in \Upsilon \Longrightarrow V + V' \in \Upsilon$ *(Closure).*
2. $V, V', V'' \in \Upsilon \Longrightarrow V + (V' + V'') = (V + V') + V''$ *(Associativity).*
3. $\exists 0 \in \Upsilon \mid V + 0 = 0 + V = V \quad \forall V \in \Upsilon$ *(Existence of identity).*
4. $\exists (-V) \in \Upsilon \mid V + (-V) = (-V) + V = 0 \quad \forall V \in \Upsilon$ *(Existence of inverses)*

The new model $V \pm V'$ is equivalent to V and V' since it inherits the transformation T by definition 11, and since the new attribute vector $\boldsymbol{a}' + \boldsymbol{a}$ is a member of the attribute space Λ, due to the closure of the Λ under $+$.

As we have seen, the ordinary addition and subtraction operations are easily applied to equivalent models where the attribute vectors are members of a group. A set of models with properly defined arithmetic operators will be called *difference* models.

Many of our familiar mathematical structures are indeed groups, like the integers, the reals, the rationals, all under the common addition, vectors and matrices under their respectively defined addition and functions of various kinds.

In the next section, we will investigate operations of a more complex nature, the approximation and the refinement.

6.2.3 Approximation and refinement

In section 5.1.3, we studied a set of three variants of a function simulating tidal variations in a primitive manner. They were regarded as models of different resolution or scale. Let us investigate the tidal function in a slightly different setting.

Consider the function f and the data reduced, or approximated variant[7] $f^\star$, see figure 6.1. Both functions may be considered as digital models, represented as two curves with point vectors $\boldsymbol{a}$ and $\boldsymbol{a}^\star$, with 9 respectively 5 points in $\mathbb{R}^2$. Both models share the transformation T_{plc}, which interprets the vectors as piecewise linear curves. Note that this transformation is able to handle an arbitrary vector of points $\{q_1, \ldots, q_n\}$. The two models are clearly

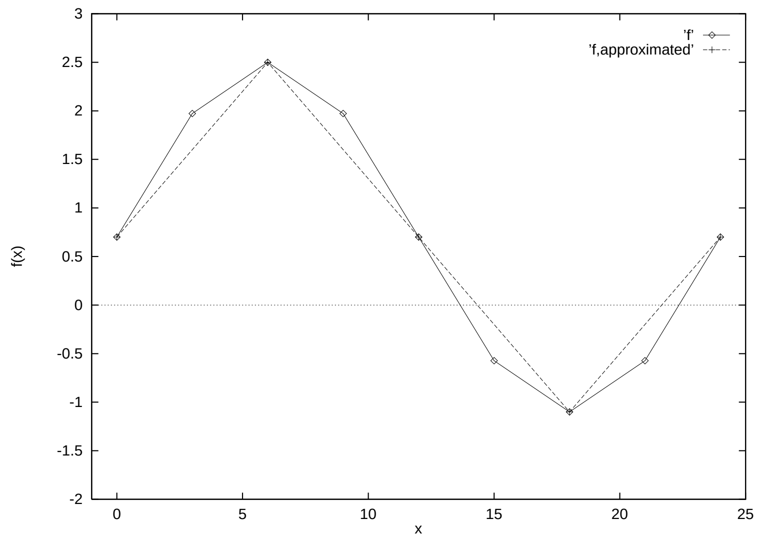


Figure 6.1: Function an its approximation

compatible according to definition 10. However, the functions are not equivalent, since the attribute vectors are elements in different attribute spaces. The initial model f, is represented

[7]Throughout the thesis, the notation $\cdot^\star$ will be used to emphasize that the model in question is approximated.

by nine points representing samples of the tidal function,

$$\boldsymbol{a} = \{(0, 0.70), (3, 1.97), \cdots, (21, -0.57), (24, 0.70)\},$$

and the approximated and data-reduced variant has five attributes,

$$\boldsymbol{a}^\star = \{(0, 0.70), (6, 2.50), (12, 0.70), (18, -1.10), (24, 0.70)\}.$$

We may define the attribute space Λ, from which $\boldsymbol{a}$ is an element, as follows:

$$\Lambda = \{(0, g(0)), (3, g(3)), \cdots, (21, g(21)), (24, g(24))\},$$

where g is any univariate function defined on the interval $[0, 24]$, and the attribute space $\Lambda^\star$ as spanned by

$$\Lambda^\star = \{(0, g(0)), (6, g(6)), (12, g(12)), (18, g(18)), (24, g(24))\},$$

where again g is any arbitrary function on the interval in question.

The process of approximating and reducing the data of the model f, may be formalized as a mapping

$$\mathcal{P} : \Lambda \to \Lambda^\star,$$

that picks every second point from f, starting with the first, thus reducing or approximating the attribute vector of f to a truncated vector in another attribute space of 'lower' dimensionality. The notation $\mathcal{P}$ indicates that the approximation process is to be regarded as a *projection* from one attribute set down to another set consisting of vectors of less length.

The models f and and the approximant $f^\star$ are not compatible, and we are unable to e.g. take the difference $\boldsymbol{a} - \boldsymbol{a}^\star$. In order to make $f^\star$ equivalent to f, we introduce the refinement operator $\mathcal{R}$ as the mapping

$$\mathcal{R} : \Lambda^\star \to \Lambda.$$

This refinement[8] operator may be defined by linear interpolation, such that

$$\mathcal{R}\boldsymbol{a}^\star = \boldsymbol{a}^\diamond = \{a^\star{}_1, \tfrac{a^\star{}_1 + a^\star{}_2}{2}, a^\star{}_2, \cdots, \tfrac{a^\star{}_4 + a^\star{}_5}{2}, a^\star{}_5\}.$$

Figure 6.2 illustrates the insertion of points resulting in a refinement of the approximant $f^\star$. The attribute set Λ is spanned by all possible function sampled on the given argument values. Since $\mathcal{R}\Lambda^\star$ consists of the same functions, but with some of the function values restricted to the linear interpolation scheme outlined above, we clearly have that $\mathcal{R}\Lambda^\star \subset \Lambda$.

We now generalize the examples above to a definition of decomposable models:

Definition 13 (Decomposable model) *A model is said to be decomposable, if we are able to approximate and refine it, as follows:*

[8]We will use the notation $\cdot^\diamond$ to emphasize that the model in question is refined.

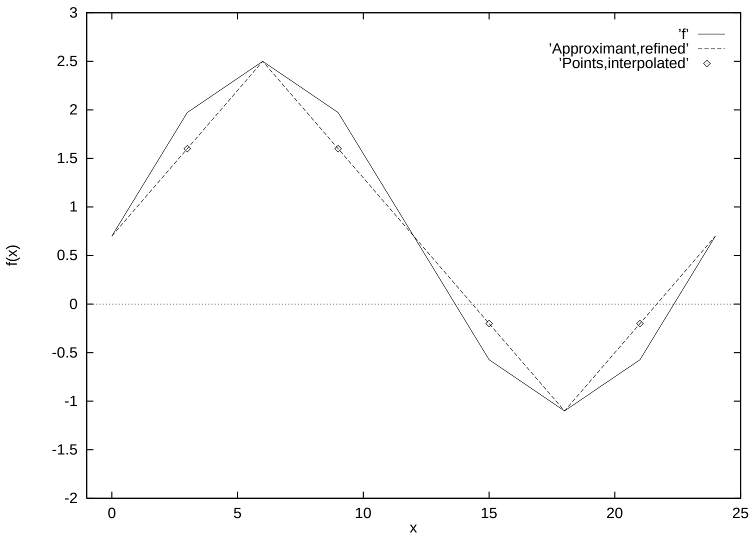


Figure 6.2: Refinement of approximant

⊕ Approximation *of a digital model $V = \langle \boldsymbol{a}, T \rangle$ is a mapping of the attribute vector* **a** *as an element of the attribute space Λ_i, to the attribute space Λ_j, under the assumption that the transformation T are valid for elements in Λ_j,*

$$\mathcal{P}_i^j : \Lambda_i \to \Lambda_j,$$

such that the approximant $V^\star$ is generated as

$$V^\star = \langle \mathcal{P}_i^j \boldsymbol{a}, T \rangle.$$

$\mathcal{P}_i^i$ is the identity mapping. The approximant is compatible, according to definition 10, to the original model, but not equivalent.

⊕ Refinement *is a 'lifting', or mapping, of a digital model $V = \langle \boldsymbol{a}, T \rangle$ of the attribute set Λ_j to a 'larger' attribute space Λ_i,*

$$\mathcal{R}_j^i : \Lambda_j \to \Lambda_i,$$

under the assumption that T is valid in Λ_i. The refined model $V^\diamond$ is produced as

$$V^\diamond = \{\mathcal{R}_j^i \boldsymbol{a}, T\}.$$

Note that the approximated and then refined model $\mathcal{R}_j^i \mathcal{P}_i^j V$ is equivalent to model $V \in \Lambda_i$, but not necessarily equal to this, since $\mathcal{R}_j^i \mathcal{P}_i^j \Lambda_i \subset \Lambda_i$.

The design of approximation and refinement operators is indeed not a trivial task. In the example of the tidal function, the operators were quite simple. More sophisticated and complex operators will be briefly investigated in section 7.3 and 7.2.

In many applications, it is interesting, and frequently necessary, to estimate how 'good' a certain approximation is. That is, we need a tool for 'measuring' the 'distance' between two models V and V', which is to be considered as the error of the approximation. For instance, in the case of the tidal function, the error might for instance be defined to be the largest absolute value of the differences of all corresponding function values (let $a_i \in \boldsymbol{a} = (x_i, y_i)$):

$$\text{distance } (f, \mathcal{R}f^\star) = \max_{i=1}^{9} |y_i - y^\diamond{}_i|.$$

As a tool for measuring the error of an approximation, we use the notion of the *metric* to define a set of *metric models*:

Definition 14 (Metric models) *A set of models Υ is said to be* metric *if there exists a function $\rho : V \times V \to \mathbb{R}$ such that for all V, $V'$and V'' in Υ the following holds:*

1. $\rho(V, V') \geq 0$.
2. $\rho(V, V') = 0 \Longleftrightarrow V = V'$.
3. $\rho(V, V') = \rho(V', V)$.
4. $\rho(V, V') \leq \rho(V, V'') + \rho(V'', V')$.

The distance function distance $(f, \mathcal{R}f^\star)$ defined above is easily verified to be a metric. In a set of metric, decomposable models, we will be able to generate approximations and estimate the associated error as the 'distance' ρ between the original and the approximated model. In many application areas, and indeed GI science, this ability is of vital importance.

The discussion and various definitions concerning the digital models will play an important role in the elaboration of the Multimodel concept during the next few sections.

6.3 Digital Multimodels

In section 6.1 definition 8 gave a description of Multimodel that was not restricted to the digital domain. Since we in section 6.2 have studied the digital model, we can now give a more detailed description of the digital Multimodel.

A digital Multimodel is essentially a finite, discrete and indexed set of digital models, $\{V_0, V_1, \cdots, V_n\}$ associated with a set of index parameters $X = \{0, 1, \cdots, n\}$. The function I is the trivial mapping

$$I(x) = V_x.$$

In other words, the digital Multimodel is an indexed set of models, which may be directly accessed according to the index. In many applications, as in the example with the scale intervals in section 6.1, we have to introduce an additional mapping from an arbitrary set of parameters to the index set X. In the further treatment of the Multimodel, we will assume that such transformations are provided.

Since the mapping I is trivial, the digital Multimodel is reduced to:

$$W = \langle V_0, V_1, \cdots, V_n \rangle.$$

Note that the model set W may be *represented* in various ways, assuming that the semantics of the ordered set are maintained, i.e. that it is possible to perform operations as insert, delete, access and so on.

6.3.1 Aspects of Multimodels

In this section, we take a closer look at some aspects of Multimodels that will come at hand when developing the concept at a more detailed level.

Multimodel operations

In an application, a MultiModel.has to offer a broad range of functionality. However, at this level we will only be interested in three kinds of procedures, which all are highly dependent on the internal representation of the set of models.

Assume that we have a Multimodel W, we may want to design procedures for performing basic operations such as initialize, insert, delete, append, merge and so on. However, to illuminate certain fundamental aspects of Multimodels, we will restrict the investigation to concern procedures for inserting new models, for accessing a particular model, and for updating or changing an existing model:

- `insert(`V`)`: For simplicity, we restrict the insertion to append the model V to the existing set W, such that we get the new set $W' = \{V_0, V_1, \cdots, V_n, V_{n+1} = V\}$.
- `reconstruct(`k`)`: The reconstruction, or accessing, procedure takes a parameter $k \in \{1, 2, \cdots, n\}$, and based on this it shall produce and return the model V_k from W. This is to be considered as the most fundamental operation on a Multimodel.
- `update(`k, V`)`: This procedure performs an updating of the model corresponding to the parameter value k, such that model V replaces the old model[9]. Since k corresponds to model V_k, we get the new model set $W' = \{V_0, V_1, \cdots, V_k = V, \cdots, V_n\}$.

Based on the characterization in section 6.2.1 of models as non-compatible, compatible or equivalent, we will propose three main classes of Multimodels.

Consistency and compactness

In this section, we introduce the notions of *consistency* and *compactness*. In definition 8, a Multimodel is said to be dependent if a change in one model is supposed to imply changes in other models. In the digital domain, we make a restricted definition of dependency. The digital Multimodel $W = \{V_i\}$ is said to be dependent if a change in model V_i is supposed to affect the consecutive models $\{V_{i+1}, V_{i+2}, \cdots, V_n\}$, according to some predefined rules.

Further, W is said to be *consistent* if the representation is such that a change in V_i implies the necessary updates of the consecutive models, without taking any explicit actions against

[9] Note that this kind of change is irreversible. If it is important to maintain the 'history' of a set of changes, it would be better to perform the update as the introduction of a new *edition* of the model.

each of these models. This is in many applications a desirable ability, recall e.g. the example of the railway in section 5.1.2. See section 6.4.2 and 6.4.3 for further details on dependency and consistency.

Another important issue, certainly in GI systems but also in other areas, is to represent the information handled by the system as compact as possible, in the sense that the data takes as little storage space as possible.

The most straightforward way to store the data in a Multimodel, is to represent each model explicitly. If a certain representation of the set of models takes less storage space than the explicit representation, we say that the representation is more *compact* than the explicit model. More formally, let

$$||V||_{bit}$$

be the number of bits required to store the *representation* of the model V digitally, and let

$$||W|| = \sum_{i=1}^{n} ||V_i||_{bit}$$

be a measure of the space required for the storage[10] of W. The representation $W_{alternative}$ is more *compact* if

$$||W_{alternative}|| < ||W_{explicit}||,$$

where $W_{explicit}$ is the explicit representation of W.

As experienced in section 5.1.2, a certain representation, the delta model scheme, yielded a structure where the models of the curves contained a large amount of *zeros*. There exist many techniques for storing such objects using less space than storing an arbitrary corresponding model, see e.g. [Nel91]. Thus, if a certain Multimodel structure contains representations with a larger content of zero or void attributes than the originals, we consider such a structure to be inherently compact.

Another potentially compact Multimodel is a representation where the attributes on the average are of 'smaller magnitude' than in the original. In the parametric curve example in section 5.1.2, we saw that the delta representation of the point vectors were of small magnitude compared to the explicitly represented vectors.

Based on the characterization of models as non-compatible, compatible or equivalent, we propose three main classes of Multimodels, the *explicit Multimodel*, the *multi-edition model* and the *multi-scale model*. The last two structures may be considered as variants of the *implicit Multimodel*. These Multimodels will differ in the way the model set is structured and represented, and in the algorithms associated to the three basic operations insert, reconstruct and update. We will also make comments on the degree of consistency and compactness they offer.

6.4 Some Categories of Multimodels

The collection of the models in W may be represented and implemented in various ways, as long as the representation obeys the semantics of the ordered set, i.e. that it supplies the

[10]With this definition, we don't take in account the additional overhead introduced by storing a number of models as a collection.

needed operators to maintain the indexed set. In our case we have limited the operations to insertion, reconstruction and updating.

It is possible to identify three main classes of representation of a Multimodel W:

- *Explicit* representation is a direct representation of each and one of the models, without any relations between the representations of the variants. This will also be referred to as the *trivial* representation of a Multimodel.
  Implicit representation is the characterization of any method that do not rely exclusively on explicit techniques, but represent the models using some sort of relations between the variants. We have two main categories of implicit Multimodels:
- A *multi-edition model* is a collection of *equivalent* models, i.e. sharing the same transformation and having attribute vectors from the same space. The multi-edition structure is useful when it is interesting to maintain a set of variants of an initial model, all described with the same degree of accuracy, or with the same resolution. Within geographic modeling, the multi-edition models should be useful for representing the time and edition variants produced in a cartographic generalization procedure (recall definition 4 in section 2.3.4).
- *Multi-scale models* are essentially sets of compatible and non-equivalent models. The different models are to be considered as variants of an initial model with decreasing accuracy, represented with less data than the original model. Such structures are particularly interesting when maintaining the essentially same geographic information in different scales or resolutions.

The trivial Multimodel and variants of multi-edition and multi-scale models are outlined in the next sections. The list of the techniques described is by no means exhaustive, many other approaches may be equally interesting or important. The main goal is to suggest a few quite different directions which may lead to practical and efficient Multimodel structures.

We pay special attention to the design of the operations insertion, reconstruction and updating, and of the degree of compactness and consistency they offer. For all the examples, we assume the Multimodel is indexed as

$$W = \langle V_0, V_1, \cdots, V_n \rangle.$$

We start with a closer look at the trivial Multimodel structure.

6.4.1 Explicit Multimodel

The explicit Multimodel will be useful when the collection of models in question is a set of ‘chalk and cheese’, i.e. that, in spite any apparently likeness of the models, they may neither be regarded as delta-models nor decomposable models.

We do not have much freedom in the design of a Multimodel of such inhomogeneous models. The text case in section 5.1.1 was an example of this class of models, even if the different pieces of text in fact were compatible.

Assume we have a set of $n+1$ explicitly represented models

$$W = \langle \{V_i\}_{i=0}^{n} \rangle .$$

The insertion, reconstruction and updating procedures become quite trivial, and may be expressed algorithmically as follows[11]:

Algorithm 6.1 *Insertion in explicit Multimodel*

```
insert($V$)

1.    if $W == \emptyset$
      1.2   $V_0 = V$;      $n = 0$;
2.    else
      2.1   $V_{n+1} = V$;
      2.2   $n = n + 1$;
```

Algorithm 6.1 simply appends the new model explicit to the Multimodel. Generating a particular model is equally simply performed as a direct access of the model:

Algorithm 6.2 *Reconstruction in explicit Multimodel*

```
reconstruct($k$)

1.    return $V_k$;
```

If the Multimodel is dependent, according to definition 8, the updating procedure has to check all successors to maintain consistency. The procedure
`check(this_model, changed_model)` performs the check and the possible changes:

Algorithm 6.3 *Update in explicit Multimodel*

```
update($k, V$)

1.    $V_k = V$;
2.    if <W is dependent>
      2.1   for $i = k+1, \cdots, n$
            2.1.1 check($V_i, V$);
```

Since all the models V_i are explicitly represented, this is indeed not a compact representation. Any update of one of the models implies a corresponding check of all consecutive models V_i, $i > k$, and the model is clearly not consistent.

The explicit Multimodel is a primitive construction, but may be useful in cases where non-compatible models are to be handled in a homogeneous way. The existence of the trivial Multimodel assures that any set of digital models may be handled as a Multimodel.

[11]In the algorithms presented in the thesis, we assume that the data structures used in the algorithms to follow allows dynamic expansion, i.e. that elements may be added to existing structures. Checking of invariants, initializing of data structures or other details may be omitted to stress the main structural issues.

6.4.2 Multi-edition models

Assume we have a set of $n+1$ equivalent models $\{\boldsymbol{a}_i\}_{i=0}^{n}$, i.e. that all the models share the transformation T, and are elements of the same attribute set Λ, where all attribute vectors have m elements, $\boldsymbol{a}_i = [a_{i1}, a_{i2}, \cdots, a_{im}]$.

In addition, assume that it is possible to represent the *difference*[12] between two models V_j and V_i expressed as Δ_i^j. Then we may express the representation of the Multimodel W as

$$\langle \{\delta_i\}_{i=0}^{n}, \Delta_i^j, T \rangle,$$

where $\delta_i = \Delta_{i-1}^i$, and $\delta_0 = V_0$. In other words, W is represented by the initial model V_0 and a sequence of differences of consecutive models. This special kind of Multimodel will be referred to as a *multi-edition model.* Temporal variations of geographic information, and generalized editions of a geographic entity are candidates for this particular Multimodel structure. The railway case in section 5.1.2 supplied examples of both a temporal change and a change due to the construction of a new edition, which were shown to be well suited for multi-edition representation by the means of a delta operator.

With the help of the difference operator Δ_i^j it is possible to make a more precise interpretation of dependency and consistency. W is said to be *dependent* if a *change* of model V_k to $\hat{V}_k$, is supposed to yield the new Multimodel representation

$$\langle V_1, V_2, \cdots, \hat{V}_k, \hat{V_{k+1}}, \cdots, \hat{V}_n \rangle,$$

such that $\Delta_i^{\hat{i}} = \Delta_k^{\hat{k}}$ for all $i > k$. With $\Delta_i^{\hat{i}}$ we mean the difference between $\hat{V}_i$ and V_i. In other words, the change $\Delta_k^{\hat{k}}$ in model k should be 'added' to all the following models.

A dependent multi-edition model is *consistent* if the representation and the associated update procedure is designed in such a manner that no explicit actions has to be taken to the models following the updated model, i.e. that the change will *automatically* propagate through the structure.

We will now investigate two classes of multi-edition models, the *pseudo delta model* and the *delta model*, differing in how the operator Δ_i^j is realized.

Pseudo delta model

In the pseudo-delta model, the delta-, or difference-operator Δ_i^j is defined as

$$\Delta_{i-1}^i = \delta_i = \{d_{i1}, d_{i2}, \cdots, d_{im}\},$$

where d_{ij} is defined as

$$d_{ij} = \begin{cases} 0 & \text{if } a_{ij} = a_{i-1j} \\ a_{ij} & \text{if } a_{ij} \neq a_{i-1j} \end{cases}$$

The element '0' is to be interpreted as an empty, or void, attribute. In appendix A we find an example of a pseudo-delta model in the implementation of the record.

[12]This will not necessarily imply that we have a fully defined arithmetic.

Insertion of a model V with an attribute vector $\boldsymbol{a}$ of m elements, $\{a_1, a_2, \cdots, a_m\}$, is performed as outlined below. Note that since all models in a pseudo-delta model are equivalent, the $n+1$ attribute vectors corresponding to $\{\delta_0, \delta_1, \cdots \delta_n\}$ including the one to be inserted, $\boldsymbol{a}$, are all elements in Λ.

Algorithm 6.4 *Insertion in pseudo-delta model*

```
insert($V$)

1.      if $W == \emptyset$
        1.2     $V_0 = V$;        $n = 0$;
2.      else
        2.1.    for $j = 1, \cdots, m$
                2.1.1  $i = n$;
                2.1.2  while $d_{ij} == 0$ and $i \geq 0$
                       2.1.2.1  $i = i - 1$;
                2.1.3  if $d_{ij} \neq a_j$
                       2.1.3.1  $d_{n+1j} = a_j$;
                2.1.4  else
                       2.1.4.1  $d_{n+1j} = 0$;
        2.2.    $n = n + 1$;
```

Algorithm 6.4 runs through the attributes of the model to be inserted, and checks each attribute against the first non-zero corresponding attribute (iterating 'backwards' from the last to the first model) in the existing structure. If the attributes are identical, a '0' is inserted as the $n+1$'th attribute of this kind, else the attribute of the new model is inserted.

The reconstruction algorithm works in a similar fashion:

Algorithm 6.5 *Reconstruction in pseudo-delta model*

```
reconstruct($k$)

1.      for $j = 1, \cdots, m$
        1.1     $i = k$;
        1.2     while $d_{ij} == 0$ and $i \geq 0$
                1.2.1  $i = i - 1$;
        1.3     $a_j = d_{ij}$;
2.      return $\boldsymbol{a} = \{a_1, a_2, \cdots, a_m\}$;
```

Algorithm 6.5 will for every V attributes work its way 'backwards' from the k'th level in the structure until encountering a non-zero attribute, which will be inserted as the corresponding attribute in the vector to be returned.

Finally we present the updating algorithm:

Algorithm 6.6 *Update in pseudo-delta model*

```
update(k, V)

1.     for $j = 1, \cdots, m$
       1.1   if $d_{kj} \neq a_j$
       1.2   $d_{kj} = a_j$
       1.3   <Check (in worst case all) prior attributes
             in the model such that no successing
             attributes are identical>;
```

The attributes in the vector on the k'th level are compared to the corresponding attributes in the updating vector, and replaced if they differ. In addition, all the attributes of this kind (included the updated) have to be checked and possibly updated.

If not all the models in the pseudo-model structure are completely different from the successor[13], the differences will contain a certain amount of zero elements, thus clearly yielding a compact representation. The step 1.3 in algorithm 6.6 shows that the structure is not consistent.

Some models permit a more sophisticated implementation of the delta-function, and this will be investigated in the following section.

Delta model

Assume we have a set of equivalent models with a well defined arithmetic, i.e. a set of difference models (see section 6.2.2). In addition, we restrict the models not only to be elements of a group (see definition 12) under '+', but the group shall also be abelian (or commutative)[14]. The identity element will be noted as 0, and the inverse elements will be noted $-V$.

The difference operator Δ^i_{i-1} is in this case trivially defined as $\delta_i = V_i - V_{i-1}$. The implicit representation will be the initial model and the sequence of differences:

$$\langle \{\delta_i\}^n_{i=0}, +, T \rangle,$$

where $\delta_0 = V_0$.

The insertion algorithm appends a model V by generating the model V_n and append the new difference $V - V_n$:

[13] A set of completely different models would indeed represent a pathological example of a multi-edition model.

[14] A group Υ under '+' is abelian if for any $V, V' \in \Upsilon$ we have that $V + V' = V' + V$. The term 'abelian' honors the Norwegian mathematician Nils Henrik Abel, 1802 - 1829, most famous for proving the impossibility of solving quintic (or higher degree) equations by means of ordinary arithmetic operations including root extraction.

Algorithm 6.7 *Insertion in delta model*

```
insert(V)

1.      if $W == \emptyset$
        1.2    $V_0 = V$;        $n = 0$;
2.      else
        2.1    $\mu = \delta_0$;
        2.2    for $i = 1, \cdots, n$
               2.2.1  $\mu = \mu + \delta_i$;
        2.3    $\delta_{n+1} = V - \mu$;
        2.4    $n = n + 1$;
```

To verify that the insertion is correct, we have to check if the inserted difference, $\delta_{n+1} = V - \mu$, is equivalent to $V - V_n$, i.e. that $\mu = V_n$. In the algorithm 6.7, μ is computed as $V_n = V_0 + \sum_{i=1}^{n} \delta_i$. We will now perform the computation, using the definition of the abelian group and some trivial implications[15]:

$$\begin{aligned}
\mu &= V_0 + \textstyle\sum_{i=1}^{n} \delta^i \\
&= V_0 + [(V_1 - V_0) + (V_2 - V_1) + \cdots + (V_{n-1} - V_{n-2}) + (V_n - V_{n-1})] \\
&= V_0 + [-V_0 + (V_1 - V_1) + (V_2 - V_2) + \cdots + (V_{n-1} - V_{n-1}) + V_n] \\
&= V_0 + [-V_0 + 0 + 0 + \cdots + 0 + V_n] \\
&= V_n
\end{aligned}$$

The reconstruction algorithm uses a similar multiple addition scheme as in algorithm 6.7, starting with the initial model and adding differences up to the level corresponding to the given parameter k.

Algorithm 6.8 *Reconstruction in delta model*

```
reconstruct(k)

1.      $\mu = \delta_0$;
2.      for $i = 1, \cdots, k$
        2.1    $\mu = \mu + \delta_i$;
3.      return $\mu$;
```

The updating algorithm 6.8 generates the model V_{k-1}, and replace the old difference δ_k with the difference between the new model and V_{k-1}:

[15] The scheme will work with non-abelian groups too, but with a little ‘awkward’ definition of the deltas as $\delta_i = -V_{i-1} + V_i$.

Algorithm 6.9 *Update in delta model*

```
update(k, V)

1.    $\mu = \delta_0$;
2.    for $i = 1, \cdots, k-1$
      2.1    $\mu = \mu + \delta_i$;
3.    $\delta_k$ = $V - \mu$;
```

It is straightforward to verify algorithm 6.9 and 6.8 using the same procedure as for verifying 6.7.

As with the pseudo-delta model, this structure is compact, since the differences will consist of a number of zero elements, under the assumption that no model completely differs from its successor.

The delta model is consistent with respect to the dependency. Note that the delta model scheme will not work correctly if such a dependency is not wanted. Any update of a certain model will automatically propagate to the successing models, as demonstrated in algorithm 6.9. To verify this, we observe that after an update of V_j to $\hat{V}_j$, such that $\hat{V}_j - V_j = \hat{\delta}_j$, we have that a model number k, $k \geq j$, is reconstructed as follows:

$$\begin{aligned}
\mu &= V_0 + \sum_{i=1}^{k} \delta^i \\
&= V_0 + \sum_{i=1}^{j-1} \delta^i + \delta^j + \sum_{i=j+1}^{k} \delta^i \\
&= V_0 + \sum_{i=1}^{j-1} \delta^i + \hat{V}_j - V_{j-1} + \sum_{i=j+1}^{k} \delta^i \\
&= V_0 + \sum_{i=1}^{j-1} \delta^i + V_j + \hat{\delta}_j - V_{j-1} + \sum_{i=j+1}^{k} \delta^i \\
&= V_0 + \sum_{i=1}^{k} \delta^i + \hat{\delta}_j \\
&= V_k + \hat{\delta}_j
\end{aligned}$$

Thus the change $\hat{\delta}_j$ applied to the model V_j propagates to all following models.

We have now outlined two methods for structuring collections of equivalent models. The next sections investigates corresponding structures for handling models which are compatible, differing in having attribute vectors from attribute spaces of different 'size'.

6.4.3 Multi-scale models

Assume we have a set of $n+1$ compatible models $\{\boldsymbol{a}_i\}_{i=0}^{n}$, i.e. that all the models share the transformation T, but may have attribute-vectors of different types, i.e. $\Lambda_i \neq \Lambda_j$ for $i \neq j$, where Λ_i is the set in which $\boldsymbol{a}_i$ is an element. The models are ordered after increasing complexity, such that the number of attributes in V_j is larger than in V_i when $j > i$.

In addition, assume that we have an *approximation* operator $\mathcal{P}_n^i$ (see section 6.2.3) defined such that

$$\mathcal{P}_n^i V_n = V^\star{}_n = V_i,$$

i.e. that the models $V_0, V_1, \cdots V_{n-1}$ are approximations of the model V_n represented with increasing complexity, i.e. represented with an increasing amount of data.

The representation of a multi-scale model is characterized by the collection

$$\langle \{\boldsymbol{a}_i\}_{i=1}^{n}, \mathcal{P}_n^i, T \rangle.$$

A set of compatible models with attribute vectors from different attribute sets, may be regarded as models of essentially the same phenomenon represented in different accuracies or resolutions. In cartography and GIS, it is common to associate the notion of scale to accuracy or resolution (see e.g. the *Introduction*). Of this reason, this particular kind of a Multimodel will be termed a *multi-scale model*.

The notion of consistency should be different interpreted when it comes to multi-scale models. It is not natural to define dependencies among the various models, other than the inherently dependency defined by the fact that each model is an approximation of an initial model. The approximation operator is part of the Multimodel, and we will therefore consider all multiscale models as consistent. This is leaving compactness as the main aspect when investigating multi-scale structures.

Before investigating multi-scale structures, we have to make some comments on the `insert` and `update` operations.

- `insert(...)`: In the case of the multi-edition model, the insertion was restricted to append the model V, given as the parameter to the procedure. In a multi-scale structure, insertion will be reformulated to append a new approximant. The approximant may be generated either by specifying the attribute space Υ_k which we want the approximant to be a member in, or, if we deal with metric models, by giving the tolerance σ.

- `update(`k, V`)`: The multi-scale structure is a sequence of successively data-reduced models, constructed by applying a well defined approximation operator on the original model. Any update of the models apart from the original model is out of the question, since such an updating would potentially violate the assumption that the model should be an approximant. Of this reason, the update operation becomes uninteresting in the context of multi-scale models.

The reconstruction procedure will be identically defined and interpreted as for the multi-edition structure.

We will now study three different variants of multi-scale models, mainly differing in properties of the approximation and refinement operators.

Selection model

We may sometimes encounter a special kind of the multi scale model, where the approximation operator is restricted to select a collection of the attributes of the initial model to yield a model of lower precision. This is a simple, but nevertheless important scheme widely used in e.g. line simplification algorithms, see section 7.2.

The approximation operator may be defined as follows:

$$\mathcal{P}_n^i V = \mathcal{P}_n^i [a_1, a_2, \cdots, a_m] = [a_{i1}, a_{i2}, \cdots, a_{i\mu}] = V^\star,$$

where $\mu \leq m$ and every a_{i_j} is an element in $[a_1, a_2, \cdots, a_m]$, and where the ordering of the original vector is maintained. $\mathcal{P}_n^n$ is defined as the identity operator.

This special Multimodel may be modeled extremely compact by using an integer array β of the same length as the original attribute vector $\boldsymbol{a}_n = \boldsymbol{a}$, and storing only the original model V_n.

The representation of selection model is expressed as

$$\langle \boldsymbol{a}, \beta, \mathcal{P}_n^i, T \rangle,$$

where $\boldsymbol{a} = [a_1, a_2, \cdots, a_m]$.

The insertion algorithm runs as follows, under the assumption that $W \neq \emptyset$, and that β is initialized to zeros if only one single model is represented by W, i.e. that $n = 0$. The notation $\mathcal{P}_n^{\cdots}$ indicates that the approximation is dependent of the input of the procedure, as discussed in the previous section.

Algorithm 6.10 *Insertion in selection model*

```
insert(...)

1.      $\boldsymbol{a}^\star = \mathcal{P}_n^{\cdots}\boldsymbol{a}$;
2.      for $i = 1, \cdots, m$
        2.1     if $a_i \in \boldsymbol{a}^\star$
                2.1.1  $\beta_i = \beta_i + 1$;
3.      $n = n + 1$;
```

We assume that the approximated model is element of a space Λ_{n+1} of lower 'dimensionality' than Λ_n.

The reconstruction use the information embedded in the β-vector, and picks the elements in $\boldsymbol{a}$ accordingly. In appendix A we implement piecewise linear curves as selection models.

Algorithm 6.11 *Reconstruction in selection model*

```
reconstruct($k$)

1.      $\nu$ = 0;
2.      for $j = 1, \cdots, m$
        2.1     if $\beta_j \geq n - k$
                2.1.1  $\nu = \nu + 1$;
                2.1.2  $\alpha_\nu = a_j$;
3.      return $\{\alpha_1, \alpha_2, \cdots, \alpha_\nu\}$;
```

The selection model is extremely compact. In addition to the original model, only the β-vector consisting of m integers is needed. Further, the storage demand is not dependent on the number of models represented.

There exists indeed more time-efficient representations of the selection model. In our case, the running time of the generation algorithm is $\mathcal{O}(m)$, where m is the number of attributes in the original model. A tree-like organization of pointers to the data, as the line-tree described in [JA86], may reduce the average running time to $\mathcal{O}(\log m)$, but as pointed out earlier, our emphasis is towards compactness rather than run-time efficiency.

Note that variations can made over the selection model scheme. The different models may be produced as stepwise approximations, such that $V_i = \mathcal{P}^i_{i+1}$, in contrast to $V_i = \mathcal{P}^i_n$, as we suggested. However, such variations requires only minor changes in the different procedures associated with the structure.

Note that the selection model did not require a refinement operator, such as the decomposed model outlined in the next section.

Decomposed model

This section is an adaptation of the methods and algorithms in Dæhlen and Lyche [DL92]. The article investigates aspects of decomposition of well defined mathematical structures, thus we have to make some modifications to fit a more general Multimodel concept.

Assume that we have a set of decomposable models $\{V_i\}^n_{i=0}$, i.e. that we have both an approximation operator and a refinement operator. The corresponding attribute vectors are elements in $\{\Lambda_0, \Lambda_1, \cdots, \Lambda_n = \Lambda\}$, where we may have that $\Lambda_i \neq \Lambda_j$.

Let each model be the result of an approximation that 'projects' the initial model V_n to a space Λ_i of 'lower dimensionality', i.e. attribute vectors of less length,

$$V_i = \mathcal{P}^i_n V_n,$$

such that the decomposition model essentially is an ordered set of models of increasing precision.

The refinement operator (see section 6.2.3) is defined such that

$$\mathcal{R}^j_i V_i = V^{\diamond}{}_j, \quad j > i,$$

where $V^{\diamond}{}_j$ is element of a subset $\Lambda^{\diamond}_j$ of Λ_j. Note that $V^{\diamond}{}_j$ is equivalent to V_j, but most often not identical.

Since we assumed that the models were decomposable, and thus having addition and subtraction at hand, we may represent the models as an explicit model $V = V_0$ and a set of differences δ_i:

$$\delta_i = V_i - \mathcal{R}^i_{i-1} V_{i-1},$$

where the refined model $\mathcal{R}^i_{i-1} V_{i-1}$ is element in $\Lambda_i{}^{\diamond} \subseteq \Lambda_i$.

The representation of the decomposition model may formally be defined as

$$\langle \{\delta_i\}^n_{i=0}, +, \mathcal{P}^j_n, \mathcal{R}^j_i, T \rangle,$$

where $\delta_0 = V_0$, thus consisting of the initial model in the coarsest resolution, a set of approximated difference models, arithmetic operators, approximation and refinement operators, and finally a common transformation.

The appending of a new approximant will essentially involve steps from the `reconstruct` routine in addition to book-keeping actions, and will not illuminate significant aspects of the decomposed model. Thus, we will not outline any details of the `insert` operation for this multi-scale structure, but rather concentrate on the reconstruction of a certain approximant.

The reconstruction in a decomposed model is to be considered as a variant of algorithm algoReconstructDelta, the corresponding operation in a deltamodel:

Algorithm 6.12 *Reconstruction in decomposed Multimodel (a)*

```
reconstruct($k$)

1.    $\mu = \delta_0$;
2.    for $i = 1, \cdots, k$
      2.1   $\mu = \mathcal{R}^i_{i-1}\mu + \delta_i$;
3.    return $\mu$;
```

Piecewise linear curves with a special kind of approximation operator are implemented as decomposition models in appendix A.

The algorithm follows since V_{i+1} can be constructed from V_i in the following way:

$$\begin{aligned} \mu &= \mathcal{R}^{i+1}_i V_i + \delta_{i+1} \\ &= \mathcal{R}^{i+1}_i V_i + V_{i+1} - \mathcal{R}^{i+1}_i V_i \\ &= V_{i+1} \end{aligned}$$

If the refinement operator satisfy

$$\mathcal{R}^j_i \mathcal{R}^k_j = \mathcal{R}^k_i, \quad i \le j \le k$$

and

$$\mathcal{R}^j_i (V \pm V') = \mathcal{R}^j_i V \pm \mathcal{R}^j_i V'$$

where $V, V' \in \Lambda_i$, we may design a variant of the reconstruction algorithm. Here, the additions are all performed in the 'largest' attribute-space Λ_n, and not on levels of increasing complexity as in algorithm 6.12.

Algorithm 6.13 *Reconstruction in decomposed Multimodel (b)*

```
reconstruct($k$)

1.    $\mu = \mathcal{R}^k_0 V_0$;
2.    for $i = 1, \cdots, k$
      2.1   $\mu = \mu + \mathcal{R}^k_i \delta_i$;
3.    return $\mu$;
```

The correctness of 6.13 is verified by performing the addition up to a level k:

$$
\begin{aligned}
\mu &= \mathcal{R}_0^k V_0 + \textstyle\sum_{i=1}^k \mathcal{R}_i^k \delta_i \\
&= \mathcal{R}_0^k V_0 + \textstyle\sum_{i=1}^k \mathcal{R}_i^k (V_i - \mathcal{R}_{i-1}^i V_{i-1}) \\
&= \mathcal{R}_0^k V_0 + \textstyle\sum_{i=1}^k \mathcal{R}_i^k V_i - \mathcal{R}_i^k \mathcal{R}_{i-1}^i V_{i-1} \\
&= \mathcal{R}_0^k V_0 + \textstyle\sum_{i=1}^k \mathcal{R}_i^k V_i - \mathcal{R}_{i-1}^k V_{i-1} \\
&= (\mathcal{R}_0^k V_0 - \mathcal{R}_0^k V_0) + (\mathcal{R}_1^k V_1 - \mathcal{R}_1^k V_1) + \cdots + \\
&\quad\; (\mathcal{R}_{k-1}^k V_{k-1} - \mathcal{R}_{k-1}^k V_{k-1}) + \mathcal{R}_k^k V_k \\
&= V_k
\end{aligned}
$$

The decomposed structure is more compact than an explicit representation under the assumption that the 'magnitude' of the differences is less than the 'magnitude' of the corresponding approximants, see section 6.3.1.

As with the selection model structure, variations can made over the decomposed scheme, for example by producing the approximations as $V_i = \mathcal{P}_{i+1}^i V_{i+1}$, instead of $V_i = \mathcal{P}_n^i V_{i+1}$. See section 2 in Dæhlen and Lyche [DL92] for further details on variations over a similar scheme.

In the selection model and the decomposition model the approximation operator projects the models from one attribute space down another *known* attribute space of lower dimensionality. Such operators could for example be implemented as $\mathcal{P}_{2^n+1}^{2^{n-1}+1}$, i.e an approximation from a space of $2^n + 1$ attributes to a space of $2^{n-1} + 1$ attributes by just picking every second attribute starting with the first. In this way, we do not have any control of how 'good' the approximation is, i.e. the distance between the original and the approximant, or in other words, the error of the data reducing process. In the next section we investigate a multi-scale structure for metric models with approximation operators that depends on given *tolerances*.

Multi-resolution model

A multi resolution model is essentially a multi-scale model associated with a set of tolerances, a metric (see definition 14), and an approximation operator capable of performing constrained data reduction according to the corresponding tolerances.

Assume we have a set of metric models $\{V_i\}_{i=0}^n$, a set of tolerances $\{\epsilon_i\}_{i=0}^n$ and a constrained approximation operator. A multi-resolution model is then the set of models such that V_i, is the result of the stepwise approximations

$$V_i = \mathcal{P}_{i+1}^i(\epsilon_i)\ V_{i+1},$$

such that $\rho(\mathcal{R}_i^{i+1} V_i, V_{i+1}) \leq \epsilon_i$. V_n is the original model with highest precision.

We may have the alternative definition where we have a set of approximations all generated from the original model:

$$V_i = \mathcal{P}_n^i(\epsilon_i)\ V_n$$

where the error in the approximation is measured as $\rho(\mathcal{R}_i^n V_i, \mathcal{R}_{i+1}^n V_{i+1})$.

The notation $\mathcal{P}_j^i(\epsilon_i)$ indicates that the approximation maps the model from the attribute set Λ_j to the unknown set Λ_i which depends on the tolerance ϵ_i.

Both the selection model and the decomposed model may easily be formulated as multiscale models, with minor adjustments of the structures and the operations.

Composite models

Until now, we have only been concerned with Multimodels varying according to one single parameter, typically, in a GIS setting, representing scale, edition or time.

However, in *Part I*, we stressed that geographic information is characterized by varying according to all these parameters simultaneously. This fact motivates the introduction of the *composite Multimodel*, i.e. a Multimodel that vary according to more than one single parameter.

We will not give any further details on composite Multimodels in the thesis, but we observe that such Multimodels represent additional challenges, particularly regarding consistency and compactness.

We have in the last sections outlined various aspects of Multimodels and investigated to some detail a few main categories of structures for multiple modeling.

In the next section, we will summarize the chapter with the description of an object model of a generic Multimodel class library.

6.5 MULTIMOD: A generic object model

We will now summarize the elaboration of the Multimodel concept in an object model[16] of a generic library structure. The library will be called MULTIMOD. The main structure of MULTIMOD is displayed in figure

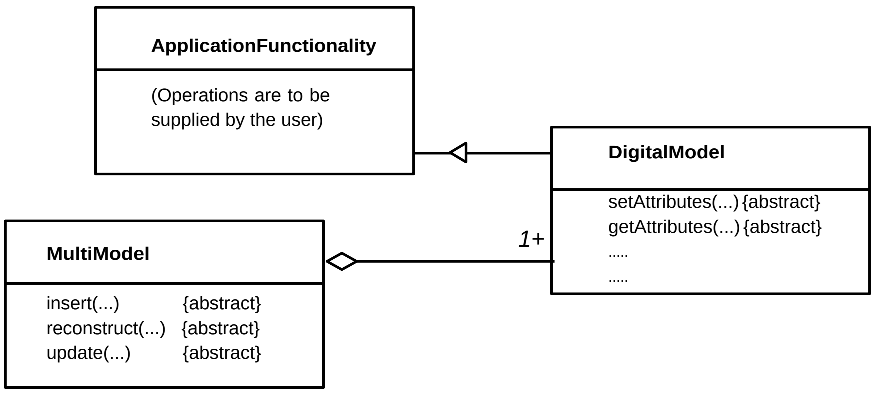

ApplicationFunctionality
(Operations are to be supplied by the user)
DigitalModel
setAttributes(...){abstract}
getAttributes(...){abstract}
.....
.....
1+
MultiModel
insert(...) {abstract}
reconstruct(...) {abstract}
update(...) {abstract}

Figure 6.3: Main structure of MULTIMOD

The figure simply states that we have a class of objects named `Multimodel`, which consists of a number (one or more) of objects of the class `DigitalModel`. The `DigitalModels` are specializations of the class `ApplicationFunctionality`.

[16] Rumbaugh et. al. gives the following description of an object model [RBP$^+$91]:

> An *object model* captures the static structure of a system by showing the objects in the system, relationships between the objects, and the attributes and the operations that characterize each class of objects.

In the presentation of the 'Object Modeling Technique' (OMT), Rumbaugh et. al. further stress that an object model is the most important description of a system. We will use a limited subset of the OMT-notation in our object diagrams, and we assume that the reader are familiar with such diagrams.

We see that `Multimodel` has a set of operations needed to maintain an ordered set of variants, represented by the operation `reconstruct` (see section 6.3 for details on such operations). The class `Multimodel` is abstract, and derived specializations need to implement the details of operations.

The `DigitalModel` is also an abstract class. It is a subclass of `ApplicationFunctionality`, which is to be considered as an interface to the application in which the Multimodels are to be used. Initially, this class has no operations, the intention is that the operations should be provided by the user. In our example, we have furnished the `ApplicationFunctionality` with the operation `printMap`. With this structure, we are ensured that regardless of how the subclasses of the `DigitalModel` and the `Multimodel` are implemented, we are always able to make the call

```
AnyMultimodel.reconstruct(index).printMap(...),
```

i.e. that is possible from any Multimodel to reconstruct a certain variant according to the parameter `index` and print that particular model in a certain fashion specified by `printMap(...)`.

If the MULTIMOD was to be used in a GIS, we should indeed have designed a composite Multimodel that were customized to handle variants according to time, scale and edition, as described briefly in section 6.4.3. Such an extension would essentially involve modeling of relations between three single Multimodels, and is omitted in order to highlight the main structural issues of the implementation of a Multimodel.

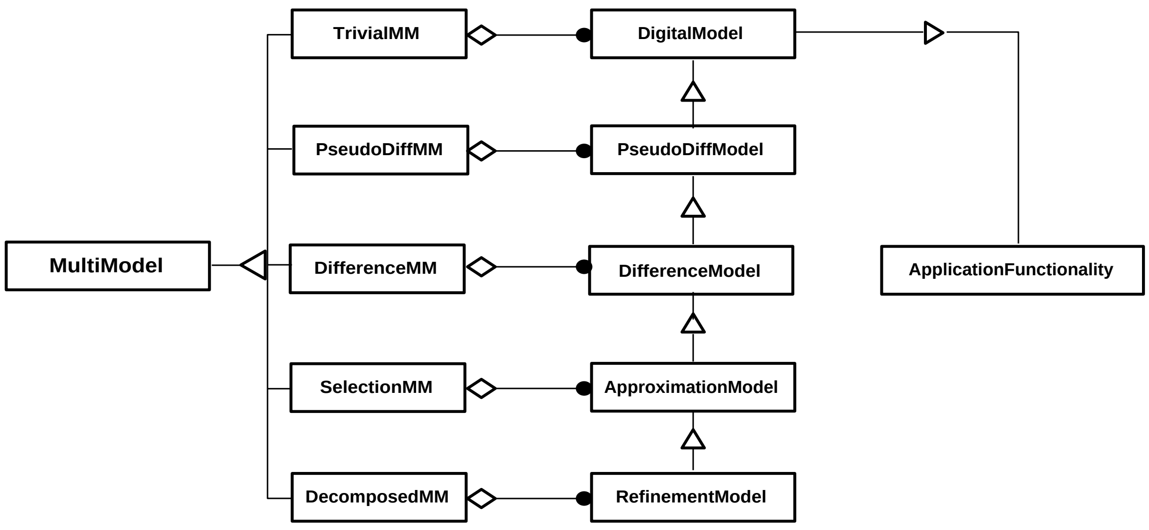


Figure 6.4: MULTIMOD - a generic Multimodel library

In figure 6.4, the main framework of MULTIMOD is extended to incorporate the different categories of digital models and Multimodels outlined in section 6.4. The initial `DigitalModel` is successively specialized into four types:

- The `PseudoDiffModel` is characterized by a simple Δ-operator.
- The `DifferenceModel`, which is supplied with a fully developed arithmetic. This implies that the models to be derived of this class must form groups (recall definition 12) under the particular kind of addition implemented.

- The `ApproximationModel` has an approximation operator enabling the model to generate approximations of itself, either to an attribute space of known 'size', or according to a given tolerance. In the last case we assume that there exists a well defined metric, or 'distance-measure'.
- The `RefinementModel` provides an refinement operator, making it possible to 'lift' the model into a 'larger' attribute space.

Note that the various models inherit the operations from their respective superclasses, such that the refinement model has both well defined arithmetic operators and an approximation operator in addition to the refinement operator of its own. Also note that the model categories are abstract classes, and can not be instantiated without being specialized into classes which implement the various operations.

Based on the model categories, we have designed five subclasses of the `Multimodel`:

- The `TrivialMM` is simply a collection of any type of `DigitalModel`, and is to be considered as a 'chalk and cheese' Multimodel. The basic operations are implemented according to algorithms 6.1, 6.2 and 6.3.
- The `PseudoMM` integrates equivalent models with Δ-operators, as defined in the class `PseudoDiffModel`, such that it is possible to express differences between models. Note that this not imply a full set of arithmetic operators. The algorithms 6.4, 6.5 and 6.6 are implemented in the class.
- The `DifferenceMM` is handling a set of specializations of the `DifferenceModel`, under the assumption that they are equivalent. Basic operations are implemented as outlined in 6.7, 6.8 and 6.9.
- The `SelectionMM` integrates compatible models, or specializations of `ApproximationModel`. They are restricted to be the result of an approximation that selects a subset of the original attribute-vector. The `insert` and `reconstruct` procedures are implemented according to algorithms 6.10 and 6.11.
- The `DecomposedMM` is the most 'completely' equipped Multimodel, being a set of decomposable models, or subclasses of `RefinementMod`. We have chosen to use a stepwise reconstruction procedure as given by algorithm 6.12.

Note that the different Multimodels are *not* abstract classes, and may be instantiated as they are.

In chapter 7, we will propose an informal methodology for designing specific Multimodels based on the framework outlined in this section. We will also give details on the design of some selected digital models.

A limited implementation of the MULTIMOD, incorporating the examples given on digital models in chapter 7, is carried through in appendix A.

Chapter 7

Examples of Multimodels

In this chapter, we will use the discussion and results from chapter 5 and 6 to design Multimodels of *piecewise linear curves, PLCs* for short, and *piecewise linear surfaces defined over triangular, irregular networks*, called *PLS*s.

Both these geometric structures are examples of representations of spatial information. The PLC is frequently used in GIS to model railway networks, borders, shorelines and other curve-like features.

The PLS is frequently used in *digital elevation models*, *DEM*s for short, to model the terrain as an explicit, linear bivariate function. In addition, the use of the PLS to represent thematic information, such as the variation of a certain parameter over a given area, is increasing.

Before investigating the PLC and the PLS, we will outline a general strategy for Multimodeling.

7.1 Designing Multimodels

The design of a particular Multimodel[1] should start with a thorough analysis of the objects in question and their properties as digital models. Following the characterizations and definition in chapter 6, the analysis may go as follows:

- Characterize the likeness models according to definition 10, as
  - non-compatible,
  - compatible or
  - equivalent.
- If the models are non-compatible, we have no other options than structure them as an explicit Multimodel, which may be trivially implemented.
- If our models are compatible, we have two main subclasses to investigate, the models in a multi-edition setting, and in a multi-scale setting.

[1]We will restrict the investigation to single Multimodels, and will not give any details of design methodology for composite Multimodels.

- The multi-edition structure requires at least a definition of a difference operator to be able to represent the difference between two models. We are perhaps able to define a pseudo-delta model, or we may find that our models in fact have well defined arithmetic operators, and possibly members of the same group. Section 6.4.2 outlined two implementations of multi-edition models depending on the arithmetic properties of the models.
- If the models are to be accessed in variants of different accuracy, we have to examine possible approximation and refinement operators. Section 6.4.3 described alternative designs of multi-scale structures, based on certain properties of the approximation operators (and possibly the refinement procedures).

We will now apply the described methodology on the PLC and the PLS.

7.2 Piecewise Linear Curves

Definition

A piecewise linear curve may be considered as a digital model. The attribute vector is an ordered set of an arbitrary number of points in the plane:

$$\boldsymbol{p} = [p_1, p_2, \cdots, p_m], \quad p_i = (x_i, y_i) \in \mathbb{R}^2).$$

The transformation of the model may be considered essentially as the following definition of the curve C as

$$C = \bigcup_{i=1}^{m-1} s_i,$$

where s_i is the segment defined as the convex combination of two consecutive points:

$$s_i = \{x \in \mathbb{R}^2 \mid x = (1-\alpha)p_i + \alpha p_{i+1}, \quad \alpha \in [0,1]\}.$$

The formulation of the PLC as a digital model is then

$$PLC = \langle \boldsymbol{p}, C \rangle.$$

Arithmetics

The PLC is fully supported with arithmetic operators. The set of all such curves $\mathcal{C}_m$ of length m is indeed an abelian group (recall definition 12) under the addition defined as

$$C + C' = [p_i + p'_i]_{i=1}^m,$$

where $C, C' \in \mathcal{C}_m$ and $p_i + p'_i = (x_i + x'_i, y_i + y'_i)$. The zero-curve is $[(0,0), \cdots, (0,0)]$, and the inverse element of C is $-C = [(-x_i, -y_i)]_{i=1}^m$.

The verification of $\langle \mathcal{C}, + \rangle$ as a commutative group is quite trivial and thus omitted.

We may also trivially define a metric on this group, as:

$$\rho(C, C') = \max_{i=1}^{m} \sqrt{(x_i - x'_i)^2 + (y_i - y'_i)^2},$$

i.e. the largest Euclidian distance between corresponding points of the curves. This is easily verified as a metric according to definition 14.

Since piecewise linear curves of same length are elements of an abelian group, it will be convenient to structure a Multimodel of such PLCs as a delta model, see section 6.4.2 and the algorithms 6.7, 6.8 and 6.9. We will then benefit from the compactness offered by this structure, and also from the consistency of the scheme[2]. An implementation of the PLC as a delta model is carried through in appendix A.

Decomposition

There exist a multitude of methods designed for line simplification in computer aided cartography, see e.g. [McM86] for an overview and discussion of some of these algorithms. We will give some details on two such methods, focusing on features related to the application of the procedures as approximation operators in a multi-scale setting.

The *Douglas-Peucker* method [DP73], also known as the anchor-and-buoy algorithm, is one of the traditional line simplification procedures widely used in CAC. Details on the algorithm will not be given here, we will only look into aspects concerning the use of the operator in a Multimodel.

If we denote all the set of all possible PLCs represented with m points as $\mathcal{C}_m$, the Douglas-Peucker may be formulated as the selection approximation (see section 6.4.3) $\mathcal{P}_{\mathrm{DP}} : \mathcal{C}_m \rightarrow \mathcal{C}_{\mu(\epsilon)}$, where $\mathcal{C}_{\mu(\epsilon)}$ is the set of all curves represented with $\mu(\epsilon) \leq m$ points. The notation $\mathcal{C}_{\mu(\epsilon)}$ indicates that the 'size' of the space is dependent on the tolerance ϵ. The approximant to the curve C is generated as $C^\star = \mathcal{P}_{\mathrm{DP}} C$, such that the the point vector $[p^\star{}_i]_{i=1}^{\mu}$ is an ordered *subset* of the points in the original curve C.

Further, the distance between the two curves, or the error of the approximation, should be less or equal to the tolerance: $\rho(C, \mathcal{R}C^\star) \leq \epsilon$.

The refinement operator $\mathcal{R}$ is defined as follows: The 'missing' points in $C^\star$ are generated as the perpendicular projections of the corresponding points in C down to the approximant.

Arge and Dæhlen [ADWM92], have proposed a variant over the Douglas-Peucker scheme, where basically the points in the approximant are allowed to be slightly perturbed in order to yield higher data reduction. In this case the approximated vector is *not* a subset of the original. We will term this approximation operator $\mathcal{P}_{\mathrm{AD}}$. The associated refinement operator is the same as for $\mathcal{P}_{\mathrm{DP}}$.

Since we have the choice of two approximation operators, we get some freedom in choosing a multi-scale structure. We have a well defined metric, thus we may extend our multi-scale structure to a multi-resolution structure, and obtain a range of models corresponding to a set of tolerances, which in turn refer to certain ranges of scales.

If want to use the $\mathcal{P}_{\mathrm{DP}}$-operator, which picks a subset of points from the original curve, the selection-model in section 6.4.3 becomes a natural choice. This scheme offers a particularly compact and simple structure, see algorithms 6.10 and 6.11.

The $\mathcal{P}_{\mathrm{AD}}$-operator may be used in a decomposed model, where the given level is generated essentially as a sum of the coarsest representations and the corresponding differences, see the

[2]We assume that we want the set of PLCs to be dependent, i.e. that changes in a certain variant should propagate to the following curves.

two variants of reconstruction algorithms, 6.12 and 6.13. The latter scheme assumes that the refinement operator satisfy $\mathcal{P}_i^j \mathcal{P}_j^k = \mathcal{P}_i^k$, which is not needed in the former.

In appendix A we find an implementation of a selection model using $\mathcal{P}_{\text{DP}}$, and a decomposed model with $\mathcal{P}_{\text{AD}}$.

7.3 Piecewise Linear Surfaces

Definition

We will now study a surface defined as a piecewise linear function defined over a triangulated, irregular network (TIN). The surface will be termed *PLS*. Before we give a definition of the PLS, we need the definition of a *triangulation*. There exist many formulations of the triangulation problem, and the one given here is adapted from [DLR90].

Definition 15 (Triangulation) *Assume we have a set of distinct points in* $\mathbb{R}^3$*,*

$$W = \{v_i\} = \{(x_i, y_i, z_i)\}, \quad i = 1, 2, \cdots, m,$$

We denote the orthogonal projection of W *to* $\mathbb{R}^2$ *as* $V = \{(x_i, y_i)\}, \quad i = 1, 2, \cdots, m.$

Let $\Omega \subset \mathbb{R}^2$ *be a region with a polygonal boundary*[3] $\partial\Omega$ *so that* $V \subset \Omega$*. The set* $T = \{T_i\}_{i=1}^t$ *of non-degenerate, disjoint triangles is a* triangulation *of* Ω *if each* $(x_i, y_i) \in V$ *is on a vertex of some triangle* T_j *and if* $\Omega = \bigcup_{i=1}^t T_i$*.*

The linear surface S is defined as the set of triangular patches:

$$S = \bigcup_{i=1}^{t} s_i,$$

where s_i is the planar segment defined as the barycentric combination of three points:

$$s_i = \{p \in \mathbb{R}^3 \mid p = uT_{i1} + vT_{i2} + wT_{i3}, \quad u + v + w = 1, \quad u, v, w \geq 0\},$$

where T_{i1}, T_{i2} and T_{i3} are the points in W which projections in V define the triangle T_i of the triangulation T.

The triangulation of V is indeed not unique, but vary according to underlying triangulation method and in some cases the ordering of the points in V. See [Sch87] for a discussion of triangulation methods. Let us assume we use the well known Delauney method, see e.g [LS80] for an outline of certain properties of this triangulation and decriptions of two algorithms for constructing such a TIN.

One formulation of the PLS as a digital model could be

$$PLS = \langle W, D, S \rangle,$$

where W is the point set, D is a the (Delauney) transformation that *order* W into a triangulation, and finally S which is the definition of the surface. We observe that the transformation of the model is actually composed of two separate operations.

[3] Note that this boundary need not be convex.

Another possibility is to consider the PLS as a completed triangulation and the surface definition:

$$PLS = \langle \{T_i\}_{i=1}^{t}, S \rangle,$$

where $T = \{T_i\}_{i=1}^{t}$ is the set of triangles, and S the transformation into a surface.

Arithmetics

We observe that the first formulation of the PLS as

$$PLS = \langle W, D, S \rangle$$

allows a trivially definition of arithmetic operators, as the attribute vector is a set of 3D points. The addition and subtraction operators can be defined as we did with the PLC in 7.2 (assumed that the point sets are of equal size), and the structure yields an abelian group.

However, the definition

$$PLS = \langle \{T_i\}_{i=1}^{t}, S \rangle$$

is not so straight forward regarding the design of arithmetic operators.

We observe that the triangle patches T_i's are highly correlated, as a point in a triangle most often is shared with several other triangles, and that changes in the point set V may violate the triangulation requirements. However, further investigation on arithmetic operators on such PLSs is beyond the scope of the thesis (a good point to start would be [Flo89]).

An implementation of a multi-edition PLS as a delta model based on the formulation

$$PLS = \langle W, D, S \rangle,$$

is carried through in appendix A.

Decomposition

During the last 10 years, several schemes for data reduction of PLSs has been proposed. Floriani [Flo89] outlines two fundamentally different approaches, concerning the relations between the approximant and the original.

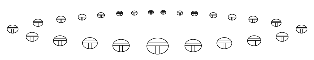
In the *subdivision* approach, the triangulation is constructed by recursively inserting points in triangles. The insertion of a new point splits the old triangle into three new triangles.

The approximation of such a triangulation is basically performed by successively removing vertices until some tolerance requirement is met, thus yielding a coarser triangle network.

The other approach, which in Florianis work is based on the Delauney triangulation, is the construction of a pyramidal structure of surfaces of successively finer resolutions. In each step, a set of points are inserted such that the Delauney requirements are maintained.

Both approaches are to be considered as constrained approximation operators, i.e. that they produce data reduced models of an original, where the error of the approximation is less than a given tolerance.

The two methods would certainly require different multi-scale schemes. However, the discussion will not be carried through in this thesis, and we restrict ourselves to model multi-scale PLSs as trivial Multimodels, as implemented in appendix A. The approximation operator to be used is based on the so-called *data dependent* triangulation. In this method, the triangulation is not restricted to a planar process, but takes into account that a surface is to be generated over the triangulation. Thus, the data, or the 3D component of each point, becomes important. The principles of datadependent triangulation is outlined in [DLR90].



Summary

In this part, the Multimodel was introduced and developed to provide a general, integrated method for multiple modeling, i.e. the structuring and management of several variants of an initial model.

Differing in the properties of the variants regarded as digital models, two main classes of Multimodels were proposed (in addition to the trivial, explicit Multimodel). The multi-edition model handles a variety of models which are considered to be of the same resolution or accuracy. The multi-scale structure encompasses variants of the same model, but differing according to resolution or scale.

The multi-edition model was shown to be an inherently compact structure. Two specializations of the multi-edition model were proposed, the pseudo-delta model and the delta model. The former was to be considered as an inconsistent structure, while the latter indeed was consistent.

The multi-scale structure was considered inherently consistent. Two variants were introduced, the extremely compact selection model, and the decomposed model, which compactness may vary according to the models involved.

According to the proposed, informal methodology of Multimodeling, details were given on modeling piecewise linear curves (PLCs) and piecewise linear surfaces (PLSs). It was straightforward to design both multi-edition and multi-scale models of the PLC, but the PLS was shown to represent more complex problems, which we did not attempt to solve. To overcome the problems, one might resort to the flexibility of the recursive property of the digital model that allows a model to be partitioned into simpler structures, which in turn may be more easily managed in the Multimodel setting. However, a 'semi' multi-edition model was trivially designed.

One might design other variants of Multimodels than those proposed in this part, and indeed the compactness and consistency properties should be more detailed studied. Empiric research involving a variety of types of objects, both spatial and non-spatial, should be carried through to gain more firm insight in Multimodeling and its potential.

However, we have managed to outline a mechanism which is capable of handling sets of arbitrary digital models in a homogeneous way. Within the same framework, we are able to model multiple representations of both 'chalk and cheese' objects, such as texts of varying length and contents, and gradually more well-structured models, such as curves and surfaces. We have proposed a set of alternative representations, differing in what types of models they are able to handle, and in the degree of compactness and consistency they offer.

The Multimodel is indeed motivated from, but not restricted to, geographic information.

The Multimodel is a general approach hopefully useful in other areas characterized by the need to manage multiple models, i.e. a base model along with its variants. In computer aided geometric design, CAGD, hierarchical surface editing is a technique where a base-surface is edited by adding difference-surfaces in order to create new versions of the initial surface. It is a growing and promising research area, see e.g. [KW92] and references therein, and is an example on multiple modeling outside the GI domain.

In *Part III* we will take advantage of the Multimodel concept, and incorporate it in a suggested augmented map concept termed *Metamap*.

Part III

METAMAP

Outline

In this part, we propose an augmented map concept and call it Metamap. The development is based on the discussions and results in *Part I* and *Part II.*

We open the part with a brief discussion of information integration in GIS, i.e. basically the process of linking spatial and non-spatial information.

We then claim that the notion of topology, i.e. the set of various relations interconnecting objects of different kinds, is covering central aspects of information integration.

The elaboration of Metamap is started with a general description at a conceptual level. We introduce the notion of the *geographic element*, an abstract representation of the real world phenomenon to be modeled, independent of any spatial or non-spatial descriptions. We stress the importance and implications of the geographic duality, i.e. that a geographic element have both a topographic description and a thematic interpretation.

Then we design the Metamap *element* as the main building block in *a Metamap*, our contribution to the augmentation of the map concept. We supply the Metamap with a set of *topologies* as a flexible structure supporting information integration.

The topographic and thematic objects managed in a Metamap are supposed to be Multimodels customized for use in a cartographic setting.

Metamap is summarized as an object model, and we then discuss the compliance of the Metamap concept according to the definition of the augmented map concept.

Based on a proposed informal methodology for Metamap modeling, we design a simple and limited Metamap called *MINIMAP.* In appendix B, a modest implementation of MINIMAP will be carried through to demonstrate key aspects of both the Multimodel mechanisms and the Metamap principles.

Chapter 8

Integration of Geographic Information

One way of characterizing a GI system, is from the *information integration* point of view. This brings our attention to how different pieces of information, perhaps extremely heterogeneous and varying over time, are being linked to spatial objects. In a broader setting, information integration covers interconnections of the spatial objects, and of the various thematic objects. Recalling that the definition 7, section 4.2, of the augmented map model encompasses both spatial and non-spatial features, information integration also becomes an important issue in computer aided cartography based on such a map model. With the support of the Multimodel structures developed in *Part II*, the information to be integrated will span a range of scale or resolutions, may vary over time and be accessible in different editions as a result of generalization processes. One of the main goals in the Metamap development is to impose some coherence and consistency to this somewhat chaotic set of fragments of information.

Before describing some details on the integrating mechanisms, a brief review of trends in spatial information integration is given.

8.1 Strategies in spatial information integration

Shepherd characterizes information integration in GIS like this [She91]:

> Current thinking and research in GIS tends to ignore the information richness of the real world.

In the same article, Shepherd gives an overview of the field of geographic information integration. He classifies 'the classic approaches to information integration in GIS' as the two main directions outlined below.

- *The composite map model.* This is equivalent to the raster concept, see section 3.3.1, where a map is a 'stack' of regularly gridded sheets (overlays) superimposed to the same geographic area. Each cell is given a value corresponding to an attribute (thematic information). Spatial and thematic information is in this way inseparably bundled within the same overlay. The different overlays are integrated indirectly in the sense that the values of the corresponding cells together composes the information associated with that particular location.

- *The geo-relational model.* In this model, topographic and thematic information is separated and stored in different tables[1], linked together by a common key, which is the unique identifier of the object. In this manner, spatial features may be accessed through the non-spatial features and vice versa. The model has been adopted by many vector based systems.

It is natural to draw the parallel to the object/field dichotomy described in 3.3. The composite map model is clearly based on a field model, where each point in space is assigned thematic information (attributes). The object approach, viewing the world as distinct spatial objects assigned various information, corresponds to basic idea behind the geo-relational model.

Shepherd finds these current approaches somewhat limiting, and suggests three directions to follow in search of more consistent and coherent integration strategies:

- *Multimedia databases*
- *Interactive hypermedia*
- *Virtual reality systems*

Details of these approaches will not be given here.

The Metamap development will be founded on an approach which to a certain extent is related to the geo-relational model. We find that the separation of spatial and non-spatial issues reflects the notion of geographic duality, and in section 3.3.1 we expressed preferences towards the object approach, which the geo-relational model is based on. However, we find the dependency to relational databases severely limiting, as Shephard also does, and we will in the next section propose an approach to information integration based on the notion of *topology.*

8.2 Topology as information integration

In a wide perspective, one may view topology as a set of relations between objects. In several application areas, more specialized interpretations do indeed exist:

- In mathematics, topology is a discipline of its own, studying e.g. how certain geometric properties are conserved during continuous transformations, see e.g. [Des88] for an introduction to the subject.
- In data communication theory, topology refers to how various nodes in a network are connected, e.g. star topology, ring topology and tree topology, [Sta88]. The relations between the nodes are mainly of one kind, the `connected_to` relation.
- The topology concept as represented in the GIS standard VPF, [VPF92], section 5.2.2.3.1, *VPF Topology*, may stand as an example of a common interpretation of topology in

[1]As suggested by the name, the geo-relational model is founded on the principles of the relational database.

CAC. VPF recognizes four levels of topology, which describes relations between the geometric entities nodes, edges and faces. The lowest level, level 0, or the spaghetti structure, is in fact characterized by the *lack* of structure, as only nodes and edges and their coordinates are represented. Level 1 is essentially a non-planar graph, like Level 2 is a planar graph. The richest description is given on Level 3 where the surface is partioned by a set of mutually exclusive and exhaustive faces, where edges only meet at nodes. The relations used in building these topologies are typically `start_node`, `end_node`, `right_face`, `left_face`, `right_edge`, `left_edge`, and so forth.

Applying the topology concept in information integration yields a rich variety of associations, and it is possible to distinguish between different classes of relations, and different collections of relations constituting different topologies.

Consider the following example of this special use of topology. The two city objects, `Oslo` and `Trondheim`, has each a relation `is_part_of` to the object `Norway`. `Oslo` is related to `Trondheim` by the `is_south_of` association. The cities have each two `is_represented_as` relations, giving each a spatial description and a non-spatial description. Note that the relation `is_south_of` is not only supporting spatial inquiries, but also acts as a constraint barring `Oslo` to be relocated north of `Trondheim` during a generalization procedure (if the scale is small enough this is more likely to happen than one should believe at first glance). Such constraining relations may become important tools in various generalization procedures.

The main motivation behind the design and utilization of topological structures is to make operations on related objects or sets of them as efficient as possible. A sparse topology may cause the operations to take too much time, in contrast to a rich topology where computations are rapidly executed. This is an example of the classical time/space tradeoff in computer science. The topological structures requires extra storage space (and time to build). The tradeoff implies that the design of the topological 'utility system' should start with a thorough analysis of the wanted performance of the system, e.g. recognization of the most frequent used operations and how long time they are allowed to take.

The example indicates that there is indeed possible to build large and complex topological structures dealing with spatiotemporal information, and in section 10.2.4 an attempt is made to classify different topologies useful in an augmented map concept.

The next chapter initiates the construction of the Metamap framework by giving a description of the model at a conceptual level.

Chapter 9

Metamap: The Conceptual Model

In this chapter, we present an augmented map concept, according to definition 7. The model will be called Metamap, and is based on discussions and results achieved so far in the thesis.

Metamap will be treated on two levels in this chapter. First, a general description is given of the main structures. At this level, Metamap is to be considered as an abstract model, and could also be classified as a *metamodel.* From this metamodel it will be possible to derive a multitude of related specific models.

At the next level, the metamodel is further developed as additions and specifications are added. The model will no longer be abstract, and will become suitable as a basis for the implementation to come in appendix B. To emphasize that this is one of many possible realizations of the Metamap concept, this particular version will be called *MINIMAP.*

All inn all, Metamap will be treated at three different levels, since it will be realized as an implementation in appendix B. There are many degrees of freedom in this process. The Paper Map Model may be augmented in several ways, the description of a specific augmentation may be formulated as several different object models, and finally, one object model certainly implies a variety of implementations. Figure 9.1 illustrates the multitude of possibilities.

The first step in the Metamap development, is to identify the main building block, the *geographic entity.*

9.1 Geographic entities

According to the discussion on the Kantian/Descartian dichotomy in section 3.3.1, and the preferences expressed towards the object view of the world, a fundamental characterization of Metamap will be that it is possible to uniquely identify objects existing in the real world. Such objects will be called geographic elements in the thesis. Note that on this level, the only assumption made is that it is possible to give the object a spatial[1] characterization, included a location. Further details on how to perform the spatial description of the entity is not given,

[1]The concept of *space* is not so straightforward as it may seem at first glance. Gatrell [Gat91] reviews and discusses different approaches to the space problem, distinguishing between *metric spaces*, such as the familiar *Euclidian space*, a space based on the so called *Manhattan* metric, and *non-metric spaces*, e.g. *topological spaces* and *conceptual spaces.*

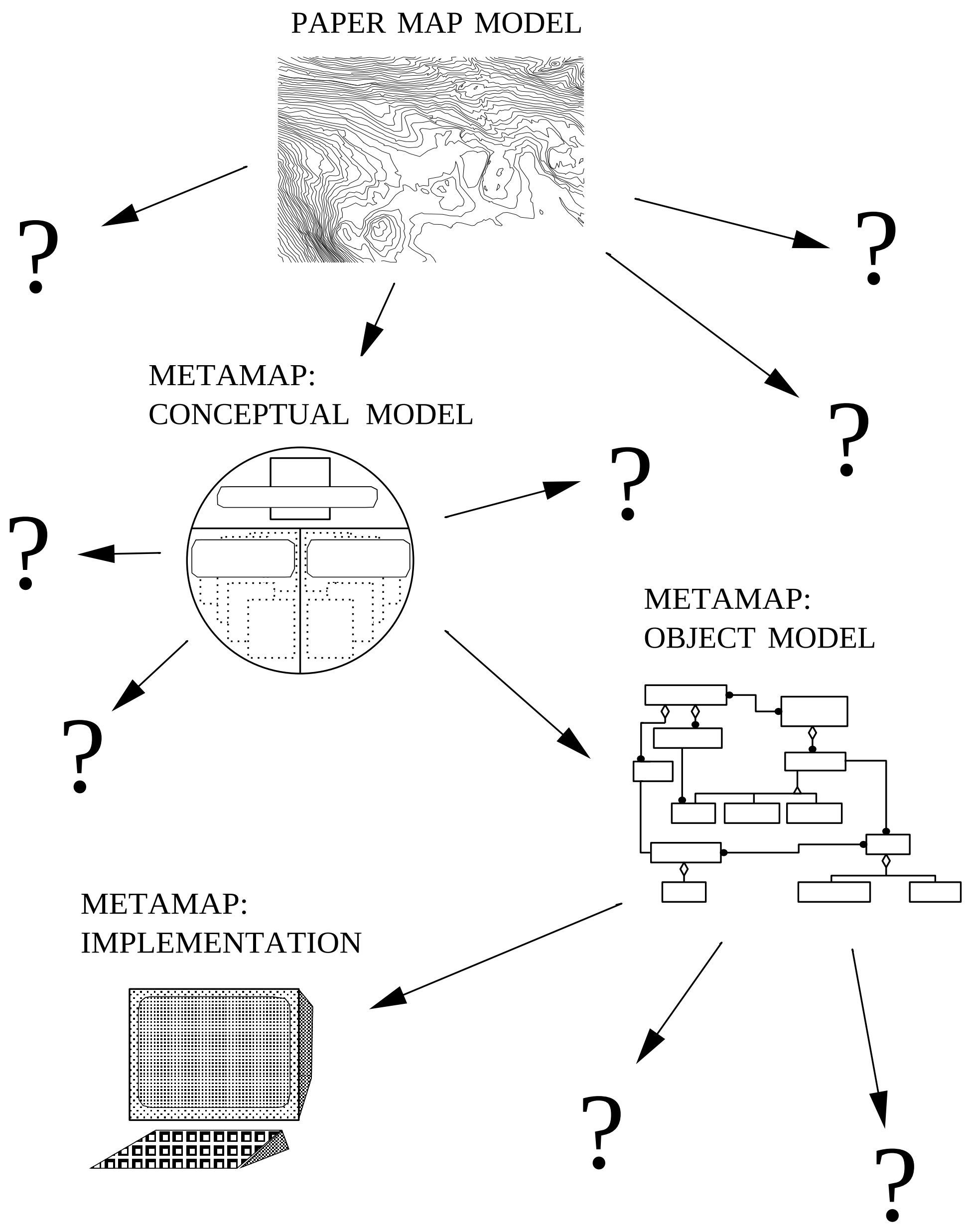


Figure 9.1: The degrees of freedom in modeling

GEOGRAPHIC ENTITY	NON-SPATIAL ENTITY
Notre Dame	Liturgy
The rise and fall of Communism in USSR	Communism as ideology
Bank of Switzerland	Your bank account
Norway	Utopia
University of Oslo	Geographic Information Science

Figure 9.2: Geographic and non-spatial entities

neither is it stated anything concerning how to classify, interpret or describe its non-spatial aspects, e.g. which thematic information that is assigned to it.

The geographic entity is a neutral entity, representing the most abstract description of a phenomenon in the real world possible to locate in space and time. In its most primitive implementation, the geographic entity is equivalent to a unique identifier, typically a name or some sort of code.

Stretching the idea to its limits, one may reach to the conclusion that practically all phenomena in the real world could be formulated as geographic elements. This is certainly not true, even though a large number of real world entities indeed is of geographic character. To illuminate the distinction between geographic entities and other real world entities, some examples are given in figure 9.2.

An important feature of the geographic entity is that it may be decomposed into a set of other geographic entities. Norway may typically be decomposed into a set of 19 counties. Going the opposite direction, Norway, Sweden and Denmark may be aggregated into a new entity, Scandinavia. Note that the decomposition/aggregation takes place on an abstract level, and is independent of any topographic or thematic description.

9.2 Geographic duality

The traditional map concept (definition 3, section 2.2) is strongly spatially oriented, and in section 3.3.3 it is claimed that this also applies to many GI systems.

As embedded in the geo-relational model (section 8.1), geographic information is characterized by the fact that it has both a topographic (spatial) and a thematic (non-spatial) component. This is what the term geographic duality is referring to in the thesis.

Shepherd [She91] underlines that access to the information from the spatial domain or the thematic domain, or a combination, implies increased efficiency in navigation and retrieval of spatiotemporal information. A philosophy paper from the European standardisation organization CEN [Com93], states that the new European GI standard in progress shall be 'based upon a conceptual scheme capable of handling spatial and non-spatial identifications'. In Metamap, spatial and non-spatial information will be treated as truly equal aspects of a geographic entity.

Both the spatial and non-spatial information is mandatory, even though the description may take a minimalistic form, such as a point in space or a one-letter thematic code. However, the topographic description has to be unique, while there may exist several thematic classifications.

As an example, consider the geographic entity `Svenner lighthouse`. Indeed, it is possible to give a unique description of the building, the ground plan, the height of the tower, the conical shape, the exact location in geographic coordinates and so on. The thematic interpretation may very well be of two categories, both as a navigational aid and a building lodging the lighthouse keeper. These two thematic descriptions would most probably be disjoint in most aspects.

Integration of the geographic entity, the notion of geographic duality and the Multimodel concept in *Part II*, yields what will be called the *Metamap element*. This will become the main object in the Metamap concept.

9.3 The Metamap element

The Metamap element[2] is the fundamental building block in the Metamap construction, as it is the *representation* (or model) of a real world geographic entity.

The Metamap element is defined at a conceptual level as follows:

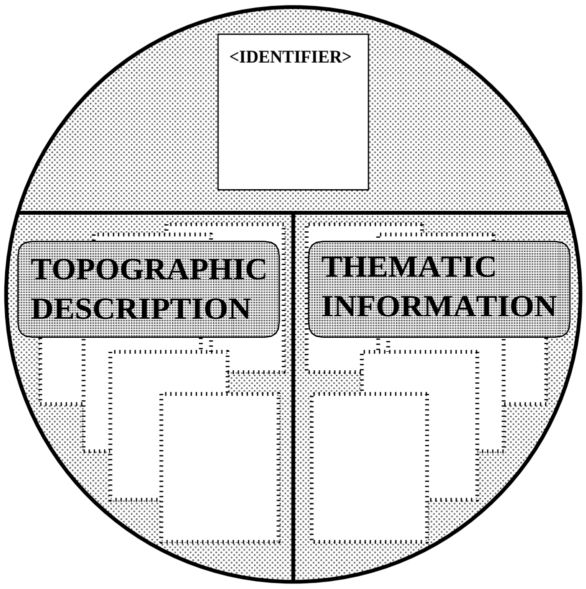


Figure 9.3: The Metamap element

Definition 16 (Metamap element) *A Metamap element is representation of a geographic entity. It is supplied with an unique identifier and a mandatory dual description of both its*

- unique *topography, or set of spatial features, and the*
- *(possibly)* multiple *thematic (or non-spatial) classification.*

[2]A Metamap element will, if allowed by the context, occasionally be referred to as just an 'element'. In the MINIMAP implementation in appendix B, the Metamap element will be referred to as a *geographic* element of implementational reasons.

These descriptions are Multimodels, integrating a set of variants according to

- *different scales (or resolutions),*
- *different editions (generalizations) and represented at*
- *different moments or intervals in time.*

A Metamap element may be an aggregation of several elements.

Figure 9.3 illustrates the definition.

The Metamap element may not seem particularly useful, standing all alone. Some structure has to be imposed to make different elements play together. The structure that will glue together a collection of Metamap elements is provided by topologies, as further addressed in section 9.4.

We will give some details on the topographic and thematic descriptions of a Metamap element in the next two sections.

9.3.1 Topographic elements

The topographic description of a Metamap element is essentially an entity of its own, a topographic, or spatial, element. It is a unique geometric definition of the shape and size of the geographic entity, independent of possible relations to other spatial or non-spatial objects.

Initially, there are indefinitely many types of spatial elements, so there is a need for a classification of main categories of such objects. The classification will be influented by the map purpose and the available technology for representation and manipulation of spatial entities. Nowadays, the field of computer aided geometric design (CAGD), is likely to be the main contributor of relevant methods, experiences a vigorously development. For an overview of the field, see e.g. [Far88].

The degree of reality characterizing an augmented map concept will heavily depend on how close to reality the spatial objects are modeled. This is due to the fact that peoples perception of reality is oriented towards to the physical domain, and in particular towards spatial features[3].

The planar projection of rivers, coastlines and county borders in the Paper Map Model is clearly not close to our perception of reality, but rather a highly abstract derivation. Still, we are so used to this map model that most people feel quite comfortable with the abstract representation.

The first steps towards a more realistic model, are to liberate ourselves from the planar projections and to introduce 3D modeling. There are many solutions and many ways to accomplish this task, and we will suggest and implement only modest refinements, see section 10 and appendix B. The topographic elements will be restricted to a few and quite simple geometric structures.

9.3.2 Thematic elements

Ideally, the thematic elements encompass information on any format that may be handled digitally. The various multimedia data types, such as plain text, images, audio and video

[3]This is indeed not a trivial question, how to perceive the reality. In fact, the question has through a couple of millenniums been a favorite theme among philosophers.

fit naturally as thematic elements. Today, most GI systems only support strongly formatted text, such as records, and occasionally digital images as e.g. satellite recordings.

However, according to the scope of the thesis, there will be paid little attention to the rich domain of non-spatial information. To ensure the conceptual comprehensiveness of Metamap, we merely assume that it is possible to design and implement realistic thematic models.

9.4 Topological structures

As mentioned in section 8.2, the term 'topology' will in the thesis refer to the structures of relations between various entities in a map (included augmented maps).

In the Metamap context, four main classes of topology will be recognized:

- *Map topology.* This is simply the relations linking Metamap elements together into a Metamap. These may be primitive relations, typically constituting an unordered collection.

- *Primary topology.* The definition of the Metamap element implies that it may be aggregated of other Metamap elements. This gives rise to a set of `is_aggregated_of` relations, that will be referred to as primary topology.

- *Secondary topology.* The Metamap element has a mandatory and unique spatial description, and a mandatory (possible multiple) non-spatial description. The structure of the relations between the topographic and the thematic domains are characterized as secondary topology.

- *Tertiary topology.* This is describing the optional relations between spatial entities, and between non-spatial entities, thus yielding two specializations,
  - *topographic topology*, which is how the spatial objects are related to each other, and
  - *thematic topology* describing the various possible non-spatial interconnections.

In the next sections, some details will be given on the web connecting the elements in a Metamap. Certain aspects of relations as topology is treated in [MS93] and [BS93].

We will now give some details on the different topologies.

9.4.1 Map topology

The map topology is simply the collection of Metamap elements constituting a Metamap. The most simple realization is the unordered set, but more efficient structures as an ordered set defined as a linked list may be preferred in real applications.

We stress that on this level, anything but the unique identifiers of the elements are known. The map topology is therefore to be considered to be the first door to open when dealing with a Metamap, and a mechanism to traverse the elements at the most abstract level.

In order to make definition 17 of Metamap sensible, the map topology is mandatory.

9.4.2 Primary topology

The primary topology is a more useful structure than the map topology, describing the associations between several Metamap elements. Unlike the map topology, the primary topology is optional in the sense that the Metamap may consist of single Metamap elements without any relations.

9.4.3 Secondary topology

The secondary topology defines relations between the topographic (spatial) description and the thematic (non-spatial) information. In other words, the secondary topology is the key to the duality of geographic information.

There are two main approaches to the design of secondary topology. The topology may be indirectly derived from the common geographic entity, or it may be explicitly modeled as associations between the spatial and non-spatial domain. Since the derived secondary topology always is available, the explicit secondary topology is optional.

An explicit topology will consist of relations from one spatial object or one or more thematic objects.

9.4.4 Tertiary topology

Despite that the tertiary topology is of an optional nature, a well designed structuring of the spatial respectively non-spatial domain is potentially one of the most important challenges in the development of a Metamap realization.

Spatial topology

The spatial description of a geographic entity yields a geometric object. The relations between such objects will together form the spatial topology of a Metamap.

The main scope with topology in general and spatial topology in particular, is to provide a 'utility' structure to support a variety of operations on the objects. The majority of such operations, like generalization procedures, computation of volumes, areas and distances and finding shortest paths in a network, will probably include some kind of spatial searching or sorting. To speed up computations like these, preprocessing algorithms are widely used. Such methods are essentially constructing more or less sophisticated topologies of relations, e.g. hierarchies and tree-structures, making it easier and faster to perform traversals among the objects. This has become a discipline of its own, 'computational geometry', see e.g. [PS85] for an introduction.

The relations may not exclusively support computational procedures, but for example express different *constraints*.

Other associations may be more directly expressing geometric properties, e.g. if an object 'inherits' some features from another object. The MINIMAP development in chapter 10 will utilize such relations.

Thematic topology

As the thematic elements fall outside the scope of the thesis and are modeled and implemented in a primitive manner, the same limiting approach is taken when modeling the tertiary topology involving non-spatial entities. As a matter of fact, there will not be introduced such relations at all in the development of MINIMAP, thus leaving the non-spatial features as independent objects only linked to the spatial entities by the secondary topology, see section 9.4.3.

9.5 The Metamap

In the Metamap context, a map, or *a* Metamap, is essentially a collection of Metamap elements, as described in section 9.3, supplied with a set of utility mechanisms for e.g. presentation purposes. To defend this suspiciously simple approach, we try to describe a traditional topographic map as a Metamap.

Assume that we have a collection of Metamap elements. Let the main element be the topographic surface. The other elements, points, curves, and areas are 'resting' on this model of the terrain. According to a certain projection and a view given e.g. as a rectangle in geographic coordinates, we project the spatial description of the Metamap elements to a plane. The height contours may be derived from the topographic 3D surface as curves defined by sectioning the terrain at given constant elevation values.

The projections are made according to a specified scale (resolution), a certain moment in time and an edition. This is accomplished since both the spatial and non-spatial description of a Metamap element are Multimodels.

So far, only spatial objects are included in the map, and only as points, curves and areas. How to draw, or present, these geometric is not yet given. We then supply the Metamap by a legend, which on basis of the thematic classification associated with each spatial entity, gives rules on how to present the object, e.g. that roads of a given category, represented in a certain scale and edition, should be drawn e.g. as dashed lines with a given thickness and color. The legend should also give similar information on how to print e.g. names associated to objects, i.e. how to present the non-spatial information.

Finally, the Metamap should supply information on the formatting of the projected objects, e.g. such that it would be possible to export the information according to a standard, say VPF (see section 3.3.3).

Based on the example, a more precise definition of a Metamap is given as follows:

Definition 17 (A Metamap) *A Metamap is essentially a collection of Metamap elements with the following additional information and functionality:*

- *Information:*
  - *Topological structures, as outlined in section 9.4.*
  - *Map legend, i.e. a description or dictionary on how to present, depending on export format and Multimodel parameters, both spatial and non-spatial information.*

- *Functionality:*

- *Edit: Insert, update, delete, and access elements, change topology and so on, given certain Multimodel parameters*[4].
- *Select a part (window) of the Metamap, according to map window and Multimodel parameters.*
- *Present a window*[5] *of the Metamap, according to legend and presentation format, e.g a 3D visualization system or a traditional paper map projection.*
- *Export a window of the Metamap, according a given export format, e.g. a certain GIS standard like the VPF.*

Note that there are no other relations between the Metamap elements than that they all are part of the same unordered set. The necessary structures for traversal and retrieval are provided by topologies between the elements, as explained in the next section.

Also note that the Multimodel basis of the topographic and thematic descriptions in each Metamap element ensures that the Metamap has access to and may benefit from mechanisms embedded in the Multimodel structure, e.g. data reduction operators and other generalization utilities.

This is the conceptual definition of the augmented map concept suggested in the thesis. The sections to come will give some more details, and eventually, in appendix B a modest implementation is carried through based on definition 17.

In the next section Metamap is formulated as an object model.

9.6 Object model

Based on the results and discussions in this part, we illustrate the Metamap concept with an object model. The model may be considered to be a *metamodel*, since we are able to design a variety of more detailed and specialized object models. These models may vary significantly. In appendix B we will present one of many possible instantiations of the metamodel shown in figure 9.4.

The metamodel is quite simple. The main class, the `Metamap`, is essentially an aggregation of one or more `GeographicElement`s. This object is the most simple and abstract representation of a geographic phenomenon, as described in section 9.1. The aggregation constitutes the map topology of the Metamap (see section 9.4.1).

The various `GeographicElement`s have some sort of relations with a `TopographicElement`, which basically is related to a Multimodel spatial description. Likewise, it is associated to a `ThematicElement`, the non-spatial part of the information, which also is based on a Multimodel. We assume that some sort of Multimodel library is provided, e.g. the generic MULTIMOD developed in appendix A.

Between the thematic and topographic elements, we may optionally have defined a secondary topology, see section 9.4.3. This topology is considered as a utility structure enhancing traversals and navigation in the Metamap structure. Note that we also have an

[4] By 'Multimodel parameters' we refer to specification of which scale, edition and moment/interval in time that is wanted.

[5] A map window is a description of the view of the Metamap that is wanted, given e.g. as a 3D box, a rectangle in geographic coordinates or a description of a vertical section.

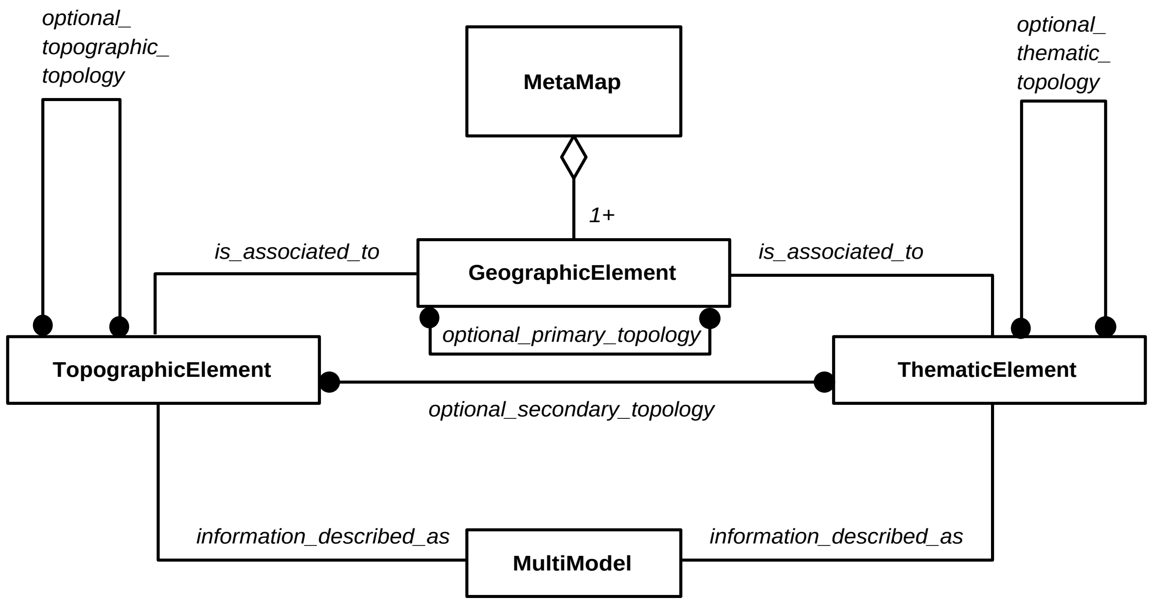


Figure 9.4: Metamap as metamodel

implicit secondary topology, as the spatial and non-spatial descriptions both are related to the `GeographicElement`.

Between the `TopographicElement` objects, and between `ThematicElement` objects, we have optional tertiary topologies (see section 9.4.4), i.e. relations between the objects to facilitate operations such as generalization operators.

We close the thesis by briefly evaluating Metamap as an augmented map concept.

9.7 Metamap as an augmented map concept

In order to ensure compatibility with definition 7 of an Augmented Map Model, a verification of the Metamap as defined in 17 is carried through as follows:

- *Realistic representation:* The Metamap concept supports spatial modeling in 3D and time, and is able to incorporate a wide variety of thematic models. In this way, a Metamap may be characterized as fairly realistic. However, by not taking advantage of the potential of the Metamap, degenerated models close to the Paper Map Model can be designed based on the Metamap concept.
- Decomposition and duality: A Metamap is a collection of Metamap elements, which are unique geographic entities associated with both spatial and non-spatial descriptions.
- Advanced and flexible data structures: Metamap is indifferent to the structures representing topographic and thematic representation. However, a well designed Metamap should obviously take advantage of the state of the art in available technology. If not, yet another instance of the Ptolemiac Paradox, as outlined in the *Introduction*, is generated.
- *Unique location combined with multiple thematic interpretations:* A Metamap is based on geographic entities with uniquely defined spatial features, included locational de-

scription, and associated with multiple thematic descriptions.

- *Different scales, moments or intervals in time and editions:* The Multimodel basis of the spatial and non-spatial information ensures the ability to handle these variations.
- *Seamless representation:* The concept of the map window, specifying an arbitrary part of the reality model to be presented, clearly implies a seamless representation.
- *Presentation independence:* One of the characterizations of a Metamap is that it should be able to support a multitude of presentations, or export formats, such as 3D perspective views, traditional 2D maps and profiles. The concept of the map legend hides much of the formatting information that characterizes a specific presentation, adding yet another dimension of presentation independence.

As a conclusion, we may say that the Metamap complies well with the Augmented Map Model. Still, degenerated instances can be derived that may not be characterized as an AMM, but rather as close to the Paper Map Model.

In the next chapter, we take a step further towards an implementation, with the presentation of the MINIMAP model.

Chapter 10

MINIMAP: Towards an Implementation

In this chapter we give an example of the use of the Metamap concept in designing a more specific map model, which we will call MINIMAP. We start by proposing an informal modeling methodology.

10.1 Metamap modeling

The design of a map model should be based on a thorough analysis of the actual real world phenomena to be modeled. In addition, well defined specifications of the functionality and capability of the systems based on the map model are of vital importance in the design process.

The process of Metamap modeling may be described by an informal methodology:

- Identify the geographic phenomena to be modeled, according to the purpose of the map and the world view the application is based upon. Specify a set of geographic elements to represent the real world geographic entities.
- Design a set of classes of spatial descriptions to cover the selected geographic entities.
- Outline the various types of thematic descriptions needed to supply non-spatial information.
- Investigate and decide what kind of topological structures that will ensure a certain level of performance ability.
- Decide the cardinality, or 'dimension' of the Multimodel structure, e.g. if the information should be allowed to vary according to both scale, time and edition.
- Design the Multimodel structures (see section 7.1).
- From the system specifications, operations should be defined covering
  - edition of the Metamap, including generalization operators, and
  - presentation and export routines, including specification of how to present the map (the map legend).

Based on the informal methodology, we design the very simple map model MINIMAP.

10.2 MINIMAP

We start by presenting the very banal world view which MINIMAP is based upon.

10.2.1 Geographic elements

We restrict the MINIMAP to model basically three classes of geographic phenomena, the terrain, or the surface of earth, the ocean, and buildings. Note that on this level we do not say anything about the spatial or non-spatial characteristics of these classes of objects.

Some details of the spatial descriptions are given in the next section.

10.2.2 Topographic elements

The main idea behind the spatial structures in the Metamap of this thesis, is that the reality may be decomposed into a main object, a unique *topographic surface*, representing the spherical surface of the Earth, and a collection of objects, the buildings, that are scattered throughout this surface. The surface may be said to *support* all the other objects.

The surface is understood to be the *solid ground* of the Earth. This term has several geological interpretations, referring to various layers of rocks and sediments. In this approach, the surface is defined to be the topography after removing:

- Man made features such as buildings and roads,
- vegetation, e.g. forests, and
- water bodies, i.e. lakes, glaciers, rivers, canals and the entire ocean.

Further, for the case of simplicity, the topography is assumed to be an *explicit surface*. The topography may be expressed as a bivariate function in spherical coordinates, $S : \mathbb{R}^2 \rightarrow \mathbb{R}$, such that

$$S(\phi, \theta),\ \phi \in [-90, 90],\ \theta \in [-180, 180].$$

The value of the function is equivalent to the elevation at the geographic point (ϕ, θ). $\phi \in [0, 90]$ corresponds to latitudes, in degrees, in the *northern* hemisphere, and $\phi \in [-90, 0]$ corresponds to latitudes in the *southern* hemisphere. Correspondingly, $\theta \in [-180, 0]$ is the longitude in degrees *east* of the Greenwich meridian, and $\theta \in [0, 180]$ is the longitude *west* of the zero meridian.

Attached to this naked topographic surface, various spatial features may be identified. In the thesis, the selection of such objects is limited to features of two different classes. The surface may be said to *support* these objects:

- *The ocean*
  The feature ocean, O, will simply be characterized by a constant radius $z \in \mathbb{R}$, thus neglecting tidal variations. It will be assumed that every point of the topography below this level is part of the sea floor.
  If the earth is defined as the volume $E = \{(\phi, \theta, \rho) \mid S(\phi, \theta) \leq z\} \subset \mathbb{R}^3$, and define O as the volume limited by the ocean level, $O = \{(\phi, \theta, \rho) \mid S(\phi, \theta) \leq z\} \subset \mathbb{R}^3$, we have the body of the ocean is implicitly defined as the volume "between" the sea level and the sea floor defined as
  $$(\mathbb{R}^3 \setminus E) \cap O$$

- *Boxes*
  The box object is ment to symbolize man made features such as buildings. A box B, is parametrized by four 2D points representing the 'ground floor' as an arbitrary quadrilateral, and a parameter h representing the height of the building,

  $$B = \langle p_1, p_2, p_3, p_4, h \rangle, \quad p_i \in \mathbb{R}^2, \quad h \geq 0.$$

  Note that this gives a unique 3D description of the box, assuming the 'walls' and 'ceiling' to be perpendicular respectively parallel to the 'ground floor'.
  However, the box is defined on a zero level plane. To give the box a correct elevation, we need a simple relation to the supporting surface. The box B is supposed to derive the elevation z from the surface S in the following manner:

  $$z = \min_{i=1}^{4} S(p_i),$$

  assuming the points are in the interval $[-90, 90] \times [-180, 180]$.
  The box may be considered as a consistent representation with respect to changes in the surface.

Note that the spatial elements should be multiply modeled as described in *Part II.* This implies that within one object, several variants of essentially the same spatial object is handled. The variants may differ according to scale, time and edition.

10.2.3 Thematic elements

According to the scope of the thesis, there will be paid relatively modest attention to the rich domain of non-spatial information. Still, to make the Metamap conceptually comprehensive, we include one class of thematic element:

- *Text*
  A text is defined to be an arbitrary collection of characters, assuming they are defined in one of the standard character sets.

As with the topographic elements, the text is assumed to be a Multimodel.

10.2.4 Topological structures

We have in fact already defined a tertiary topological relation, in describing how the box derives its elevation from the surface.

We will not design any primary topologies, i.e. relations between geographic elements, neither any secondary relations, i.e. topology between thematic and topographic elements. Still, we have an implicit secondary topology in the sense that a geographic element have both a thematic and a topographic description.

We introduce one more tertiary topology between the spatial objects, by the concept of *supporting* and *supported* elements.

An topographic element is said to support a collection of other elements, if a generalization of the elements should effect the supported elements in a specified manner.

An example of such a topology could be that a translation generalization of a defined area of the surface should imply a corresponding translation of objects supported by the area, say a group of buildings.

This special kind of dependencies are frequently used in 2D and 3D illustration applications ('grouping' and 'ungrouping' of objects), and is a well established concept in computer graphics in general (see e.g. [FDFH90], chapter 5).

In our case, the surface of the Earth is default supporting all other spatial objects. In fact, the derivation of the elevation of the Box class is an example of a specified relation between supporting/supported elements.

We close the chapter with an object model formulation of MINIMAP.

10.2.5 Object model

By extending and specializing the metamodel in figure 9.4, according to the discussion in this section, we get the following object model:

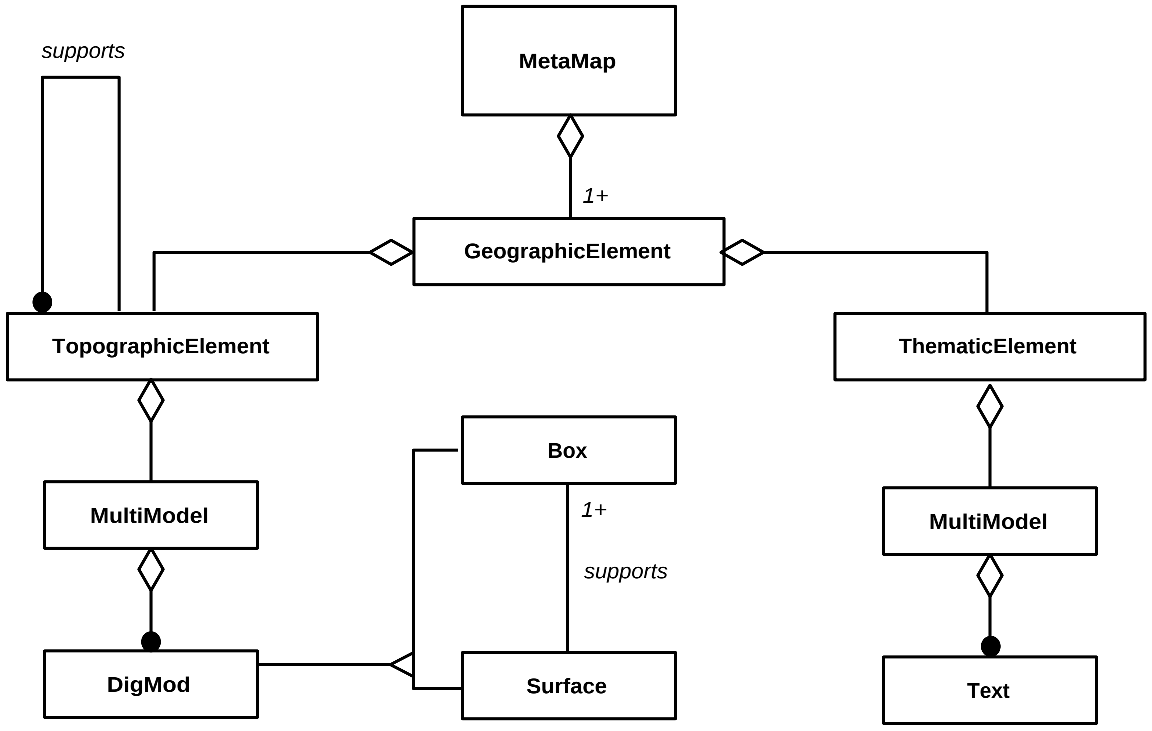
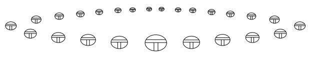


Figure 10.1: MINIMAP - an object model

Note that we do not represent the ocean explicitly, but implicitly as a given z-value. For more details on the object model, see the MINIMAP implementation in appendix B.



Summary

In this part, we have accomplished to develop an example of an augmented map concept, as proposed in *Part I*, definition 7.

We emphasized the importance of the information integration aspect in modeling geographic information. It was claimed that the use of the topology concept is an efficient tool in the process of interconnecting the various spatial and non-spatial objects.

The Metamap was designed as an augmented map concept, basically defined to be a set of Metamap elements with some additional functionality, e.g. to permit various visualizations of a certain view of the Metamap.

The Metamap element is the representation of the abstract geographic entity, the neutral 'thing', including a unique spatial description and one or several thematic interpretations. The two latter elements are both assumed to be Multimodels, as described in *Part II*, thus allowing the Metamap element to be represented in a multitude of scales, editions and moments or intervals in time.

The Metamap was shown to be compliant with definition 7 of the augmented map concept.

We proposed a Metamap modeling methodology in order to design a particular map model. We then used the methodology to develop MINIMAP as a limited example of a Metamap. MINIMAP will be implemented to some extent in appendix B.

Part IV

CONCLUSIONS AND IMPLEMENTATIONS

Chapter 11

Results

The main achievement of the thesis is that we have shown that it is possible to augment the traditional map concept to meet new challenges provided by computer aided management of spatiotemporal information.

We have accomplished to develop an augmented map concept, starting on a conceptual level and resulting in a computer implementation, as a conceptually comprehensive model. We have followed an original approach in the sense that similar frameworks, to our knowledge, are not suggested in contemporary literature.

In this chapter we summarize some of the results achieved during the process.

Cartography and GIS

We identified and defined the Paper Map Model, which has its roots in ancient cartography, and showed that it is the core in any GI system today. We claimed that this Ptolemaic paradox represents a bottleneck in computer based handling of geographic information.

In contrast to some trends in GI science, which tend to drift away from cartography and into the realms of database theory, primitive geometric modeling and low level algorithmic optimization, we claim that GI science may benefit from not discarding the notion of the map. By regarding the long and rich traditions of cartography as a firm foundation, we proposed to augment the Paper Map Model as a step towards a new core model of the real world for use in GI applications.

The concept of cartographic generalization inherently generates multiple representations of geographic objects, varying according to scale, edition and moments or intervals in time. This problem has been only partially addressed in GI research, frequently with emphasis on multi-scale representations of simple spatial structures. We suggested a more general and integrated approach by introducing the Multimodel.

Multimodels

The Multimodel is basically a high level conceptual mechanism for managing collections of model variants in a homogeneous manner, i.e. that we may perform the same operations on the various sets, without paying attention to details concerning the actual representation of the objects.

We designed a few specialized Multimodels, differing basically due to the properties of the models they could handle. Key operations associated with the Multimodels were algorithmically outlined. The different Multimodels was shown to represent the variants in various degrees of compactness and consistency.

An implementation of a generic Multimodel customized to handle generalized geographic information was carried through. We defined Multimodels to some detail for piecewise linear curves and piecewise linear surfaces.

The Multimodel concept should not be restricted to the domain of spatiotemporal information. We may experience a spinoff effect due to the versatility of our approach, in the sense that the Multimodel concept may be adopted and adapted in other application areas, such as computer aided geometric design (CAGD) and general information systems.

Metamap

Metamap was developed as one of many possible realizations of the augmented map concept.

The two main ideas behind the Metamap concept are:

- To provide structures for integration of the duality of geographic information, i.e. that a 'thing' has both a spatial and non-spatial description.
- To manage the multitude of generalized variants.

The integration of spatial and non-spatial information is maintained by a set of topologies, or relations between various objects. The topologies may also be used to impose constraints and to facilitate navigation in collections of geographic elements.

Metamap is incorporating the Multimodel concept, in the sense that both the spatial and the non-spatial objects are assumed to be Multimodeled.

We presented an informal methodology of Metamap modeling, and applied it in the design of a limited Metamap called MINIMAP. MINIMAP was rudimentary implemented to demonstrate key features of the augmented map model.

Chapter 12

Future Research

The interdisciplinary nature of cartography and GI science is the background for the somewhat grand scope of the thesis.

We have accomplished to design and develop structures which we have used to augment the traditional map concept. However, even if the framework is conceptually comprehensive, the lack of details in several areas is obvious.

We realize the substantial amount of research needed for developing the ideas in the thesis into well functioning tools in a GI setting. In this chapter we will try to sort out some main areas of interest in possible future research on augmented map concepts. Due to the wide span of research areas, we see the advantages of integrated and coordinated efforts. A research strategy implying a number of single, independent projects is likely to yield yet another instance of the Ptolemaic paradox.

Cartography and GIS

In *Part I*, we briefly investigated cartographic generalization. We claimed that this is a fundamental issue in geographic modeling, resulting in variants of different kinds. However, we did not go into detail concerning different generalization operators. A thorough understanding of the nature of the generalization process is of vital importance in GI modeling. A successful development of the Multimodel concept, as described in *Part II*, is e.g. quite dependent of such knowledge.

There are several promising ongoing research projects in this field, see e.g [BM91] for an overview of contemporary trends. The structures and functionality of generalization should supply some of the main premisses in development of the Multimodel concept, which would benefit from incorporating results from this area.

Multimodels

In *Part II* we proposed the Multimodel as a mechanism for management of model variants, and claimed that the various specialized Multimodels provided certain degrees of consistency and compactness. However, we did not carry through any empirical experiments to support or verify these assertions. Such investigations should be of central interest in further research on Multimodels.

We only made realistic implementations of multirecords, multicurves (PLCs) and multisurfaces (PLSs). Clearly, to investigate the versatility of the Multimodel concept, a wide variety of digital models should be implemented. The thesis have a geometric approach, and special attention should therefore be paid to validate the Multimodel approach in the thematic domain of geographic modeling.

Such implementations may also reveal other classes of Multimodels than those proposed in the thesis. Certainly, detailed algorithms, covering a complete set of operations, should be the result of more extensive research in Multimodeling.

A detailed survey of related approaches should be carried out. Adaptations has to be made to incorporate existing techniques, such as approximation methods within CAGD.

Questions concerning database issues will sooner or later have to be answered. Such considerations are absent in the thesis, and efficient Multimodel implementations will to a large extent be dependent of the database implementations.

We have occasionally claimed that the Multimodel approach may become useful in other areas than GI science. A survey and study of alternative application areas, like facelifting techniques in CAGD (see [KW92]), should be of interest.

Metamap

The proposed realization of the augmented map concept, Metamap, was rudimentary implemented in appendix B, and key aspects of the augmented map concept was demonstrated.

We believe that many alternative formulations of the AMM may be successfully carried through, and such alternatives should be investigated. However, it will require extensive research to reveal the strengths and weak points of an augmented map model.

An investigation of Metamap should imply an extensive exploration of a wide range of 'real world' cases. Such experiments would be extremely useful in the process of refining and adjusting the concept.

The notion of topology as an information integration tool would perhaps represent the most challenging part of a Metamap study. The design of such topological structures implies a large degree of freedom, and assumes thorough knowledge in both the application areas and their performance demands, data modeling in general and algorithmic design in particular.

An important issue to consider, is how to integrate the Metamap approach with existing concepts and methods in GI science. Special attention should be paid to investigate the compliance with selected GI standards. The Metamap concept should also be adjusted to make smooth integration with other areas possible, like visualization and computer graphics.

At last, one should not exclude the possibility of applying the Metamap concept in other areas than GIS.

Chapter 13

Epilogue

In chapter 1, we asked some questions which initiated the quest resulting in the concepts of the Multimodel (*Part II*) and the Metamap (*Part III*). In *Part IV*, we carried through some rudimentary implementations of the concepts, and performed some experiments to highlight some selected aspects of Metamap and Multimodels.

The initial questions in chapter 1 have been an inspiration throughout the thesis. We now close the thesis by giving some direct remarks on how we have managed to solve the problems outlined by the questions 1, 2, 3 and 4.

Answer 1 Crossing contours.
No direct solution of this problem has been given. However, an implicit answer may be found in the 3D capability of an augmented map concept. If we represent the topography (or terrain) as an explicit surface, we may derive the height contours by horizontally sectioning the surface. Since the terrain may be represented as a Multimodel with varying resolution (or scale), and the surface is explicit, *the sectioning process will guarantee that crossing of contours will not occur. Figure 13.1 shows a contour map of the terrain visualized in figures B.1 and B.2, after reducing the number of points in the triangulation from 974 to 56. We do not observe any crossing contours*[1]*, such as displayed in figure 1.2 and 1.3.*

Answer 2 Dislocation.
The dislocation problem, as illustrated in figure 1.4, may be approached with the help of the topology mechanisms outlined in 9.4. By defining a tertiary topology as a relation constraining the road object to be located inside *the polygon representing the island, we may prevent such dislocations.*

Answer 3 Multi scale structures.
We have indeed made an attempt to suggest a solution to the problem outlined in question 3. In Part II *we introduced the Multimodel concept to address this and related problems. Implementations and experiments were performed in appendix A to highlight selected features of the Multimodel, and we showed that under certain assumptions the Multimodel is able to integrate a set of variants in a compact and consistent manner.*

[1]However, the contours are not optimally shaped from a cartographic point of view. Thus, to design algorithms for data reduction of PLSs yielding satisfactory contours remains a challenge.

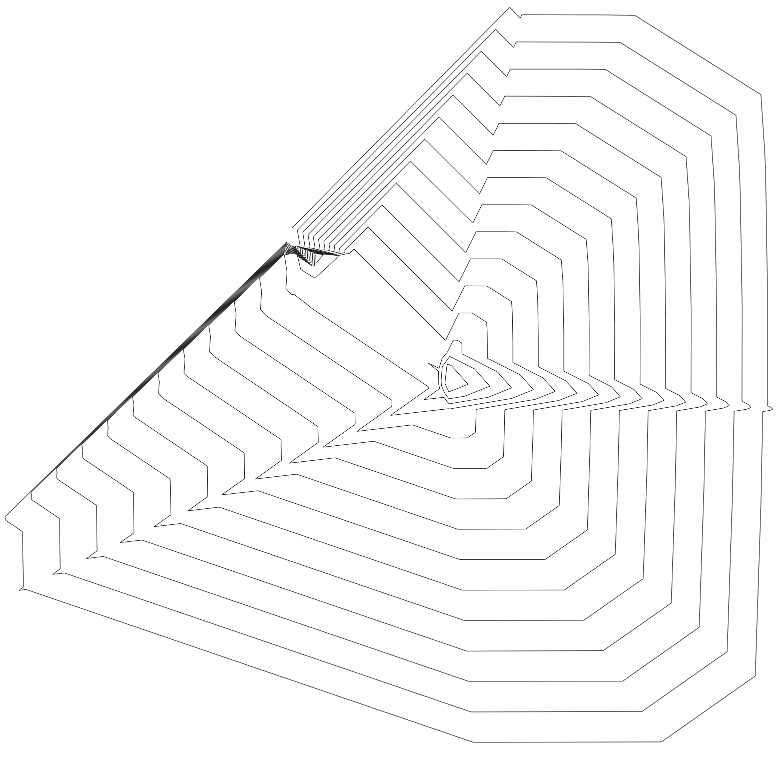


Figure 13.1: Contour map after heavy data reduction of terrain

Answer 4 Augmentation of the map concept

In Part I, *we claimed that the traditional map model, called the Paper Map Model, PMM, is inadequate as a basis in GI systems.*

An augmentation of the PMM was proposed, and in Part III *we developed such an augmented map concept, called Metamap. Later, in appendix B, we carried out a minor implementation that indicated the potential of the concept as a core map model in GI systems.*

In the thesis, a lofty framework has been developed, and some mechanisms have been specified. Within the framework, some relatively limited and simple problems have actually been solved. We may clearly have found solutions by more straight forward and simple approaches. Still, we believe that the more comprehensive approach will pay off when the complexity of the problems increases.

However, we realize that both the Multimodel and the Metamap concepts need substantial amounts of refinements, enhancements and corrections to become useful in 'real-world' applications. Still, we hope we have accomplished to establish a starting point from where it could be possible to develop an augmented map model that may become the core of future GI systems.

Bibliography

[Aas92] Rune Aasgaard. *Automated Cartographic Generalization, with Emphasis on Real-Time Applications.* PhD thesis, Norwegian Institute of Technology, 1992.

[AD91] Erlend Arge and Morten Dæhlen. Simplification of piecewise linear curves. Technical report, Center of Industrial Research, Oslo, 1991. ISBN 82-411-0339-9.

[AD92] Erlend Arge and Morten Dæhlen. On Generalization of Geometric Objects. In *Neste Generasjon Geografiske Informasjonssystemer 1992*, pages 91 – 99, Trondheim, Norway, Desember 1992. Institutt for Kart og Oppmåling, Universitetet i Trondheim, Norges Tekniske Høgskole.

[ADWM92] E. Arge, M. Dæhlen, G. Westgaard, and G. Misund. Efficient numerical line generalization. Technical report, Center of Industrial Research, Oslo, February 1992. ISBN 82-411-0373-5.

[Ans88] R. W. Anson, editor. *Basic cartography*, volume 2. ICA (International Cartographic Association), 1988. ISBN 1-85166-232-4.

[AS81] Øystein Andersen and Kari Strande, editors. *Kart og Kartbruk.* Universitetsforlaget AS, 2. edition, 1981. ISBN 82-0018150-2 (In Norwegian).

[AS89] Jostein Amlien and Leif Sørbel. Automatisert generalisering av geomorfologiske temakart. Meddelelser fra Geografisk Institutt. Naturgeografisk serie. Rapport nr. 13, Geografisk Institutt, University of Oslo, 1989. (In Norwegian).

[Ass84] ICA (International Cartographic Association), editor. *Basic cartography*, volume 1. ICA (International Cartographic Association), 1984. ISBN 90 70 310 05 8.

[ATB90] Khaled K. Al-Taha and Renato Barrera. Temporal Data and GIS: An Overview. In *Proceedings of GIS/LIS '90*, November 1990.

[ATSS92] Khaled K. Al-Thala, R.T Snodgrass, and M.D Soo. Bibliography on spatiotemporal databases. Anonymous ftp from cs.arizona.edu, bib/spacetime.bib., 1992. Latex format.

[BK91] Bud P. Bruegger and Werner Kuhn. Multiple Topological Representations. Technical Paper Series no. 17, National Center for Geographic Information and Analysis (NCGIA) and Department of Engineering, University of Main, 1991.

[BM91] Barbara P. Buttenfield and Robert McMaster, editors. *Map generalization: Making rules for knowledge representation.* Longman Scientific and Technical, 1991. ISBN 0-582-08062-2.

[Bro49] A. Lloyd Brown. *The Story of MAPS.* Little, Brown and Company, 1949.

[BS93] Olav Mork Bjørnås and David Skogan. MultiModels and spatio-temporal modeling in Object-Oriented GIS. Master's thesis, Norwegian Institute of Technology (NTH), December 1993. (to be published).

[Bur90] P. A. Burrough. *Principles of Geographical Information Systems for Land Resources Assessment.* Monographs on soil and resources survey no 12. Oxford University Press, 1990. ISBN 0-19-854592-4.

[Bur92] P.A. Burrough. Are GIS data structures too simple minded? *Computer & Geosciences*, 18(4):395 – 400, 1992.

[Com93] Comité Européen de Normalisation (TC 287 - WG 3). Philosophy paper revised, June 1993.

[Cur88] James P. Curran, editor. *Compendium of Cartographic Techniques.* Elsevier Applied Science Publishers, on behalf of International Cartographic Association, 1988. ISBN 1-85166-229-4.

[Des88] J.V. Deshpandé. *Introduction to Topology.* Tate McGraw-Hill Publishing Company Limited, 1988.

[DL92] Morten Dæhlen and Tom Lyche. Decomposition of Splines. In Tom Lyche and Larry L. Schumaker, editors, *Mathematical Methods in CAGD and Image Processing*, chapter 1, pages 135 – 160. Academic Press, 1992. ISBN 0-12-460510-9.

[DLR90] Nira Dyn, David Levin, and Samuel Rippa. Data dependent triangulations for piecewise linear interpolation. *IMA Journal of Numerical Analysis*, 10(1):137–154, 1990. ISSN 0272-4979.

[DP73] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112 – 122, 1973.

[Far88] Gerald Farin. *Curves and Surfaces for Computer Aided Geometric Design.* Academic Press, Inc, 1988. ISBN 0-12-249050-9.

[FDFH90] J.D. Foley, van A. Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics - Principles and Practice.* Addison-Wesley, second edition, 1990. ISBN 0-201-12110-7.

[Flo89] Leila De Floriani. A pyramidal data structure for triangle-based surface description. *IEEE Computer Graphics and Aplications*, pages 67–78, March 1989.

[FM91] A. U. Frank and D. M. Mark. Language Issues for GIS. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 1: PRINCIPLES*, chapter 11, pages 147–166. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[Fra92] A.U. Frank. Spatial concepts, geometric data models, and geometric data structures. *Computer & Geosciences*, 18(4):409 – 417, 1992.

[Gat91] A. C. Gatrell. Concepts of Space and Geographical Data. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 1: PRINCIPLES*, chapter 9, pages 119–134. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[Goo92] M.F. Goodchild. Geographical data modeling. *Computer & Geosciences*, 18(4):401 – 408, 1992.

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

[Gup92] Stephen C. Guptill. Multiple representations of geographic entities through space and time. In *Proceedings of the 4th International Symposium on Spatial Data Handling*, volume 2, pages 859 – 868, 1992.

[HB92] Fred Holroyd and Sarah B. M. Bell. Raster GIS: models of raster encoding. *Computers & Geosciences*, 18(4):419 – 426, 1992.

[Her75] I. N. Herstein. *Topics in Algebra.* John Wiley & Sons, 2nd edition, 1975.

[Int90] International Hydrographic Organization (COE Working Group). Provisional specifications for chart content and display of ECDIS. Special Publication No. 52, International Hydrographic Bureau, Monaco, May 1990.

[JA86] Cristopher B. Jones and Ian M. Abraham. Design considerations for a scale-independent cartographic database. In *Proceedings of the 2th International Symposium on Spatial Data Handling, Seattle, WA*, 1986.

[Kot92] Cifford A. Kottman. Some Questions and Answers About Digital Geographic Exchange Standards. Technical report, Intergraph Corporation, November 1992.

[KW92] Johannes Kaasa and Geir Westgaard. Modeling of CAD Models from Measured Data. Technical report, Center of Industrial Research, Oslo, 1992. ISBN 82-411-0421-9.

[LS80] D.T. Lee and B.J. Schachter. Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer and Information Sciences*, 9(3):219–241, 1980.

[Mag91] David J. Maguire. An Overview and Definition of GIS. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 1: PRINCIPLES*, chapter 1, pages 10–11. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[Mag92] David J. Maguire. The Raster GIS Design Model - A Profile of Erdas. *Computers & Geosciences*, 18(4):463 – 470, 1992.

[McM86] Robert B. McMaster. Automated line generalization. *Cartographica*, 24(2):74 – 111, 1986.

[McM91] Robert McMaster. *Map generalization: Making rules for knowledge representation*, chapter 2 (Conseptual frameworks for geographical knowledge). Longman Scientific and Technical, 1991. ISBN 0-582-08062-2.

[McoCI73] E. Meynen (chairman of Comission II, International Cartographic Association), editor. *The Multilingual Dictionary of Technical Terms in cartography.* Franz Steiner Verlag, 1973.

[MGR91] David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors. *Geographic Information Systems, Vol. 2: Applications.* Longman / Wiley, 1991.

[MS93] Gunnar Misund and Bjørn Skjellaug. MetaMap - en modell for håndtering av stedfestet, tidsvarierende informasjon. SINTEF-SI rapport STF33 A93010, SINTEF-SI, Juni 1993. ISBN 82-595-7762-3 Preliminary version (In Norwegian).

[Mul91] J-C Muller. Generalization of Spatial Databases. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 1: PRINCIPLES*, chapter 30, pages 457–475. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[Nel91] Mark Nelson. *The Data Compression Book.* M & T Publishing, Inc., 1991. ISBN 0-13-202854-9.

[PS85] Franco T. Preparata and Michael Ian Shamos. *Computational Geometry - An Introduction.* Springer-Verlag New York Inc., 1985. ISBN 0-387-96131-3.

[Rai38] Erwin Raisz. *General Cartography.* McGraw-Hill Book Company, Inc, 1. edition, 1938.

[RBP$^+$91] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorensen. *Object-Oriented Modeling and Design.* Prentice-Hall, Inc., 1991. ISBN 0-13-630054-5.

[RGM91] David W. Rhind, Michael F. Goodchild, and David Maguire. Language Issues for GIS. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 2: APPLICATIONS*, chapter Epilogue, pages 313–327. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[RM92] Jonathan F. Raper and David J. Maguire. Design models and functionality in GIS. *Computers & Geosciences*, 18(4):387 – 394, 1992.

[RSM78] Arthur Robinson, Randall Sale, and Joel Morrison. *Elements of cartography.* John Wiley and Sons, Inc, 4. edition, 1978. ISBN 0-471-01781-7.

[Sch87] Larry L. Schumaker. Triangulation Methods. In *Topics in Multivariate Approximation*, pages 219 –231. Academic Press, Inc, 1987. ISBN 0-12-174585-6.

[sch92] The Schlumberger Data Model, version 4.0. Schlumberger Technology Corp., December 1992.

[She91] I. D. H. Shepherd. Information Integration in GIS. In David J. Maguire, Michael F. Goodchild, and David W. Rhind, editors, *Geographic Information Systems, Vol. 1: PRINCIPLES*, chapter 22, pages 337–360. Longman / Wiley, 1991. ISBN 0-582-05661-6.

[Sta88] William Stallings. *Data and Computer Communications*, chapter 11, Local Networks. Macmillian Publishing Company, second edition, 1988. ISBN 0-02-415451-2.

[Str91] Bjarne Stroustrup. *The C++ Programming Language.* Addison-Wesley, second edition, 1991. ISBN 0-201-53992-6.

[TB89] R.V Tooley and C. Bricker, editors. *Landmarks of Mapmaking.* Phaidon Press Ltd, 1989. ISBN 1-85326-936-0.

[VPF92] Military Standard, Vector Product Format. Department of Defense, USA, April 1992. MIL-STD-600006.

Appendix A

MULTIMOD - A Simple Multimodel Library

Based on the object model described in section 6.5 and the methodology and results from chapter 7, we will in this appendix carry out an implementation using the object-oriented programming language C++. For details on the syntax and semantics of C++, see e.g. [Str91]. We call the implementation MULTIMOD.

The goal is to establish a generic class library. By generic, we mean that the classes should be of a general nature, able to act as a basis for many types of Multimodels. This implies that some details has to be added when it comes to use of MULTIMOD in a specific application context.

We will only present the class definitions and their *public* operations. Data structures and operations that are *private* or *protected* will not be shown, following a common tradition in object-oriented development.

The implementation is academic in scope, we just want to suggest one of many possible directions to follow, and to demonstrate some key aspects of the Multimodels. Thus, minimal attention is paid to robustness and optimalization of the code.

The code, including the data dependent triangulation routines, is written by the author[1].

A.1 The generic library

We will first present the three main classes in MULTIMOD, and then give some details on specializations of them.

Main structure

The heart of the library is the abstract `DigMod` class, implementing the notion of a digital model, according to definition 9 in section 6.2.

[1]However, some core subroutines in the AD-approximation of linear curves are provided by Arge and Dæhlen, and the Delauney triangulation is based on a public domain Fortran version of the Cline-Renka method.

```
class DigMod
   : public AppFunc
{
public:

   DigMod(void);

   // Access of private data
   void                setAttNo        (const int);
   int                 getAttNo        (void) const;
   void                setType         (const TransType);
   TransType           getType         (void) const;
   virtual void        setAtt          (void*, const int) = 0;
   virtual void*       getAtt          (const int) const = 0;
   char*               whatAmI         (void) const;
};
```

The operations enables us to manipulate the attribute vector and the transformation that maps the attributes to a 'real world' model.

`DigMod` is a subclass of the `AppFunc`. We observe that this class initially is empty. By adding customized operations depending on the application context, `AppFunc` act as an interface between the Multimodel and the application. This will be shown later in the chapter.

```
class AppFunc
{
public:

   AppFunc(void);
};
```

The `Multimodel` is basically a collection of objects which are generalized by the `DigMod` class.

```
class Multimodel
{
public:

   Multimodel(void);

   // Index = 0,1,..,EDITIONS-1
   virtual      DigMod* reconstruct     (const int) = 0;
   virtual      void    insert          (DigMod*) = 0;
   virtual      void    update          (DigMod*, const int) = 0;

   virtual      void    dump            (void) = 0;
};
```

The class should provide all the necessary operations to maintain the ordered set of variants. In our case, we only have paid attention to operations as described in section 6.3.1. We have also include a `dump()` procedure which is supposed to reveal the internal structure of the Multimodel, especially how the variants are represented. Note that `Multimodel` is an abstract class.

Having established the main structure of MULTIMOD, as illustrated with the object model in figure 6.3 in section 6.5 we will now design some specializations of the `DigMod` class.

Specializations of `DigMod`

The subclasses proposed in this section follow the categorization of digital models as outlined in section 6.2. We design the classes with special attention to possible operations on sets of models. All following classes are abstract, and need specialization before taken in use in an application.

Class `DigMod` represents the most primitive digital model, with no operations associated. From this class, we successively derive subclasses with increasing degree of complexity and supplied operations. We start with class `PseudoDiff`, with the Δ-operator, called `sub(...)`, a copy-operator and two operations for comparing attributes in two models. These operations will be needed in the the algorithms 6.4, 6.5 and 6.6, which will be used in the Multimodel associated to the `PseudoDiff` models. We choose to use functions when implementing arithmetic operations, we think that the alternative of overloading existing operators may confuse the reader.

```
class PseudoDiff
   : public DigMod
{
public:

   PseudoDiff(void);

   virtual void        copy        (void*)           = 0;
   virtual void        sub         (void*, void*) = 0;

   virtual MmBool      equalAtt    (PseudoDiff*, const int, PseudoDiff*, const int) = 0;
   virtual MmBool      equalZero   (PseudoDiff*, const int) = 0;
};
```

The next class, `DiffMod`, will be furnished with an addition operator, and inherits all the operations in `PseudoDiff`. The `DiffMod` models are assumed to form an abelian group (see definition 12).

```
class DiffMod
   : public PseudoDiff
{
public:

   DiffMod(void);

   virtual      void    add     (void*, void*) = 0;
};
```

The class `ApproxMod` is supplied with an approximation operator, and we assume that a metric (see definition 14) is available in the specializations of this class, such that we will be able to build true multiresolution structures.

```
class ApproxMod
   : public DiffMod
{
public:

   ApproxMod(void);

   virtual      void    approx          (ApproxMod*, const double) = 0;
};
```

We end the design of model classes with **RefineMod**, which adds a refinement operator to all the procedures supplied by the superclasses. In addition, we have designed a set of utility procedures.

```
class RefineMod
   : public ApproxMod
{
public:

   RefineMod(void);

   virtual      void refine     (RefineMod*, int*, int)  = 0;

   void         setParamNo      (int);
   int*         getParamList    (void);
   int          getParamNo      (void);
   void         setParamListNo  (int, int);
};
```

In the next section we design a suit of Multimodels capable of handling the various categories of digital models.

Specializations of `Multimodel`

We start with the most simple Multimodel, the **TrivialMM**. This is, however, a useful Multimodel, allowing collections of arbitrary kinds of digital models to be handled in a homogeneous manner. We observe that the virtual operations from class **Multimodel** now is implemented, and thus it is possible to instantiate objects of the class **TrivialMM**.

```
class TrivialMM
   : public Multimodel
{
public:

   TrivialMM (void);
   TrivialMM (DigMod*);

   DigMod*      reconstruct     (const int);
   void         insert          (DigMod*);
   void         update          (DigMod*, const int);
};
```

The PseudoMM is an integration of variants of objects with the common superclass PseudoDiff. Note that the public part of the class is almost identical to the class TrivialMM. The main differences are hidden in the private parts of the class. Of this reason, we will not display the definitions of the classes DifferenceMM, SelectionMM and DecomposedMM, which all are quite similar in terms of public operations.

```
class PseudoMM
   : public Multimodel
{
public:

   PseudoMM(void);
   PseudoMM(PseudoDiff*, PseudoDiff*);

   DigMod*      reconstruct     (const int);
   void         insert          (DigMod*);
   void         update          (DigMod*, const int);

};
```

The generic part of MULTIMOD is then completed, and complies well with the object model in figure 6.4 in section 6.5. We will now take a look at a possible customization of the library for use in a primitive GI application.

A.2 Customizing MULTIMOD

The first thing to do, is to design the operations that we want all models in the application to support.

Application functionality

We assume that our application is going to handle geographic information. We want to be able to produce printed maps, i.e. planar projections, and supply the interface class AppFunc with the the virtual procedure printMap (MmPoint& max, MmPoint& min) which performs a planar projection of the model given a rectangular map window (see definition 7 of the augmented map concept). In addition, we want to be able to generalize the object (see definition 4 of cartographic generalization). The virtual operation generalize (...) performs a generalization specified as an affine transformation, and/or specified by a given tolerance (scale).

```
class AppFunc
{
public:

   AppFunc(void);

   virtual void printMap       (...) const = 0;
   virtual void generalize     (...)       = 0;
};
```

We are now ensured that every model in our system support these two operations (at least as a dummy procedure if the operation do not have any meaningful interpretation in a certain model).

In the next sections, we implement a variety of classes of objects that will be needed in our application.

Arbitrary text

In our application we want a class for the management of thematic objects of type 'arbitrary text'. It is not possible to impose any structure of these objects, which may vary in contents and length, and our only choice is to implement the class as a derivation of the plain `DigMod` class:

```
class Text
   : public DigMod
{
public:

   Text(void);
   Text(char* tt);

   Text& operator=            (const Text&);

   void printMap              (...) const;
   void generalize            (...);

};
```

We are now able to instantiate our first Multimodel, by the following code segment:

```
Text t1("This is");              // Call the Text constructor
Text t2("our first");
Text t3("MlutiModlle:");
Text t4("Hello world!");
TrivialMM tmm(&t1);              // Initialize a TrivialMM with t1

tmm.insert(&t2);                 // Insert models
tmm.insert(&t3);
tmm.insert(&t4);

tmm.reconstruct(0)->printMap(0,0,0, max, min);  // Access models and
tmm.reconstruct(1)->printMap(0,0,0, max, min);  // print to map
tmm.reconstruct(2)->printMap(0,0,0, max, min);
tmm.reconstruct(3)->printMap(0,0,0, max, min);
```

Running the code yields[2]:

```
Printing model of type "Text" to map file:    "This is"
Printing model of type "Text" to map file:    "our first"
Printing model of type "Text" to map file:    "MlutiModlle:"
Printing model of type "Text" to map file:    "Hello world!"
```

[2]The example may seem a little far out in a GIS setting, but according to the academic scope, we allow ourselves this kind of freedom.

Observing an error in the fourth variant, we correct it by an update and check the result:

```
Text correction("Multimodel:");                 // Construct new model

tmm.update(&correction, 2);                     // Update old model

tmm.reconstruct(0)->printMap(0,0,0, max, min);
tmm.reconstruct(1)->printMap(0,0,0, max, min);
tmm.reconstruct(2)->printMap(0,0,0, max, min);
tmm.reconstruct(3)->printMap(0,0,0, max, min);
```

The output of the application is as

```
Printing model of type "Text" to map file:    "This is"
Printing model of type "Text" to map file:    "our first"
Printing model of type "Text" to map file:    "Multimodel:"
Printing model of type "Text" to map file:    "Hello world!"
```

Encouraged by this minor achievement, we carry on with the slightly more advanced 'record' class.

Records

We model the 'record' type as a number of arbitrary words. This class should comply with the definition of the abstract PseudoDiff class, and we design the specialization Record. This implements the subtraction operator defined as a virtual procedure in the superclass PseudoDiff in addition to utility operations inherited from other superclasses.

```
class Record
   : public PseudoDiff
{
public:

   Record (void);
   Record (const char*);

   Record&      operator=    (const Record&);
   void         copy         (void*);
   void         sub          (void*, void*);

   void printMap             (...) const;
   void generalize           (...);

   MmBool       equalAtt     (PseudoDiff*, const int, PseudoDiff*, const int);
   MmBool       equalZero    (PseudoDiff*, const int);
   void         setAtt       (void*, const int);
   void*        getAtt       (const int) const;

};
```

We will now make a Multimodel of the four five-words records written to the files:

```
record1.dta:    "This"    "is"    "a"    "long"      "paper"
record2.dta:    "This"    "is"    "a"    "boring"    "paper"
record3.dta:    "This"    "is"    "a"    "long"      "thesis"
record4.dta:    "This"    "is"    "my"   "master"    "thesis"
```

We run a little example by constructing the records and collect them in a `PseudoMM` Multimodel. We then dump the *representations* of the variants to highlight the internal structure of the Multimodel. At last, we call the `printMap` operation for all the variants to check if the reconstruction procedure works as it should.

```
Record rec1("record1.dta");      // Call the Record constructor
Record rec2("record2.dta");
Record rec3("record3.dta");
Record rec4("record4.dta");
Record util_rec("record1.dta");

PseudoMM pmm(&rec1, &util_rec);  // Initialize a PseudoMM

pmm.insert(&rec2);               // Insert variants
pmm.insert(&rec3);
pmm.insert(&rec4);

pmm.dump();                      // Dump representations

pmm.reconstruct(0)->printMap(0,0,0, max, min); // Reconstruct and print to map
pmm.reconstruct(1)->printMap(0,0,0, max, min);
pmm.reconstruct(2)->printMap(0,0,0, max, min);
pmm.reconstruct(3)->printMap(0,0,0, max, min);
```

Executing the code, we first get a dump of the representations:

```
Dumping representation of variant nr. 1:  "This"  "is"    "a"     "long"    "paper"
Dumping representation of variant nr. 2:  "ZERO"  "ZERO"  "ZERO"  "boring"  "ZERO"
Dumping representation of variant nr. 3:  "ZERO"  "ZERO"  "ZERO"  "long"    "thesis"
Dumping representation of variant nr. 4:  "ZERO"  "ZERO"  "my"    "master"  "ZERO"
```

We observe the 'ZERO's where no change has taken place relative to the last variant. As assumed in section 6.4.2, we may store zeros or sequences of zeros more compact than an explicit representation. The models are reconstructed correctly as:

```
Printing model of type "Record" to map file:  "This"  "is"  "a"   "long"    "paper"
Printing model of type "Record" to map file:  "This"  "is"  "a"   "boring"  "paper"
Printing model of type "Record" to map file:  "This"  "is"  "a"   "long"    "thesis"
Printing model of type "Record" to map file:  "This"  "is"  "my"  "master"  "thesis"
```

In the two next sections we will look at Multimodels of piecewise linear curves.

Piecewise linear curves: DP-approximation

In section 7.2 we presented a formulation of the piecewise linear curve (PLC) as two kinds of digital models, differing in their approximation operator. The Douglas-Peucker operator gave rise to the very compact selection Multimodel. Based on the results from section 7.2, we design a class `PlDpCurve`, as a subclass of `ApproxMod`. The class implement operations derived from the superclasses, and some utility procedures. The class is aggregating an instance of the class `Data3D` to hold the points, but we find that this only includes uninteresting details, and omit the description of this class.

```
class PlDpCurve
   :  public ApproxMod
{
public:

   PlDpCurve (void);
   PlDpCurve (char* fname);
   PlDpCurve (const int pnt_no);

   PlDpCurve&   operator=  (const PlDpCurve&);
   void         copy       (void*);
   void         add        (void*, void*);
   void         sub        (void*, void*);

   void         approx     (ApproxMod*, const double);

   void         printMap   (...) const;
   void         generalize (...);

   MmBool       equalAtt   (ApproxMod*, const int, ApproxMod*, const int);
   void         setAtt     (void*, const int);
   void*        getAtt     (const int) const;
   void         setNo      (const int);

   Data3D*      getCurve   (void) const;
   void         setCurve   (Data3D*);
};
```

We make some tests with the `PlDpCurve` class to demonstrate some features of the selection Multimodel.

We use the sine-like curve we studied in section 5.1.3, represented by eleven points as illustrated in figure A.1. The curve is approximated according to a set of three tolerances. We then dump the representation, i.e. the initial explicit represented curve and the book-keeping β-vector. The curves are then printed in a map format.

```
PlDpCurve dp_crv  ("f_11");                     // Initialize original
PlDpCurve util_crv("f_11");                     // and utility curve

double tol[3];                                  // Set tolerances
tol[0] = 1.0;
tol[1] = 0.25;
tol[2] = 0.05;

SelectionMM smm(&dp_crv, &util_crv, tol, 3);    // Construct Multimodel,
                                                // including approximants.
smm.dump();                                     // Dump representation

smm.reconstruct(0)->printMap(0,0,0, max, min); // Reconstruct and
smm.reconstruct(1)->printMap(0,0,0, max, min); // print to map
smm.reconstruct(2)->printMap(0,0,0, max, min); // all variants
smm.reconstruct(3)->printMap(0,0,0, max, min);
```

This code generates the following output (in addition to map files containing the curve-data:

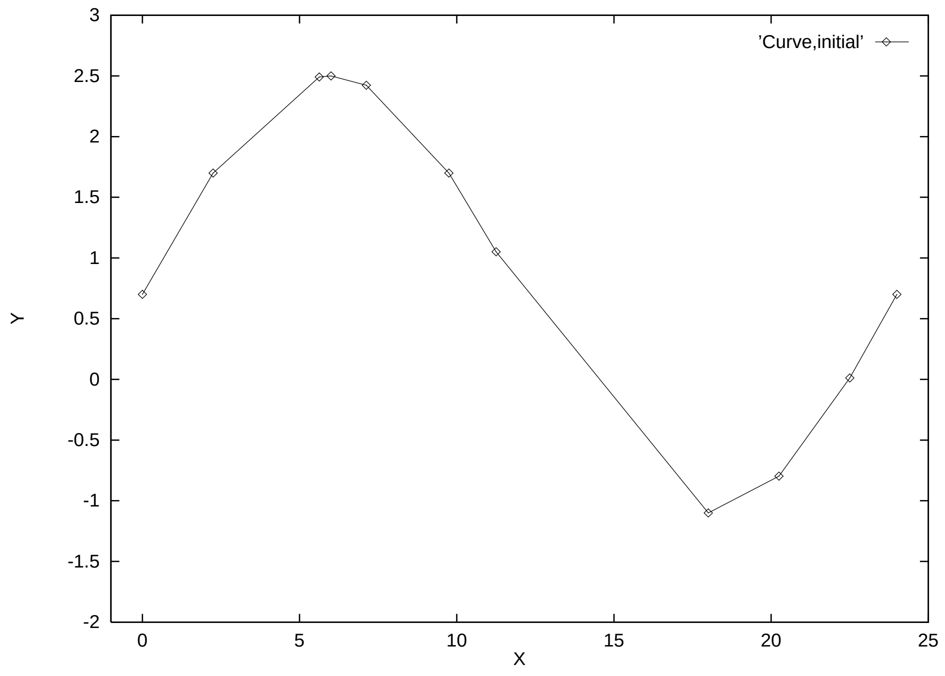


Figure A.1: Initial PLC curve

```
Running Douglas Peucker approximation scheme.........
        PlDpCurve :: approx: DP stat = 0, old_no = 11, new_no = 4
Running Douglas Peucker approximation scheme.........
        PlDpCurve :: approx: DP stat = 0, old_no = 11, new_no = 7
Running Douglas Peucker approximation scheme.........
        PlDpCurve :: approx: DP stat = 0, old_no = 11, new_no = 9

Dumping representations of variants:
Attribute vector of initial model,  11 attributes:
(0.0 0.7)  (2.2 1.7)   (5.6 2.5)   (6.0 2.5)  (7.1 2.4)  (9.8 1.7) .../
(11.2 1.1) (18.0 -1.1) (20.2 -0.8) (22.5 0.0) (24.0 0.7)
Beta vector:
3          2           0           3          1          2         .../
1          3           2           0          3

Printing PlDpCurve to map file...
        Printing model of type "Data3D" of 4 points to map file.
Printing PlDpCurve to map file...
        Printing model of type "Data3D" of 7 points to map file.
Printing PlDpCurve to map file...
        Printing model of type "Data3D" of 9 points to map file.
Printing PlDpCurve to map file...
        Printing model of type "Data3D" of 11 points to map file.
```

We see that the approximations yields curves of 4, 7, and 9 of the original 11 points. By checking the β-vector with original attribute vector and the approximants showed in A.2 we see that this representation is correct.

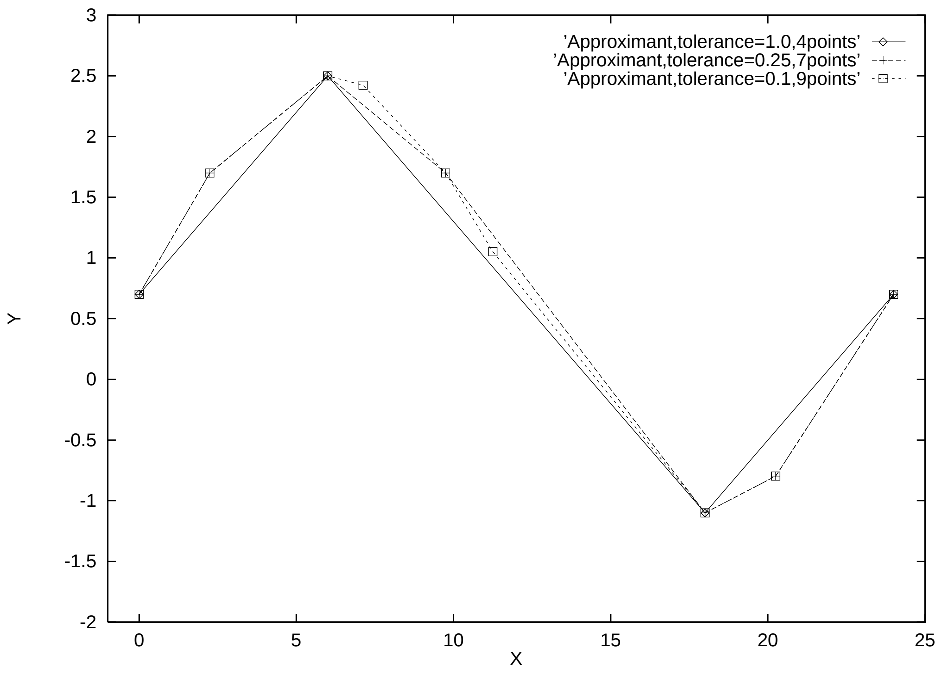


Figure A.2: Approximants of initial PLC

Piecewise linear curves: AD-approximation

In section 7.2 we designed a decomposed Multimodel of PLCs based on another approximation operator, the Arge-Dæhlen algorithm. Basically, this method allows minor perturbations of the points in order to optimize the data reduction performance. The class `PlAdCurve` is derived from the `RefineMod` class, which basically add a refinement operator to the superclass `ApproxMod`.

```
class PlAdCurve
   :  public RefineMod
{
public:

   PlAdCurve (void);
   PlAdCurve (char* fname);
   PlAdCurve (const int pnt_no);

   PlAdCurve&   operator=  (const PlAdCurve&);
   void         copy       (void*);
   void         add        (void*, void*);
   void         sub        (void*, void*);

   void         approx     (ApproxMod*, const double);
   void         refine     (RefineMod*, int*, int);

   void         printMap   (...) const;
   void         generalize (...);

   MmBool       equalAtt   (ApproxMod*, const int, ApproxMod*, const int);
   void         setAtt     (void*, const int);
   void*        getAtt     (const int) const;
   void         setNo      (const int);

   Data3D*      getCurve   (void) const;
   void         setCurve   (Data3D*);
};
```

We then integrate a set of `PlAdCurve`s in a `DecomposedMM` class in order to study some aspects of this Multimodel. We will not display any code for these examples, since it is essentially the same statements as in the DP-approximation example. However, we include an output from the reconstruction of the original curve:

```
        Reconstructing variant no. 4...
        Refining PlAdCurve from 4 to 15 points...
        Adding PlAdCurves...
        Refining PlAdCurve from 15 to 31 points...
        Adding PlAdCurves...
        Refining PlAdCurve from 31 to 65 points...
        Adding PlAdCurves...

Printing PlAdCurve to map file...
        Printing model of type "Data3D" of 65 points to map file.
```

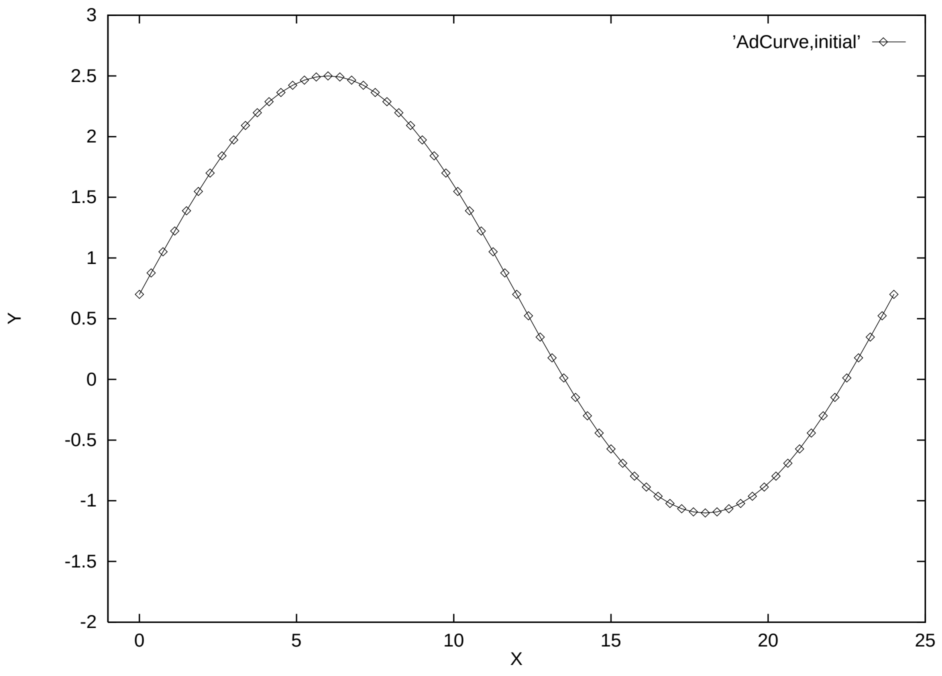


Figure A.3: Approximants of initial PLC

We observe that the procedure follows algorithm 6.12 with stepwise refinement and addition of differences.

We use again the sine-function A.2, but now represented with 65 points in the original curve, see figure A.2. We use the tolerances $0.5, 0.05$ and 0.01, which generates curves of $4, 15$ and 31 points, respectively. The set of approximants are shown in figure A.4.

To highlight the ability of AD-approximation to move points in order to increase the data reduction rate, an enlargement of the original curve an two of the approximants are shown in A.5.

In figure A.6 we have plotted the *representation* of the decomposed Multimodel. We observe the coarsest approximation as an explicit representation, while the three other variants are given as successively differences, thus clustering around origo.

To study the the difference vectors in more detail, we have made two close-ups in figure A.7 and A.8.

We observe that the 'magnitudes' of the points representing the differences are small. By inspection, we see that the majority of the points in the difference according to the original curve is in the interval $[-0.01, 0.01] \times [-0.01, 0.01]$, which is not unexpected since the tolerance of the first approximation was indeed 0.01. By the use of standard compression techniques. see e.g. [Nel91], this phenomenon may yield very compact representations, using significantly less storage than compared to an explicit representation.

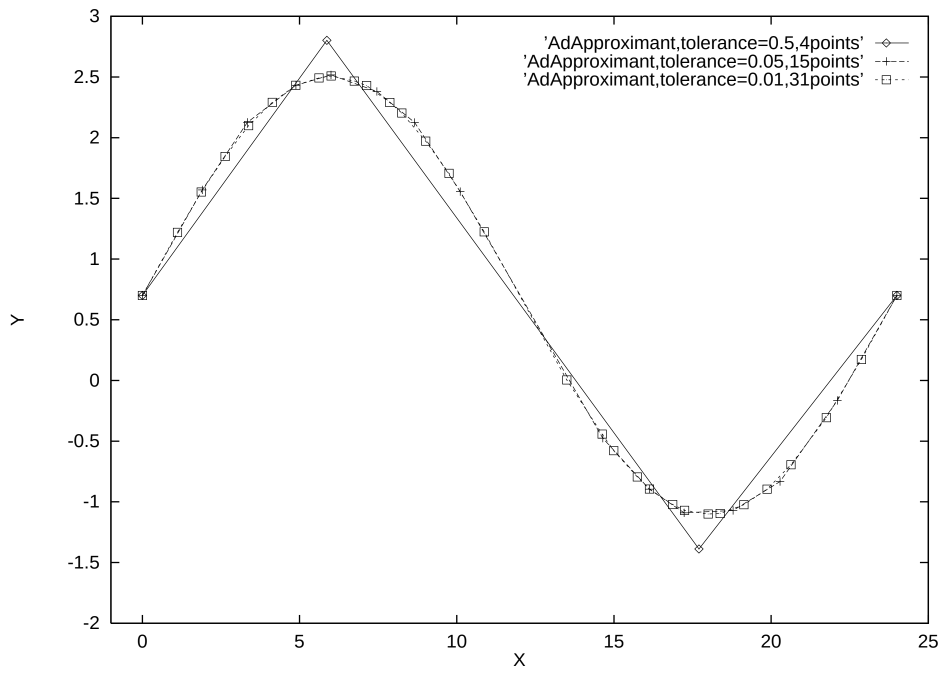


Figure A.4: Approximants of initial PLC

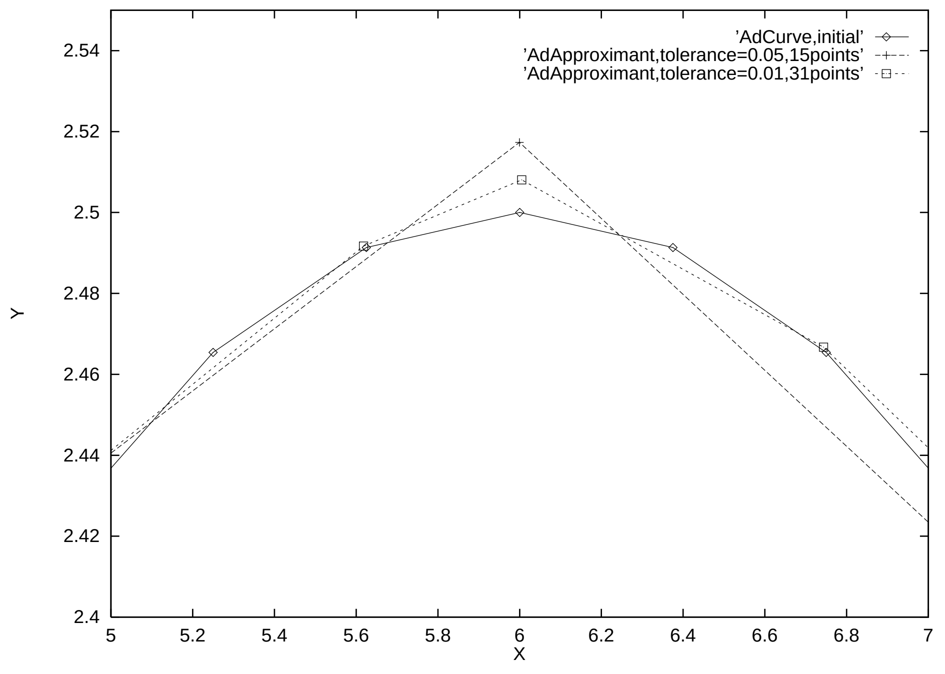


Figure A.5: Detail of AD approximation

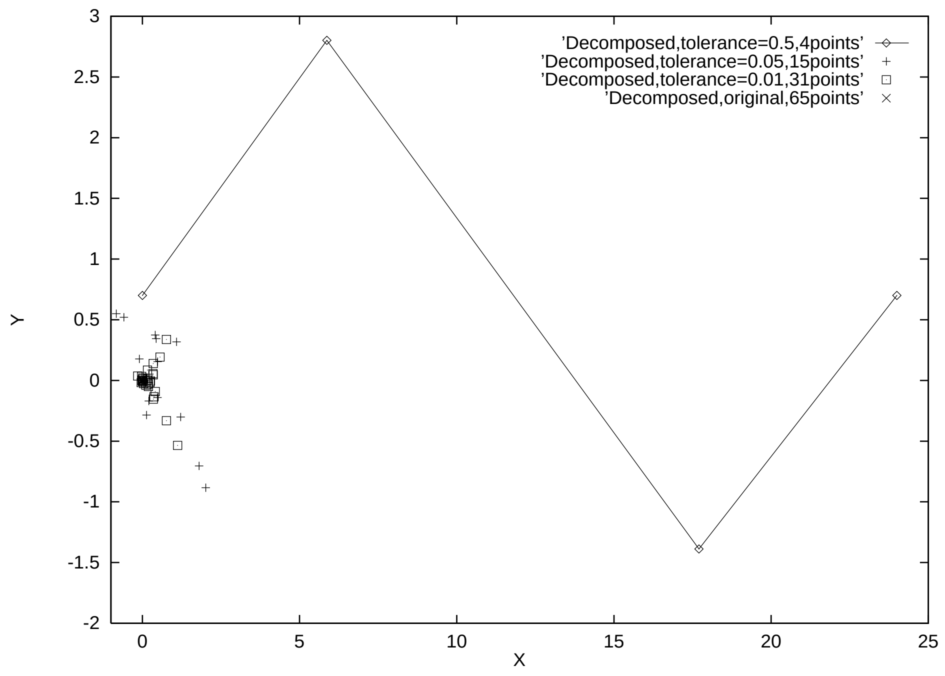


Figure A.6: Decomposed delta representation of PLC

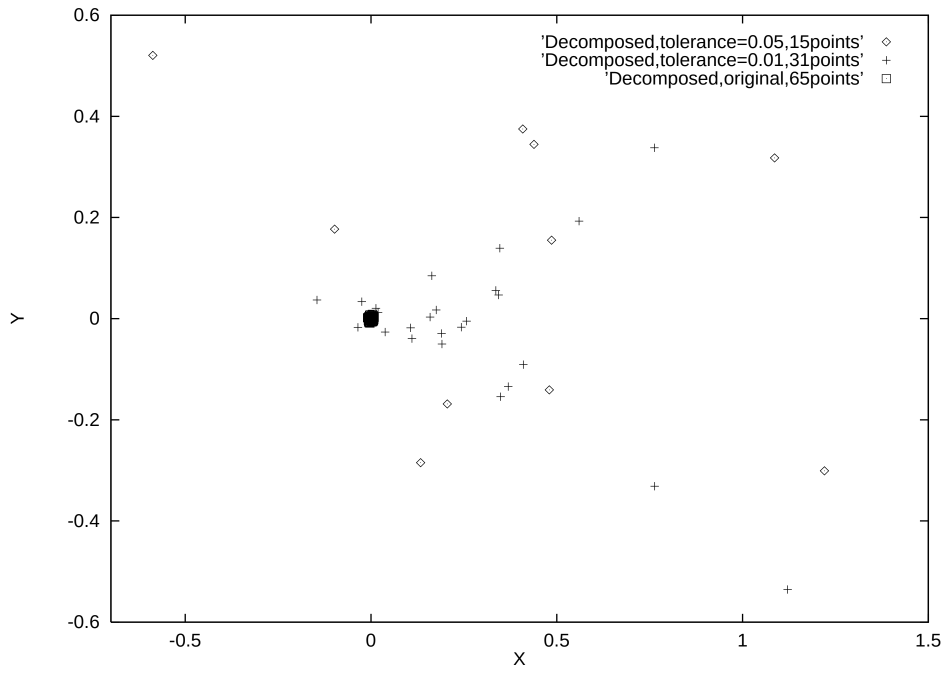


Figure A.7: Decomposed delta representation of PLC, detail I

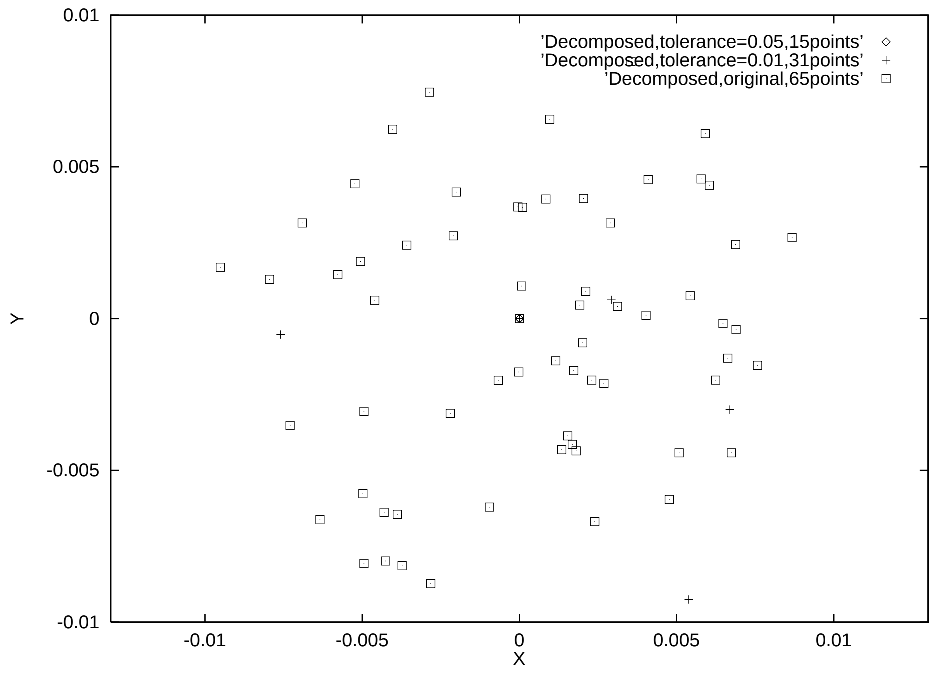


Figure A.8: Decomposed delta representation of PLC, detail II

Piecewise linear surfaces

In section 7.3, we briefly discussed piecewise linear surfaces defined over triangulations. We did not propose any implicit storage scheme for multiscale PLSs, due to certain problems associated to this task. Still, we make a trivial Multimodel representation based on the class `PlSurface`. The definition of the class is not displayed here, since it is very similar to the `PlDpCurve`. The approximation operator implements the datadependent procedure mentioned in 7.3.

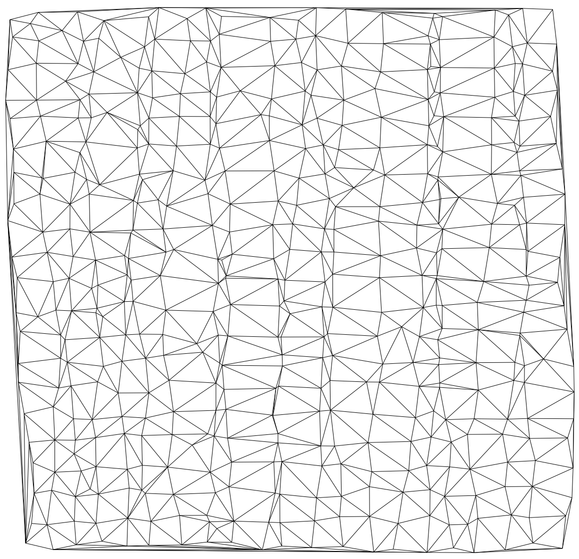


Figure A.9: Original Delauney triangulation

We make a test based on 477 point samples. The initial triangulation, shown as XY-plot in figure[3]A.9, without any data reduction, is a Delauney triangulation. The following output gives some statistics on the various triangulations:

```
Points in triangulation..............: 477
Edges in triangulation...............: 1409
Triangles in triangulation...........: 933
Tolerance............................: 0.000000
Points in original triangulation.....: 0
Reduction in percent of original.....: 0.000000
Nr of edges swapped..................: 0
LOP criterion........................: Delauney

Points in triangulation..............: 256
Edges in triangulation...............: 746
Triangles in triangulation...........: 491
Tolerance............................: 0.050000
Points in original triangulation.....: 477
Reduction in percent of original.....: 53.668763
```

[3]The 2D and 3D surface visualizations in the thesis is generated by software written by Per Øyvind Hvidsteen, SINTEF SI.

```
Nr of edges swapped..................: 472
Nr of edges rejected to swap.........: 2359
LOP criterion........................: Angle between normals

Points in triangulation..............: 36
Edges in triangulation...............: 86
Triangles in triangulation...........: 51
Tolerance............................: 0.150000
Points in original triangulation.....: 477
Reduction in percent of original.....: 7.547170
Nr of edges swapped..................: 54
Nr of edges rejected to swap.........: 219
LOP criterion........................: Angle between normals
```

The approximated triangulations are illustrated in figures A.10 and A.11. The points omitted in the triangulations are marked as single dots. We observe, especially in A.10, the long and thin triangles that are characteristic for datadependent triangulations, in contrast to the more 'well-formed' triangles of the Delauney triangulation in A.9.

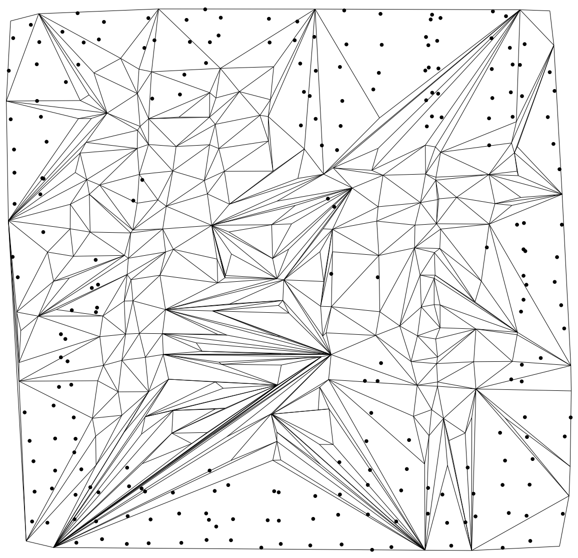


Figure A.10: Reduced triangulation with 256 points

The surfaces defined over the triangulations are shown in figures A.12, A.13 and A.14. The surfaces are rendered to visualize the triangle patches structure. We observe that the approximation with the largest tolerance, 0.15, is quite distorted compared to the original. This should not be surprising since the z-values of the surface vary between 0 and 0.2.

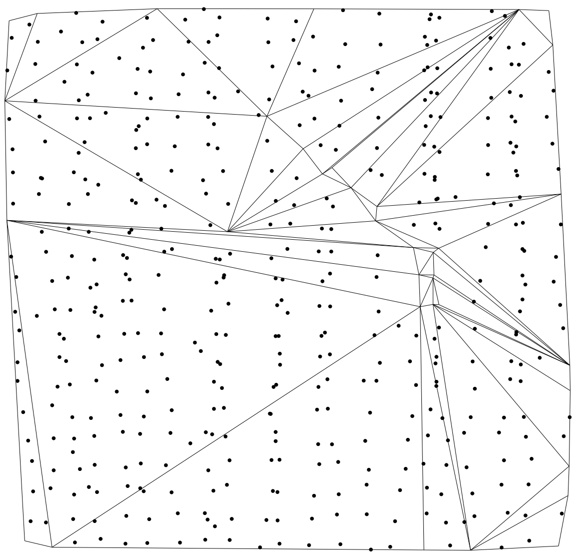


Figure A.11: Reduced triangulation with 36 points

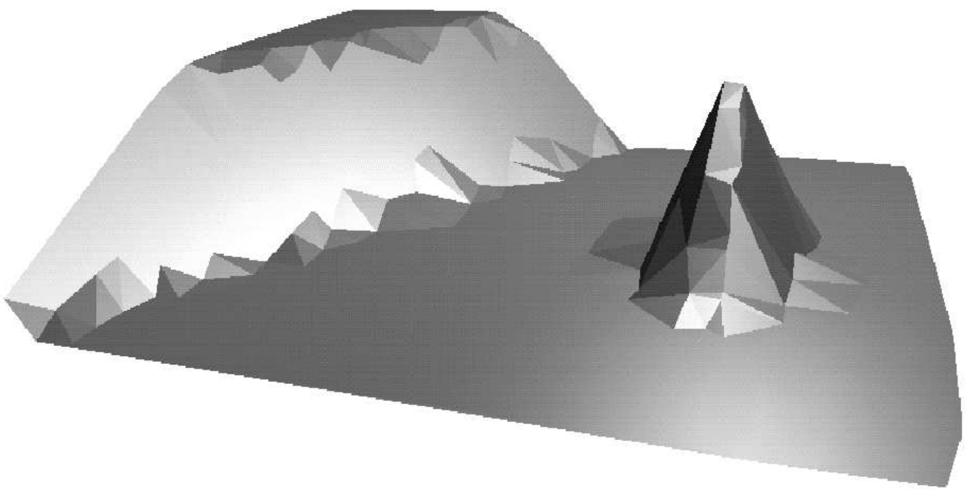


Figure A.12: Surface defined over 477 points

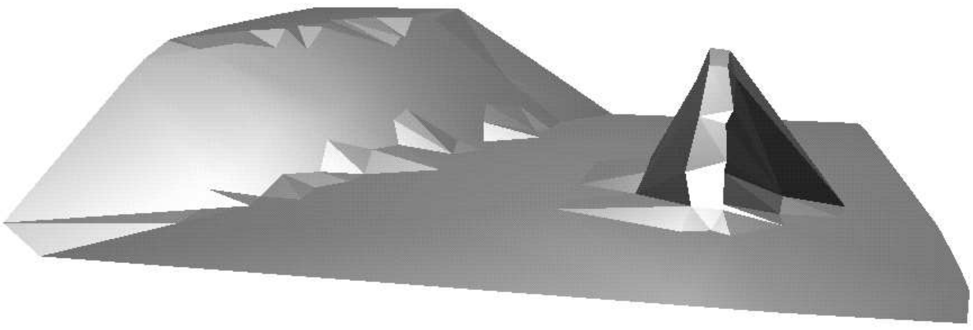


Figure A.13: Surface defined over 256 points

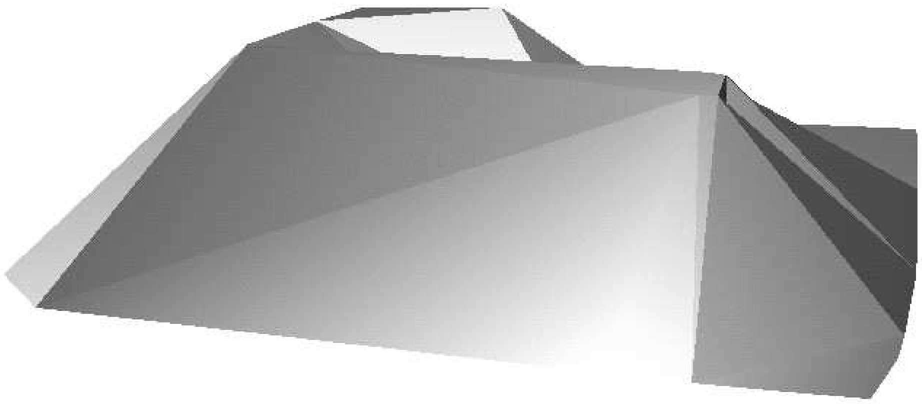


Figure A.14: Surface defined over 36 points

Chalk and cheese

To study the homogeneous aspects of the Multimodel, we consider the following examples.

```
Multimodel* divMM[2];        // Make an array of two Multimodel
                             // of arbitrary kind
divMM[0] = &tmm;             // Initialize the array with the trivial Multimodel
divMM[1] = &pmm;             // of text and the pseudo multi model of records,
                             // which were constructed in earlier examples

                             // Reconstruct and generate the fourth element in
                             // each of the Multimodels
divMM[0]->reconstruct(3)->printMap(0,0,0, max, min);
divMM[1]->reconstruct(3)->printMap(0,0,0, max, min);
```

We have two basically different Multimodels, a selection model and a trivial model, containing different types of digital models. We see that the Multimodels are handled on a superclass level, thus independent on which specific specializations the application deals with. Since the superclass `AppFunc` ensures us that the operation `printMap(...)` is implemented in all subclasses of `DigMod`, and that the procedure `reconstruct(index)` is an abstract operation in `Multimodel`, we are able to make the reconstruction/printing call without any further considerations. The code produces the following printout.

```
Printing model of type "Text" to map file:     "Hello world!"
Printing model of type "Record" to map file:   "This"  "is"  "my"  "masters"  "thesis"
```

We just experienced an example of integrating different Multimodels in a homogeneous manner. We now take a look on how different types of digital models are integrated in the very same Multimodel. To achieve this, we have to resort to the trivial Multimodel:

```
             // Initialize the chalk-and-cheese Multimodel with
             // a variant from the ‘Text’ Multimodel from the last example
TrivMM chalk_and_cheese (divMM[0]->reconstruct(3));
             // Insert a variant from the ‘Record’ Multimodel from the last example
chalk_and_cheese.insert (divMM[1]->reconstruct(3));

             // Generate and print the two variants in the Multimodel
chalk_and_cheese.reconstruct(0)->printMap(0,0,0, max, min);
chalk_and_cheese.reconstruct(1)->printMap(0,0,0, max, min);
```

We should expect the output to be identical of the last example, and to our fortune, it is:

```
Printing model of type "Text" to map file:     "Hello world!"
Printing model of type "Record" to map file:   "This"  "is"  "my"  "masters"  "thesis"
```

We summarize the elaboration of the generic Multimodel library with an object model.

Final object model

We close the MULTIMOD development with an object model of the customization of the generic library, see figure A.15[4]. A.15.

[4]Some minor deviations in the class names and operations may occur.

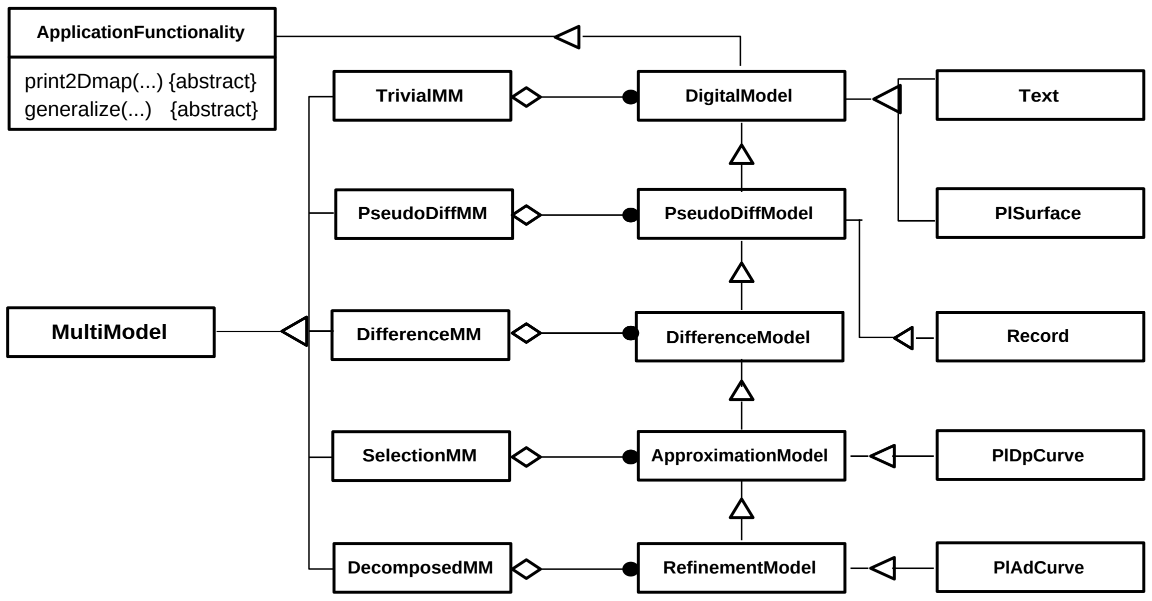


Figure A.15: Customization of MULTIMOD

Appendix B

MINIMAP - A Simple Metamap Library

In this chapter we make a limited implementation of the MINIMAP as it is described in section 10. We start with the development of a library based on the metamodel in figure 9.4. We use the library to implement the Metamap outlined in the object model given in figure 10.1. The chapter is closed with some test examples.

Like the MULTIMOD development in appendix A, the scope of the implementation is indeed academic. We just want to demonstrate some selected high level mechanisms in a Metamap, and from a GIS point of view, the examples are far from realistic.

The entire code is written by the author.

B.1 The library

Following the object model given in figure 10.1, we start the implementation with the main class, the `MINIMAP`.

```
class MINIMAP
{
public:

   MINIMAP (void);
   MINIMAP (GeographicElement*);
   MINIMAP (const char*, double, const char*);

   void                printMap        (const int,  MmPoint&, MmPoint&);
   void                printMap        (const int,  const int,
                                         MmPoint& , MmPoint&   );

   void                generalize      (const double, const double,
                                         const int, const int       );

   GeographicElement*  acessGeoElement (const int);
   void                addGeoElement   (GeographicElement*, const int);
};
```

The class aggregates a set of objects of the class `GeographicElement`. The constructors initialize the object either based on a `GeographicElement`, which is assumed to be the surface of the terrain in our map, or by a file containing the surface description and a text string representing the thematic information. We have two `printMap` operations, one printing the entire map according to a given Multimodel parameter, the other printing a selected object and all its supported objects (see section 10.2.4 for the discussion of topologies of MINIMAP). The `printMap` produces both 2D and 3D presentations. We also implement a very simple generalization operator that performs translations of spatial objects.

To maintain the the set of `GeographicElements`, we have supplied the class with the operators `accessGeoElement`, which returns a certain geographic element, and `addGeoElement`, which appends a geographic element to the Metamap, and set the support-topology according to a given parameter.

The definition of the class `GeographicElement` is given as follows:

```
class GeographicElement
{
public:

   GeographicElement (void);
   GeographicElement (TopographicElement*, ThematicElement*);
   GeographicElement (TopographicElement*, const char*);

   void                printMap        (const int, MmPoint&, MmPoint&);

   void                generalize      (const double, const double,
                                        const int                  );

   GeographicElement*  getSupported    (const int) const;
   void                setSupported    (GeographicElement*);
   int                 getSupportedNo  (void);

   TopographicElement* getTopo         (void);
   ThematicElement*    getTheme        (void);
   void                setZ            (PlSurface*, const int);
};
```

We observe that the class aggregates objects of the classes `TopographicElement`, and `ThematicElement`. The constructor takes two such objects as input, or alternatively the thematic element given as a plain text string. We find the printing and generalization operators corresponding to those given in the `MINIMAP` class. In addition, we have a set of utility procedures for maintaining the support-topology.

The two classes `TopographicElement` and `TopographicElement` are quite identically defined, and we only display the `TopographicElement`.

```
class TopographicElement
{
public:

   TopographicElement (void);
   TopographicElement (Multimodel*);

   void         printMap        (const int, MmPoint&, MmPoint&);
   DigMod*      accessVariant   (const int);
};
```

The class is constructed based on a Multimodel input, and we only have two additional operations. The print procedure generates 2D and 3D presentations, and `accessVariant` reconstructs a given variant in the Multimodel.

We use the framework to produce a few examples.

B.2 Examples

According to the academic scope of the implementation, we do not attempt to simulate a realistic GIS application. We will only focus on aspects of the Metamap. Issues concerning Multimodels will not be illustrated, such examples are given appendix A.

Initialization of the MINIMAP

We start our experiments by initializing a simple MINIMAP consisting of a surface and a single box[1].

```
        // Construct a MINIMAP based on a file with a
        // surface description, and a text string.
        // The surface is initially approximated according to
        // the given tolerance
MINIMAP map("world.dta", 0.12, "This is our world");


        // Initialize a box (‘building’)
Box box1(0.4,0.7, 0.42,0.7, 0.42,0.75 , 0.4,0.75, 0.04);
        // Make a trivial Multimodel of the box
TrivMM mm_b1(&box1);
        // Construct a topographic element based on the Multimodel
TopographicElement b1(&mm_b1);
        // Construct a geographic element based on the topographic object
        // and a text string
GeographicElement gbox1 (&b1, "box1");

        // Add the complete description of the ‘building’ to the map
        // and let the surface support it
map.addGeoElement (&gbox1, 0);
        // Print the the complete map represented by the first and
        // initial variants in the Multimodels (here: only one variant)
map.printMap(0, max, min);
```

[1]The implementation of the `Box` class is quite trivial an is not explained in any detail.

Presentation independence

By executing the code, we first get some information on the construction of the surface:

```
Points in triangulation.............: 974
Edges in triangulation..............: 2892
Triangles in triangulation..........: 1919
Tolerance...........................: 0.120000
Points in original triangulation.....: 3481
Reduction in percent of original.....: 27.980465
Nr of edges swapped.................: 3213
Nr of edges rejected to swap.........: 12892
LOP criterion.......................: Angle between normals
```

The `printMap` procedure gives the following output:

```
Starts printing MINIMAP with 2 elements.....

        Printing geo-element no 0 ...
        Printing model of type "Piecewise linear surface" to map file...
        Printing model of type "Text" to map file:   "This is our world"

        Printing geo-element no 1 ...
        Printing model of type "Box" to map file...
        Printing model of type "Text" to map file:   "box1"
```

The spatial output of the `printMap` procedure is a 2D contour plot shown in figure B.1, and the 3D rendering in figure B.2. This is an example of the presentation independence of a Metamap. Note that the 'ocean' is not explicitly modeled, but merely hard-coded as a certain z-level.

Generalization

To demonstrate the simple translation generalization, we move the box with the following statement:

```
map.generalize(-0.1, 0.05, 0, 0);
```

The call implies a translation of all spatial objects supported by the surface, geographic element no. 0. The change is performed in the initial Multimodel variant (no. 0). The result is shown in figure B.3. Observe that the box automatically derives the elevation given by the surface.

Grouping of objects

We now want to group a collection of boxes, such that a generalization of the supporting element automatically propagates to the collection of supported objects. For this purpose, we construct an 'empty' geographic element, only having the mission to aggregate a set of supported boxes.

```
        // Construct MINIMAP
MINIMAP map("world.dta", 0.12, "This is our world");

        // Construct 4 box elements
```

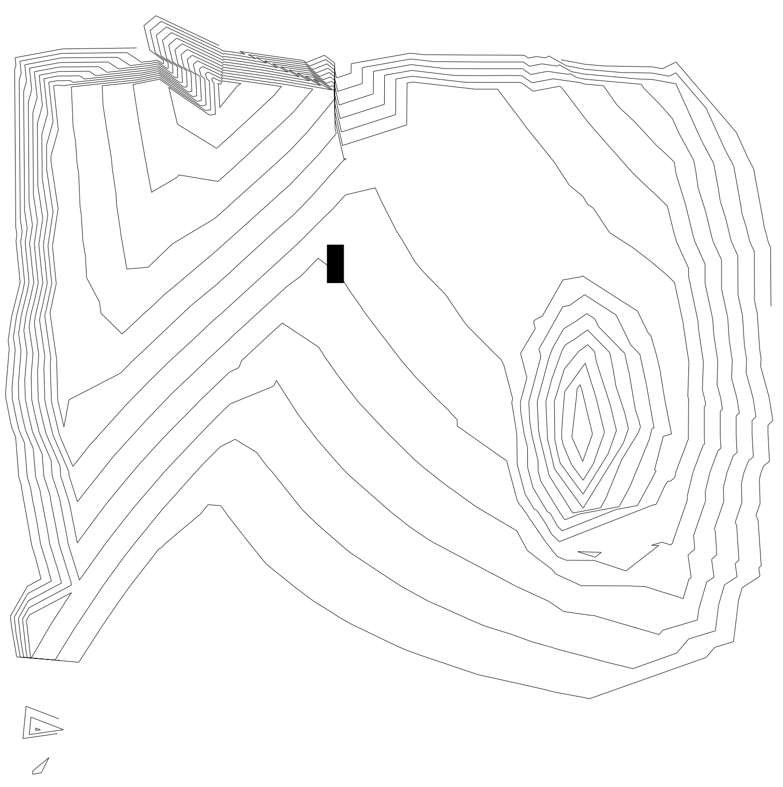


Figure B.1: Contour plot of the Metamap

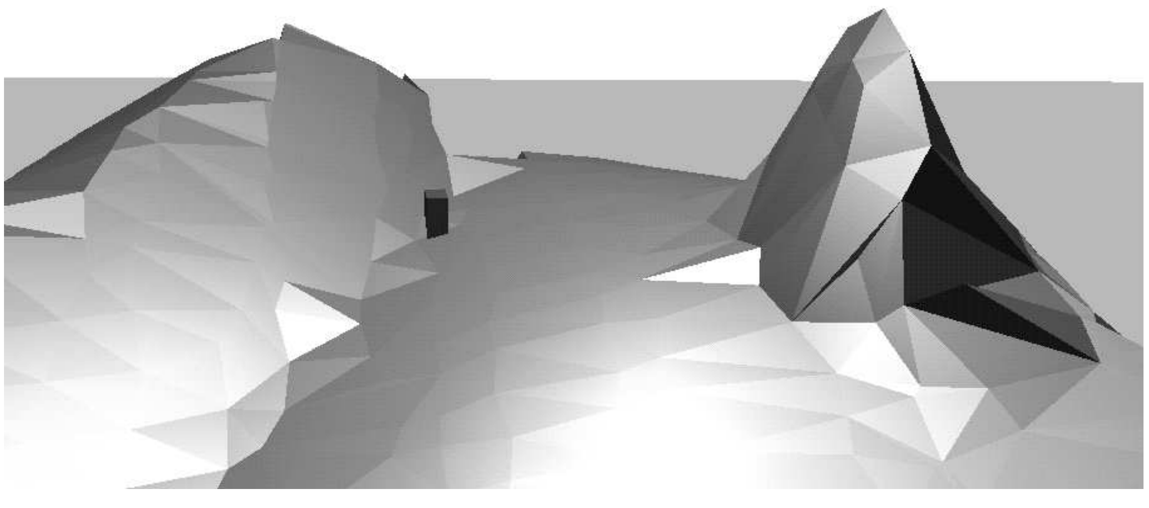


Figure B.2: 3D visualization of the Metamap

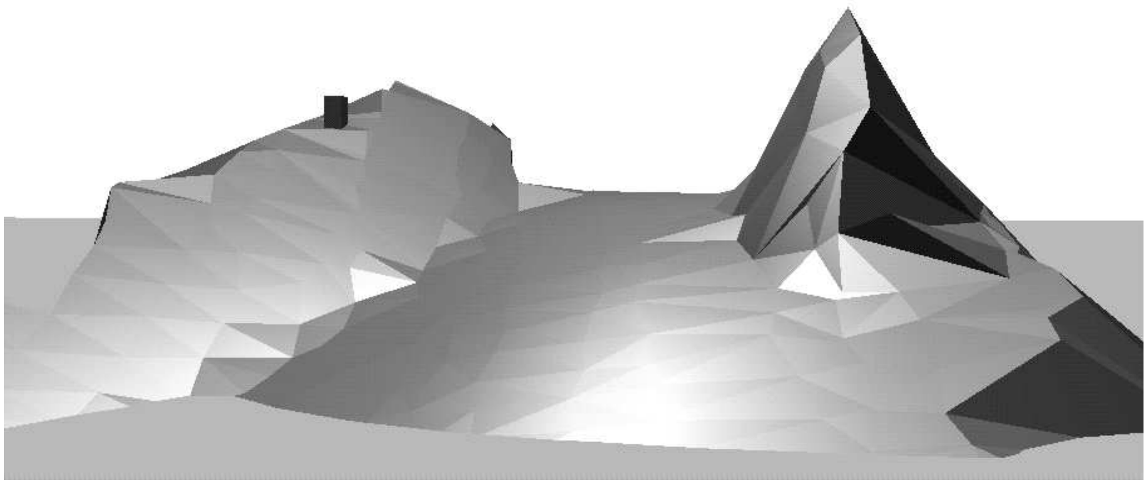


Figure B.3: Simple generalization

```
Box box1(0.4,0.7, 0.42,0.7, 0.42,0.75 , 0.4,0.75, 0.04);
TrivMM mm_b1(&box1);
TopographicElement b1(&mm_b1);
GeographicElement gbox1 (&b1, "box1");
...
GeographicElement gbox2...
...
GeographicElement gbox3...
...
GeographicElement gbox3...

        // Construct the ‘empty’ geographic element
GeographicElement empty_support;

        // Add the ‘empty’ to the surface
map.addGeoElement (&empty_support, 0);
        // Add the boxes to the supporting ‘empty’ element
map.addGeoElement (&gbox1, 1);
map.addGeoElement (&gbox2, 1);
map.addGeoElement (&gbox3, 1);
map.addGeoElement (&gbox4, 1);

        // Print the map
map.printMap(0, max, min);
```

Figure B.4 shows the new scene with four boxes.

By performing a generalization of the ‘empty’ object, which is indexed as 1, all the supported boxes should be affected:

```
map.generalize(-0.1, 0.05, 1, 0);
```

Comparing the original scene to the generalized version in figure B.5, we observe that the whole group of boxes has been translated, as it was supposed to.

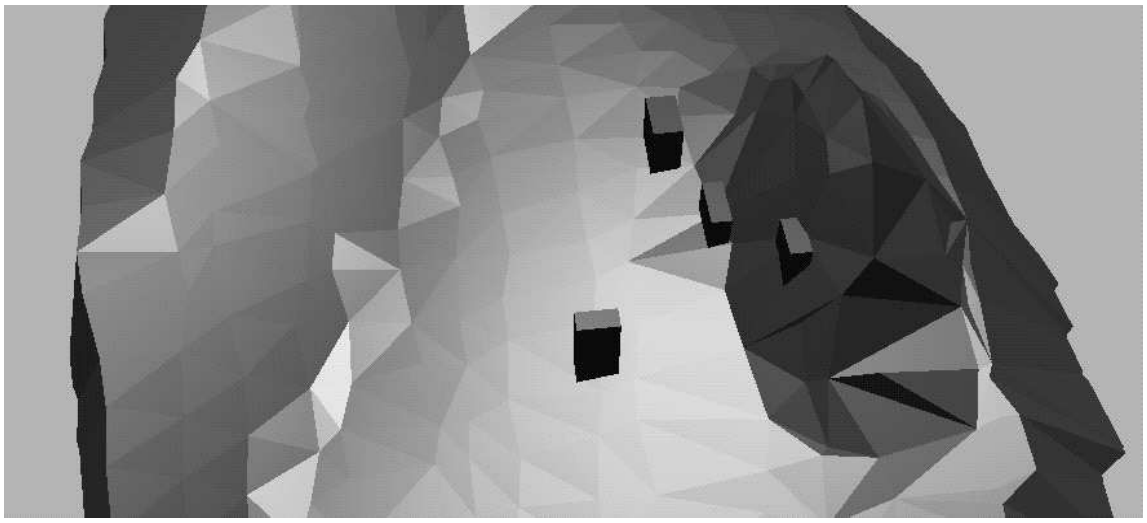


Figure B.4: Group of elements

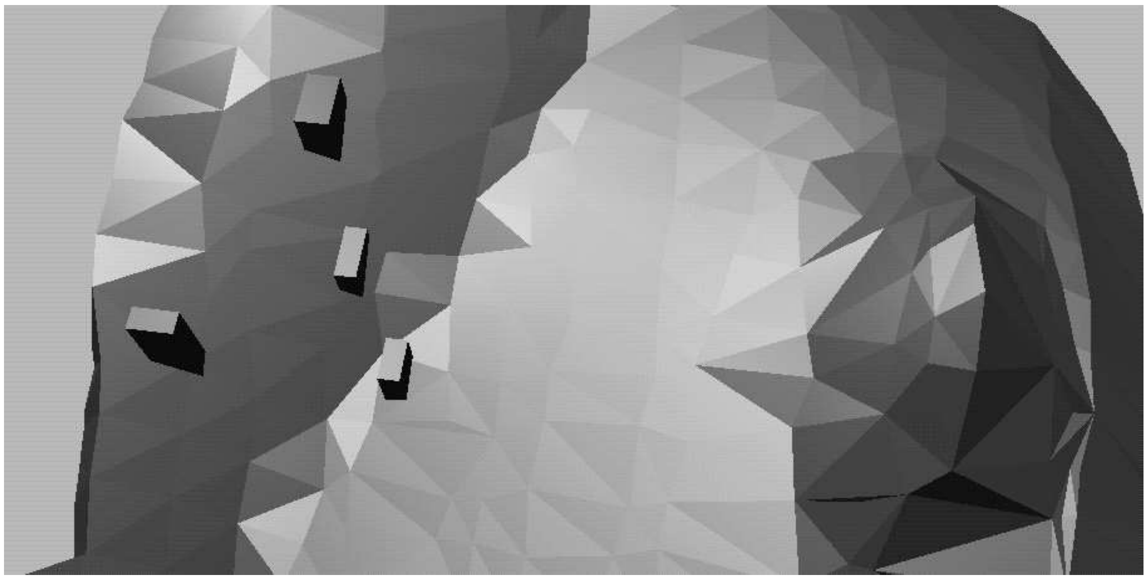


Figure B.5: Propagation of generalization