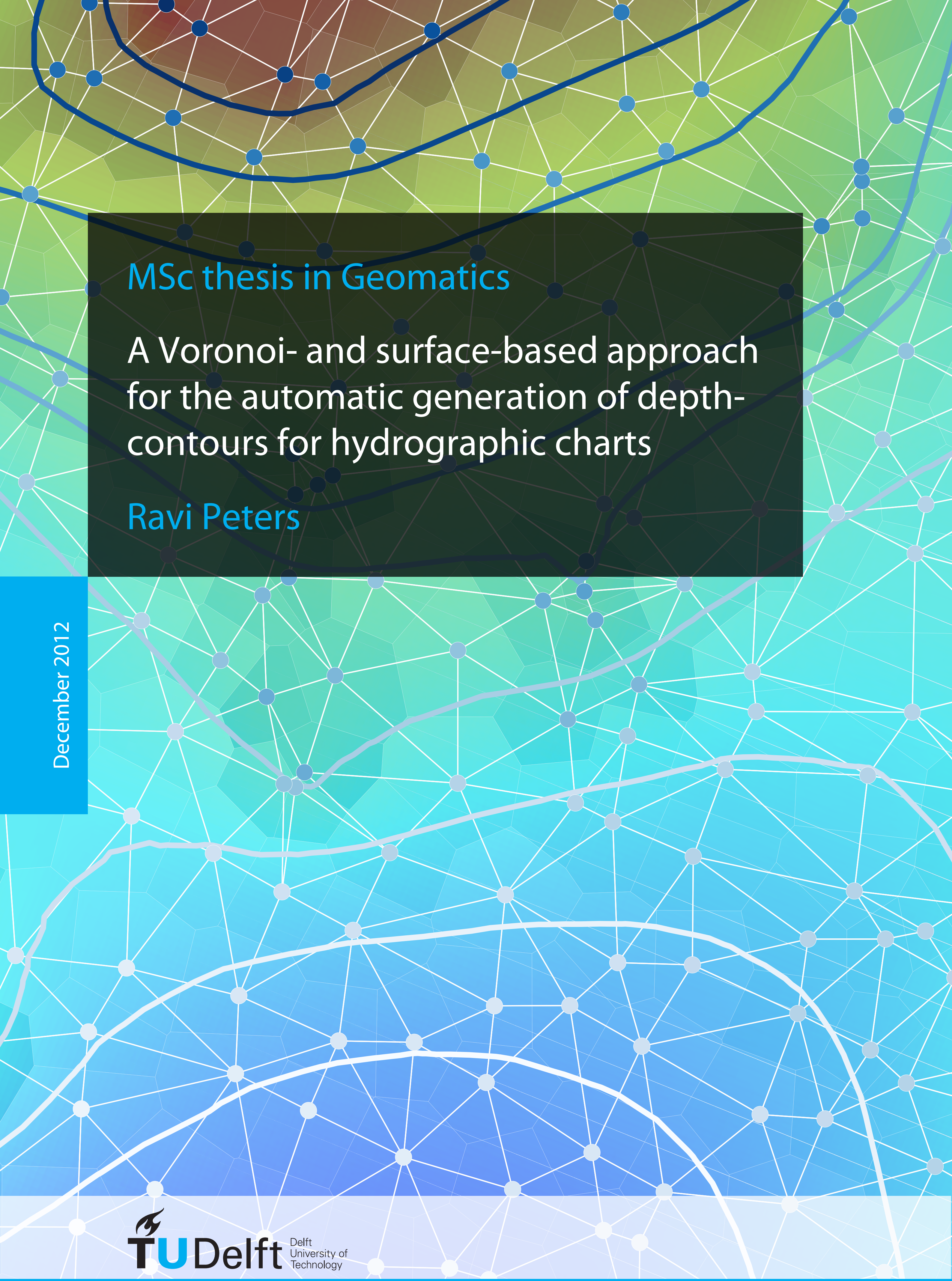


MSc thesis in Geomatics

A Voronoi- and surface-based approach for the automatic generation of depth-contours for hydrographic charts

Ravi Peters

December 2012

TU Delft Delft University of Technology

Department of GIS Technology, OTB Research Institute for the Built Environment

ON COVER:

Composite image of the different elements in the Voronoi- and surface-based approach when applied to the Australia dataset.

A Voronoi- and surface-based approach for the automatic generation of depth-contours for hydrographic charts

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of

Master of Science in Geomatics

by

Ravi Ylan Peters

December 2012

Supervising Professor:	Prof. dr. ir. Peter van Oosterom
Supervisor:	Dr. Hugo Ledoux
Co-reader:	Dr. Roderik Lindenbergh

ABSTRACT

Depth-contours are an essential part of any hydrographic chart—a map of a waterbody intended for safe ship navigation. Traditionally these were manually drawn by skilled hydrographers from a limited set of surveyed depth measurements. Nowadays this process of map making is shifted towards the digital domain, not in the last place because of the huge amounts of data resulting from modern surveying techniques. Furthermore, the task of automating the process of cartographic generalization that depends on subjective criteria is challenging. The produced depth-contours should comply with the four hydrographic generalization constraints of safety, legibility (smoothness), topology and waterbody morphology.

I show that grid-based approaches to generalize depth contours that are currently used in practice do not always comply with those fundamental generalization constraints. Most notably, the safety constraint, that ensures that a map never indicates an area as being shallower than measured, is often violated. But also the legibility and morphology constraints are not always optimally respected.

Furthermore, heterogeneous datasets (that contain a transition of very sparse to very dense data), can lead to unwished interpolation artifacts, when the popular Inverse Distance Weighting (IDW) spatial interpolation method is used. Part of this problem is the non-adaptive nature of IDW, that requires the user to re-set the interpolation parameters when the spatial distribution of the input point changes.

I present and prototype a novel surface-based approach for the generalization of hydrographic depth-contours that is based on the Voronoi Diagram (VD) and performs generalization on the *surface* that defines the contours, rather on the contour lines individually. Through the VD, a fully adaptive, automatic and smooth spatial interpolation method known as the Laplace interpolant is coupled with a Delaunay Triangulation (DT) data structure that contains all data points with their exact planimetric coordinates. Using this concept a number of operators is defined that are able to perform the relevant cartographic generalization operations for hydrographic contours: simplification, smoothing, aggregation, omission and enlargement.

The significance of the proposed approach lies herein that it honors all four hydrographic generalization constraints, most notably: it is guaranteed to be safe. As opposed to current automated approaches, it does therefore not require any form of manual safety verification. And, because all of the employed algorithms are local, it is also well scalable to big datasets in principle.

ACKNOWLEDGEMENTS

I would like to express my utmost gratitude to the people who supported me either directly or indirectly, while I was working on this thesis. First, I would like to thank my main supervisor Hugo Ledoux for his guidance and the many pleasant and constructive discussions that we had. I would like to thank Peter van Oosterom and Roderik Lindenbergh for their in-depth feedback on my writings, and Martijn Meijers for the thorough proof-reading of my thesis and his role in the final period of my graduation.

I also would like thank all the other people from the GISt section. In particular Edward Verbree and Tjeu Lemmens for their interest in my work and support with the graduation procedure and Elfriede Fendel for her administrative role. I would like to thank Ken Arroyo Ohori for his technical support and practical tips and Filip Biljecki for borrowing me his bible of typography.

Furthermore, I would like to express my gratitude to George Spoelstra for sharing his knowledge as a hydrographer and for handing me an extensive collection of datasets.

Of course I would also like to thank my office mates Lei Niu, Hua Liu, Josafat Guerrero and Radan Šuba for the friendly atmosphere in office.

Finally, I thank my parents Saskia and Levity, my brothers and the rest of my family and friends for their continued support during my student life.

Thank you all.

CONTENTS

1 INTRODUCTION 1
- 1.1 Objectives & Research questions . . . 6
- 1.2 Scope of research . . . 8
- 1.3 Thesis outline . . . 8

2 THEORY 11
- 2.1 Modeling reality . . . 11
- 2.2 Digital representations of fields . . . 12
  - 2.2.1 Rasters . . . 13
  - 2.2.2 Triangulated Irregular Networks . . . 15
- 2.3 Spatial interpolation . . . 18
  - 2.3.1 Kriging . . . 20
  - 2.3.2 Inverse Distance Weighting . . . 21
  - 2.3.3 Linear TIN interpolation . . . 22
  - 2.3.4 Natural Neighbours . . . 23
- 2.4 Contouring and contouring algorithms . . . 25
  - 2.4.1 Contouring a raster . . . 29
  - 2.4.2 Contouring a TIN . . . 30
- 2.5 Generalization . . . 30
  - 2.5.1 Hydrographic generalization constraints . . . 31
  - 2.5.2 Object based generalization operators . . . 32
  - 2.5.3 Generalization of the field . . . 34
  - 2.5.4 The generalization process as a whole . . . 36

3 CURRENT APPROACHES IN HYDROGRAPHIC CONTOURING 37
- 3.1 Point-based methods . . . 38
  - 3.1.1 Virtual gridding . . . 40
- 3.2 Raster-based methods . . . 42
  - 3.2.1 Max rasterization . . . 42
  - 3.2.2 IDW interpolation to a raster . . . 42
  - 3.2.3 Raster coarsening . . . 43
- 3.3 TIN simplification . . . 43
- 3.4 Line-based methods . . . 44
  - 3.4.1 Double buffering . . . 45
  - 3.4.2 Spline snake model . . . 46
- 3.5 Other related work . . . 46
  - 3.5.1 Multi-agent systems and Feature trees . . . 46
  - 3.5.2 The navigational surface . . . 47
- 3.6 Conclusions . . . 48

4 A VORONOI- AND SURFACE-BASED APPROACH 51
- 4.1 Overview . . . 52

4.1.1 Terms and definitions 53
4.1.2 Properties of the surface 53
4.2 TIN simplification 54
4.3 Operators on the surface 55
4.3.1 Smoothing 57
4.3.2 Reshaping 58
4.3.3 Densification 61
4.4 Automation of the surface-based approach 62
4.5 Scalability of the surface-based approach 63

5 IMPLEMENTATION AND EXPERIMENTS 65
5.1 The implemented prototype 65
5.2 Datasets 65
5.3 Metrics 66
5.3.1 The root mean square of differences 66
5.3.2 Angularity 66
5.3.3 Point differences 67
5.4 Experiments 67
5.4.1 The smoothing operator 68
5.4.2 The reshaping operator 72
5.4.3 Densification 72
5.4.4 The price of being safe 76
5.4.5 Heterogeneous data 77
5.4.6 Comparision with existing methods 80

6 CONCLUSIONS AND FUTURE WORK 85
6.1 Conclusions 85
6.1.1 Research questions 85
6.1.2 Summary of contributions 86
6.1.3 Discussion 87
6.2 Future Work 88

A A MODIFIED DROP HEURISTIC ALGORITHM 91

B SOFTWARE IMPLEMENTATION 93
B.1 Voronoi- and surface-based approach 93
B.2 Additional tools 94

C DATASETS 95
C.1 Overview 95
C.2 Antilles 97
C.3 Australia 98
C.4 London 98
C.5 Zeeland 99

D A 3D VIEW OF SMOOTHING 101

LIST OF FIGURES

Figure 1 The process of map making 1
Figure 2 Contours by Nicolaus Cruquius 3
Figure 3 Generalization in a hydrographic chart 4
Figure 4 Multi Beam Echo Sounding 5
Figure 5 IDW interpolation artifacts 6
Figure 6 Representing irregular points with a raster . . 13
Figure 7 Representing irregular points with a TIN 15
Figure 8 Empty circle criterium 17
Figure 9 Duality between the VD and the DT 17
Figure 10 Co-circularity 17
Figure 11 Dual relationships between DT and VD 18
Figure 12 Filtering high frequency detail with Kriging . 20
Figure 13 IDW interpolation 21
Figure 14 Lack of adaptiveness with IDW interpolation . 22
Figure 15 Linear TIN interpolation 23
Figure 16 VD-based interpolation 23
Figure 17 Preserving surface morphology in contours . . 26
Figure 18 Contouring a raster 27
Figure 19 Contouring configurations for a raster cell . . . 28
Figure 20 Contouring configurations for a triangle 28
Figure 21 Hydrographic generalization operatores 33
Figure 22 Safe contour generalization 34
Figure 23 Generalization using structure line models . . 35
Figure 24 Current hydrographic processing scheme . . . 37
Figure 25 Generalization by random filtering 38
Figure 26 Gridding irregular points 39
Figure 27 Schematics of gridding approaches 40
Figure 28 Safety violation in gridding approaches 41
Figure 29 Line-based contour generalization 45
Figure 30 The feature tree 47
Figure 31 Overview Voronoi- and surface-based approach 52
Figure 32 Safety violation in TIN simplification 55
Figure 33 Smoothing operator 58
Figure 34 Reshaping operator 59
Figure 35 Densification operator 61
Figure 36 Streaming computation of contours 64
Figure 37 Metrics used for analysis 67
Figure 38 RMS of differences with respect to the initial conceptual surface as a function of the number of smoothing passes 69
Figure 39 Line angularity for the contour of Figure 42 . . 69

Figure 40 The effect of the smoothing operator on the extracted contours . . . 70
Figure 41 The effect of the smoothing operator on the conceptual surface . . . 71
Figure 42 The effect of the smoothing operator on a single contour . . . 73
Figure 43 The reshaping operator performs aggregation 74
Figure 44 Surface change caused by reshaping . . . 75
Figure 45 Approximation error of TIN interpolation with respect to the initial the conceptual surface . . 76
Figure 46 Densification illustrated . . . 77
Figure 47 Comparison of TIN interpolated fields with conceptual surface . . . 78
Figure 48 The effect of incorporating the safety constraint into the Laplace interpolant . . . 79
Figure 49 Dealing with heterogeneous data . . . 81
Figure 50 Boxplots of the sample point differences with respect to the interpolated fields . . . 82
Figure 51 Comparison between hydrographic contouring approaches . . . 83
Figure 52 Multi-resolution TIN . . . 90
Figure 53 Dependency graph prototype . . . 93
Figure 54 OpenGL-based viewer application . . . 94
Figure 55 Locations of datasets . . . 95
Figure 56 Histograms of triangle aspect ratios . . . 96
Figure 57 Dataset Antillen . . . 97
Figure 58 Dataset Australia . . . 98
Figure 59 Dataset London . . . 98
Figure 60 Dataset Zeeland . . . 99
Figure 61 A 3D view of smoothing . . . 101

LIST OF TABLES

Table 1 Summary of generalization methods in the hydrographic contouring pipeline 48
Table 2 Details of datasets 96

LIST OF ALGORITHMS

1 A simple contouring algorithm 29

2 The Laplace interpolant 56
3 A safe Laplace interpolant 56
4 The smoothing operator 57
5 The reshaping operator 60
6 The densification operator 62

7 A TIN simplification algorithm 91

ACRONYMS

CGAL	Computational Geometry Algorithms Library
DCEL	Doubly Connected Edge List
DEM	Digital Elevation Model
DT	Delaunay Triangulation
ECDIS	Electronic Chart Display Information System
ENC	Electronical Navigational Chart
GDAL	Geospatial Data Abstraction Library
GEOS	Geometry Engine Open Source
GIS	Geographic Information Science
GPS	Global Positioning System
GUI	Graphical User Interface
IDW	Inverse Distance Weighting
IHO	International Hydrographic Organization
INS	Inertial Navigation System
LIDAR	LIght Detection And Ranging
MBES	Multi Beam Echo Sounding
RMS	Root Mean Square
SBES	Single Beam Echo Sounding
TIN	Triangular Irregular Network
VD	Voronoi Diagram
ZOC	Zone Of Confidence

1 INTRODUCTION

The way maps are made is rapidly changing. Traditionally, the process of transforming spatial measurements into a relevant graphical representation (a map) has involved a lot of manual work, as well as the trained eye of a cartographer. Nowadays the entire map making process has shifted towards the digital domain. This shift comes with great benefits, but it also requires a fundamental rethinking of the entire map making process (as depicted in Figure 1). In order to fully understand this we first need to ask ourselves two elementary questions: *Why do we make maps?* and *What makes a good map?* After answering those questions, I will discuss the more practical challenges in modern map making, after which I state the subject and goal of this thesis, which is in a way a fundamental rethinking of the map making process of hydrographic charts.

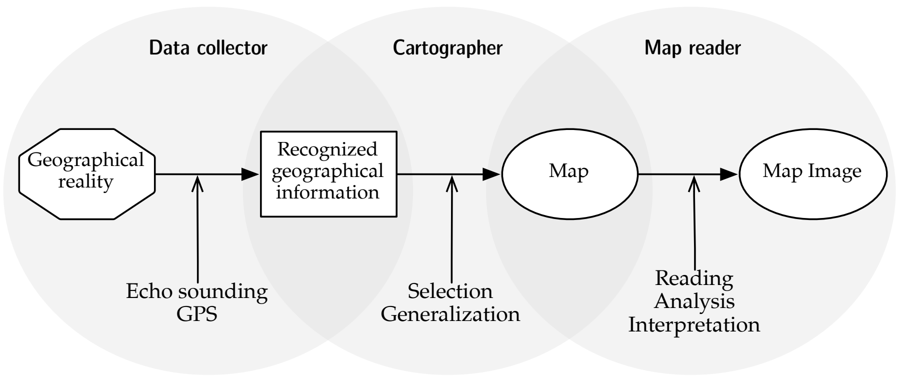


Figure 1: The different steps in the process of map making. Figure adapted from (Kimerling and Muehrcke, 2009).

Why do we make hydrographic charts?

Kimerling and Muehrcke (2009) write that the power of a map lies in its ability to visualize environmental patterns. The ultimate goal of that visualization is to optimally convey a message to the map reader in a timely and reasonable fashion. The contents of that message—and therefore the form of visualization—entirely depends on the map purpose. It is the map purpose that sets the *constraints* on the process of *cartographic abstraction*: the complete process of transforming data that have been collected about our environment into a visualization, i.e. a graphical representation, of features relevant to the purpose of a map (Kimerling and Muehrcke, 2009).

Hydrographic charts are maps of the underwater world, specifically intended for safe ship navigation[1]. The purpose of a hydrographic chart is thus to facilitate safe and efficient ship navigation, consequently the main constraints of a hydrographic chart are:

1. The *safety constraint*. A hydrographic chart is primarily a depth-map of the underwater surface. At every location, the indicated depth may never be deeper than the depth that was originally measured at that location. This is to guarantee that a ship never runs aground because of a faulty map.

2. The *legibility constraint*. A shipper should be able to quickly grasp an good impression of the waterbody morphology by looking at the map for only a short period of time. Superfluous and insignificant information for ship-navigation will only slow down the map-reading process, thus only the essential information should be present on the map in a form that is clearly and efficiently apprehensible.

3. The *topology constraint*. The topology of the depicted map elements must be correct. Contour lines for instance may not intersect.

4. The *morphology constraint*. The map should also be as realistic and accurate as possible, i.e. the overall shape of the morphology of the underwater surface should be clearly perceivable.

A map that is able to deliver this knowledge intuitively to the map reader is clearly of great value to a shipper. That is why hydrographic charts are made.

What makes a *good* hydrographic chart?

A good map is a usable map, thus a map that fulfills its purpose by respecting all of its constraints. Making a good map however, is a tedious task. It is not without reason that cartography is often referred to as an art. Successfully integrating all of the required map constraints, requires the ability to purposefully select, adapt and depict the raw and often imperfect samples of reality. Especially with small scale maps, where available space is limited and elements tend to get

1 Both paper and digital variants of such charts exist and the digital variant is named Electronical Navigational Chart (ENC), as defined in the S-57 IHO Transfer Standard for Digital Hydrographic Data. ENCs are used onboard in a so-called Electronic Chart Display Information System (ECDIS) and, as opposed to paper charts, also have the ability to dynamically set the color scale of depth areas, using current water levels and the ship dependent draft (the water depth needed to float the ship). ENCs are often integrated with other systems such as radar and Global Positioning System (GPS), that complete system can also automatically warn for possible collisions. (Wright and Bartlett, 2000)

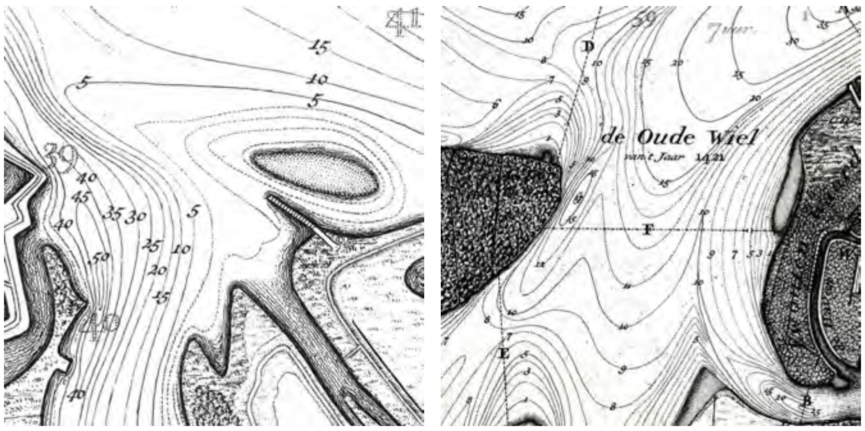
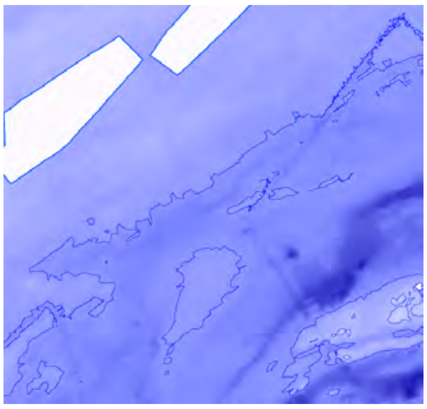


Figure 2: Details from two maps of the river Merwede in 1729 by the Delft engineer and cartographer Nicolaus Cruquius. One of the earliest uses of depth contours.

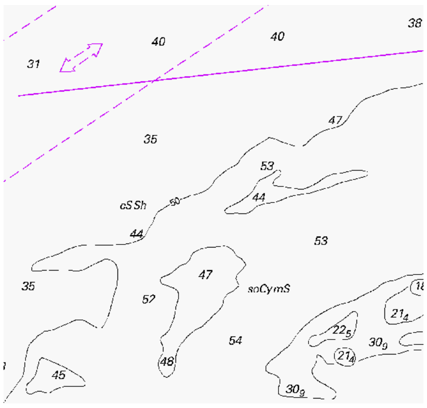
so small that they are no longer visibly perceivable, it is a challenging task to devise a map that still conveys the right message and still is pleasant to read.

The first step in process of cartographic abstraction is *cartographic selection*, it is to choose the relevant bits of information to include on a map, and thus also determining what should be left off. It is effectively reducing the abundance of available data to its sheer essence.

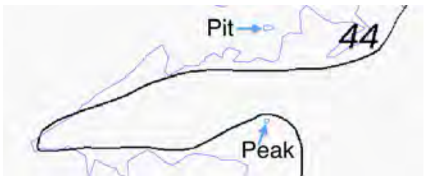
A prime example of cartographic selection on a hydrographic chart is the *depth contour* (§ 2.4). As Figure 2 demonstrates, these are smooth looking lines that connect points of equal depth and aim to give the map reader a clear impression of the seabed morphology. The fact that every sample point that is not on a contouring depth is omitted, reduces the amount of information that is depicted on a map, but in case of good contours the reader will still be able to deduct the general seabed morphology. It is even so that the use of contours will speed up the map reading process, as it conveys just that relevant bit of data to the map reader rather than 'flooding' the reader with information which essentially makes the user do his own cartographic selection. Creating good depth-contours requires *generalization* (see § 2.5), i.e. the process of meaningfully reducing detail. Ultimately the depth-contours need to look simple and smooth (Figures 2 and 3b) so that the shipper can quickly and surely consider his surroundings (the legibility constraint). This, in strong contrast to the raw and cluttered *un*generalized contours of Figure 3a. Only by deviating from original measurements one can achieve these smooth looking contours. But since hydrographic charts are intended for safe navigation (the safety constraint), this deviating can only happen towards deeper areas. In other words, while generalizing the original measurements to achieve the smooth or legible look it is only allowed to do this by artificially moving the underwater surface upwards and



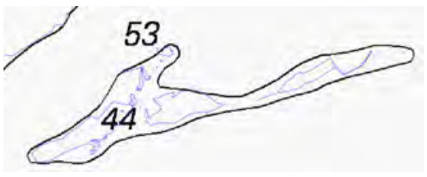
(a) TIN interpolation from raw data points and corresponding raw depth contours.



(b) Hydrographic map product of the same area.



(c) Detail. Pits are removed, while peaks are preserved or integrated.



(d) Detail. Groups of nearby contour lines are aggregated

Figure 3: Comparison of raw data and a hydrographic chart from the Royal Australian Navy of the Torres Strait north of Australia. Raw depth contours are blue, generalized depth contours are black.

never downwards. Besides, there is also the morphology constraint to consider: the overall (original) feature shapes need to be preserved, which further limits the range of surface alterations that can be made for the sake of legibility. Figures 3c and 3d illustrate some aspects of the process of generalizing depth-coutours.

In brief, the constraints for hydrographic charts are conflicting; optimally satisfying one constraint will violate another constraint. It is the art of cartography to obtain a sort of optimal equilibrium between the map constraints. I use the word art because it implies a certain subjectivity, i.e. there is no well defined set of formal metrics to assess a map's quality in all its various aspects. And, although serious efforts are made to tackle this problem by for instance Stoter et al. (2009), it remains extremely difficult to fully automate a problem that is dependent on subjective procedures.

Data

Naturally, a map cannot be made without data, i.e. samples of geographical reality (this is explained in § 2.1). In modern hydrographic

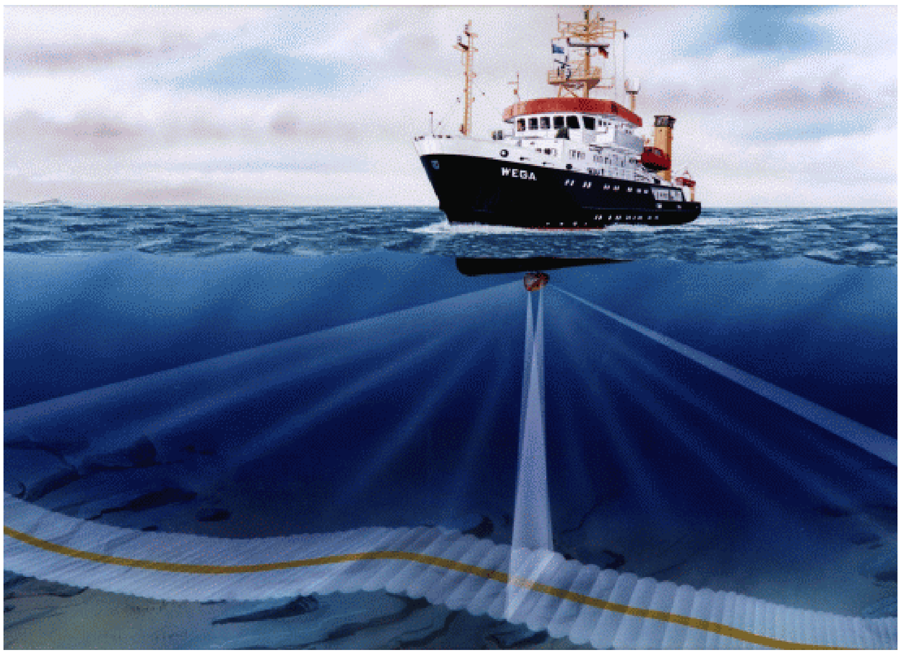


Figure 4: Multi Beam Echo Sounding (MBES) is based on the principle of measuring the time of flight—that corresponds to twice the distance to the reflecting object—of hundred of individual signal pulses. By keeping track of the angle of the transmitted signal and by correcting for ship motion using an Inertial Navigation System (INS), accurate depth soundings are obtained.

map making the two main challenges in automatically processing these data are:

1. There is a lot of data.
2. Data is imperfect.

During the past century, the acquisition-rate, density and accuracy of the underwater surface measurements have been continuously increasing. Long ago these measurements (referred to as depth soundings) were collected by sinking a leadline from a ship. Resulting soundings were sparse and inaccurate. A more modern technique is Single Beam Echo Sounding (SBES) (sonar). This is using single beams of sound to measure depth more quickly and more accurately. Today the most widely used system is Multi Beam Echo Sounding (MBES) (see Figure 4), which swaps the underwater surface with hundreds of narrow beams. With the use of a Global Positioning System (GPS) and by correcting for the ship's motion this results in accurate and very dense depth soundings. The resulting massive and dense point clouds arguably lead to higher quality products, but these point clouds have also proven to be challenging to process efficiently, since they do not fit in a computers' internal memory.

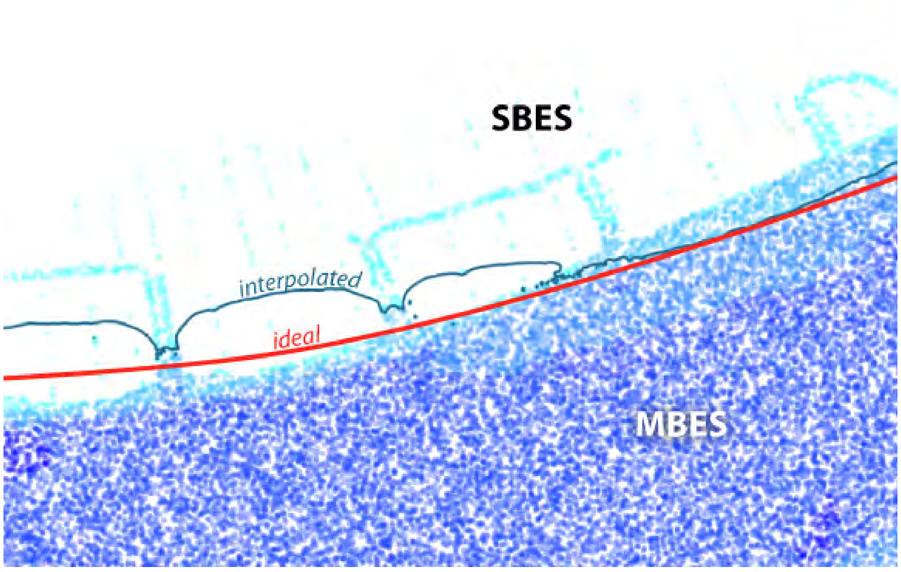


Figure 5: This dataset of the Thames (thinned to 10%) is composed of both SBES and MBES surveys. The dents in the Inverse Distance Weighting (IDW) interpolated depth contours are caused by the varying point densities. Darker blue points indicate a deeper area.

Datasets also tend to have flaws. In the case of using bathymetric data to make a hydrographic chart, the main problem is the mixture of surveys with strongly varying point densities. Large areas of the sea are only sparsely surveyed and for some areas only old and extremely sparse leadline soundings are available, yet other adjacent areas are densely surveyed. I call this a heterogeneous distribution of sample points. It is at the boundaries between areas of different surveying densities that existing automated contouring approaches have problems (I elaborate on that in Chapter 3), which results in unacceptable contours (see Figure 5). In fact this was a specific problem of a former major commercial player in hydrographic software, that partly initiated this thesis work. That company was the Dutch company Atlis, which regrettably went bankrupt during the writing of this thesis[2].

1.1 OBJECTIVES & RESEARCH QUESTIONS

This thesis explores an alternative approach to perform hydrographic depth-contouring, which thus includes generalization. This approach

2 Atlis has been declared bankrupt as of April 2012 (www.gismagazine.nl/blog/laatste-nieuws/transfer-solutions-neemt-atlis-medewerkers-over)

is specifically aimed to be robust and scalable with respect to massive and heterogeneously distributed datasets, but it also respects all of the hydrographic map constraints. It builds on the preliminary work of Ledoux (2009), who proposed the original idea of using a Voronoi-based spatial interpolation method to safely generalize hydrographic contour lines. Through the implementation of a prototype software and by studying current solutions that idea is improved and extended and it is shown that it forms a viable approach for the generation of hydrographic depth-contours from real-world datasets.

Methodology and significant findings

The Voronoi- and surface-based approach that is presented in this thesis is a fundamentally different approach from current hydrographic contouring solutions. Rather than obtaining contour lines in a series of sequential operations that may imply loss of significant information with every step (see Chapter 3), it is a unified approach that intrinsically respects the hydrographic chart constraints, is completely adaptive to the spatial distribution of sample points and deals with the entire processing chain: from sample points to generalized depth contours. Furthermore the sample points are stored with their exact coordinates and the only form of discretization takes place at the very final processing step. A conceptual surface is defined that is guaranteed to be smooth. Also, a number of local, thus in principle well scalable, generalization operators on that conceptual surface are defined that can perform every relevant cartographic generalization operation and respect the safety constraint by definition. Chapter 4 discusses this proposed approach in more detail.

After closely studying the complete hydrographic contouring process of current methods, it is concluded that none of the methods that are known to be used in practice are truly respecting the hydrographic safety constraint. Chapters 3 and 5 further elaborate on this.

The main research question of this thesis is defined as follows:

- Is the Voronoi- and surface-based approach a viable option for the automatic generation of depth-contours for hydrographic charts?

Additionally the following set of sub-research questions are defined:

1. What characterizes surfaces that lead to good depth contours for hydrographic charts and what is needed in terms of interpolation and generalization to achieve such a surface?
2. Are those characterizations respected in the Voronoi- and surface-based approach?
3. Does the Voronoi- and surface-based approach perform well for heterogeneously distributed input data?

4. To what extent can the Voronoi- and surface-based approach be automated?

5. Is the Voronoi- and surface-based approach well scalable to big datasets?

In order to obtain answers to all these questions 1) an extensive literature review is done, 2) a prototype software application is developed as a proof of concept and 3) this application is tested and compared to existing methods using a number of real-world datasets.

1.2 SCOPE OF RESEARCH

In order to maintain a clear research focus and to more strictly define the scope of this research, the following points are explicitly made:

1. The sample points that form the input of the methodology described in this thesis, are considered to be statistically preprocessed and reliable. By absence of any metadata on point quality in the available datasets, it is assumed that input points are error-free. Note that Arge et al. (2010) both give an overview of the different noise types and demonstrate an effective and efficient way to remove it from MBES datasets.

2. It is not considered how to deal with any temporal changes in morphology, for instance time series of point clouds.

3. The implemented prototype serves only as a proof of concept. It is not aimed to be production ready.

4. Coordinates are assumed to be Euclidean. This effectively means that all data is projected before it is used.

5. While the work done in this thesis fits in the hydrographic generalization problem as a whole, the aim of this research is not to solve the complete hydrographic map making problem. Primarily, it only deals with depth-contours, which is only a part of a hydrographic chart.

1.3 THESIS OUTLINE

The next five chapters are structured as follows:

- In Chapter 2 I introduce the reader to the relevant THEORY for this thesis. It covers the fundamentals of the digital representation of field-based spatial information and the preparation of that information for map-use using generalization.

- Chapter 3 describes and analyses CURRENT APPROACHES IN HYDROGRAPHIC CONTOURING from hydrographic practice and literature. Difficulties with those approaches are illustrated and a comparison of general characteristics is made.
- In Chapter 4 I motivate and describe A VORONOI- AND SURFACE-BASED APPROACH, the alternative method for hydrographic contouring that I propose. The chapter includes several algorithms and schematics.
- Chapter 5 continues with the IMPLEMENTATION AND EXPERIMENTS of the proposed approach. A set of objective metrics is defined that are used to quantify the effectiveness of the proposed approach with respect to its fundamental requirements (the hydrographic generalization constraints). Noteworthy aspects are highlighted and a comparison with existing methods is made.
- Chapter 6 gives a summary of the most significant CONCLUSIONS AND FUTURE WORK. The main contributions of my work are summarized and I answer the research questions.

Following are a number of appendices:

- Appendix A describes a modified drop-heuristics algorithm which is related to § 4.2.
- Appendix B gives more details on the implementation of the prototype software that was developed.
- Appendix C gives an overview of the datasets that were used.
- Appendix D illustrates the smoothing operator that is defined in § 4.3.1 with a series of 3D renderings.

2 THEORY

This chapter provides an overview of the relevant theory related to what is to come in the following chapters. In § 2.1 the difficulties in modeling the infinite complexity of the tangible world around us is discussed. Following is § 2.2 that explains how to do this, specifically using a digital model. § 2.3 is about spatial interpolation, § 2.4 elaborates on the idea of contouring and in § 2.5 the concept of generalization is introduced.

2.1 MODELING REALITY

The geographical world in which we live is a complex one. It is a world of infinite detail and continuous change, human built or otherwise. And even though it seems impossible to have a complete and accurate model of all this, we are able to measure and approximate useful properties such as the shape of the earth's surface with some detail. Such an approximation is bound to be limited, since we can only take a finite number of measurement samples with a finite amount of accuracy. And storing this approximation in a *data structure*[1] on a computer leads to even further abstractions and generalizations of the real world (Goodchild, 1992). Still, as the plethora of Geographic Information Science (GIS) applications illustrates, even such a limited approximation or *data model*[2] of a property of the earth can be of great use (Maguire et al., 1991). How can this be?

Goodchild (1992) uses the term *geographical reality* to refer to the empirically verifiable facts about the real world. He also describes a way to define the fundamental element of geographical information as the tuple $T = \langle x, y, z_1, z_2, ..., z_n \rangle$. This allows us to give every two-dimensional location (x, y) a set of n spatial variables. T could also be extended to include a third dimension and time. For this thesis however it suffices to include just one spatial variable next to the dimensions x and y:

$$T = \langle x, y, h \rangle \tag{1}$$

h being the depth at location (x, y). Because both x and y are continuous, the number of tuples is infinite. The infinite set of tuples is defined as a *field* (Molenaar, 1998). Every unique location, defined by

1 A set of guidelines for the representation of the logical organization of the data in a data base consisting of named logical units of data and the relationships between them. (Tsichritzis and Lochovsky, 1977)

2 The continuous surface implied by a data structure. (Kumler, 1994)

two *independent* variables (x and y), has one *dependent* variable (h) associated with it. An implication of this is that the field, as defined here, can not be used to represent some true 3-dimensional topographic features such as caves or overfolds. For this reason the field is said to be single-valued or $2.5D$; only two of its spatial dimensions are represented truly.

As stated earlier we can not obtain the full (infinite) set of tuples of a field, since that would take infinite time to do (not to mention that fields in reality are continuously changing in time as well). In addition it would be impossible to store all those tuples in a digital data store, which only has finite storage capacity.[3].
Luckily there is the notion of *spatial autocorrelation* that holds for many spatial variables. It is perhaps best explained by citing Tobler's First Law of Geography:

> "Everything is related to everything else, but near things are more related than distant things" (Tobler, 1970)

So if we take two spatially autocorrelated tuples T_1 and T_2 the similarity of their spatial variables, in our case only the depth h, increases as their (x, y) locations converge.

This has two major consequences. Firstly, even with a finite number of samples of a geographic phenomena it is still possible to create a good description of its field. And secondly, the local similarity in spatial variables of nearby tuples can be exploited to predict those spatial variables at locations where no sample was taken.

In other words: Tobler's First Law explains *why* we can sample and interpolate. It explains why it is valid to represent geographical reality in abstracted, generalized and discretized form. In that sense it is fair to say that Tobler's First Law lays at the foundation of Geographical Information Science (Goodchild, 2004). Yet, while Tobler's First Law justifies the use of sampling and interpolation techniques, it is still very important to apply caution in the use of such techniques. As Fisher (1997) and Goodchild (1992) explain, the nature of a geographical phenomena itself and the method of measuring are always to be considered in the subsequent process of storage and processing.

2.2 DIGITAL REPRESENTATIONS OF FIELDS

In GIS, a field is commonly represented as a piecewise tessellation of the plane (Goodchild, 1992). By dividing (or discretizing) the plane in

3 Note that if *depth* in geographical reality is a signal of limited frequency, it could be represented with a finite number of samples. That is according to the Nyquist–Shannon sampling theorem: A signal with a highest frequency f can be completely reconstructed from a discrete series with a sample rate of at least $2f$. The question then is of course what is the magnitude of f, and if $2f$ is a technically achievable sampling rate.

pieces it becomes possible to represent it in digital form. The shape and size of these pieces depends on the data model that is chosen. One possibility is to tessellate the plane into uniform and rectangular pieces. Rasters (§ 2.2.1) fall in this category. An alternative approach is to tessellate the plane into triangular pieces of variable size. The Triangular Irregular Network (TIN) (§ 2.2.2) can be classified as such.

The choices for a data model and a data structure are interwoven, and cannot always be treated independently. When choosing for a particular data model, one should also consider how it can be implemented as a data structure and what are the related implications for scalability, speed and storage efficiency. Inversely, when choosing for a particular data structure instead, consider how this might limit the actual field representation.

Note that the Voronoi- and surface-based approach proposed in this thesis is centered around the use of the TIN. However, since many of the existing contouring methods (as described in Chapter 3) are based on rasters and to motivate the preference for the TIN, a description is still included.

2.2.1 Rasters

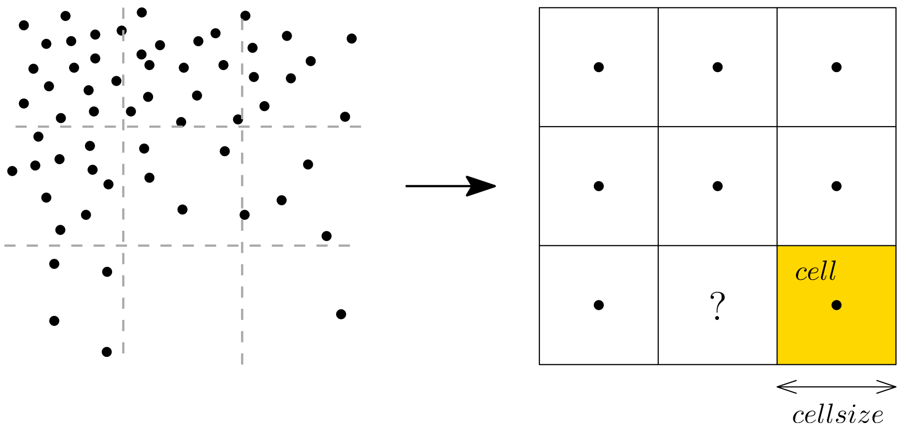


Figure 6: Representing a set of irregularly distributed points using a raster

The regular square grid or raster is essentially a two-dimensional matrix of (depth) values (see Figure 6). The exact meaning of every cell or pixel in this matrix varies in literature and is not uniquely defined (Fisher, 1997; Goodchild, 1992). First of all, it could represent a value at some systematic location within the area of the pixel. In this case one could argue that the raster structure itself does not imply a piecewise tessellation of the plane. It merely represents a set of regularly spaced points. Representative values for positions in between these points however, could be obtained by applying spatial interpolation (Goodchild, 1992; van Kreveld, 1997). Alternatively the value of a pixel could be representative for the complete area of the pixel. Possible metrics to assign this value could be the mean, median, minimum or maximum of the field data available within the area of the pixel. Because of the discrete jumps between the pixel values, the raster can not represent a truly continuous field in this case (unless the pixels are infinitely small, which is impossible).

A natural way to pick a spatial data structure for a field would be on the basis of the data acquisition method. The hardware used in many remote sensing techniques, such as multi-spectral imaging, is already physically arranged in a two-dimensional array. For storing the spatial information captured by such imaging devices a raster is therefore a very natural choice. And in this case even the exact meaning of a pixel, which is mapped directly to the fundamental (physical) sample unit, is likely to be precisely specified in the sensor's design specification (Fisher, 1997).

On the contrary, other data acquisition methods such as LIght Detection And Ranging (LIDAR) or echo sounding do not necessarily output data in a regular arrangement. When such data is represented with a raster, the pixel will not directly correspond to the fundamental sample unit. This is certainly the case for irregularly distributed samples, whose exact locations will thus be lost in the raster representation (observe Figure 6). In addition, even significant geographical features might be lost when a large cell size is chosen, while other cells might not even contain any samples at all. Variability in geographical reality can differ greatly from place to place. This is disregarded by the arbitrary rectangular and regular tessellation of space that is imposed by a raster. Similarly, a heterogeneous distribution of data samples would also be disregarded.
For these reasons it might be argued that the fixed resolution of a raster should, at least to some degree, coincide with the data samples it represents.

The regular structure that is enforced by a raster constraints its geographical applicability. However, for its implementation as a digital data structure it is anything but a constraint. Because rasters are essentially two-dimensional arrays, they are extremely straightforward to implement on a computer system. In many programming languages such as C or C++ an array is available as a basic data structure. Iterating over its elements is as simple as moving to the next memory unit. And given the proper spatial definitions, a very simple relation exists between a pixel's position in the array (the *index*) and its geographical extent. In other words: the raster data structure is a spatial index on its own. It also implies that the geographical coordinates for each pixel do not even need to be stored, which reduces storage capacity requirements.

The fact that a raster is so naturally and efficiently represented on a computer has given rise to many trivial raster-based algorithms that perform spatial analysis and manipulation (image processing). It is probably also an important reason why it has such a strong presence in GIS.

2.2.2 Triangulated Irregular Networks

A Triangular Irregular Network (TIN) is a triangular subdivision of the plane (see Figure 7). It is in many ways the counterpart of a raster. Other than with a raster, little freedom is left in the interpretation of how it relates to geographical reality. Each triangle or *face* corresponds to an area in reality. The *vertices* of a triangle are points in reality. And the boundaries (excluding the vertices) or *edges* would correspond to lines in reality. Often, points in between vertices are assigned a value by linear TIN interpolation. Furthermore, vertices do not need to lie in a particular pattern and the density may vary: a TIN is *adaptive*. This has two consequences. First of all, the geographical location of a sampled point can be *exactly* represented with a vertex. Secondly, it makes the TIN a suitable structure to represent geographic regions with varying scales of detail or any geographical point data set with a strong heterogenous distribution. A TIN implies a topology

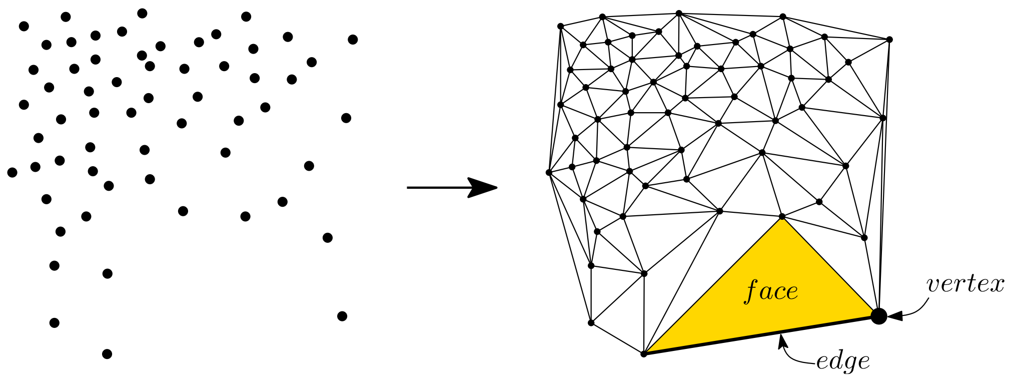


Figure 7: Representing a set of irregularly distributed points using a TIN

between nearby faces, edges and vertices. When this topology is adequately implemented, like in a Doubly Connected Edge List (DCEL) (de Berg et al., 2000), it can be used to efficiently walk through the triangulation. Obviously, a raster also has implicit neighbour relations between pixels. Depending on how it is counted, every pixel has four or eight direct neighbours. However, these are only neighbours in structural sense, and do not necessarily correspond to the morphology of the geographical surface. Contrarily, neighbour relations in a TIN could feature such morphological correspondence. Combined with the fact that a triangle is the simplest way to model an area—so it causes least degeneracies , it encourages the design of many interesting algorithms (van Kreveld, 1997). This is despite the slightly more complex implementation details compared to a raster (Kumler, 1994).

2.2.2.1 *Delaunay and Voronoi*

A crucial choice in the construction of a triangulation is *how* to connect vertices into the formation of triangles. In this regard different types of triangulations exist. An interesting one is the data-dependent triangulation in which the topology of the triangulation is chosen based on the three-dimensional surface fit through the sampled points (Garland and Heckbert, 1995; Verbree and van Oosterom, 2003). This generally leads to a low approximation error with respect to the initial triangulation. However, the most commonly used method to construct a TIN is according to the Delaunay or empty circle criterion: a triangulation is said to be Delaunay if the interior of the circumscribed circle of any triangle does not contain any vertices.

The resulting triangulation is unique, if the vertices are in *general position*, that is no three vertices are co-linear and no four vertices are co-circular[4], as depicted in Figure 10. The Delaunay Triangulation (DT) has a number of valuable properties, most notably:

ANGLE OPTIMALITY: the DT maximizes the minimum angle. As a result, triangles are as 'fat' or equilateral as possible. Skinny, elongated triangles are avoided.

LOCALITY: if triangles are locally Delaunay, the global triangulation is also Delaunay. This for instance means that the operation to delete or insert a point to an existing Delaunay triangulation is *incremental* and does not require a global re-triangulation. Because of this property, the construction and maintan ce of a Delaunay triangulation is quite efficient.

These properties have made the Delaunay TIN quite ubiquitous. In favor of the DT, in a comparison with the data-dependent triangulation Garland and Heckbert (1995) argue that natural terrains for which the data-dependent triangulation excels (in terms of approximation error) are statistically less common, after which they conjecture that data-dependent triangulation does not yield significantly higher quality approximations than DT for natural terrains in general. This statement is also supported by the studies of Rippa (1990) and Wang et al. (2001), who both conclude that the DT is in fact the best triangulation to model terrains according to their respective criteria of triangulation quality and surface roughness.

A *graph* that is closely related to the DT is the Voronoi Diagram (VD). The VD is a different (non-triangular) subdivision of the plane. It

4 If one would perform a one-to-one mapping from raster to TIN, where every grid cell is subdivided in two equally sized triangles, the result would be a triangulation that is not in general position, which is thus not uniquely defined.

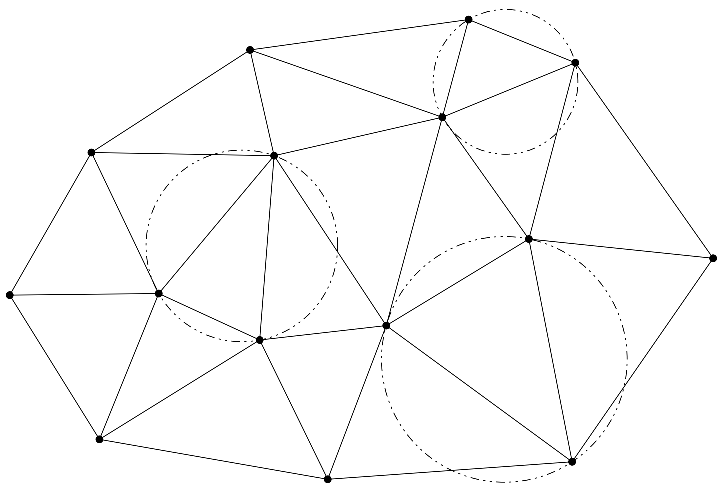


Figure 8: The empty circle criterium holds for every triangle in a DT

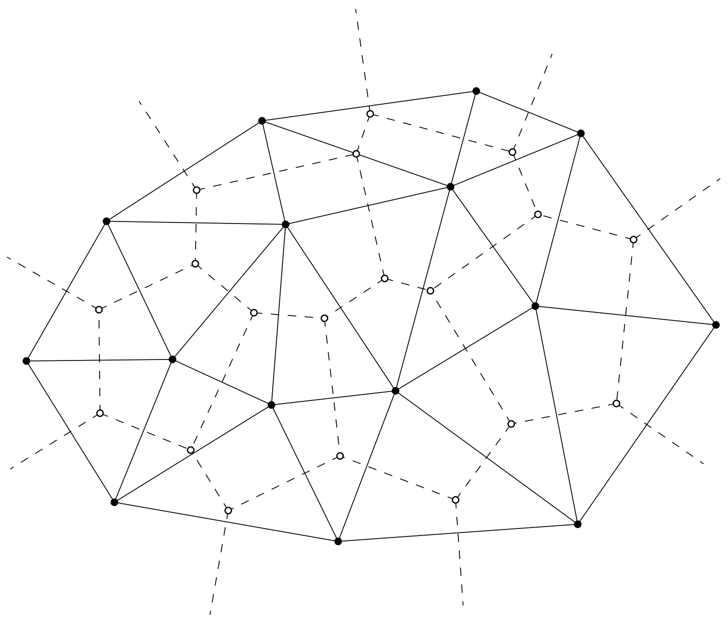


Figure 9: The DT (solid lines) is the dual of the VD (dashed)

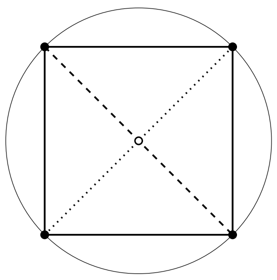
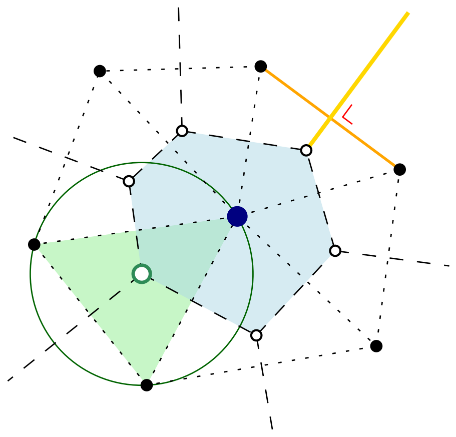


Figure 10: Co-circularity results in a non-unique triangulation; i.e. the dashed and the dotted edges both results in a triangulation of the four co-circular points that respects the Delaunay criterium.



DT		VD
face	↔	vertex
vertex	↔	face
edge	↔	edge

Figure 11: Duality between the DT (dotted) and the VD (dashed)

consists of the union of its *Voronoi cells*. Every vertex $p \in S$, the set of all vertices defined in $\mathbb{R}^2$, corresponds to one Voronoi cell. A Voronoi cell $\mathcal{V}_p$ of a vertex p is defined as the set of points x that are closer to p, than to any other vertex q. Formally:

$$\mathcal{V}_p = \{x \in \mathbb{R}^2 \mid \|x - p\| \leq \|x - q\|, \forall\, q \in S\} \tag{2}$$

So every cell is a sort of 'proximity region' around its vertex.
The VD is in fact the *dual graph* of the DT. As a result a unique one-to-one mapping of the data structure of a Voronoi diagram to the data structure of a DT exists (Ledoux, 2006). This works in both directions, meaning that you can both derive a DT from a VD as the other way around. The one-to-one mapping between the constructs of both data structure concepts is illustrated in Figure 11. A Delaunay face maps to a Voronoi vertex, while a Voronoi face maps to a Delaunay vertex. And finally, a Delaunay edge corresponds to a Voronoi edge, and is perpendicular to it. Consequently the VD also shares the properties of adaptiveness and neighbour relations with the Delaunay TIN, which lets it serve as an excellent basis for spatial interpolation methods.

2.3 SPATIAL INTERPOLATION

Many methods exist to come to a reasonable estimate of a spatial variable at locations in the field were no sample point was taken. These are generally referred to as interpolation methods. According to Watson (1992) "Interpolation, using a computer, is the performance of a numerical procedure that generates an estimate of functional dependence at a particular location, based upon knowledge of the functional dependence at some surrounding locations. It is only an informed estimate of the unknown." Clearly this indicates a heavy dependence on the notion of spatial autocorrelation. What sets the different interpolation methods apart is how they model or approximate spatial autocorrelation. Some methods, like Kriging, explicitly assume certain properties in the modeled field. Other methods less so (but implicit assumptions might be made), instead they will produce a more deterministic and local estimate. These methods need to search for nearby points, either based on a (Euclidian) distance (e.g. Inverse Distance Weighting (IDW)) or adjacency relationships (e.g. natural neighbours). Some points of interest are the computational and implementation efficiency, but of course also the quality of approximation in general.

Watson (1992) lists the properties of his ideal interpolation method, these are presented here with slight alterations (similar to Ledoux (2006)):

EXACT: the interpolation method should return the exact value, rather than some estimate, of a sample point when it is queried at that precise location. Note that an inexact interpolation method may thus violate the hydrographic safety constraint at the locations of sample points, if the interpolated depth is deeper than the original depth.

CONTINUOUS: 1) every (x, y) location in the interpolated region corresponds to exactly one value and 2) there are no discrete jumps. Such a surface is also said to be C^0.

SMOOTH: also the derivative of the interpolated surface is continuous (this is called C^1). Possibly, also higher derivatives are continuous (C^n with $n > 1$). However Sibson (1997) says C^1 is sufficient for a surface to be perceived as smooth by the human eye.

LOCAL: the method should only use a local subset of data for the interpolation of a point. This prevents the widespread propagation of dominant feature values throughout the surface. More importantly it limits computational cost and supports efficient addition or removal of new data points.

ADAPTIVE: the method performs well for varying configurations and density patterns of sample points.

AUTOMATIC: although not in Watson's original list, it is relevant here. An interpolation method is said to be automatic if it requires no manual configuration.

One could argue that there is no such thing as an ideal interpolation method, since it strongly depends on the application and the distribution and origin of sample points. However the above list is certainly applicable in the quest for the ideal interpolation method in the context of this thesis. With these requirements in mind, the following sections discuss a number of different interpolation methods.

2.3.1 Kriging

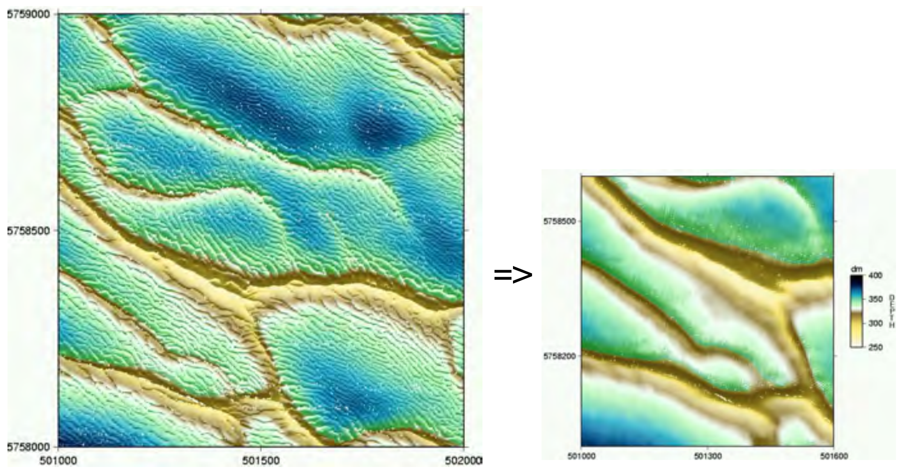


Figure 12: Kriging can be used to filter certain frequencies in the field. Note how in the right image, which is a detail of the left image, the high frequency dunes have been removed, while the general (low frequency) structure is preserved (Lindenbergh and Hooper, 2010).

With Kriging (Oliver and Webster, 1990) the spatial autocorrelation is modeled by fitting a mathematical function through the observed covariance (which is the similarity in value between sample points as a function of the distance between them). Through this function a stochastic model of the data set is obtained, which can then be used to predict values anywhere in the field. One of the nice things about this is that it can now be proven mathematically that these prediction are the *best*, in the sense that it minimizes the expected value of the squared difference between the predicted and the 'true' value (through least-squares adjustment). Kriging is also exact and continuous, although these properties are lost when it needs to correct for noise in the samples (by including a *nugget*-model), and it also gives a well defined error description for the predicted points. Kriging is a global method; a predicted point is a weighted sum of all sample points. Variants with local support do exist, but as proven by Meyer (2004) these can lead to discontinuities in the terrain model. Because of its solid theoretical foundation Kriging is a highly regarded interpolation method in geostatistics, and it has a diverse range of applications. Using Kriging, it is for example possible to filter certain frequencies in the field. This might have an interesting application in the contouring of sandbanks that typically have wavy features at different frequencies: one could simple filter out the high frequency waves that have no meaning for ship navigation, while preserving the overall structure of the sandbanks (Figure 12).

As with all interpolation methods, the quality of the predicted surface is at most as good as the quality of the sampled data. More importantly, fitting a suitable covariance function is challenging and can not be automated to satisfaction (Watson, 1992). Clearly, if the covariance is not adequately modeled, the predicted points have little meaning. In terms of computational expenses the main disadvantage is the huge matrix inversion—it includes covariances between all sample points—that needs to be performed (Lindenbergh and Hooper, 2010). This generally makes it unattractive to use in practice for big datasets.

2.3.2 Inverse Distance Weighting

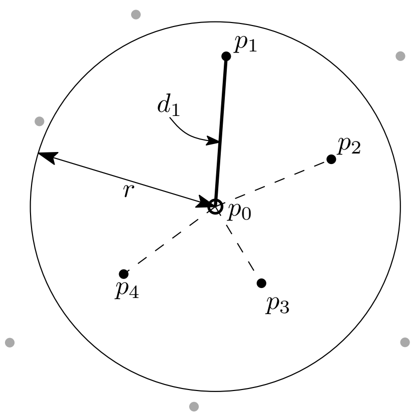


Figure 13: Inverse Distance Weighting (IDW) interpolation

An ubiquitous group of interpolation methods are the Inverse Distance Weighting (IDW)-methods. In principle IDW is based on a distance weighted sum of near points. Observe Figure 13. Let p_0 be the point of which the depth $\hat{h}_0$ needs to be predicted and $d_i = d(p_0, p_i)$ be the Euclidean distance between p_0 and p_i. The points $p_1, ..., p_n$ are the n closest around p_0 with depths $h_1, ..., h_n$. Now the predicted depth $\hat{h}_0$ equals

$$\hat{h}_0 = \frac{1}{\sum_{i=1}^{n} w_i} \cdot \sum_{i=1}^{n} w_i h_i \tag{3}$$

with the weights

$$w_i = \frac{1}{d_i^{\alpha}} \tag{4}$$

where α is configurable power, usually $\alpha \geq 1$. Higher powers would give closer points relatively more weight. Many variants to IDW exist and they mostly differ in the procedure to select the closest points. One option is to equal n to the total number of sample points, this will have a smoothing effect. But to exploit locality and subsequently speed up the interpolation process, n is usually much smaller. It may be dependent on a search radius r around p_0, in which all points or

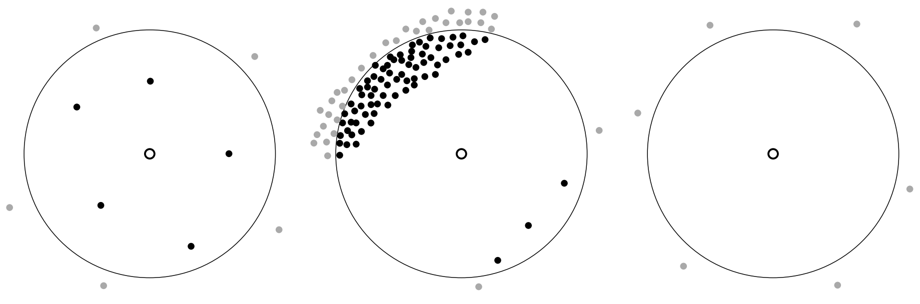


Figure 14: Overview of different point configurations and how IDW interpolation would choose its points.

just a closest subset is selected. And in order to ensure points are not taken from only one direction, the search disk may be subdivided in four quadrants, after which some number of points is selected from each quadrant. The resulting surface is often quite smooth and constrained by the sample points, but it is also discontinuous at those sample points (Watson, 1992). The effectiveness of IDW very much depends on the chosen parameters and the distribution of sampled points. For example, when a distance d_i approaches 0, the weight becomes disproportionally heavy if n is not sufficiently large. The extrema that subsequently show up in the interpolation have been dubbed the 'bull's eye' effect. In order to prevent this, n or r should somehow relate to the point density. Moreover, when the distribution of sample points is heterogeneous, a fixed value for n is ineffective. In a region with locally very little points on one side and a lot of points on the other side, a common depth-value of the dense side may be disproportionally represented in the interpolated value (Figures 14b and 49c, p. 81). Even worse would be when r is chosen such that the search disk contains no sample points at all in areas of low point density (Figure 14c). Sadly, these type of sample point configurations are not uncommon in bathymetric data sets. Thus smoothness and continuity are not guaranteed.

2.3.3 Linear TIN interpolation

Linear interpolation in a Triangular Irregular Network (TIN) is both trivial and fast. This interpolation method always uses the three points that form the triangle that contains the point p_0, whose depth $\hat{h}_0$ needs to be predicted. The value of the depth will lay on the plane that is spanned by the triangle's vertices (consider depth as the third coordinate). Observe Figure 15 The equation for linear triangle interpolation is simply

$$\hat{h}_0 = \alpha h_2 + \beta h_3 + \gamma h_1 \quad (5)$$

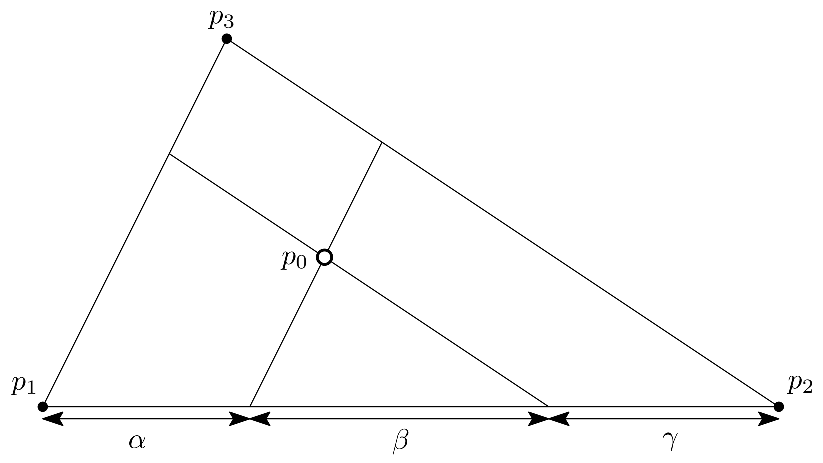
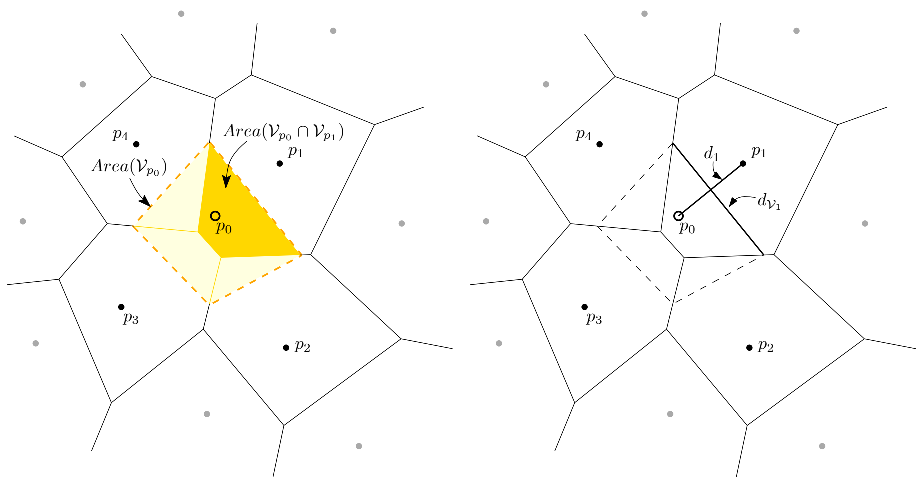


Figure 15: Linear TIN interpolation

With the scalars α, β and γ being fractions of the normalised edge $\overline{p_1 p_2}$. The continuity of the resulting surface is C^0; it is not smooth at the edges. On the plus side, linear TIN interpolation is local, adaptive, automatic and exact. It is mostly used for quick visualisation or data extraction (see for instance § 2.4), in which case a smooth (C^1) surface is not required.

2.3.4 Natural Neighbours


$Area(\mathcal{V}_{p_0} \cap \mathcal{V}_{p_1})$
$Area(\mathcal{V}_{p_0})$
p_4
p_1
p_0
p_3
p_2
d_1
$d_{\mathcal{V}_1}$

(a) Natural Neighbour interpolation **(b)** Laplace interpolation

Figure 16: Interpolation methods based on the VD. Black points indicate Natural Neighbours. The dashed region marks the Voronoi cell that would be added with the insertion of p_0 to the VD

Natural Neighbour or Sibson interpolation, proposed by Sibson (1981), works in a way that intrinsically incorporates the spatial distribution of data, it is thus highly adaptive to the sample distribution. Natural Neighbour interpolation is based on so-called *natural coordi-*

nates, that are defined through the Voronoi diagram of the sampled points. A natural coordinate is essentially a weight that is proportional to the closeness of a *natural neighbour* with respect to the local point distribution. Observe Figure 16a. The natural neighbours are the sample points in the Voronoi cells that are adjacent to the cell $\mathcal{V}_{p_0}$ that appears if the prediction point p_0 would be inserted in the Voronoi diagram. Now the weights (or natural coordinates) for each of these natural neighbours are defined as:

$$w_i = \frac{Area(\mathcal{V}_{p_0} \cap \mathcal{V}_{p_i})}{Area(\mathcal{V}_{p_0})} \tag{6}$$

Here $\mathcal{V}_i$ are the Voronoi cells of the natural neighbours with $i = 1, ..., n$ for n natural coordinates, the value of which is now strictly determined by the local spatial point distribution. The fraction indicates for each natural coordinate to what extent it is spatially related. This is measured by the amount of area overlap between $\mathcal{V}_0$ and $\mathcal{V}_i$ in relation to the total area of $\mathcal{V}_0$. Simply put: the more area in the Voronoi diagram that is 'stolen' from a particular natural neighbour, the more influence that neighbour has. Note that the natural coordinates always sum up to unity. Finally the linear combination of the weights and depth values makes up the value of the predicted depth $\hat{h}_0$ for p_0:

$$\hat{h}_0 = \sum_{i=1}^{n} w_i h_i \tag{7}$$

The resulting interpolation proves to be smooth everywhere, except at data points. But methods to work around this limitation have been described by Gold (1989); Sibson (1981); Watson (1992), for instance by using local gradient estimations. Besides, according to Sambridge et al. (1995) the lack of differentiability at the sample points is rarely a problem in practice for topographic data.

Natural neighbour interpolation is automatic, local and exact. Also the fact that it is based on the Voronoi diagram makes it an interesting option when a (Delaunay) TIN is used as the primary data structure.

On the down side the area-intersection calculation is relatively expensive in computational terms (Shokin and Afanas'ev, 2011). Luckily a computationally cheaper alternative exists. It was independently discovered by Christ et al. (1982), Belikov et al. (1997) and Hiyoshi and Sugihara (1999). This method, called non-Sibsonian interpolation or the Laplace Interpolant, is faster to compute, but yields identical properties compared to the conventional Natural Neighbour interpolation. The performance benefit is gained by replacing the area computation with distance computation. Observe Figure 16b. The weights are now defined by:

$$w_i = \frac{d_{\mathcal{V}_i}}{d_i} \tag{8}$$

where d_i denotes the Euclidean distance between p_0 and p_i, and $d_{\mathcal{V}_i}$ is the length of the Voronoi edge incident to $\mathcal{V}_0$ and $\mathcal{V}_i$. Note that this fraction becomes indeterminate when p_0 equals on of the sample points p_i. In this case the Laplace interpolant therefore simply defines that $\hat{h}_0 = h_i$. Formally:

$$\hat{h}_0 = \begin{cases} h_i & \text{if } p_0 = p_i \\ \frac{1}{\sum_{i=1}^{n} w_i} \cdot \sum_{i=1}^{n} w_i h_i & \text{otherwise} \end{cases} \tag{9}$$

Thus, the Laplace interpolant is exact by definition. In a qualitative analysis of interpolation methods for a Digital Elevation Model (DEM) Yanalak (2004) shows that Natural Neighbour and Laplace interpolation are only slightly outperformed by computationally more expensive and global interpolation methods.

2.4 CONTOURING AND CONTOURING ALGORITHMS

A contour line is an isoline, literally a 'line of equalness'. Generally speaking an isoline is a path along which some attribute of a field is constant. In bathymetry this is usually the bathymetric depth with respect to some lowest water tide. In this case the isoline is called a depth-contour or isobath. In this thesis the term contour is used to refer to the latter (and historically speaking this also seems to be how contours were originally conceived).

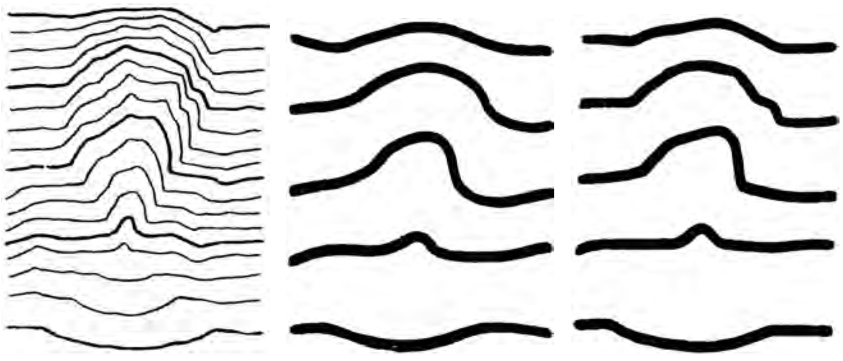
In relation to the field, as defined in § 2.1, the contour lines for a constant depth h_c are made up of the set of tuples that satisfy:

$$h = h_c \tag{10}$$

In principle, a single contouring depth h_c corresponds to a set of disconnected and—unless they are outside the data extent—closed contour lines. One particular property of contours, is that their direction is alway perpendicular to the direction of the steepest slope. Another property that follows from the 2.5D property of the field, is that contours neither intersect themselves nor each other.

The purpose of contours on a map is to reveal the shape of the underlying field. By observing the shape and interrelation of neighbouring contours, the presence and significance of surface features becomes apparent. Reading a contour map requires some skill, however as Watson (1992) points out, it is considerably easier to learn to interpret a contour map than to manually draw one from a limited set of point data. Yet this was exactly the task of many cartographers in the past couple of centuries. It was intuitively done by imagining a local triangulation of sample points (Watson, 1992). As is the case with many cartographic skills, the skill is developed through practice and experience. Consequently, the exact thought process behind the skill is difficult to capture in a fixed algorithm. And the follow-up process of evaluating the outcome of a contouring-algorithm is problematic for the same reason. On the other hand it can be argued that through the availability of more and higher quality data and the use of automated interpolation algorithms, it has become much easier to understand the morphology of the field.

In cartography a contour rarely has a one-to-one relationship with geographical reality. In fact, the choice for contours on itself, is already a form of *cartographic selection*[5]: choosing the relevant information to display on a map. By choosing to use some specific data, the other data is obviously omitted. Compare for example a continuous color map to a set of contours from the same field. Data in between contours is simply absent in the contour map. Yet, in case of good contours the reader will still be able to deduct the general field morphology. It is even so that the use of contours will speed up the map reading process, as it conveys just that relevant bit of data to the map reader rather than 'flooding' the reader with information which essentially makes the user do his own cartographic selection. Contouring is a form of discretizing the field that makes it easier to use a map. Naturally, as Goodchild (1992) points out, this comes at a price. He states that the level of approximation of the field can (dramatically) differ in between contours, the biggest error would be midway in between contour lines. But, depending on the relation between the spacing between contours (the *contour interval*) and the map scale, which in turn is dependent on the map application, this effect may be neglected.



(a) Original (b) Poorly generalized (c) Better generalized

Figure 17: Example of generalization of contour lines. Imhof opposes overly smoothed contours, as these do not adhere to the morphological nature of the sampled field (Imhof, 1965)

The relationship of a contour line with geographical reality can be distorted. Imhof (1965) argues that the contour shape, which is often

5 Cartographic selection is the first step in the process of cartographic abstraction: the process of transforming data that have been collected about our environment into a graphical representation of features and attributes relevant to the purpose of the map (Kimerling and Muehrcke, 2009).

influenced by generalization, should be true to the topographical surface that is depicted. He seems particularly against overly smoothing contour lines simply because that looks nice, see Figure 17. A (region of a) field that is not smooth from itself should thus not be represented with smooth contours, think of cliffs and rock formations for example. On the other hand, Sibson (1997) and Watson (1992) talk of smoothness as a part of the definition of a contour, on which basis they advertise the use of smooth surface models to contour from. Of course it is also dependent on the map's purpose. And Imhof (1965) himself states that in case of nautical charts, the topographic representation of the ocean floor takes second place, since everything is so much directed towards the goal of safe navigation.

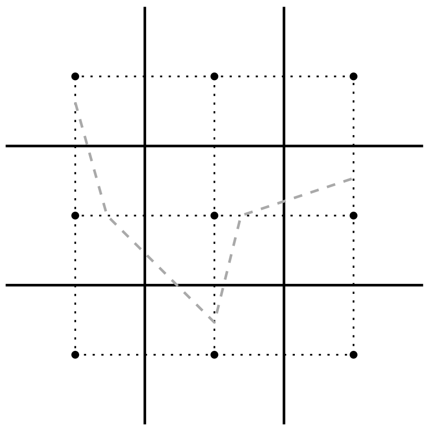


Figure 18: The 'contouring grid' (dotted lines) that is used as the basis for the linear interpolation in conventional raster contouring.

Contours are usually directly extracted from the available digital field representation. Any interpolation method that was used to construct that field representation is not relevant for the contour extraction algorithm itself, which in its basic form performs its own linear interpolation in-between the data elements of the data model (i.e. a raster or a TIN). Therefore the number and size of the line segments in the resulting contour lines are dependent on the resolution of the data representation. Consequently by refining that data representation, e.g. by performing interpolation, the resolution of the contour lines is improved.

Regardless of whether a raster or a TIN is contoured, the basic algorithm can be broken down to iteratively contouring the defining data elements, respectively the grid cell or the triangle, for one depth h_c. The algorithm GENERATECONTOURS (Algorithm 1) then boils down to recognizing the particular configuration of a data element with respect to the contouring depth, and performing the appropriate contouring action.

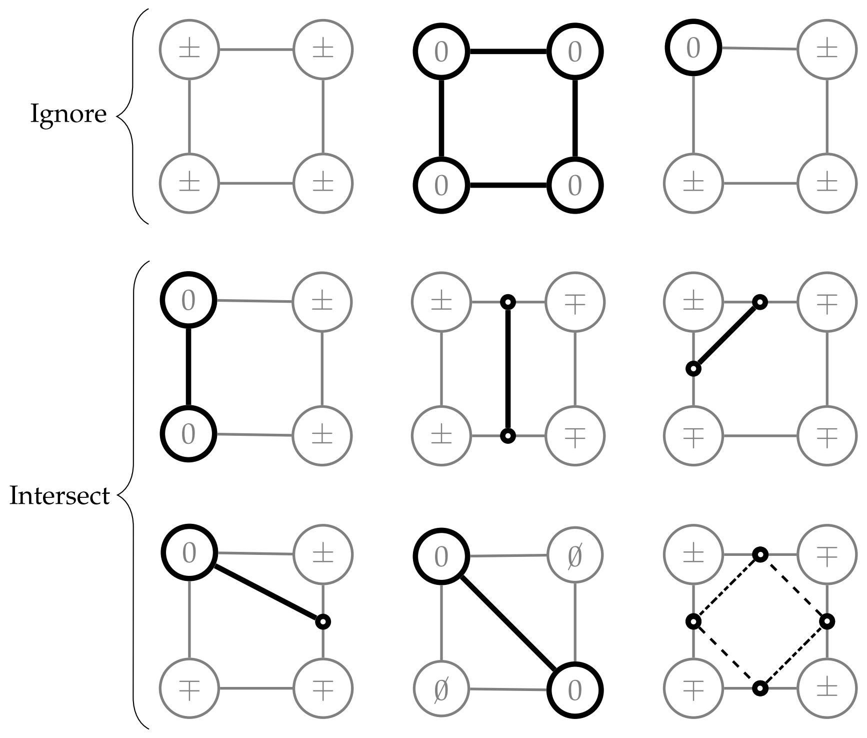


Figure 19: Possible grid cell intersections with contouring depth h_c. A '0' indicates a vertex at contouring depth h_c, '+' and '−' are vertices respectively above and below h_c. A '∅' indicates a vertex that is not equal to h_c.

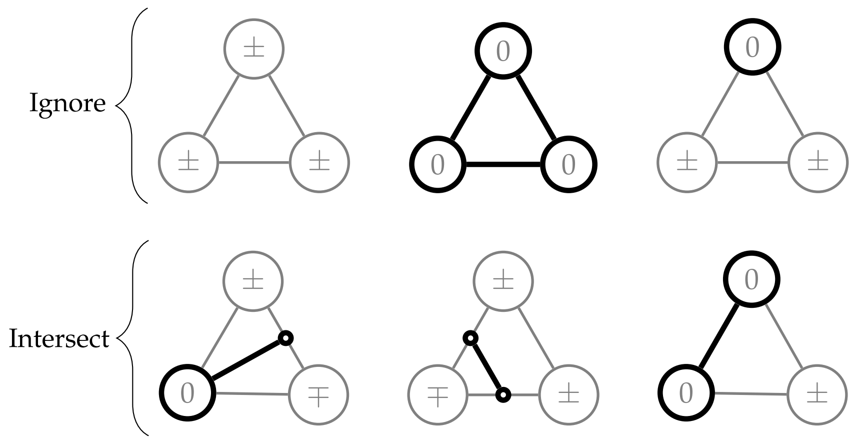


Figure 20: Possible triangle intersections with contouring depth h_c. A '0' indicates a vertex at contouring depth h_c, '+' and '−' are vertices respectively above and below h_c.

Note that since the algorithm contours every grid cell or triangle individually and requires only local information, it is very easy to parallelize. It is thus quite a scalable algorithm.

The resulting line segments—at most one for every data object—can be merged into the complete contour lines afterwards, if such connectivity is required for the intended application.

Algorithm 1 A simple contouring algorithm

Input: a planar partition of elements E (grid cells or triangles) and a list of contouring depths H_c
Output: a list of line segments that form the contour lines

```
1:  function GenerateContours
2:      $segmentList \leftarrow [\,]$
3:      for all depths $h_c \in H_c$ do
4:          for all elements $e \in E$ do
5:              if $h_c$ intersects $e$ then                ▷ See Figures 19 and 20
6:                  extract intersection $\chi$ of $h_c$ with $e$
7:                  append $\chi$ to $segmentList$
8:              end if
9:          end for
10:     end for
11:     return $segmentList$
12: end function
```

2.4.1 Contouring a raster

In the case of contouring a raster using GenerateContours a grid is used that has the data points at its intersections, observe the dotted lines in Figure 18. Intersections are computed by linear interpolation along the edges of this grid.

Figure 19 illustrates the different intersection cases. The top row indicates the cases that are not intersected, either because there is no intersection with h_c or because the intersection would already be included by one or more neighbouring cells. For the same reason of avoiding duplicate line segments and in case of the left-middle case in Figure 19, only bottom and left edges need to be extracted.

The most interesting case is the bottom-right one in Figure 19, that occurs when the two pairs of opposing points are respectively above and below the contouring depth h_c. An ambiguity arises here since there are two ways to extract a valid pair of contour line segments. This can be resolved by simply picking a random option, consistently choose one geometric orientation (the Geospatial Data Abstraction Library (GDAL) does that) or by considering what is the *safest* in terms of the hydrographic safety constraint. The latter is promoted by Zhang et al. (2008), who always choose the option that gives more area to the shallower side of the contour.

Note that when considering the complete contouring pipeline—and even when always picking the safer option in the ambiguous case—raster contouring is likely to violate the safety constraint (see § 3.2).

2.4.2 Contouring a TIN

Since a triangle has one edge less than a square grid cell, there is less possible intersection cases (observe Figure 20). There is also no ambiguous case. Otherwise the intersection cases are quite similar to the raster situation and they can be easily implemented with GenerateContours.

Van Kreveld (1994) proposes a more efficient algorithm to compute contours for a TIN. It requires an additional data structure, the *interval tree*, that aids fast localization of triangles that intersect a certain depth. Compared to the brute-force GenerateContours, this approach has two advantages. Firstly, not all triangles need to be visited during contouring, which is especially beneficial when the requested contour line does not intersect a large portion of the triangles. And secondly, the method delivers connected contour segments straight away, so no line merging needs to be performed as with GenerateContours. Van Kreveld et al. (1997) also propose a somewhat similar approach that is based on the slightly more sophisticated *contour tree* (also known as the *Reeb graph*).

Given that the auxiliary data structure is available, these approaches can compute a set of contour lines in logarithmic time, compared to linear time for GenerateContours.

2.5 GENERALIZATION

Generalization is a very comprehensive concept. There are many aspects to it and it inherently relates to many of the ideas that are discussed in this chapter. This section will introduce the basic concepts, relate them to the contents of previous sections and discuss what generalization means from a bathymetric point of view.

In essence, generalization is the process of the meaningful reduction of detail (on a map). This is not an optional process; *all* maps are reductions of reality (Kimerling and Muehrcke, 2009). As explained in § 2.1, it is impossible to capture reality in its full complexity, let alone to depict it on a map. But, as a consequence of Tobler's First Law, we can discretize reality and still come to an acceptable approximation. In this context, discretization is in effect a way to meaningfully reduce detail. It is meaningful in the sense that the resulting approximation can now be (digitally) stored, analyzed and reproduced.

Thus, discretization is a form of generalization. Gruenreich (1992) termed this type of generalization *object generalization*: building a primary model of the real world. This primary model is typically the result of physically sampling reality and storing those samples in a digital representation that contains as much information as possible. For technical reasons, like computational and storage efficiency or lowering network transfer bandwidth (Weibel, 1997), the detail in the primary data model is often further reduced systematically. For fields this is usually done by downsampling[6]. Gruenreich (1992) calls this *model generalization*, it is a process that transforms the primary model to a secondary model, which has the same general data structure but contains less data elements.

Of a more substantial nature is *cartographic generalization*. Weibel (1997) describes this as the generalization of spatial data for cartographic visualization. It is the transformation of a primary or secondary model into a final map product. He further states that the main objective of generalization, in the conventional (cartographic) sense, is to create maps of high graphical clarity so that the map image is easily perceived and the message that the map intends to deliver can be readily understood. In achieving this, map scale and map purpose are particularly important. As map scale is reduced, small map objects (symbols, points, lines and areas) may approach the limits of visual perceptibility (Weibel, 1997). To maintain legibility and preserve aesthetic quality, the map's objects are therefore altered in their visual representation. Depending on the map's purpose, some (properties of) objects are assigned higher priorities than others during the process of cartographic generalization. Besides, all map objects are related; the alteration of one map object may very well induce changes to nearby map elements. Yet, at all times the connection to geographical reality and the nature of the depicted spatial data must be respected. With the wilderness of map design choices that results, it is not hard to understand that two equally competent cartographers can come up with two different maps, nonetheless those maps will still comply with the same map objectives (Kraak and Ormeling, 2003). Cartographic generalization thus involves a good deal of subjective decisions. That sets it apart from object- and model generalization, whose processes can be automated more easily.

2.5.1 Hydrographic generalization constraints

From the map purpose a set of generalization constraints can be determined. These are to be used as guidelines during the process of generalization. For the contour generalization of hydrographic charts, the following ones are important (Zhang and Guilbert, 2011):

6 Downsampling here means to reduce the sample count of a digital field representation. For example in case of a raster this is to increase the pixel size

SAFETY: contours may not indicate a depth deeper than the measured field. Thus of any point on the map the real depth is at least the depth as interpretable from the map.

LEGIBILITY: contours must be clearly legible. Unnecessary detail is to be removed while basic geometrical form and spatial position must be maintained (Kimerling and Muehrcke, 2009).

TOPOLOGY: topological relationships are to be preserved. Contour line intersection may not occur. Depth soundings are consistent with the contour lines.

MORPHOLOGY: morphological details of the waterbody must be maintained as much as possible.

Of these the safety and legibility are the most important. Note that these constraints are sometimes conflicting. Particularly the safety constraint limits the effectiveness of many operators that improve legibility (this is also observed in § 5.4.4, p. 76).

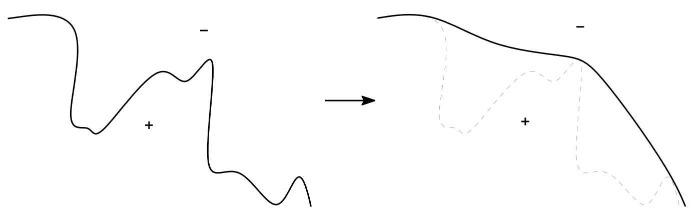
2.5.2 Object based generalization operators

The process of (cartographic) generalization is often decomposed into subprocesses: the generalization operators. Do note that generalization is more than just a sum of its parts. A good cartographic generalization operator (e.g. line simplification) may be relatively simple to implement, however to recognize when it needs to be applied and to make the different generalization operators work together without causing conflicts, is a bigger challenge. On the other hand, a decomposition into separate generalization operators does make the complex process of generalization easier to understand, as it results in some smaller but more clearly defined problems to solve. These operators may be sequentially applied on a map, in which case it should be noted that the ordering in the sequence makes a difference (Weibel, 1997). A standard sequence that works for every application does not exist, yet for instance Weibel (1997) does provide some general pragmatic guidelines on this matter.

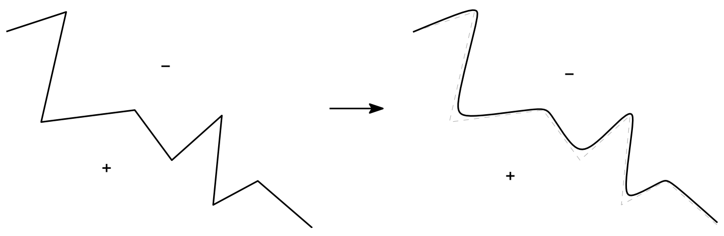
Here I present a basic list of such generalization operations that are relevant to hydrographic contour line generalization. What they have in common is that they all aim to remove insignificant details in order to improve legibility. Furthermore, they all respect the safety constraint, as can be seen in Figure 21.

SIMPLIFICATION: selectively reducing the number of points required to represent an object. Superfluous bents and shaped in a contour line are to be removed.

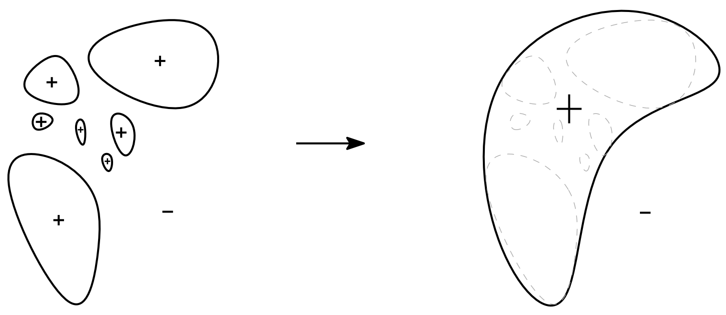
SMOOTHING: reducing the angularity of angles between lines to improve the aesthetic quality of the map.



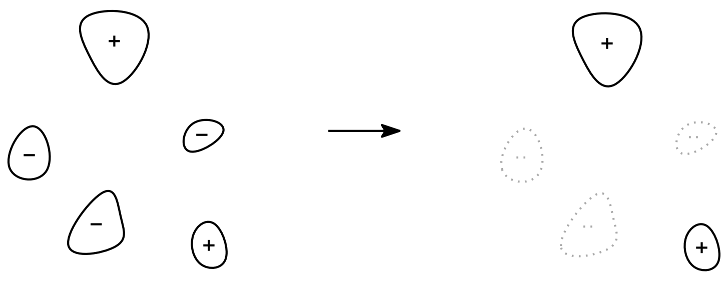
(a) Simplification



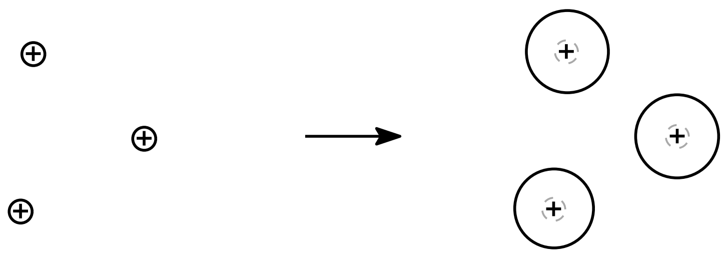
(b) Smoothing



(c) Aggregation



(d) Omission



(e) Enlargement

Figure 21: Generalization operators for hydrographic contours. The '+' and '−' symbols respectively indicate shallow and deep regions. Note how the safety constraint is always respected by only moving lines towards the deeper regions.

AGGREGATION: grouping of individual nearby features into a single larger object.

OMISSION: not including an object. Take for instance very small contours that depict a small pit in the surface. The pit may be smaller than the dimension of the navigating ship or its location may be so that it has no meaning to navigation. In either case the pit is deemed to be an irrelevant detail, and should thus be removed. Of course the opposite, a bump in the surface, should never be omitted as this would violate the safety constrain.

ENLARGEMENT: features, such as sharp peaks, that would be too small to be legible on the map, yet very significant for safe navigation, must be enlarged so that they are still noticed by the map user.

Observe that in Figures 21a-(e) the hydrographic safety constraint is respected at all times. This means that the contours are always pushed towards the deeper area as illustrated in Figure 22.

In literature the listed generalization operators are often described as object based and context-independent (Kimerling and Muehrcke, 2009; Weibel, 1997). As such, they operate on map objects, such as (contour) lines, individually, disregarding the rest of the map and how they affect each other. Also note that the object based view of reality does not really apply to contour lines. Although contours are line objects on a map, they do in fact represent fields.

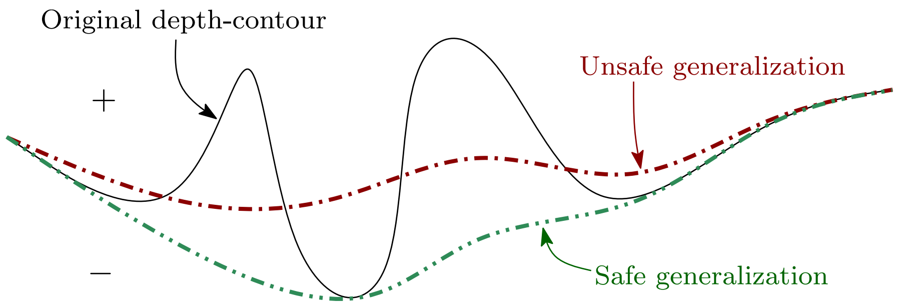


Figure 22: During generalization, contours can only be moved towards greater depth (indicated by a '−').

2.5.3 Generalization of the field

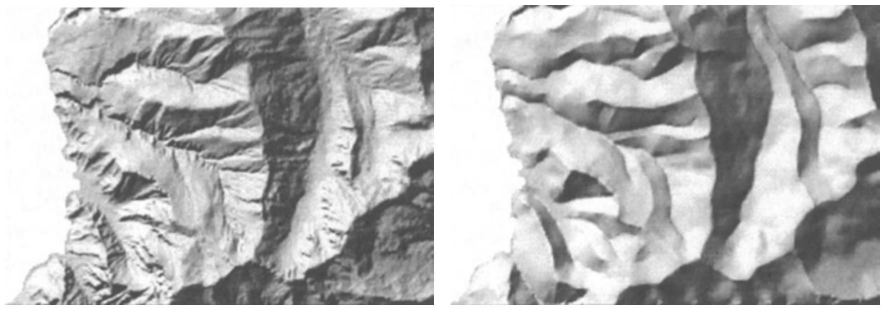
Specifically in the light of contour line generalization, the renowned cartographer Eduard Imhof makes a compelling argument for a field based approach in the first of his eight rules for contour line generalization for small scales:

> "One must never overlook the fact that (geographic) *surfaces* are being depicted with contours. A single line says

> very little. One line does not define a surface. Everything comes back, eventually, to the formation of the system of lines, that is, the surface.” (Imhof, 1965)

Where Imhof, who drew maps by hand, talks of a system of lines that forms a surface, I think that with the technological advances of GIS we should even say that a *field* forms a surface. It is simply so, that a system of lines—and even more so—a field gives a much more complete impression of geographical reality than a single line does. And, as pointed out by Imhof, in the generalization of a contour line one should also make use of this type of additional information that is related to nearby morphological features.

Field-based variants of the afore mentioned object based operators do exist in some cases. For instance smoothing a field, by averaging or removing and re-interpolating points, is quite common. Interestingly, when we look at the corresponding contour lines, the smoothing of a field often also results in some form of omission and aggregation (see Figure 40, p. 70). To control exactly where on the map the generalization is applied, *structure recognition* can be employed. Weibel (1997) mentions the use of so-called structure line models, consisting out of ridges and drainage channels, for the generalization of a terrain field. The resulting network of terrain feature lines are generalized, after which the regions in between are re-interpolated based on the generalized structure lines. This will reduce detail, while retaining significant surface lines. Figure 23 illustrates this process with an example.



(a) Original field **(b)** Generalizaed field

Figure 23: Generalization using structure line models (Weibel, 1997).

Alternatively simplification of (terrain) fields in the case of TINs can be achieved using TIN simplification (explained in § 3.3), which is in a way both model and cartographic generalization (Weibel, 1997) because it offers a meaningful way to reduce data.

In Chapter 4 a generalization method is developed that performs field based generalization of contour lines for hydrographic charts.

2.5.4 The generalization process as a whole

The availability of implementations of generalization operators on itself does not solve the complete problem of automatic generalization. The challenge remains to successfully deploy these operators in an integrated approach that fulfills the generalization objectives of a given map application. As described by Mackaness et al. (2007) three general approaches exist to tackle this problem. The first one, named *condition-action modeling*, is based on structural knowledge that is derived from map objects themselves and their relations to each other. Based on this structural knowledge a set of conditions is evaluated on basis of which certain actions (operators) are performed. For example:

Example of condition-action modeling

```
1: if $length(contour) > 10m$ AND $contour$ is a pit then
2:     remove $contour$ from map
3: end if
```

Second is *human interaction modeling*, which is based on the principle that the cognitive workload can be shared between computer and human. Tasks that can be sufficiently formalized are carried out by the computer, while the human assumes responsibility for guiding and controlling the algorithm. A variant were the computer is not limited to merely performing fixed generalization operators initiated by a mouse click of the user, is *amplified intelligence* (Weibel, 1991). Here the computer performs a structural analysis on the data and subsequently *proposes* likely solutions to generalization problems by visually highlighting them. It is still up to the user to accept them or not. Such a guided approach aims to lighten the workload of the cartographer as much as technically possible, yet still leaves him in full control of what is happening.

Third are the *constrained-based models*, where the generalization process is continually guided by a set of constraints. The system seeks to fulfill these constrains as much as possible. An example is a form of artificial intelligence named *agent-based modeling*. It is based on the definition of autonomous *agents* each of which controls a part of the map. Every agent knows its objectives and attempts to find the best solution to fulfill those. Interaction between agents plays an important role, as well as a continuous evaluation of the effectiveness as possible solutions. Zhang and Guilbert (2011) employ a multi-agent system in the generalization of hydrographic contour lines.

3 CURRENT APPROACHES IN HYDROGRAPHIC CONTOURING

The main goal of this chapter is to describe current computer-based approaches to hydrographic contour line generalization, using some of the theory that was introduced in the previous chapter. Described are mostly methods that are common practice, but also some other relevant methods from literature. In the final section of this chapter (§ 3.6), the pros and cons for each method are summarized.

The described methods have the common goal of hydrographic contour line generalization, but that goal can be achieved in different manners. Observe Figure 24 that illustrates the basic processing pipeline that is commonly used to obtain generalized contour lines from a point cloud.

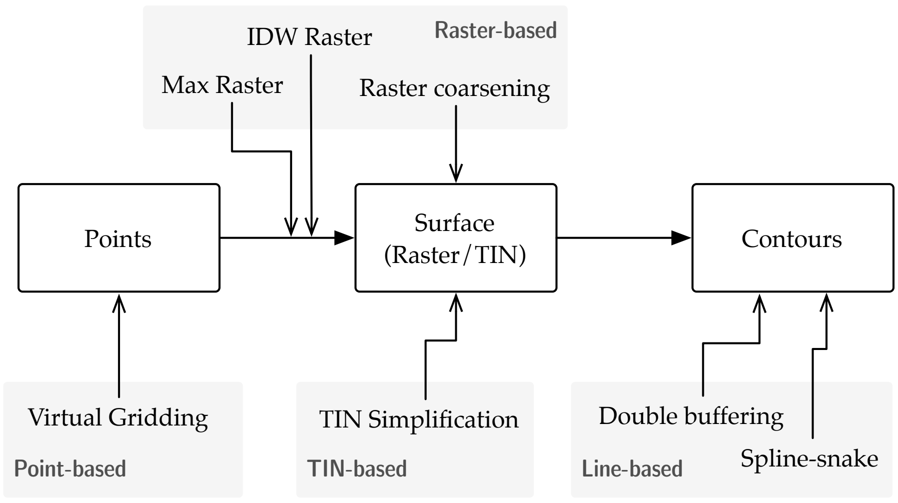


Figure 24: Basic processing scheme to obtain generalized contours from a given point cloud.

Evidently, every processing operation that is performed in this pipeline has an effect on the generated contour lines. This simple fact has two consequences. Firstly, it means that the hydrographic contouring constraints (see § 2.5.1)—most importantly the safety constraint—must be carefully considered during each operation, and any assumption made on the input of one processing operation should be valid given the preceding operations. And secondly, it means that different strategies exist to obtain those nicely generalized depth-contours. The different methods, that are described in this chapter, work on different parts of the pipeline (as seen in Figure 24). I make the following categorization of methods:

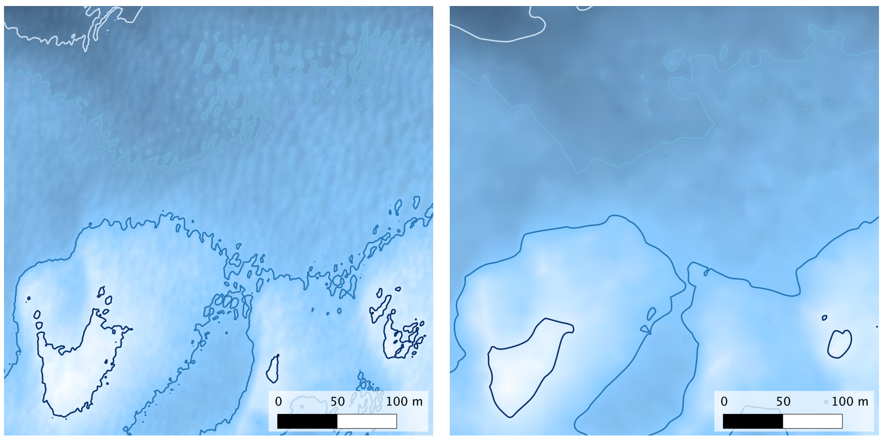
POINT-BASED METHODS: methods that work on points (§ 3.1).

RASTER-BASED METHODS: methods that work with a raster data structure (§ 3.2).

TIN-BASED METHODS: methods that work on a TIN data structure (§ 3.3).

LINE-BASED METHODS: methods that work on contour lines (§ 3.4).

Other related work, that does not fall into these categories is discussed in § 3.5. Do note that all of the presented methods are no more than mere generalization operators. In order to employ these methods in the complete process of hydrographic chart generation, that may require interaction with other operators and other types of map objects (e.g. depth soundings, symbology), more sophistication is required (also described in § 2.5.4).



(a) Using all points (b) Using a random 2% of points

Figure 25: Reduction of detail by throwing away random points. Shown is a Natural Neighbour interpolation with a cellsize of 50cm. Portion of the Zeeland dataset (see Appendix C.5).

3.1 POINT-BASED METHODS

Point-based methods perform filtering, i.e. reducing the data volume by dropping points. Point filtering is mainly done for three reasons:

1. to remove noise and statistical outliers;
2. less points take less time to process;
3. sparser data requires less generalization since it contains less detail per definition

The first reason is out of the scope of this thesis, input points are considered to be clean (e.g. Arge et al. (2010) describes how that can

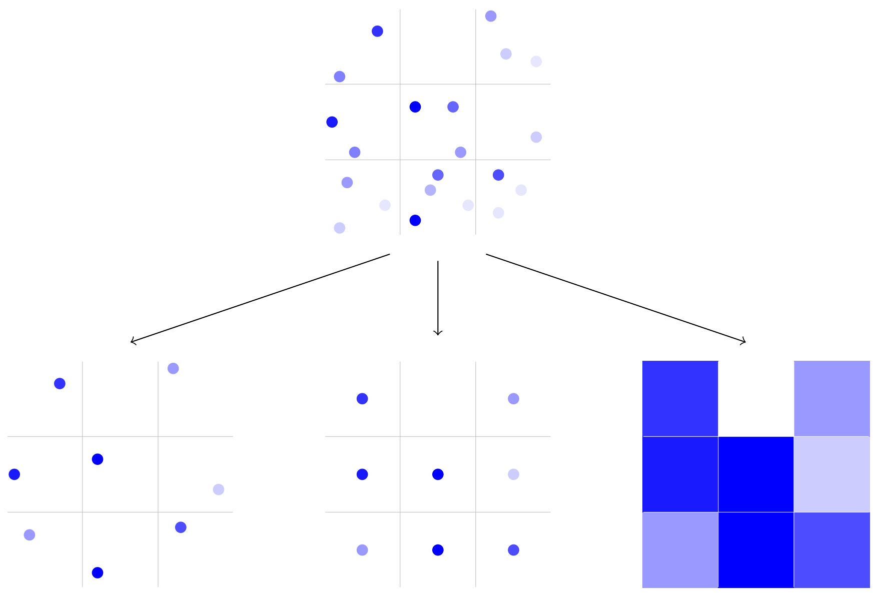


Figure 26: Virtual gridding (a) and simple rasterization (b and c) of a set of irregularly distributed points. Deeper blue indicates shallower points.

be achieved). However, the other two reasons are of interest for this thesis.

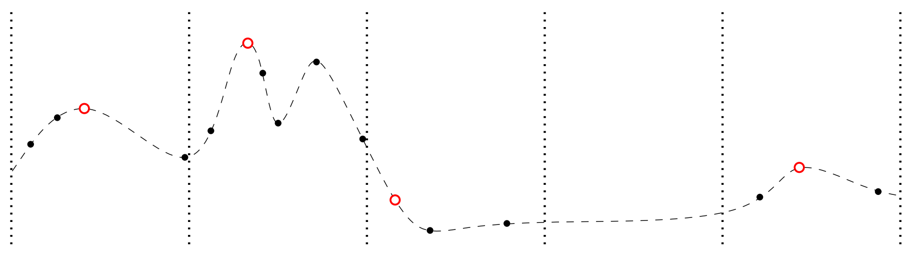
Reducing the volume of data by point filtering is a form of object generalization, where the reduction of the data volume is an objective on itself in order to save storage and processing time. As such, very simple methods like randomly selecting a subset of points or, quite arbitrarily, keeping one point out of every 100 points would serve the purpose. However, using those methods does not agree with the concept of cartographic generalization where only insignificant—rather than *arbitrary*—detail should be removed. That is especially valid for data sets with heterogeneous distribution, since such random selection methods would significantly alter relatively big areas in the very sparse regions, where one point might represent a really big portion of the geographical extent of the data set. Of course the safety constraint, which may also promote a point to being significant, is to be considered here. The removal of any point will have an effect on the eventual surface that is created. And any point that is filtered out should lay *below* that surface, otherwise the safety is violated. It does seem difficult to create such a safe surface, when points are already omitted in the very beginning of the processing pipeline. It gives rise to the question if we should remove any points at all when the safety constraint is to respected? It certainly seems *safer* not to.

However, in order to obtain well generalized depth-contours, insignificant detail—initially represented by points—must be removed.

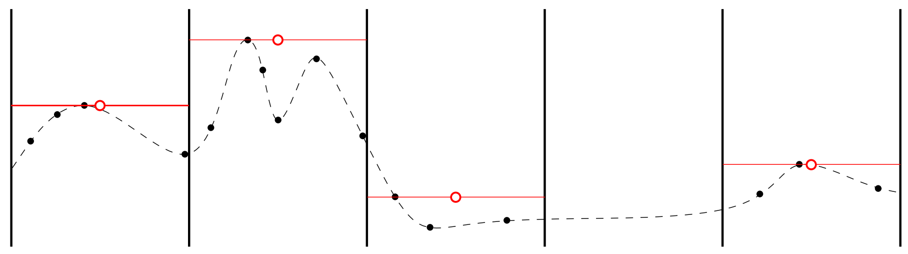
So, also for that reason it makes sense to filter out some (insignificant) points. As can be observed from Figure 25, interpolated surfaces, and thus the derived contours, indeed appear more generalized when constructed from less points.

3.1.1 Virtual gridding

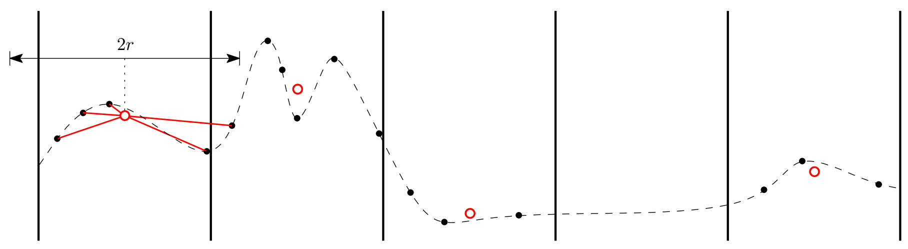
A point filtering method that is employed in practice is virtual gridding. The idea is to overlay a virtual grid on the input points and to keep one point for every virtual grid cell, as in Figure 26(a). The number of remaining points is at most equal to the number of grid cells that are used in the virtual grid. Different functions can be used to pick a point for each cell. It can for instance be the deepest, the average, the median or the shallowest point that lays within the boundaries of that cell. For the sake of the safety constraint the shallowest point is often chosen, see Figure 27a for a one-dimensional equivalent.



(a) Virtual gridding



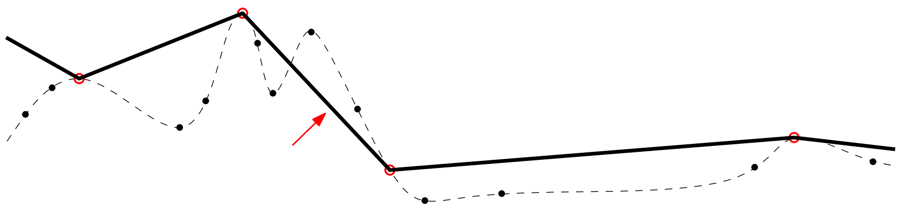
(b) Max rasterization

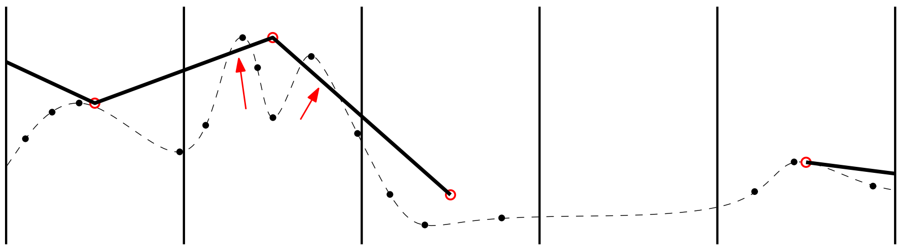


(c) IDW rasterization

Figure 27: One-dimensional schematics of different filtering and rasterization methods. Red circles indicate points that are chosen to represent the grid cell.

A slight variant of this method uses a quadtree data structure rather than the fixed regular grid. In that quadtree, cells are merged (preserving the shallowest point) if the variance of data points in each cell is below a pre-set value (Ledoux, 2009). This results in less remaining points in areas with relatively low variance. At the same time the quadtree also functions as an efficient spatial index that allows for quick extraction of a subset of the data with the required resolution. In practice the selected points are used to either create a TIN or an IDW interpolation based raster.


(a) Virtual gridding and TIN-based contour values


(b) Max rasterization

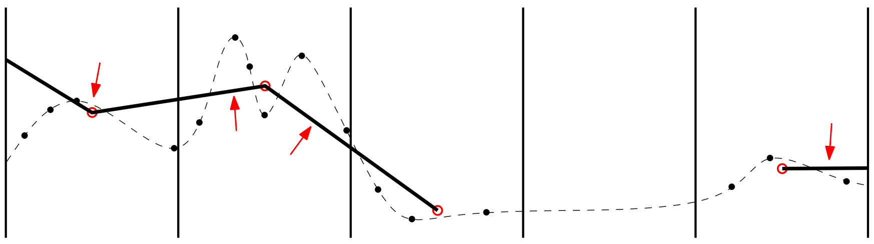

(c) IDW rasterization

Figure 28: The one-dimensional contouring surface of Figure 27 is shown in thick black lines (the 2D contours, that would go orthogonal to the paper, are points on these lines). Red arrows indicate where the safety constraint is violated. Also note that in case a grid cell contains no data, no contours can be derived.

However, picking the shallowest point per virtual grid cell does not guarantee safe contours in principle. The problem is that contour extraction algorithms (see § 2.4) perform a linear interpolation on top of the points present in the data structure. As can be observed from Figure 28a, this easily results in safety violations at 'secondary' local maxima in a grid cell. The number and severity of these violation is related to the used cellsize of the virtual grid cells. A bigger cellsize will result in more and more severely violated points.

3.2 RASTER-BASED METHODS

3.2.1 Max rasterization

This method is quite similar to virtual gridding. The difference lies herein that instead of exact points, a raster is outputted. Every cell in the virtual grid now becomes a raster cell, which is usually assumed to represent a point located at the cell's center, see Figure 26b for the two-dimensional case and Figure 27a for the one-dimensional case. This disregards the exact coordinates of the original points, and effectively moves the shallowest point in the grid cell to the center of the pixel. The result is a surface that—for the same reason as with virtual gridding—does not guarantee safety. But, aside from that it also disregards the morphology of the sampled field with these arbitrary point movements. That not only violates the morphology constraint, but it can also cause extra safety violations (compare Figures 28a and 28b). Again, the severity of these problems depends on the chosen cellsize.

An additional problem that might occur here are grid cells that do not contain any sample points. These cells, that thus can not be assigned a depth-value, lead to gaps in the eventual depth-contours. Although this is not usually a problem for multi-beam data, it might be problematic with heterogeneous datasets that also contain sparsely sampled areas.

Alternatively the assumption can be made that a grid cell represents a square *area*. This has been illustrated in Figure 26c. The resulting non-continuous surface is now guaranteed to be safe. Because of the discontinuities in this surface it can not be used to generate smooth depth-contours. One might be able to define a smooth surface on top of this blocky raster surface. To my best knowledge however, that has never been tried in the context of hydrographic contouring.

3.2.2 IDW interpolation to a raster

Another way of creating a raster from sample points is through spatial interpolation. In hydrographic contouring practice, Inverse Distance Weighting (IDW) interpolation is a popular interpolation method.

Figure 27c illustrates the process of IDW interpolation (§ 2.3.2) to gridded raster points. As a result of the averaging that takes place with IDW interpolation and the fact that grid cells seldom coincide with sample points, extrema are disregarded and subsequently the safety constraint is violated. This is also evident from Figure 28c.

Generalization can be achieved by tuning the interpolation parameters. With IDW these are primarily the radius and the power (as described in § 2.3.2). A higher radius and a lower power will result in a more generalized and smoother looking surface. On the on the other hand however, that would also lead to bigger safety violations, as the amount of averaging increases.

3.2.3 Raster coarsening

Raster coarsening is somewhat similar to max rasterization, the difference is that it takes a raster as input rather than exact points. Based on that input raster, a new raster is created that has larger cellsize. In this way small details in the surface are omitted. Subsequently the contour lines that correspond to this coarsened raster surface also contain less small details. In terms of safety this method has the same drawbacks as max rasterization.

3.3 TIN SIMPLIFICATION

The objective of TIN simplification is to minimize the the number of vertices in a TIN while staying as close as possible to the initial model. The resulting TIN has just enough vertices to model every phenomena it represents within a given tolerance. This means more vertices in regions of high variability and less vertices in (planar) areas of small variability. Less vertices overall of course means more efficient triangle traversal and less storage requirements on the one hand and it could be seen as a form of cartographic generalization on the other hand.

To my knowledge TIN simplification is not being used for the generalization of hydrographic depth-contours. Yet, I think that it could be an effective method to achieve that. Unlike the described raster-based generalization method, most TIN simplification methods take into account the geometric configuration of neighboring points. Furthermore points are not arbitrarily shifted, which does happen with the raster-based methods.

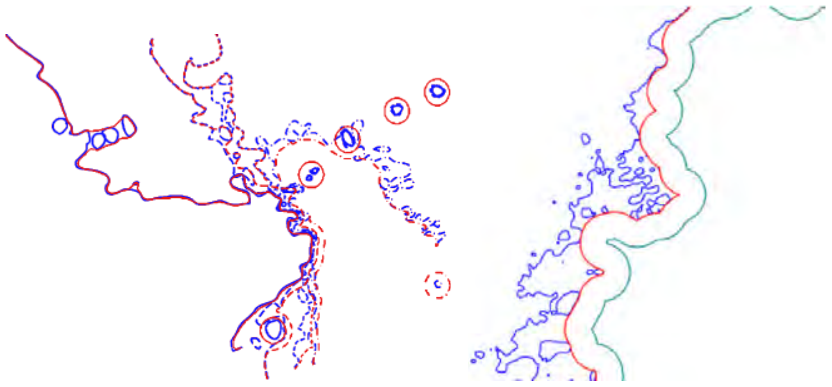
Heckbert and Garland (1997) conducted a comprehensive survey of numerous existing TIN simplification methods developed by researchers from various fields in science. Garland and Heckbert (1995) also propose their own fast algorithm to approximate height fields. It is based on a very simple local metric of judging the importance of a point, first used by Lee (1989) who proposed a comparable algorithm (he named it the drop heuristics algorithm), that in terms of approximation quality outperforms more complicated or global metrics. This metric is defined as the vertical (elevation) difference of an input point with the triangulation without that point at the same location. This metric calculated for all input points after which either the point with the biggest difference is added to an initial minimal triangulation that covers the data extent (this the *refinement* approach) or the point with the smallest difference is dropped from a complete triangulation of all points (the *decimation* approach). That step is repeated until a pre-set threshold of elevation difference is reached. As pointed out by Lee (1989), Garland and Heckbert (1995) and van Kreveld (1997) this results in a high quality approximation of the sampled field with only a limited set of vertices that is also a Delaunay triangulation.

Garland and Heckbert (1997) propose a truly three-dimensional simplification algorithm for triangular meshes, which as demonstrated in the same paper also works well for terrain models. The quadric edge contraction algorithm that they propose is based on a metric that is defined for each vertex as the sum of squared distances of a point to the planes that are spanned by the triangles incident to that vertex. For every edge in the triangulation the sum of this quadric metric for the incident vertices is calculated. And by minimizing this sum, the location of a new vertex follows that could replace both initial vertices. The next step is to perform the edge contractions (replacing its two initial vertices with a single vertex that minimizes summed quadric errors) for the edges for which the quadric error is the lowest. Evidently, every edge contraction also implies a local re-triangulation. This process is continued until some preset number of triangles is remaining. The quality of resulting geometry in terms of approximation of the initial model is arguably better than the earlier described simplification algorithm, because quadric edge contraction is based on a fully three-dimensional metric and also optimizes the location of new vertices, whereas drop-heuristics is just based on a one-dimensional metric and does not perform any optimization of vertex location.

In § 4.2 I discuss the implementation of one of these TIN simplification methods for the application in hydrographic contour line generalization.

3.4 LINE-BASED METHODS

The generalization of lines in general is a very common operation in digital cartography. Among others, Guilbert and Saux (2008) and Li and Openshaw (1992) have classified the numerous line smoothing and simplification methods according to different characteristics. Unfortunately most of them do not explicitly take into account the constraint of navigation safety, since they are not specifically developed for bathymetric charts. Only methods that do respect the safety constraint are described here.



(a) The B-spline snake model. Different line patterns indicate different contour depths. Taken from Guilbert and Saux (2008)

(b) Double buffering. The original contour line is first buffered to the green line, which is subsequently buffered back to the red line. From Smith (2003)

Figure 29: Different line-based depth contour generalization methods. Blue represents the original contour lines, red are the generalized contour lines.

3.4.1 Double buffering

One line-based approach is double-buffering. It is a popular method employed in major commercial hydrographic packages from companies such as Atlis and Caris. As illustrated in Figure 29b, it works by buffering a set of input contour lines back and forth, effectively taking into account the safety constraint as well as performing a form of aggregation. When a sphere is taken instead of a disk, the method can also be used on a 3D surface. Resulting contour lines seem to be safe, they are however not always C^1 smooth, because sharp outward pointing angles are retained (visible in Figure 29b).

Do note that the safety can only be guaranteed in case of safe input contours. If the input contours are not safe, for example if they are extracted from an IDW interpolated grid, the double buffering operation does not make them safe (but neither would it cause extra violations of the safety).

A problem with double buffering is that it uses a fixed buffering distance, which might not always be appropriate because some regions may require a different buffering distance than other regions. Furthermore, from Figure 29b it can be argued that the resulting contours imply a surface that is completely constructed from the intersection of disks, which is not necessarily morphologically correct. In other words: the hydrographic morphology constraint is not fully respected.

3.4.2 Spline snake model

A spline is a piecewise polynomial function that is by definition smooth. Guilbert and Lin (2007) use splines in combination with a snake model to perform smoothing of bathymetric contour lines. In a later paper (Guilbert and Saux, 2008) the method is extended with enlargement and aggregation operators. The spline snake model is an iterative optimization method that minimizes the total energy that is associated with a spline (or contour line). This energy is related to the shape of the spline itself (internal energy) and to constraints that are imposed externally (external energy). A smoother and straighter line has less internal energy. The external energy is lower when constraints such as a minimum line distance and the safety are met. The sum of the energies is iteratively minimized.

The method respects the safety constraint and achieves smooth contours at the same time (see Figure 29a). Furthermore, in each iteration it is checked if there are any conflicts in the contours. A conflict occurs when the distance between line segments (from the same or another contour) is below a given threshold, if that is the case the conflicting segment is removed. Also the contours are generalized in order of ascending depth, so that deformations resulting from conflicts are propagated towards the deepest contours. The method is designed to be fully automatic; parameters in the spline model are automatically set. However in practice it seems that manual intervention is still required. Firstly because the method itself does not recognize the topology of the contours (i.e. the orientation with respect to the safety constraint). And secondly, in one of the case studies a contour line needed to be split because it got stuck between two other contours. It is unclear how well the method performs on raw contours, as the input contours in the presented case studies are b-splines already and it is unclear how these b-splines can be safely obtained in the first place. The authors also note that the computational cost of the algorithm is high for complex lines, where convergence is slow because of the safety constraint.

3.5 OTHER RELATED WORK

3.5.1 Multi-agent systems and Feature trees

Guilbert (2012) introduces a method to extract (bathymetric) terrain features and to store them in a *feature tree* that provides a description of the surface at multiple levels of detail (see Figure 30). This graph structure allows for automatic recognition of certain types of surface features such as peaks or pits and their hierarchal relations. Zhang and Guilbert (2011) propose the idea to use these feature trees in a multi-agent generalization system (as discussed in § 2.5.4). Guilbert and Zhang (2012) research this further, and also combine it with several generalization operators based on the spline snake model. The size of spot soundings labels is also considered[1], and it is ensured that the contours do not intersect the label. However, how to obtain those spot soundings is not mentioned. Also, the spline-snake smoothing operator (see § 3.4.2) is not (yet) integrated in the approach.

3.5.2 The navigational surface

Smith et al. (2002) and Smith (2003) introduced the concept of the 'Navigational Surface' which aims to improve the workflow of the nautical charting process in different ways. Firstly, it focusses on obtaining an uncertainty grid of the available depth soundings by averaging the measurement-uncertainty in depth-soundings in multibeam regions and empirically estimating the uncertainty at linear TIN interpolated grid cells in regions with low data density (see § 2.3.3). Secondly, it sets rules to combine different overlapping surveys (rasters) based on the modeled uncertainty and date of acquisition. And thirdly, it proposes to keep a unified database model that contains high resolution grids, including the uncertainty models, that can then be used to derive chart product for different applications. These rasters are constructed using max rasterization (see § 3.2) in case of densely distributed data and using linear TIN interpolation in case of sparsely distributed soundings. Deriving a hydrographic chart from the Navigational Surface involves raster coarsening (see § 3.2.3) to the appropriate scale and also double buffering (see § 3.4.1). Furthermore in case of grid cells with low horizontal accuracy the surface is locally

1 Spot soundings are significant depth measurements that are explicitly indicated on a hydrographic chart.

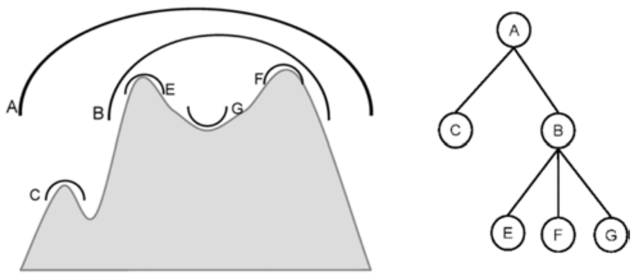


Figure 30: Structural analysis of the surface through the feature tree. From Zhang and Guilbert (2011).

'defocused', which is quite similar to the enlargement generalization operator depicted in Figure 21e.

Note that the objective of the Navigational Surface is not to fully automate the charting process, but to facilitate and support the hydrographer in his work. That however, also implies that the hydrographer is the one responsible for any violations of the hydrographic safety constraint, which will occur since the approach is based on virtual gridding and raster coarsening (Smith (2003) calls this *shoal binning*). To mitigate the danger of critical depth soundings the Navigational surface Smith (2003) lists three points. Firstly, the resolution of the raster should be chosen approximately equal to the sonar resolution. Secondly, the hydrographer can use the earlier described uncertainty models. And third, the hydrographer is given the 'opportunity' to select *Golden Soundings*, that will be honored at the nearest grid cell.

3.6 CONCLUSIONS

The most significant conclusion for this chapter is that most generalization methods are in fact not safe by definition, i.e. they are likely to violate the safety constraint for a number of input points. Furthermore, for the methods that are safe on itself, that observation is only of limited value since those methods assume safe contour lines at the input and it is not clear how to obtain those *safely*.

Table 1 summarizes the general findings of the different generalization methods that were discussed in this chapter. This is done in terms of safety (column 3), generalization (column 4) and (C^1) smoothness (column 5). These criteria respectively correspond to the hydrographic generalization constraints of safety, morphology and legibility (see § 2.5.1). The topology constraint is satisfied for all methods and is therefore not relevant in the comparison.

	Type	*Safe*	*Generalization*	*Smooth*	*Other:*
Virtual Gridding	point	–	a	–	
Max rasterization	raster	–	a	–	
Raster coarsening	raster	–	a	–	
IDW interpolated Raster	raster	–	s	√	
TIN simplification	TIN	–	s	–	
Double buffering	line	√*	s,u	–	
Spline-snake	line	√*	s	√	Computationally expensive

a = arbitrary reduction of detail
s = significant features are preserved
u = unnatural appearance of contour lines
*Only if the input contours are safe

Table 1: Summary of generalization methods in the hydrographic contouring pipeline

The first three methods in the table (virtual gridding, max rasterization and raster coarsening) have similar characteristics, which is not surprising since they are all quite similar. The fundamental problem for these methods is the fact that (significant) detail is arbitrarily omitted. It is a result of the fact that these are in essence only model generalization methods (see § 2.5), where the significance of a feature is not considered and the primary objective is reducing the data volume. Consequently the safety constraint is not respected by these methods.

The IDW interpolated raster does a better job in terms of smoothness (if an appropriate raster cellsize is chosen). However it does not respect the hydrographic safety constraint and also suffer.

Next in the table is TIN simplifcation which is a definite improvement over the previous three methods, because the significance of features is considered during the generalization process. The safety of one of these methods is further discussed in § 4.2. Last in the table are the line-based methods, which are, given valid and safe contour lines as input, both safe. However in case of double buffering the resulting depth-contours are not smooth, as certain contour configurations result in sharp joints. Furthermore the morphology of the surface is not optimally respected as explained in § 3.4.1 and ultimately the result is only as safe as the inputted contour lines. The spline snake based method is both smooth and safe, yet it appears to be slow and not robust and thus not completely automatic. Furthermore it is unclear how this method performs on raw input contours, as it is only tested with splines as input and it is unclear how these can safely be obtained from raw input points.

Note that many more contouring methods exist, yet those are aimed at the general problem of contour generalization and thus never discuss the safety constraint. For instance Matuk et al. (2006) perform contour line generalization based on skeleton retraction that would preserve topology by definition. The triangulation based method of van der Poorten and Jones (2002) delivers smooth contours that are topologically sound.

4 A VORONOI- AND SURFACE-BASED APPROACH

The key motivation behind the Voronoi- and surface-based approach that is presented here follows from the fundamental limitations of methods that are currently in use for hydrographic contour line generalization (see Chapter 3). Point- and raster-based methods do not guarantee the safety and do not always perform *cartographic* generalization. Line methods assume safe input contours, but it is unclear how to obtain those. Furthermore, the safety constraint is often not truly respected. Part of the problem is the fact that the different processes such as spatial interpolation, generalization and contouring are treated as independent processes, while they are in fact interrelated. In my opinion this calls for a radical rethinking of the existing processing pipeline as depicted in Figure 24, p. 37.

Therefore, and in consideration of the theory described in Chapter 2, the aim of the surface-based approach that is presented in this chapter is to unify the concepts of the digital field representation (data structure), spatial interpolation, generalization and contouring, while keeping a meaningful relation to geographical reality: the loss of information by discretization is minimized. Specifically, it respects the spatial distribution of the source data. But it also intrinsically respects the hydrographic safety and legibility constraints (in terms of smoothness), while guaranteeing topological consistency of the outputted contour lines. Instead of generating contours in a linear sequence of independent processing steps, the idea behind the surface-based approach is to have a single consistent field representation from which contours can be generated as a sort of *view*. Conceptually this somewhat resembles what a spatial database system does. But it is much less generic than a spatial database, since it specifically focusses on the goal of this thesis: the generalization of depth-contours for hydrographic charts.

§ 4.1 gives a general overview of the surface-based approach, explaining the different components and the terminology that is used throughout this chapter. This is followed by a section (§ 4.2) that shortly discusses the idea of incorporating TIN simplification into the surface-based approach. Then, § 4.3 describes the basic operations and algorithms that lie at the core of the surface-based approach. Finally, § 4.4 and § 4.5 discuss how the surface-based approach can be further improved in terms of automation and scalability.

4.1 OVERVIEW

Observe Figure 31. At the core of the surface based approach lies the Voronoi Diagram (VD). It serves as a natural coupling between a Delaunay Triangulation (DT) that contains all of the original sample points and a spatial interpolation method based on natural neighbors. The initial surface that is constructed is defined by the Laplace interpolation of the input points. Because Laplace interpolation is exact and smooth, this surface is safe and smooth per definition. Furthermore, input points are stored as vertices in the triangulation, meaning that their coordinates are also stored exactly. The topology of the DT also serves as a convenient base for any operators that work on the surface, both locally and otherwise. Contours are derived di-

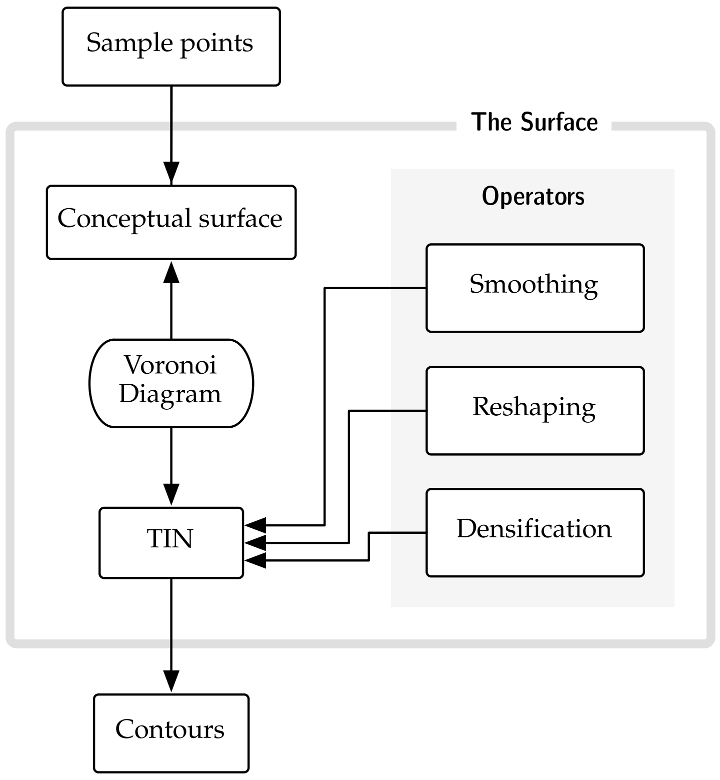


Figure 31: Overview of the Voronoi- and surface-based approach

rectly from the triangulation using the algorithm described in § 2.4.2. As a result those contours are topologically correct and will not contain any intersections. Note that this also means that discretization, caused by the linear interpolation in the contouring algorithm, is only applied at the very final step of the process. The surface itself is, at least conceptually, entirely continuous and smooth.

Generalization operators work on the surface through interaction with the DT, which is thus tightly coupled with the Natural Neighbour interpolated field by means of the VD. By never removing any points and never moving any point downward in the surface, the safety constraint is respected at all times.

4.1.1 Terms and definitions

Throughout this chapter a number of terms and definitions are used. The following list summarizes and clearly defines these terms.

SURFACE: This refers to the complete Voronoi- and Surface surface-based framework as presented in this chapter and as depicted in Figure 31.

SAMPLE POINTS: These represent the sampled geographical reality and are the input points for the surface. Typically this is a point cloud acquired through echo-sounding.

CONCEPTUAL SURFACE: This is the continuous field that is initially interpolated from the sample points using Laplace interpolation. This is assumed to be the ground truth, i.e. the best known representation of geographical reality.

TIN: This is the DT, the data model that represents the conceptual surface, it contains all of the sample points as well as the natural neighbour relations. The triangulation itself is also a linear TIN discretization of the conceptual surface.

SAFE: A representation is said to be safe if the depth of *all* of the original sample points lay under or precisely on that representation.

OPERATOR: A function or process that can be employed on the surface to perform generalization.

4.1.2 Properties of the surface

In summary, these are the principal properties of the surface:

1. Adaptive to the spatial distribution of input sample points
2. Safe, thus in accordance with the constrain of navigational safety for hydrograhpic charts
3. Smooth, meaning that the surface is C^1, i.e. its first derivative is continuous.
4. Topological consistency is guaranteed in the generated depth-contours, i.e. they do not intersect.
5. Exact, it respects and preserves the two-dimensional coordinates of sample points. In discretization, no arbitrary changes

in point coordinates are made that result in a loss of significant information.

6. Local operators, in principle all algorithms related to constructing, altering and extracting contours from the surface are local algorithms.

4.2 TIN SIMPLIFICATION

One of my original ideas was to integrate one of the TIN simplification algorithms that were described in § 3.3 into the Voronoi- and surface-based approach that is presented in this chapter. This would significantly reduce the data volume of input sample points with only a minimal effect on the morphology of the surface, while at the same time deliver a form of generalization.

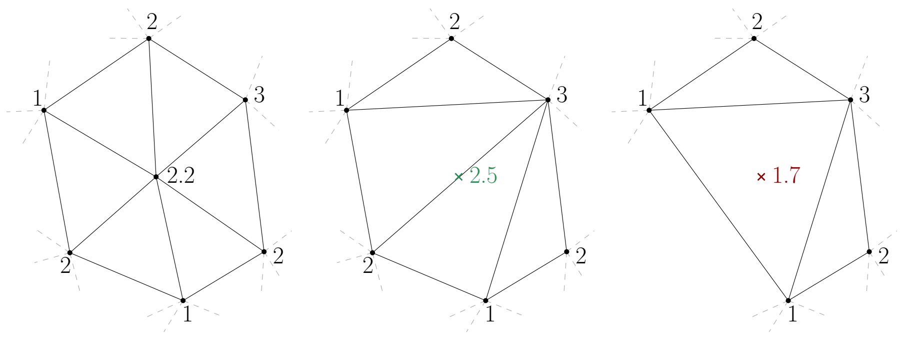
I chose to implement the drop-heuristic algorithm to be the best candidate for the following reasons:

1. It outputs a DT, which is a requirement if one wanted to employ Laplace interpolation.

2. Its metric of vertical distances is directly related to the safety constraint. This seems to make it straightforward to incorporate the safety constraint into the method, i.e. by simply only dropping points from the triangulation that would move the surface upwards.

While it may also be possible to use the quadric edge contraction algorithm for hydrographic (safe) surface simplification, possibly using the method proposed by Zelinka and Garland (2002) to incorporate the safety constraint and the algorithm of Shewchuk (2005) to obtain a Delaunay Triangulation, this would significantly complicate the implementation and probably also significantly increase the running time of the algorithm.

I have thus implemented the drop heuristic algorithm, and modified it to only drop points that would at the time of the drop not violate the safety constraint, i.e. a point is only dropped if that would directly result in an upward movement in the surface. The complete algorithm is given in Appendix A. However, after running the initial tests, I discovered that the algorithm does in fact not respect the hydrographic safety constraint. The cause of this problem is the local edge flipping—it is required for preserving the Deulaunay property of the triangulation—that takes place over multiple point drops. Figure 32 illustrates this. The assumption that the safety constraint can be respected by verifying at each iteration that a point is not moved upwards (see Figure 32b) is flawed, because as a result of future point drops the local triangulation can change in such a way that the safety is still violated (see Figure 32c).



(a) Initial configuration (b) First drop (c) Second drop

Figure 32: Due to edge flips violation of the safety constraint may occur after a series of point drops when using the algorithm of Appendix A. A lower number means a shallower point.

A possible way to circumvent this problem might be to use some sort of data-dependent triangulation instead of the Delaunay triangulation, e.g. for the case of Figure 32 a triangulation exists that is safe (but not Delaunay). However, that is unacceptable since that would mean to give up the Voronoi-based interpolation that is essential for the Voronoi- and surface-based approach. Another solution could be not to drop those points that would be violated in a later iteration of the algorithm. However that would probably significantly complicate the algorithm and also make the simplification less effective (as less point will be dropped). Therefore I ultimately decided to stop working on this idea, and focus on other more effective ways of surface generalization.

4.3 OPERATORS ON THE SURFACE

This section describes the operators that are defined in the surface-based approach. And unlike the TIN simplification algorithm that was described in § 4.2, these are guaranteed to respect the hydrographic safety constraint. The function InterpolateDepth (Algorithm 2) forms the basis for every surface based operator. It performs a Laplace interpolation (see § 2.3.4) for a vertex v that is present in the TIN using only its incident vertices (that corresponds to its natural neighbors). The local neighborhood is thus smoothed when the newly interpolated depth is assigned to v.

In order to ensure that the safety constraint can not be violated, in the function InterpolateCheckDepth (Algorithm 3), the newly interpolated depth is only returned if it is shallower than the current depth at the vertex. If the interpolated depth is deeper than the current depth, the current depth is returned.

Algorithm 2 The Laplace interpolant

```
    Input: a vertex $v$ that is to be estimated in the DT
    Output: the Laplace interpolated depth $h$ for that vertex $v$
 1: function InterpolateDepth(vertex $v$)
 2:     for all natural neighbours $v_i$ around $v$ do
 3:         $e_1 \leftarrow edge(v, v_i)$
 4:         $e_2 \leftarrow dual(e_1)$
 5:         $w_i \leftarrow \frac{length(e_2)}{length(e_1)}$
 6:     end for
 7:     $h \leftarrow 0$
 8:     for all $w_i, h_i$ from the natural neighbours $v_i$ around $v$ do
 9:         $h \leftarrow h + \frac{w_i}{\sum w_i} * h_i$
10:     end for
11:     return $h$
12: end function
```

Algorithm 3 A safe Laplace interpolant

```
    Input: a vertex $v$ that is to be estimated in the DT
    Output: the safely constrained Laplace interpolated depth $h$ for
    that vertex
 1: function InterpolateCheckDepth(vertex $v$)
 2:     $h \leftarrow$ InterpolateDepth($v$)
 3:     if $h$ shallower than current depth at $v$ then
 4:         return $h$
 5:     else
 6:         return current depth at $v$
 7:     end if
 8: end function
```

InterpolateCheckDepth, which in turn calls InterpolateDepth, is always used when the depth of an existing vertex of the TIN is to be altered. Only in cases where new vertices are inserted (this is only the case for densification), the function InterpolateDepth is directly used, to assign those vertices an initial depth in accordance with the conceptual surface.

In the following paragraphs I introduce three functions that operate on the surface, these are *smoothing*, *reshaping* and *densification*.

4.3.1 Smoothing

The operator Smooth (Algorithm 4) is the most trivial application of the InterpolateCheckDepth function, it simply calls that function for all input vertices, after which their depths are updated (see Figure 33). Thus, smoothing does not change the planimetric coordinates of vertices, it only lifts vertex depths. It can be performed either on a portion of a dataset or the whole dataset. Furthermore this operator can be applied any number of times, delivering more generalization with each pass. The reader is referred to Appendix D for a visual impression on how smoothing affects the surface.

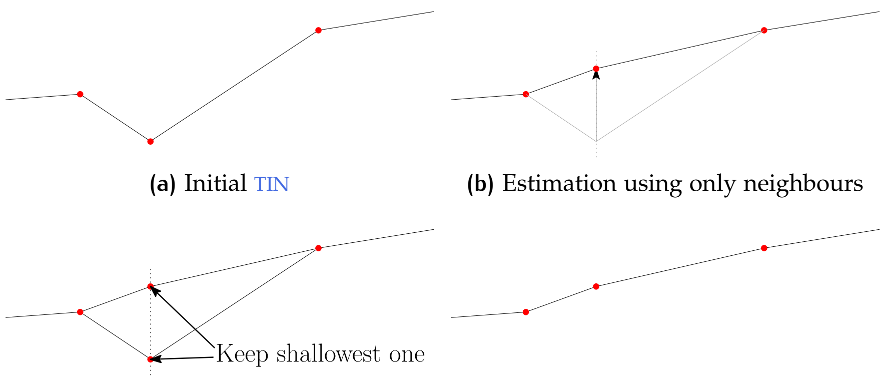
Algorithm 4 The smoothing operator

```
   Input: a DT
   Output: a smoothed DT
1: function SMOOTH(vertices $V$)
2:     for all vertices $v_i \in V$ do                ▷ Interpolate all depths
3:         $h_i \leftarrow$ INTERPOLATECHECKDEPTH($v_i$)
4:     end for

5:     for all tuples $v_i, h_i$ do              ▷ Now apply changes to DT
6:         update depth of $v_i$ with $h_i$
7:     end for
8: end function
```

The primary objective of smoothing is to generalize the surface by removing high frequency detail, while preserving the overall feature shape. This both reduces the angle between the planes spanned by adjacent triangles, which is analogues to the line based smoothing operator (where the angle between adjacent line segments is reduced), and simplifies overall shape, which is analogues to the object based simplification operator, except that the number of points stays the same. Thus, with some minor reservations, in terms of the object based generalization operators of § 2.5.2, Smooth performs:

- smoothing and



(a) Initial TIN
(b) Estimation using only neighbours
(c) Comparison of depths
(d) Resulting TIN

Figure 33: Smoothing of a single vertex for the one-dimensional case.

- simplification.

A call to Smooth performs two linear loops over the n vertices of the TIN, therefore the algorithm is $O(n)$.

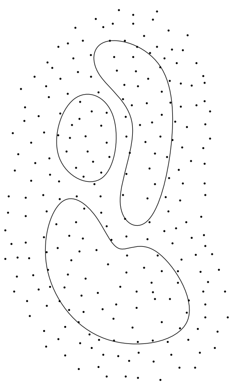
4.3.2 Reshaping

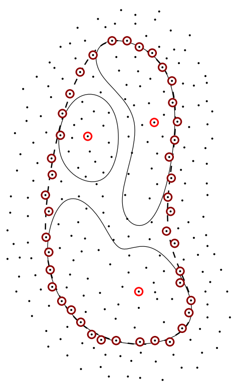
The reshaping operator aims to perform specific and aggressive surface modifications where needed. It is also based on InterpolateCheckDepth, but other than with Smooth prior to updating the depth of a vertex, the set of input vertices V are temporarily removed from the DT. After that, each input vertex $v \in V$ is individually re-added to the triangulation, assigned a depth of InterpolateCheckDepth(v) and removed again. After doing that for every input vertex $v \in V$, they are permanently put back into the triangulation, but now with new depths that can be significantly different. Observe the function Reshape (Algorithm 5).

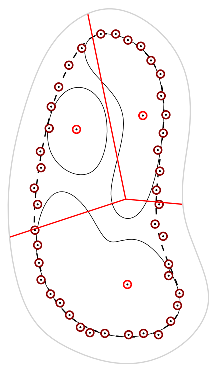
Reshape should be considered as a mechanism to perform different kinds of generalization that require specific and vigorous alteration of the surface that can not be achieved through Smooth alone. In terms of the object (contour) based operators listed in § 2.5.2, this includes:

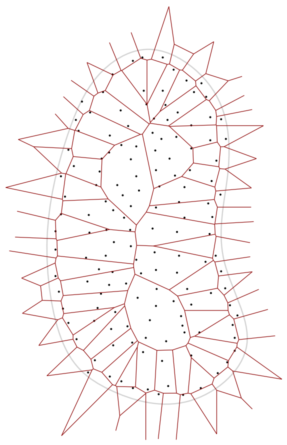
- Omission,
- enlargement and
- aggregation.

In all cases the operator Reshape is to be performed once for every (group of) contour feature(s) that require such cartographic generalization. Figure 34 illustrates the process of aggregation using Reshape in more detail. Prior to calling Reshape, the relevant feature(s) need to be identified (which is currently done manually) and a set


(a) Initial situation: a group of nearby features


(b) Selection of primary (light red) and secondary (dark red) steering points.


(c) Depth-estimation of secondary steering points using a VD of only primary steering points.


(d) Depth-estimation of non-steering points inside buffer using VD of steering points and other points outside the buffer.

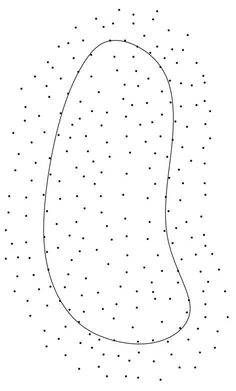

(e) The result; an aggregated feature

Figure 34: Aggregation on the surface using the reshaping operator.

Algorithm 5 The reshaping operator

```
    Input: a set of vertices $V$ that are to be reshaped
    Output: a reshaped region in the DT
 1: function RESHAPE(vertices $V$)
 2:     $vertexCache \leftarrow [\,]$
 3:     for all vertices $v \in V$ do
 4:         append $v$ to $vertexCache$
 5:         remove $v$ from triangulation
 6:     end for

 7:     for all vertices $v$ in $vertexCache$ do
 8:         insert $v$ in triangulation
 9:         update depth of $v$ with INTERPOLATECHECKDEPTH($v$)
10:         remove $v$ from triangulation
11:     end for

12:     for all vertices $v$ in $vertexCache$ do
13:         insert $v$ in triangulation
14:     end for
15: end function
```

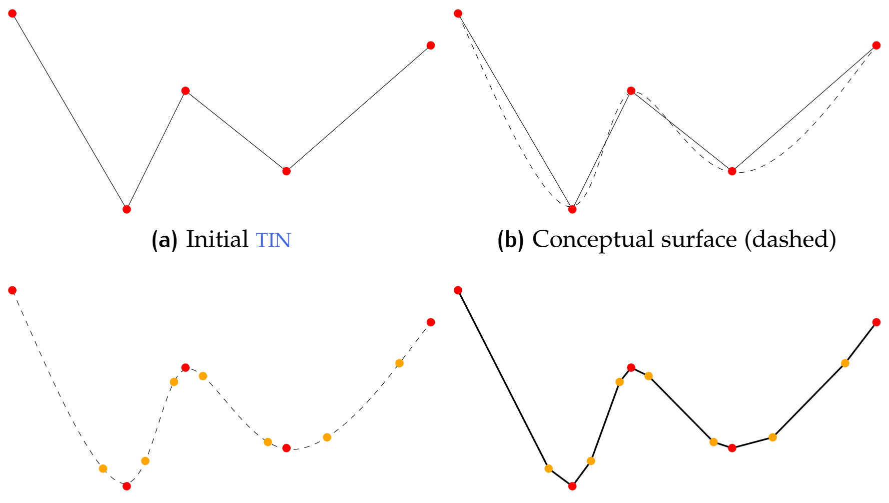
of so-called *steering points* needs to be assigned, these *steer* the process of locally reshaping the surface. Figure 34a depicts two kinds of steering points. A *primary* steering point, highlighted with a light red circle, is a local maximum of a feature. In case of aggregation, the *secondary* steering points are vertices that are on the the convex hull (or alpha-shape) of the features that are to be aggregated, drawn in dark red in Figure 34b. These secondary steering points are assigned a (shallower) depth that follows from a natural neighbor interpolation that includes only the primary steering points (see Figure 34c). Finally, and as depicted in Figure 34d, the RESHAPE operator is called using all vertices that are inside a buffer around the earlier described convex hull, but excluding the steering points. The result, shown in Figure 34e, is a locally reshaped surface with boundaries that closely follow the convex hull or an alpha-shape of the initial features. Of course, all general benefits from the Voronoi- and Surface based approach apply, meaning that the resulting surface is guaranteed to be both smooth and safe.

In case of enlargement, the secondary steering points should be picked along a circle centered at the primary steering point (local maximum). And in case of omitting a pit, no steering point need to be assigned, it suffices to simply call RESHAPE on *all* the vertices that define the feature that is to be omitted.

Note that a call to Reshape only performs changes to a very specific region of n_r vertices that is a subset of all the n vertices in the DT. Over those n_r points the scalability of the algorithm is $O(n_r)$.

4.3.3 Densification

The objective of densification is primarily to minimize the discretization error in the contours that are extracted from the Delaunay triangulation. By inserting vertices in large triangles, the resolution of the Delaunay triangulation is improved. As a result also the extracted contour lines have smoother appearance because they now have shorter line-segments. These newly inserted vertices are assigned a depth using InterpolateDepth (Algorithm 6) at the center of the circumscribed circle (that is equivalent to a node in the Voronoi diagram). The circumcenter is chosen here because that location is equidistant to its three closest points, and subsequently results in a very natural point distribution. In case of two co-circular triangles, only one vertex needs to be inserted for both triangles (see Figure 10, p. 17).



(a) Initial TIN **(b)** Conceptual surface (dashed)

(c) Interpolate at intermediate points **(d)** Resulting TIN

Figure 35: The densification operator for the one-dimensional case.

Considering that the conceptual surface is defined as the Laplace interpolated field, densification does not actually change that conceptual surface, instead it aims to improve the approximation of the conceptual surface—and, with that also the resolution of the extracted contours—i.e. the TIN. Therefore, densification is to be applied just before the extraction of the depth-contours. If applied *before* e.g. the Smooth operator, it would limit the effectiveness of that operator, since a denser triangulation smoothes more slowly. The Reshape operator would in principle be equally effective, but more points would need to be processed so there is no reason to do this either.

Algorithm 6 The densification operator

```
   Input: a DT
   Output: the densified DT
1: function DENSIFY(triangulation $T$, threshold $maxArea$)
2:     for all triangles $t \in T$ do
3:         if $area(t) > maxArea$ then
4:             insert vertex $v$ in triangulation at $circumcenter(t)$
5:             set depth of $v$ to INTERPOLATEDEPTH($v$)
6:         end if
7:     end for
8: end function
```

The DENSIFY operator as presented in Algorithm 6 uses an area-threshold that determines which triangles are densified. This way triangles that are already sufficiently small are not densified. It performs a single pass on the input triangles, thus with every call to DENSIFY the resolution of Delaunay triangulation is increased, until all triangles have reached a certain area. Alternatively, a recursive approach could be considered where every triangle is densified until the user defined triangle area is reached.

If the maximum area threshold is ignored, a single call to DENSIFY is $O(n)$, as it only requires a single pass over the n triangles of the TIN. However, when a number of t densification passes is sequentially performed, it only scales to $O(3^t n)$, since every point insertion might create three new triangles. But, because of the maximum area threshold that worst case scenario will never be reached in practice with large t.

4.4 AUTOMATION OF THE SURFACE-BASED APPROACH

The surface based operators proposed in this chapter may perform well, but as pointed out in § 2.5.4 this does not solve the complete automatic generalization problem. Consider for instance the RESHAPE operator, while it might be a very effective operator, the question remains on how and on what features RESHAPE it is to be employed. Of course that could simply be manually decided by a cartographer. Doing this automatically however, would require some form of structural and qualitative feature analysis that is not currently part of the proposed surface-based approach of hydrographic contour line generalization. I do list two plausible approaches of tackling this problem.

Firstly, there is the idea of building a Graphical User Interface (GUI) that usefully assists the cartographer in its generalization task. Recall from § 2.5.4 this is called amplified intelligence (Weibel, 1991). The GUI should be supported by a powerful and automatic structural analysis tool, such as the *feature tree* that is described in Zhang and Guilbert (2011) and Guilbert (2012) and illustrated in Figure 30. Based on that analysis that is either performed on the surface itself or on the extracted contour lines, the system proposes likely generalization actions. It is then up to the cartographer to accept these.

Secondly, a constraint-based approach could be developed that aims to fully automate the generalization task. In addition to a structural analysis tool, this would also require some qualitative analysis tool that judges how well the hydrographic contouring constraints are respected. A system like this should automatically converge to an optimal solution, that depends on a set of formally defined constraints. Zhang and Guilbert (2011) discuss such a system for hydrographic contour line generalization.

Both approaches have in common that they require structural feature analysis, yet in general the second approach seems to be the more challenging. Therefore I would propose a course of action where first the idea based on amplified intelligence is developed and tested, which should result in-depth information of the complete contour generalization process, which can then be used to define the formal constraints and further analysis tools that are required for the constrain-based approach.

4.5 SCALABILITY OF THE SURFACE-BASED APPROACH

Another issue that is not experimented with in this thesis, is the scalability of the surface-based approach in terms of computational and memory resources. However, since all algorithms that are required to construct, manipulate and extract from the surface are local, the surface-based approach should be well scalable. Two plausible ways to achieve that are chunking and streaming.

Chunking means to divide the inputs into smaller portions, that are processed in parallel. When that processing is done they are merged back into the greater dataset. Though, special care is to be paid to the boundaries of the portions.

Streaming works by performing a small number of sequential passes over the input data, and processes the data using a memory buffer whose size is a fraction of the stream length (Isenburg et al., 2006a). While this limits the range of operators that can be applied—they must be local—memory requirements are extremely low. Another benefit is that, since the processed data is immediately outputted, the next processing step can already commence before the current one is finished. As demonstrated by Isenburg et al. (2006b) streaming can be effectively applied to massive point clouds in order to e.g. generate Delaunay Triangulation (DT)s, perform smoothing and extracting (topographic) contour lines.

In principle both approaches could be applied to the surface approach, since all its operations are local. Most promising seems to be the streaming approach of Isenburg et al. (2006a) since a comparable processing pipeline, i.e. that generates contour lines from points through triangulation, was already successfully demonstrated (see Figure 36). It would be trivial to implement the smoothing and densification operators using that approach.

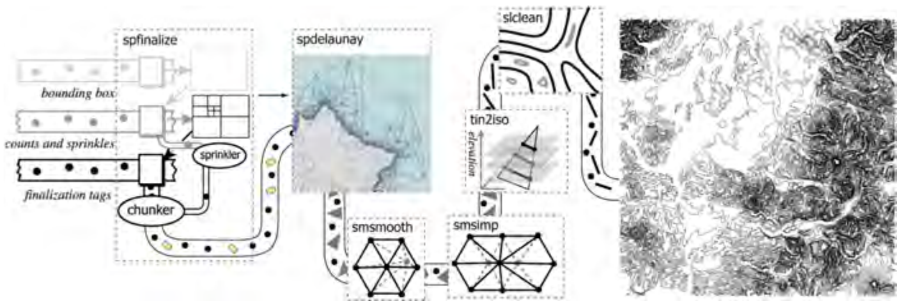


Figure 36: Highly efficient streaming computation of elevation contours from massive point clouds through triangulation. From Isenburg et al. (2006b).

5 IMPLEMENTATION AND EXPERIMENTS

This chapter demonstrates the effectiveness of the Voronoi- and surface-based approach that was introduced in the previous chapter. This is done by running a number of experiments on a set of datasets that are described in § 5.2 and using a prototype software (see §5.1). In § 5.3 metrics are defined that were used to asses the results of the experiments that are presented in § 5.4.

5.1 THE IMPLEMENTED PROTOTYPE

In order to test and analyze the algorithms presented in Chapter 4, I implemented those algorithms in the C++ programming language. The choice for C++ was made because the necessary software libraries were all available in C++ and because it allows for a high level of control on the implementation itself. The following software libraries were directly used:

CGAL: the Computational Geometry Algorithms Library; an extensive and robust computational geometry library. Of main interest for this thesis was its robust data structure for Delaunay Triangulations.

GDAL/OGR: the Geospatial Data Abstraction Library; used for import and export to vector and raster file formats.

GEOS: the Geometry Engine Open Source; primarily used for the merging of contour line segments after extraction from the TIN (as described in section 2.4.2).

All of the described software and tools are Open Source, thus freely available, including the software that I wrote. More details on the implementation of the developed software can be found in Appendix B.

5.2 DATASETS

Thanks to George Spoelstra, from the former company Atlis, a varied selection of datasets were available to use for experimentation. Appendix C lists these datasets in more detail, following are a short description for each dataset:

ANTILLES: a Single Beam Echo Sounding (SBES) dataset. Very sparse and anisotropic points from the deep sea.

AUSTRALIA: a Multi Beam Echo Sounding (MBES) dataset with homogeneous distribution.

LONDON: A portion of the Thames river in London. A mixture of SBES and MBES data (I call this heterogeneous).

ZEELAND: A very dense MBES dataset from the river Westerschelde in the Dutch province Zeeland. Its wavy sandbank patterns are a notable feature.

5.3 METRICS

As described in Chapter 1, a well-defined set of formal metrics to measure the quality of a map does not exist. However, it is possible to measure and quantify some of the underlying processes and characteristics. For the analysis of the results presented in this chapter, I introduce and apply the following formal metrics.

5.3.1 The root mean square of differences

The Root Mean Square (RMS) of differences is used to measure the difference between two fields. Consider two rasters R^a and R^b that both represent a field and their raster cells h_i^a and h_i^b, where i is the zero-based cell index. The cells of R^a and R^b are equal in number and geographical extent. If n is the total number of cells per raster, the RMS of differences between R^a and R^b is calculated as:

$$e_{R^a - R^b} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} \left(h_i^a - h_i^b\right)^2} \tag{11}$$

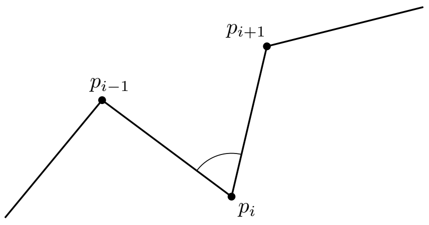
This measure is used to quantify the change in the surface as a result of generalization operations. In terms of the generalization constraints of the hydrographic chart, this is especially relevant for the morphology.

5.3.2 Angularity

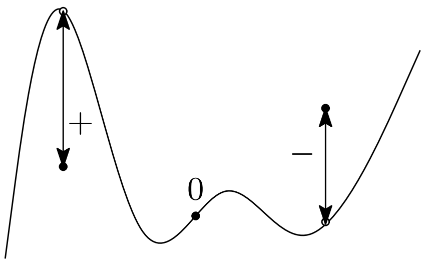
Of main importance for the legibility of the chart, is the smoothness of the contour lines. Since smoothing is defined as reducing the angularity of a line, I look at the angles between the contour line segments to say something about the smoothness of a contour line. I define the angularity α_i of a point p_i on the polyline L:

$$\alpha_i = \pi - \angle p_{i-1} p_i p_{i+1} \tag{12}$$

That is π minus the angle—it is defined on the interval $[0, \pi]$—of p_i with its two neighbors (see Figure 37a). Thus, a high angularity



(a) The angle $\angle p_{i-1}p_ip_{i+1}$ is defined as the smallest angle of point p_i with it neighbors.



(b) One-dimensional view of point differences. A sample point that lies above the field (indicated by the curve) has a negative point difference and is unsafe.

Figure 37: Metrics used for analysis

α_i means that the point p_i has a small angle. For quantifying the angularity of the complete polyline L, I simply use the average of the angularities of its n number of points:

$$\alpha_L = \frac{1}{n}\sum_{i=0}^{n-1}\alpha_i \tag{13}$$

The angularity is used to analyze how the smoothness—and therefore also the legibility—of a contour line changes as a result of generalization operations.

Note that the angularity of a polyline can be reduced by simply adding points on straight line segments, therefore I only use the angularity to compare lines that have the same number of points.

5.3.3 Point differences

The point difference of a sample point p_i with depth h_i with respect to the interpolated depth field F is defined as:

$$\delta_i = F(p_i) - h_i \tag{14}$$

As can be observed from Figure 37b, a negative value of δ_i therefore means that the depth h_i lays above the depth field F, in other words the safety constraint is violated by F. Point differences can therefore be used to verify if a field is safe.

5.4 EXPERIMENTS

The following paragraphs describe and analyze the experiments that were performed to analyze the effectiveness of the Voronoi- and surface-based approach. § 5.4.1 investigates the smoothing operator, § 5.4.2 the reshaping operator and § 5.4.3 the densification operator. In

§ 5.4.4 it is investigated how respecting the safety constraint affects the generalization result and § 5.4.5 focusses specifically on the performance with a heterogeneous input. Finally, § 5.4.6 compares the contours of different current approaches in hydrographic contouring with those from the Voronoi- and surface-based approach.

5.4.1 The smoothing operator

In terms of the object based generalization operators, the aim of the surface-based smoothing operator is to perform smoothing, simplification, aggregation and omission. The effectiveness of the smoothing operator is therefore demonstrated on a region that has very irregular and cluttered contour lines: sandbanks in the Zeeland dataset. As can be observed from Figure 40a, the raw and ungeneralized contours indeed meet those criteria. However the smoothed contours from Figure 40b have a much cleaner and less cluttered appearance. Clearly, the number of contour features has dropped. This is both because pits (local minima) have been lifted upwards by the smoothing operator, and nearby peaks have been aggregated because the region in-between has been lifted upwards. Thus omission and aggregation take place in the contour lines, this is highlighted by respectively the blue and green ellipses in Figure 40. A third effect of the smoothing operator is the enlargement of certain features as a result the uplifting of points surrounding a local maximum. This is highlighted by the red ellipse in Figure 40.

The effect on the morphology of the surface is illustrated by Figures 38 and 41. Figure 41a shows the initial Laplace interpolated field, notice the high frequency sandbank patterns in the top right of the image. After smoothing these high frequency patterns have been removed (see Figure 41b), while the general surface shape (low frequency pattern) has been preserved. This observation is supported by the difference map of Figure 41c since those high frequency patterns are also clearly visibly there.

As to be expected, the overall effect of the smoothing operator on the morphology of the surface is significant. As shown by the plot of RMS differences between the initial and the generalized surface in Figure 38 the smoothing of the surface results in a per pixel difference of several tens of centimeters. That supports the idea that the smoothing the surface works against the preservation of *all* morphological features. However by Figure 41 it is also shown that especially the high frequency features are altered, thus even though the surface is significantly altered this is done a meaningful way. Another observation of Figure 38, is that the steepness of the plotted line slowly decays. That indicates that the smoothing operator is especially effective with the earliest smoothing passes and that the amount of surface alteration decreases in the later smoothing passes.

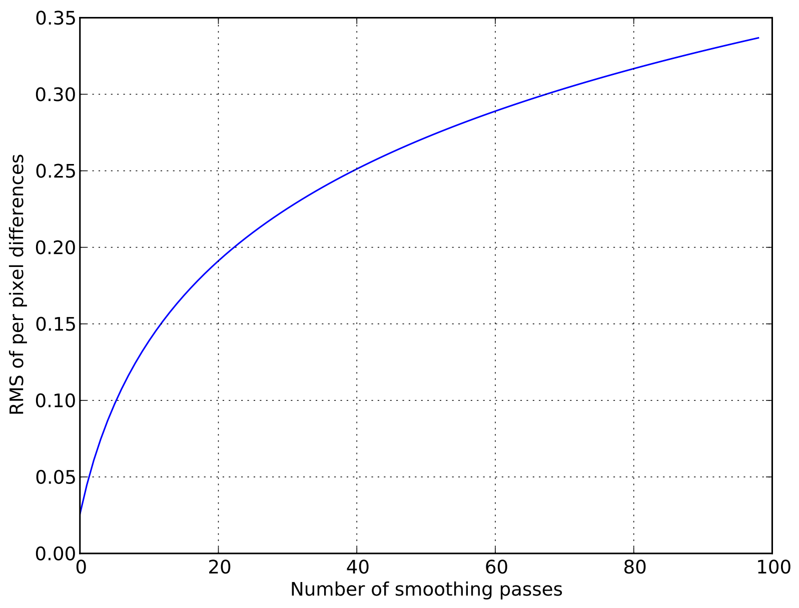


Figure 38: RMS of differences with respect to the initial conceptual surface as a function of the number of smoothing passes. Dataset Zeeland.

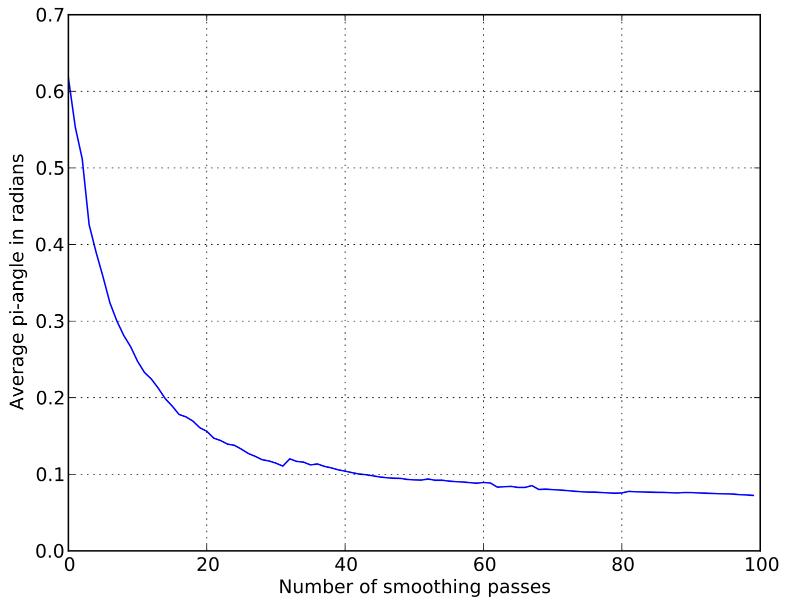
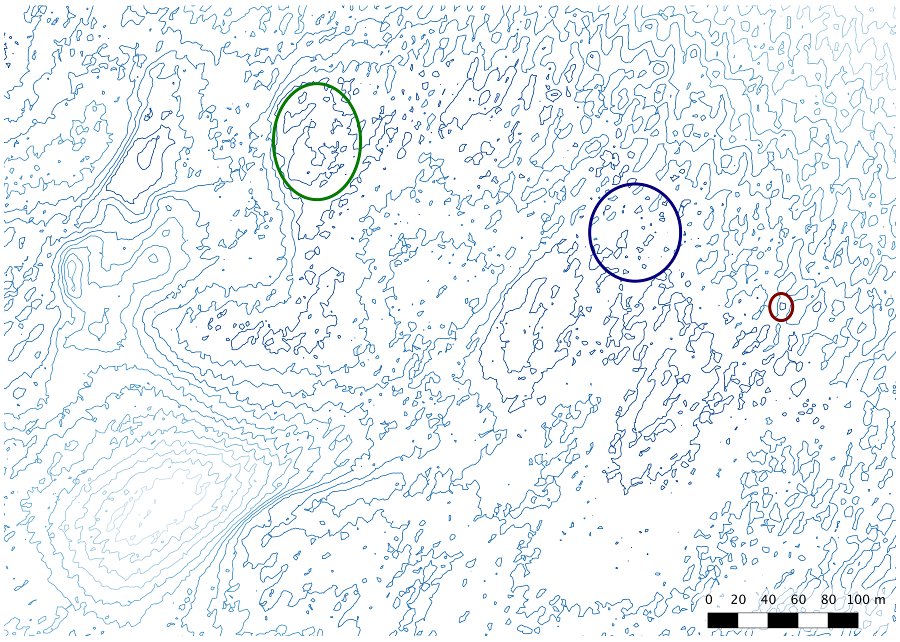
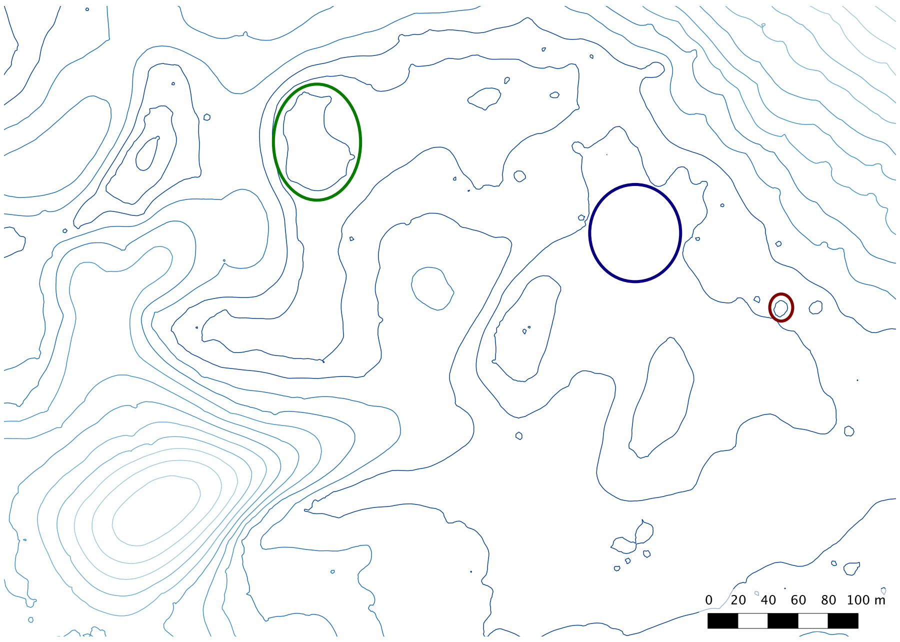


Figure 39: Line angularity for the contour of Figure 42 as a function of the number of smoothing passes. Dataset Zeeland(b)

Naturally, the smoothing operator also smoothes and simplifies the resulting contour lines. This is demonstrated in Figures 39 and 42. Figure 42a illustrates the effect of the smoothing operator on a sin-

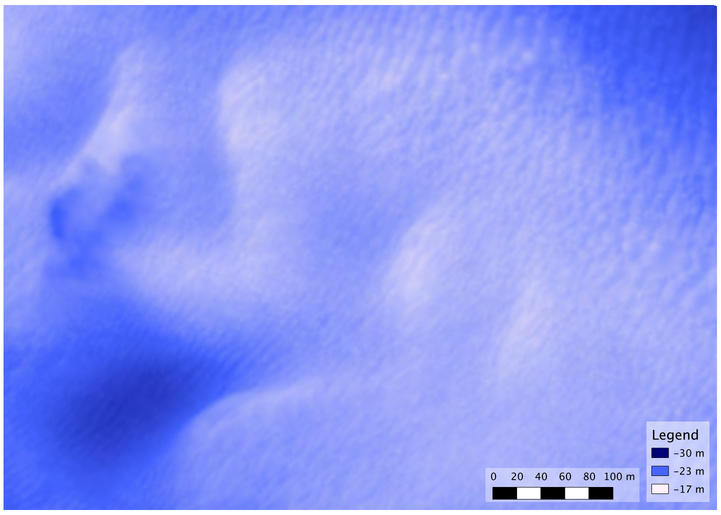


(a) Raw contours

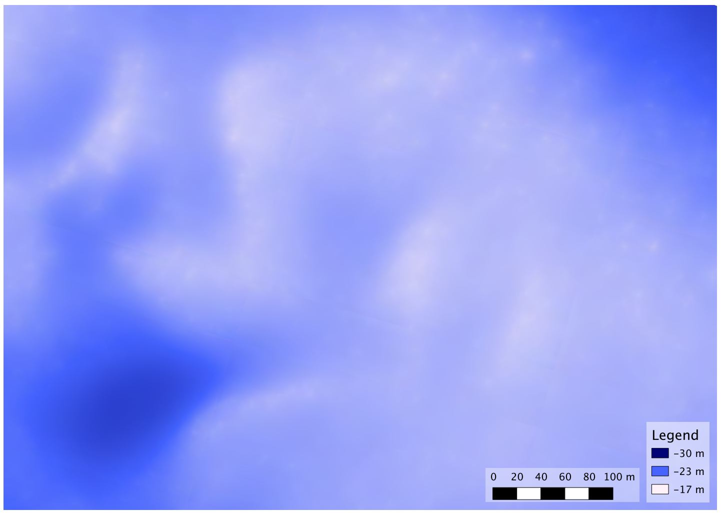


(b) 100x Smoothed contours

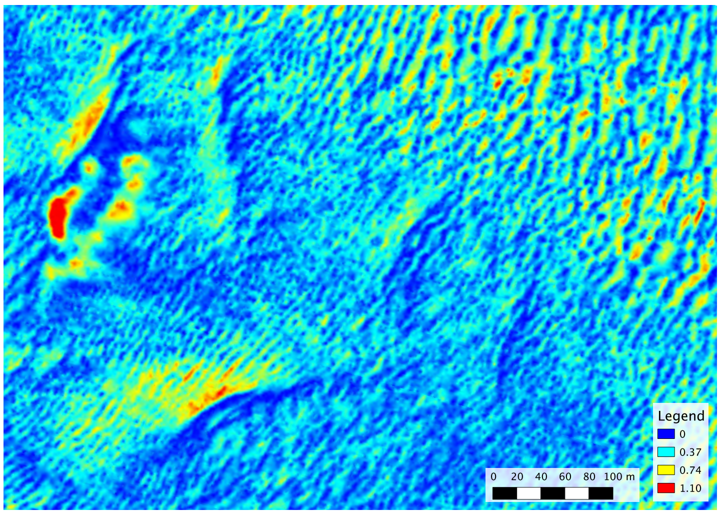
Figure 40: The effect of the smoothing operator on the extracted contours (at every 50cm). The ellipses mark areas where aggregation (green), omission (blue) and enlargement (red) take place. Dataset Zeeland(a).



(a) Initial conceptual surface



(b) Smoothed conceptual surface (100x)



(c) Difference map

Figure 41: The effect of the smoothing operator on the conceptual surface. Dataset Zeeland.

gle contour over 30 smoothing passes. It is clear that the contour line moves towards the inner region, which is the deeper side of the contour, which is to be expected since the smoothing operator is safe per definition (and only lifts the surface upwards). What can also be seen is that the line is simplified (the details on the outer rim disappear, note however that the point count stays the same) and smoothed. That is further supported by Figures 42b and 42c that shows the point angularity before and after smoothing. It is evident that the number of points with a medium to high angularity are significantly reduced, which also corresponds to a smoother and more simplified contour line.

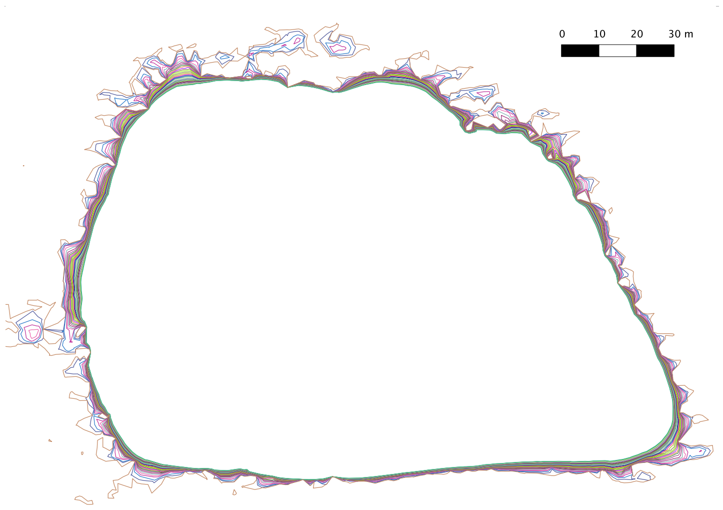
Figure 39 clearly shows that the line angularity decreases with more smoothing passes. Again the slope is the strongest with the earliest smoothing passes, further supporting the observation that the earliest smoothing passes are the most effective. After around 30 smoothing passes the line hardly becomes any smoother at all. So at that point and at least for this particular contour line there is not much reason to smoothen further.

5.4.2 The reshaping operator

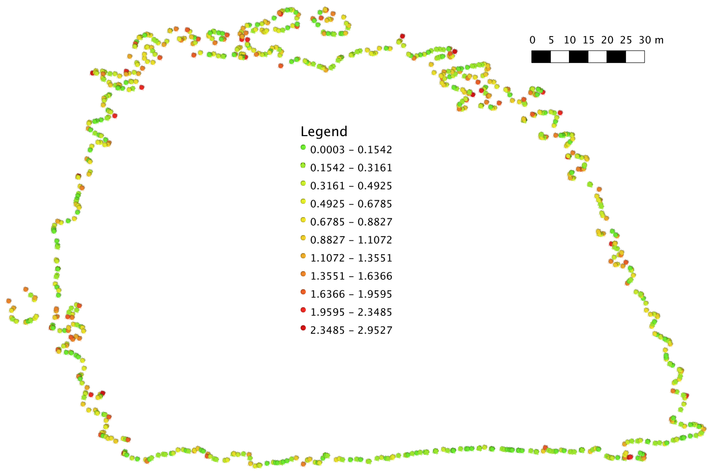
The aim of the reshaping operator is to make vigorous changes to the surface where that is wished from a cartographic point of view and not obtained by the smoothing operator. Its effectiveness is demonstrated by employing the reshaping operator on the two peaks shown in Figure 43b. As described in § 4.3.2, prior to performing the reshape operator the steering points need to be set. The primary steering points are highlighted with yellow Voronoi cells in Figure 43a. The secondary steering points are the other non-yellow points in the center of the image. These are assigned new depth values similar to the primary steering points, as indicated by the color of the corresponding Voronoi cells in Figure 43c, where also the input points of the reshape operator are removed from the VD. The result, the aggregated peaks, are shown in Figure 43d, which demonstrates that the reshape operator performs exactly as expected.
Figure 44 shows exactly how the surface was raised to aggregate the two peaks. Evidently, the region in between the peaks is significantly raised such that it has the same depth as the original peaks. In other words: aggregation is performed.

5.4.3 Densification

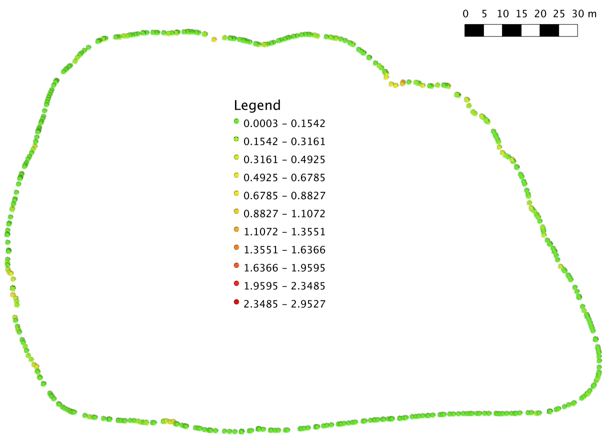
The goal of the densification operator is to improve the TIN approximation of the conceptual surface. The effect is illustrated in Figure 46. Shown are the raw, undensified, contour line in red, the densified contour line in green and a number of Single Beam Echo Sounding (SBES)



(a) From 0x smoothing (outer) to 30x smoothing (inner)

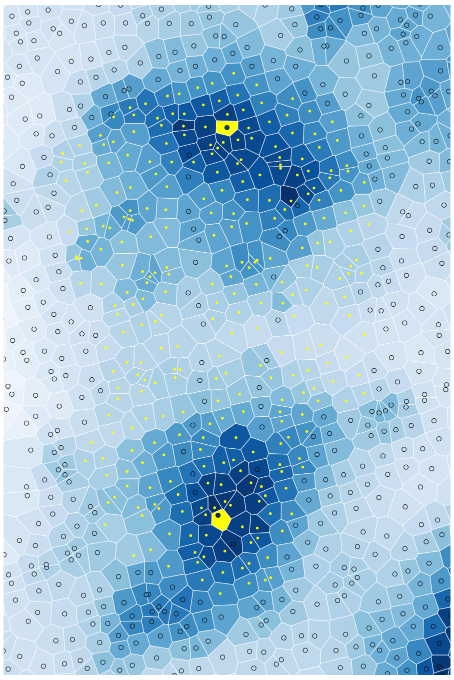


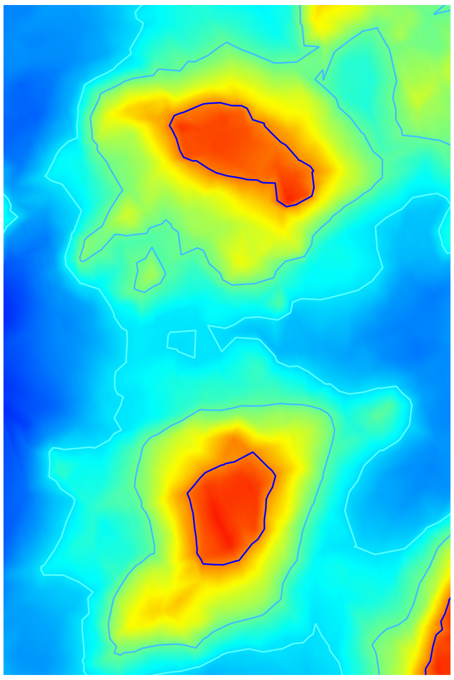
(b) Point angularity in radians at 1x smoothing

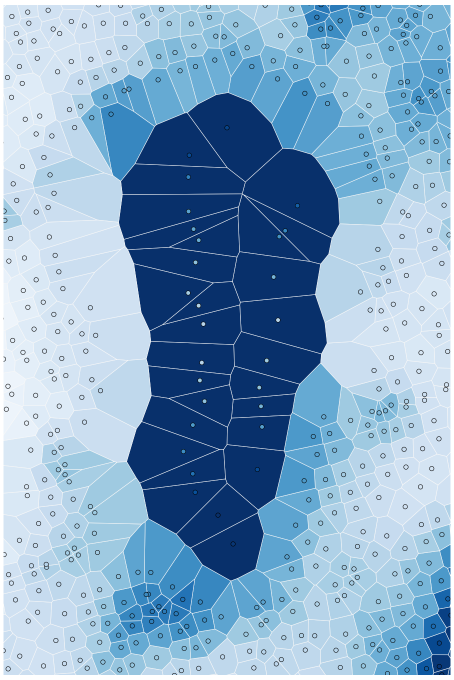


(c) Point angularity in radians at 30x smoothing

Figure 42: The effect of the smoothing operator on a single contour. Dataset Zeeland(b).


(a) Voronoi diagram of the sample points. The Voronoi cells of the primary steering points are highlighted in yellow. The yellow points are the points that are to be reshaped.


(b) The initial conceptual surface with corresponding contours


(c) Voronoi diagram and remaining points after removal of the points that are being reshaped.

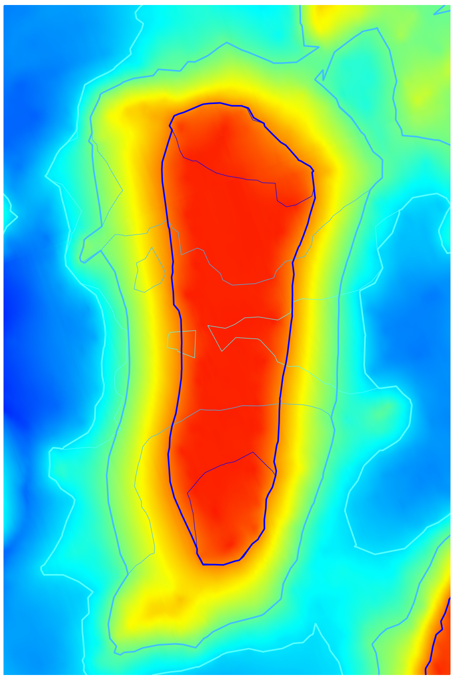

(d) The conceptual surface after reshaping. Both the initial (thin line) as the reshaped contours (fat line) are shown.

Figure 43: The reshaping operator performs aggregation. Before (upper part) and after (lower part). Dataset Australia(a).

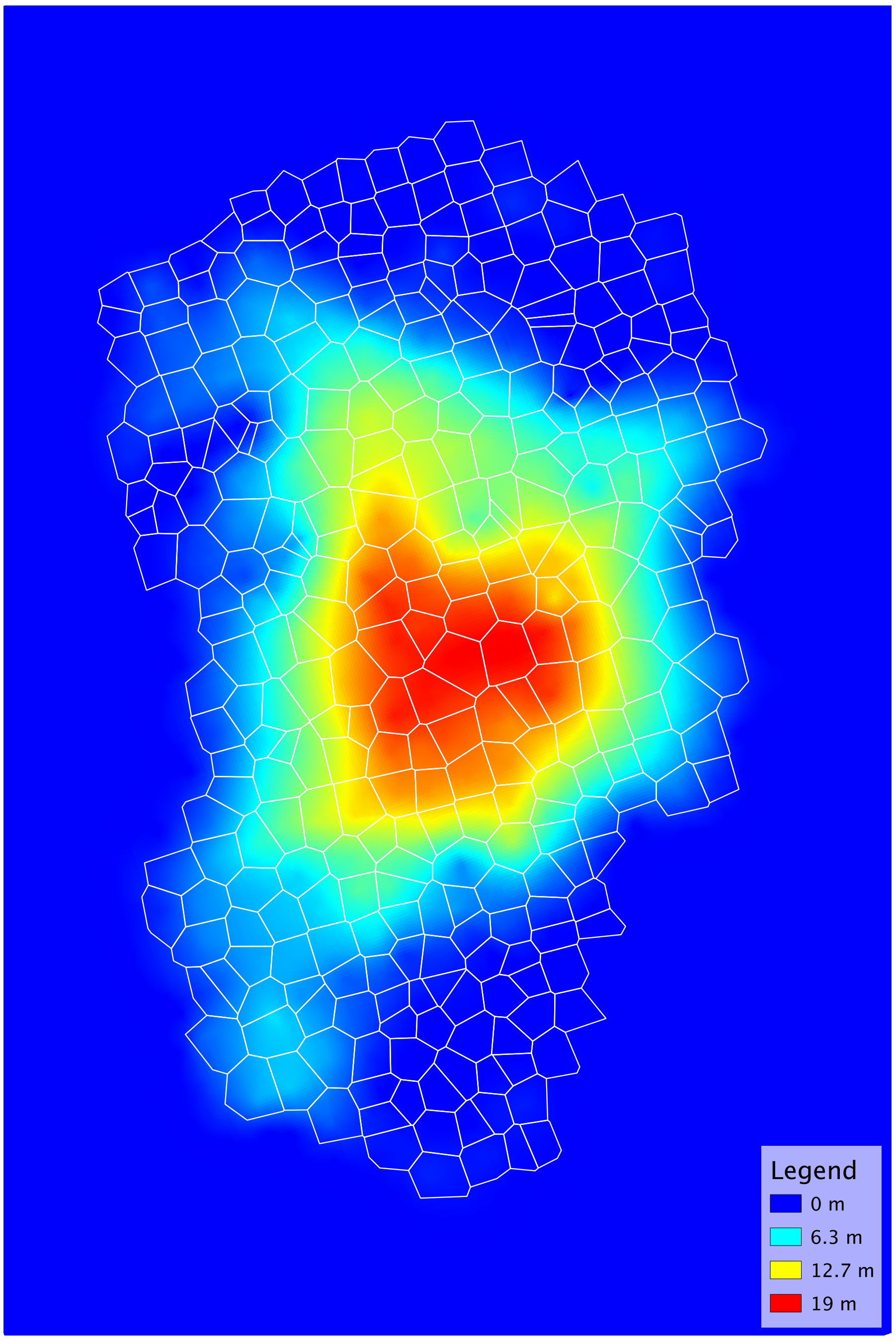


Figure 44: Difference map of Figure 43b and Figure 43d. In white lines the VD of the reshaped region is shown. Dataset Australia(a).

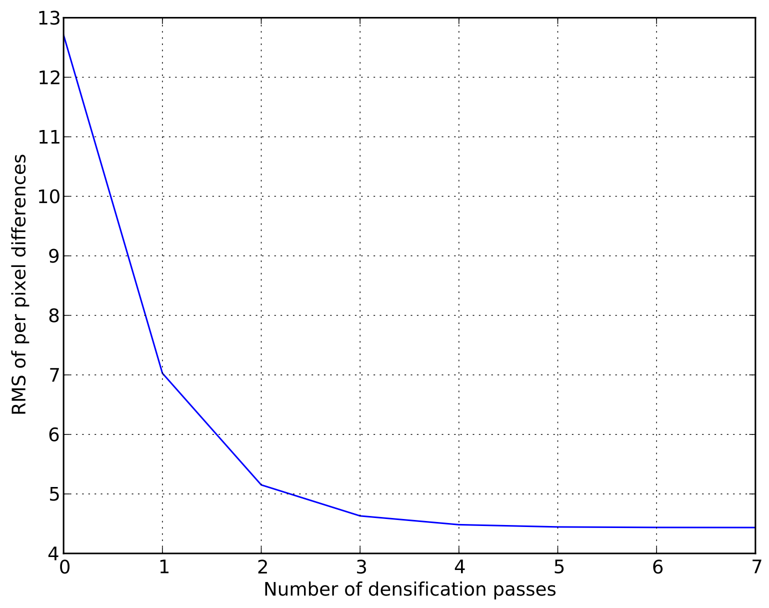
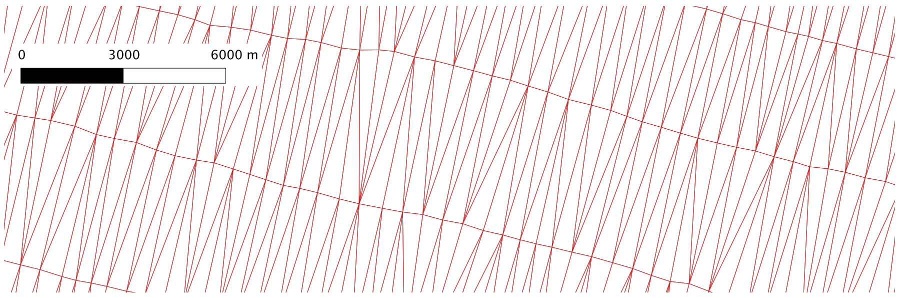


Figure 45: Approximation error of TIN interpolation with respect to the initial the conceptual surface as a function of the number of densification passes. Dataset Antilles(a).

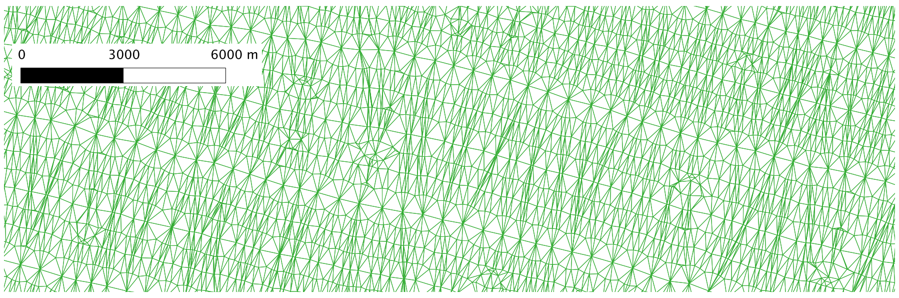
tracks. The sharp edges of the undensified line are caused by the large triangles in the initial TIN (see Figure 46a), however after densification these large triangles are subdivided into much smaller ones (see Figure 46b). The result is a much smoother and more realistic contour line. Figure 45 shows how the approximation error of the TIN with respect to the conceptual surface improves with more densification passes for the same region. After about five densification passes the approximation error stays practically constant. The remaining approximation error in this case is due to the heterogeneous nature of the input points which is further explained by Figure 47d that shows a difference map between the conceptual surface (Figure 47c) and the 7x densified TIN (Figure 47b). Visible are parabolic patterns, centered at the sample points that are visualized in overlay. It is quite interesting to see that these patterns or interpolation artifacts are not present in the densified TIN (compare Figure 47b to Figure 47c), which gives the densified TIN a more natural look (since those parabolic patterns are unlikely to be present in reality) when compared to the true Laplace interpolation (the conceptual surface).

5.4.4 The price of being safe

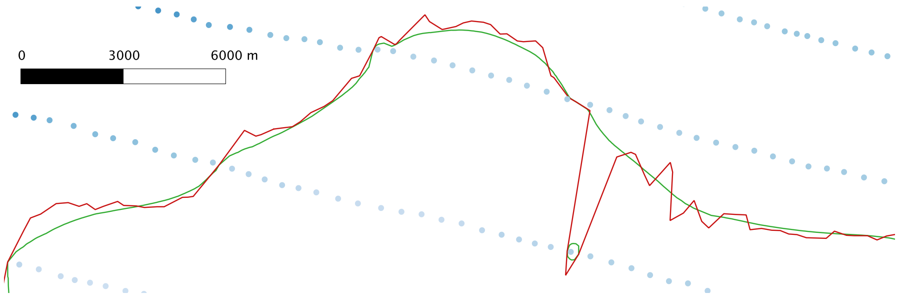
The proposed surface-based operators are all safe by definition. However, the safety constraint has implications for the other hydrographic chart constraints on legibility and morphology. This was investigated



(a) TIN before densification



(b) TIN after 3x densification



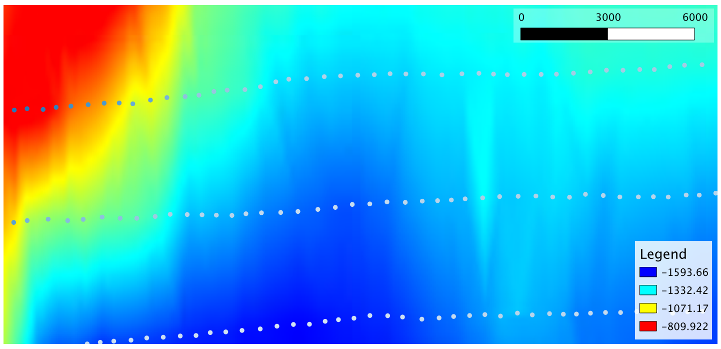
(c) Comparison of corresponding contour lines. The red line is not densified, the green line is densified 3 times. Also shown are the sample points in blue.

Figure 46: Densification illustrated on a $-1350m$ contour. Dataset Antilles(a).

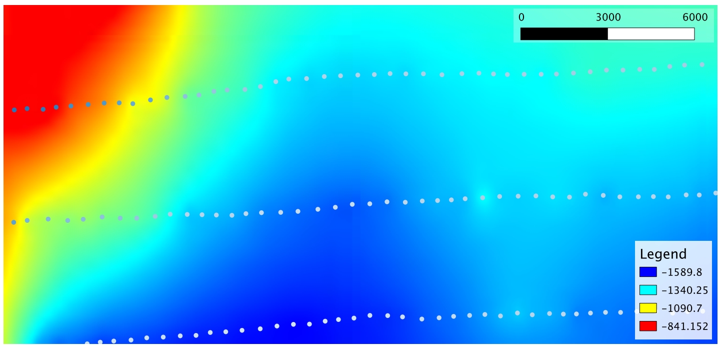
by removing the safety constraint from the operators, i.e. by replacing every call to InterpolateCheckDepth with a call to InterpolateDepth in the algorithms of Chapter 4. The result is shown in Figure 48. Clearly the green and *unsafe* lines both have a more simplified appearance and are positioned closer to the raw contours than the red and *safe* contours. The former implies better legibility and the latter implies that the morphology constraint is respected better by the unsafe contours. This not unexpected, but it should be taken into consideration when comparing to other (unsafe) methods.

5.4.5 Heterogeneous data

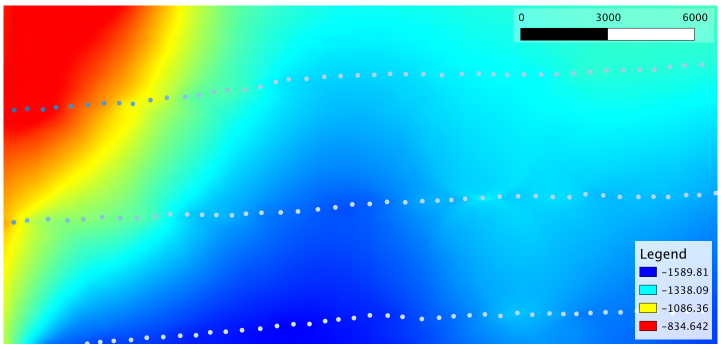
One of the sub research questions relates to the performance of the proposed Voronoi- and surface-based approach with respect to het-



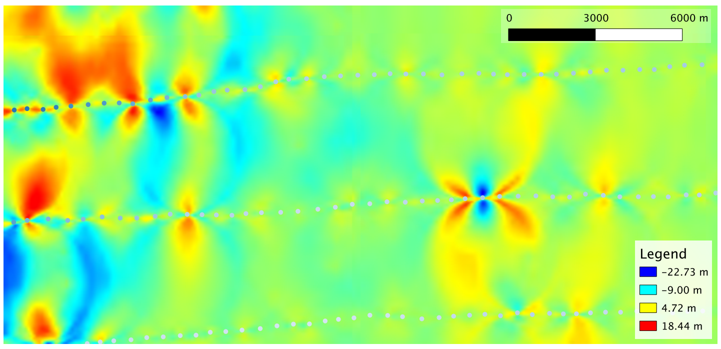
(a) TIN interpolation (0x densified)



(b) TIN interpolation (7x densified)

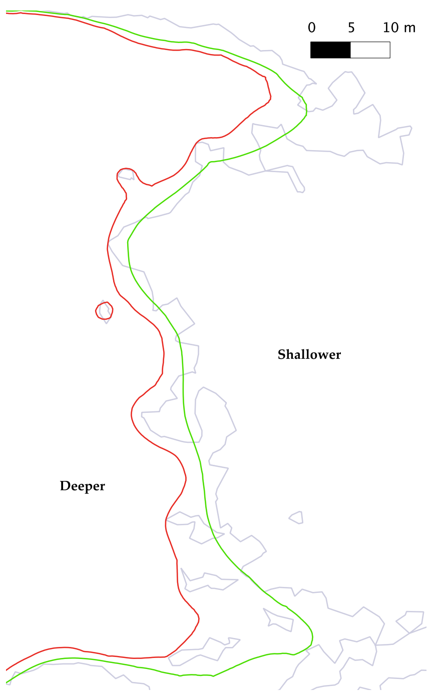


(c) Laplace interpolation

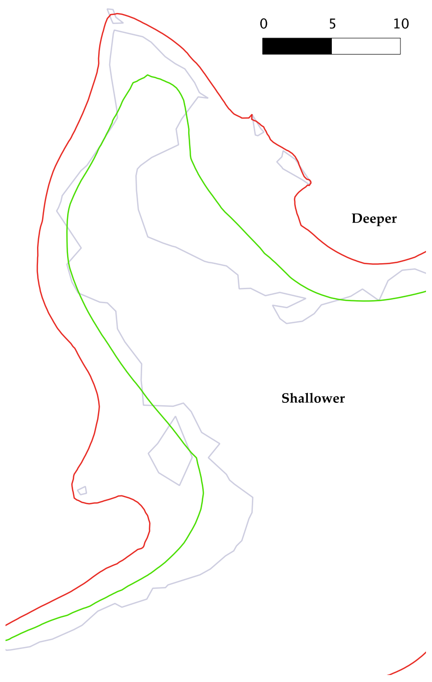


(d) Laplace interpolation − TIN interpolation (7x densified)

Figure 47: Comparison of the (densified) TIN interpolated field with conceptual surface. In overlay are the sample points. Dataset Antilles(a).



(a) Dataset London(a).



(b) Dataset London(b).

Figure 48: The effect of incorporating the safety constraint into the Laplace interpolant. Shown are the direct TIN interpolated contours (grey), the unsafely generalized contours (green) and the safely generalized contours (red). In both cases 10x smoothing and 3x densification was applied.

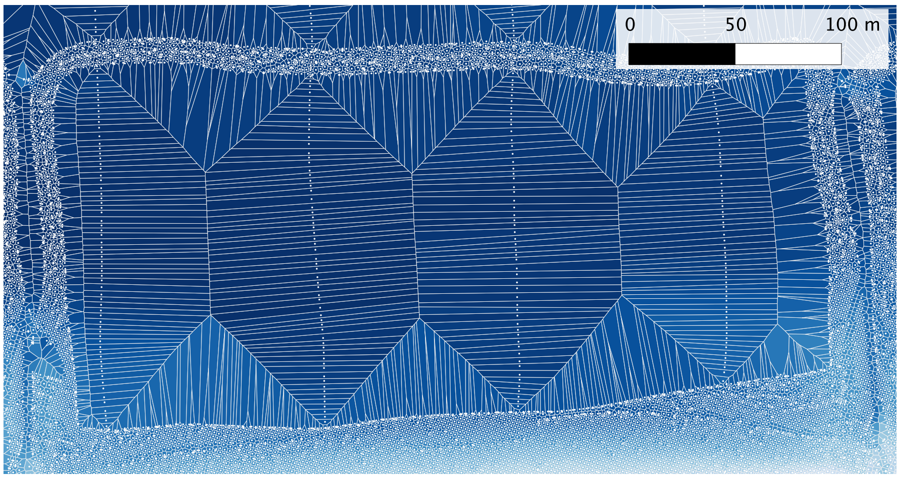
erogeneously distributed sample points. A detail of such a dataset is shown in Figure 49a, it shows the sample points and the VD of those points colored with depths. This figure illustrates why many distance-based interpolation methods have problems with such a heterogeneous data distribution. In general such interpolation methods, including IDW and the Laplace interpolant assume a homogeneous point distribution where sample points from every direction have equal importance. However, in the case of Figure 49a that assumption is invalid, because of the vertical single beam tracks that have a wide horizontal spacing. To compensate for the problematic sample distribution in such an area, the interpolant should assign higher importance to points that are along the axis through the interpolation location and perpendicular to the single beam tracks. That is what a cartographer would naturally do when drawing contours.

With that in mind, observe Figures 49b and 49c. In both cases there is clearly an overly strong influence on the contour lines by the very dense region of points on the bottom and top part of the images. In case of the Laplace interpolation this translates to contours that are somewhat unrealistically bent toward the inner part of the image, i.e. the horizontal centerline. And in case of IDW interpolation it translates to the appearance of entirely unrealistic new local maxima. Therefore I believe that, while still not perfect, the Laplace interpolation does a significantly better job in this case.

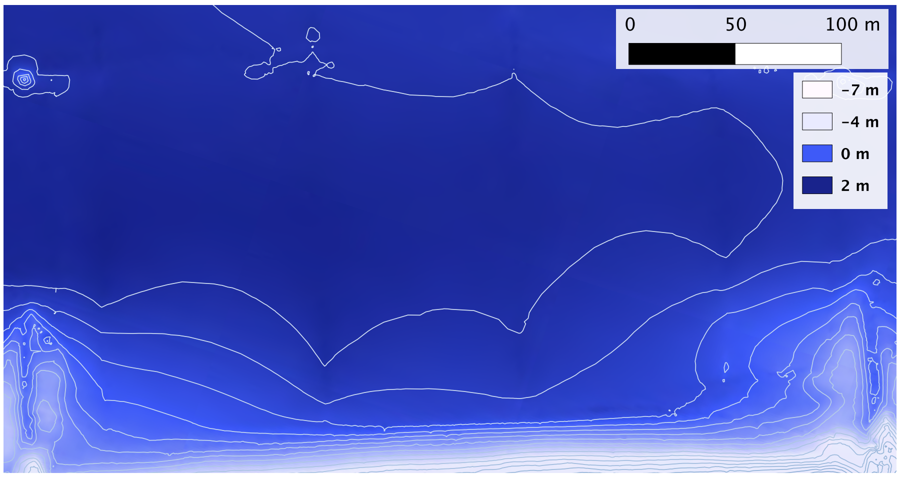
Another point of interest here is the adaptiveness of the interpolation methods. Since Laplace interpolation is fully adaptive to the density of input samples there is no distance parameter that needs to be set, that means that it can easily be applied on datasets with strongly varying density in input samples. But in case of IDW, there is a distance parameter, i.e. the radius of the search ellipse (see § 2.3.2). As a result, in order obtain an interpolation result that covers the entire region of Figure 49c, the radius was set to 150 meters. That kind of radius gives much more cluttered contours in the lower part of the image, and it results in a stronger effect of averaging, which plays against both the safety and the morphology constraints. The Laplace interpolant has none of those problems, making it the better choice for data with varying point densities.

5.4.6 Comparision with existing methods

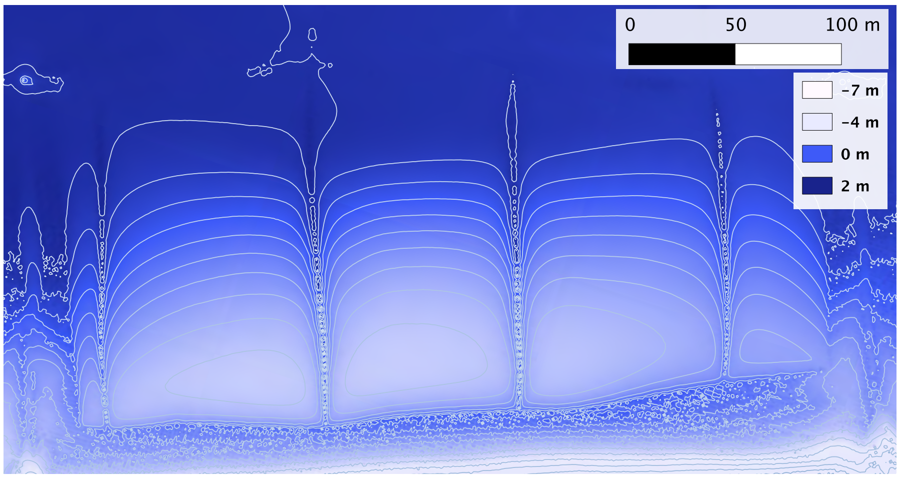
As pointed out in Chapter 3, the use of Inverse Distance Weighting (IDW) interpolation in combination with a raster which is commonly done in practice therefore not safe. As a consequence, the box plot of point differences in Figure 50 has a minimum value below zero. With different parameters for the IDW interpolation, i.e. a larger radius or a lower power, the safety constraint is violated even further. Yet with the used parameters (radius=5m and power=2) roughly half



(a) Voronoi Diagram colored with depths (darker blue is shallower) and sample points



(b) Laplace interpolant



(c) Inverse Distance Weighting (radius=150m, power=2)

Figure 49: Dealing with heterogeneous data. Contours are shown at every 50 cm. Dataset London(e).

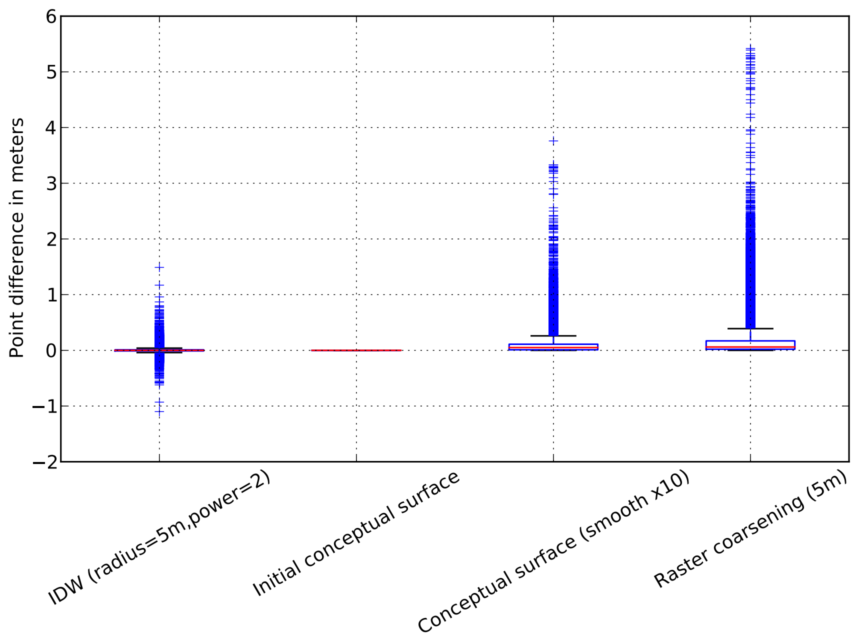


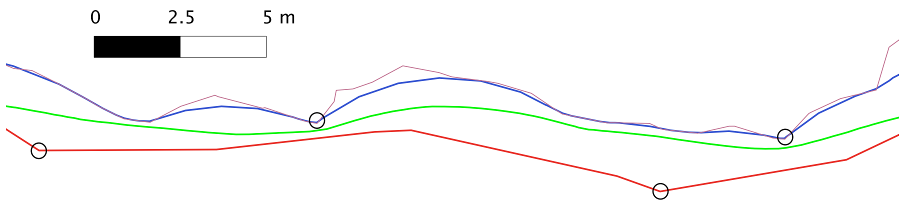
Figure 50: Boxplots of the sample point differences with respect to the interpolated fields. The blue symbols are the outliers. Dataset London.

of the sample points are violated already (with an extreme of -1.09 meter). On the other hand and quite obviously—since it is defined by the sample points *exactly*—the box plot of the initial conceptual surface is flat, thus fully safe and accurate with respect to the sample points. And as expected also after smoothing the safety constraint is respected. In case of raster coarsening the safety constraint is also respected in its field representation (note that does not mean the resulting contours are safe, see § 3.2.3). However when compared to the smoothed conceptual surface it becomes clear that there the differences are relatively larger, thus raster coarsening is less accurate than the smoothed conceptual surface.

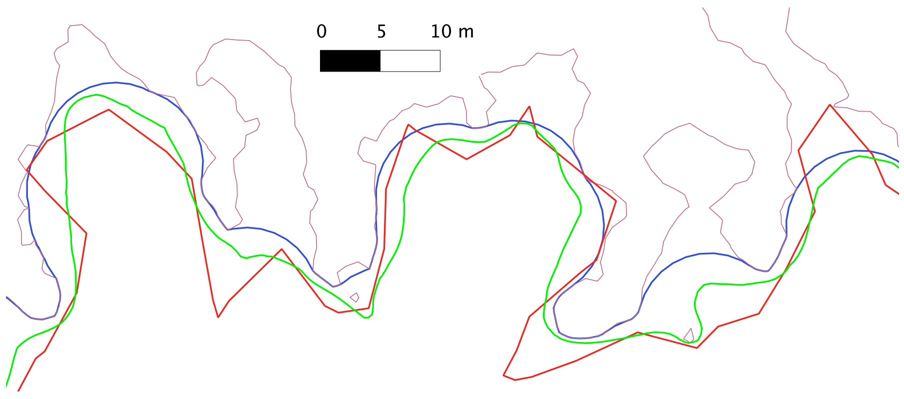
As a result of the violation of the safety constraint in IDW, there is no guarantee that any contours derived from an IDW interpolation are safe, neither can that be said of contours that are derived from a coarsened raster. And also any contours derived from those contours—i.e. using double buffering—might not be safe.

Disregarding the safety constraint for a moment (otherwise we should reject all of the contours from current methods immediately), we can still compare the contours of the different contouring methods with respect to legibility and smoothness. Observe Figure 51. From Figure 51a, it can be seen that the contours of both double buffering and raster coarsening have sharp edges. In case of double buffering this is due to the convex features (pointing south and towards the deeper region) in the input line (shown in light blue). Those features are not smoothed by the process of double buffering unless a very large buffer distance is chosen. In case of concave features in the input line, the doubly buffered line clearly exhibits circle arcs in its contour (Figure 51b), in a way that is quite an arbitrary depiction of geographic reality that is not necessarily very realistic (I would not expect the true seabed to be made up out of such circle arcs).

It is evident from both pictures in Figure 51 that the contours from the coarsened raster are indeed quite coarse and therefore not smooth at all. As expected, since the conceptual surface is smooth per definition, the contours from the Voronoi- and Surface based approach are smooth, and also have a quite natural look.



(a) Ellipses indicate sharp angles. Dataset London(d).



(b) Dataset London(c).

Figure 51: Comparison with methods that are used for hydrographic contour generalization in practice. Shown are IDW (cellsize=1m, radius=5m, power=2) in purple, double buffering (distance=6m) in blue, raster coarsening (cellsize=5m) in red and the Voronoi- and surface-based approach (smoothing 10x, densification 3x) in green.

6 CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

The conclusions can be subdivided in the answers to the research questions that were given in the first chapter (§ 6.1.1), a summary of the most significant contributions (§ 6.1.2), and a discussion of the overall value of this research (§ 6.1.3).

6.1.1 Research questions

This paragraph answers the research questions. It starts with answering the sub research questions and concludes with the answer to the main research question.

What characterizes surfaces that lead to good depth contours for hydrographic charts and what is needed in terms of interpolation and generalization to achieve such a surface?

According to the hydrographic generalization constraints, a good surface for hydrographic charts is:

- **legible**, i.e. smooth and clutterfree contours. A surface is smooth if its first derivative is continuous (C^1), which is thus a requirement for the interpolation method. Clutter can be reduced through cartographic generalization, i.e by the processes of simplification, omission and aggregation. Legibility can further be improved by the application of the generalization processes smoothing and enlargement.
- **topologically consistent**, more specially: non-intersecting contours. This can be guaranteed if the surface is continuous.
- **respecting the waterbody morphology** it represents. This translates to exactly honoring sample points and not disregarding any significant morphology during generalization, i.e. removing small high frequency detail while preserving large low frequency detail.
- **safe for navigation**, which means that the depth at the location of any sample point is never lifted downwards in the surface model. This can be implemented by preserving all sample points during the process of generalization and guaranteeing that no point is moved downwards.

Are those characterizations respected in the Voronoi- and surface-based approach?

Yes, because all of the stated requirements are incorporated into Are those characterizations respected in the Voronoi- and surface-based approach. It is also confirmed by experiments that were performed.

Does the Voronoi- and surface-based approach perform well for heterogeneously distributed input data?

For strongly heterogeneous datasets, i.e. those containing a mixture of single- and multi-beam surveys, the proposed method performs significantly better than what is currently used (IDW interpolation), although it is still not optimal.

To what extent can the Voronoi- and surface-based approach be automated?

As opposed to current methods, no (data-dependent) distance-based parameters need to be set. In the current implementation the user controls the amount of generalization (the number of smoothing passes), or where generalization needs to be applied (in case of the reshaping operator). Further automation is a subject of future research.

Is the Voronoi- and surface-based approach well scalable to big datasets?

Extreme scalability can be achieved through application of techniques such as streaming and parallelization, because all algorithms are local and do not require a global overview of the dataset. An interesting direction for future work is therefore also to test this.

And now I can answer the main research question:

Is the Voronoi- and surface-based approach a viable option for the automatic generation of depth-contours for hydrographic charts?

Yes. As far as this can be judged through the objective metrics that were used for analysis of the experimental results and comparison with current methods, it can be concluded that the Voronoi- and Surface-based approach performs well in terms of the hydrographic generalization constraints. Furthermore, it is also simple to implement and very scalable in principle.

6.1.2 Summary of contributions

The main contributions of this work are twofold:

1. I have shown that current approaches to generate and generalize contour lines from points are not truly respecting the hydrographic safety constraint.

2. I have presented an effective and novel Voronoi- and surface-based approach that uses natural neighbor interpolation at its core, covers the complete contour generation pipeline from points to generalized contour lines. The most significant properties are that the approach is:

   UNIFIED: it covers and integrates the complete processing chain to generate generalized contours from points.

   SURFACE-BASED: generalization is performed through the surface, i.e. through the complete and interrelated *system* of contours, rather than on contours individually.

   SAFE: non-violation of the hydrographic safety constraint is absolutely guaranteed. This is particularly significant because this completely eliminates the need to manually check the generated contours against the sample points for safety as is the case with approaches from practice.

   EXACT: planimetric coordinates of sample points are exactly preserved during processing.

   ADAPTIVE: it is automatically adaptive to the spatial distribution of sample points.

   SCALABLE: it is in principle extremely scalable using available techniques.

   EFFECTIVE IN GENERALIZATION: all of the relevant generalization operators for hydrographic contours are incorporated.

6.1.3 Discussion

The major criticism on the Voronoi- and surface-based approach as developed in this thesis is that it seemingly ignores the topic of modeling the effect of cartographic generalization on the geometrical uncertainty in sample points. Indeed, and as stated in § 1.2, I assume error-free sample points. Yet, strictly speaking, every measurement of reality inherently has an uncertainty associated to it that is commonly decomposed into a systematic and a random error. Minimum requirements for these errors and issues such as the minimum size of a feature that should be included, i.e. measurable by the surveying equipment, in hydrographic surveys are stated in the International Hydrographic Organization (IHO) S-44 standard on hydrographic surveying (IHO, 2008). The motivation of that document is to ensure a level of quality of every hydrographic survey, most importantly in areas that are critical to navigation. The uncertainty in points should also be recorded as part of the survey to be able to check if the survey is meeting the requirements of the S-44 standard. If that is not the case for some regions, that area should be re-surveyed with higher accuracy.

The question remains on how this information on uncertainty is used and affected by the hydrographer that draws the hydrographic chart; the process of generalization. To my best knowledge the IHO does not state anything on this, other than referring to hydrographic practices such as mentioning the geographic extent, quality description (using so-called Zone Of Confidence (ZOC)) and datedness of the surveys that were used to draw the map (sometimes that is also done for individual navigation-critical features). Still, the process of generalization—and that lies at the core of this research—inevitably causes displacements in boundaries of map features. Does that affect the error in the modeled surface? Kimerling and Muehrcke (2009) argue that saying that positional displacement caused by cartographic generalization is error, misses the whole point of generalization, i.e. to perform meaningful alterations of feature geometry to improve the overall legibility and usefulness of the map. Hydrographic chart products (e.g. from the Dutch Hydrographic Office) also state in capital letters "Always use the largest scale chart appropriate". Evidently, less generalization is applied to large scale charts, these are thus closer to reality, than to smaller scale charts, which primarily serve to provide a more simplified and clutterless overview of a large area.

This does not mean that statistical uncertainty is not to be taken into account at all during generalization. According to Smith (2003) the most significant part of the vertical error in a sample point is *shoal-biased* (i.e. directed towards the water surface). Since the safety constraint is guaranteed to be respected in the Voronoi- and surface-based approach, all alterations of the surface are shoal-biased. Assuming a normally distributed vertical error, the size of the remaining shoal-biased error (with e.g. 95% probability) after processing can thus only decrease. Therefore I would argue that the significant (shoal-biased) vertical error in a point can never get worse.

With regard to the horizontal error in a sample point, it is common practice for a hydrographer to draw a contour more widely around points with a particularly low horizontal accuracy as that increases the probability that the true location of that point lies inside that contour. Smith (2003) calls this *defocusing* the surface. Given the availability of a sample point's quality description, this is also trivial to implement using e.g. the reshaping operator that is part of the Voronoi- and surface-based approach.

6.2 FUTURE WORK

Here I list a number of interesting topics for follow-up research. I find the first two topics most interesting as these are both currently very active and interesting research areas.

MORE COMPREHENSIVE GENERALIZATION: More research can be performed in investigating how the Voronoi- and surface based approach can be integrated with (the generalization of) other elements on the hydrographic chart, such as depth-soundings and symbology. As stated in § 4.4 further automation can also be achieved through an overall generalization methodology such as amplified intelligence or a constraint-based approach. Relevant research on this topic was performed by Guilbert and Zhang (2012).

ACHIEVING SCALABILITY: As described in § 4.5 a re-implementation of the developed prototype using techniques such as streaming processing (Isenburg et al., 2006b) would greatly improve scalability.

TRACK DEPENDENT GENERALIZATION: A suggestion from shippers is to use known or likely ship routes during generalization. In many inland hydrographic charts for example it can be considered to perform more aggregation in features that are along the river axis. The reshaping operator could be used for this.

DATA-COMPRESSION IN CONTOURS: The outputted depth-contour of the current implementation contain many points, that do not significantly contribute to the shape of the contour. Using a method such as the one proposed by Meijers (2011), those points can probably safely be removed.

BETTER HANDLING OF ANISOTROPIC DATA: As can be concluded from Chapter 5, the Laplace interpolation from sample points with anisotropic sampling (i.e in case of single-beam tracks) can result in unwanted artifacts. It might be possible to adapt the Laplace interpolant in these cases to give points perpendicular to the single beam tracks more importance, which would improve the interpolation result.

ADAPTIVE DENSIFICATION: Since the densification operator aims to minimize the approximation error of the generated contours, it might prove beneficial to only densify those triangles that contribute to the requested contour line depths, quite similar to Figure 52. The difficulty in this lies in the fact that inserting a point might induce changes to the triangulation even outside the current triangle.

DEALING WITH UNCERTAINTY: As follows from the IHO-S44 standard (IHO, 2008) that is discussed in § 6.1.3, echo sounding surveys should always including information on the errors in the measurements. It would be interesting to study how these errors are affected by the generalization operators of the Voronoi-

and surface-based approach as well as those from existing approaches.

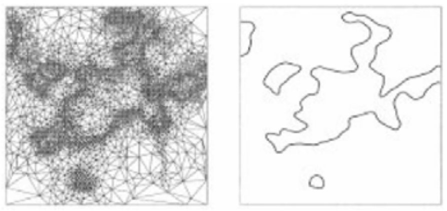


Figure 52: The TIN resolution is increased only at triangles that contribute to the contour lines. From Floriani et al. (2000)

A A MODIFIED DROP HEURISTIC ALGORITHM

This algorithm is virtually identical to the one described by van Kreveld (1997), which is an improved version of the original algorithm by Lee (1989). The reported expected running time is $O(n \log n)$.
I added the conditional statements on lines 10 and 21 in an attempt to satisfy the hydrographic safety constraint during simplification (which ultimately did not work as explained in § 4.2).

Algorithm 7 A TIN simplification algorithm

```
    Input: a Delaunay triangulation $\mathcal{T}$, a threshold $maxError$
    Output: the simplified Delaunay triangulation of $\mathcal{T}$

 1: function VerticalError(vertex $v$ with depth $h$)
 2:     temporarily remove $v$ from $\mathcal{T}$ and retriangulate hole
 3:     estimate depth $h'$ in the current $\mathcal{T}$ at the location of $v$ using
    Linear TIN interpolation
 4:     re-add $v$ to $\mathcal{T}$
 5:     Return $h - h'$
 6: end function

 7: function Simplify(triangulation $\mathcal{T}$, threshold $maxError$)
 8:     for all vertices $v \in \mathcal{T}$ do
 9:         $d \leftarrow$ VerticalError($v$)
10:         if $d > 0$ then
11:             store $d$ in binary tree $\mathcal{D}$
12:         end if
13:     end for
14:     while $d \leftarrow \text{MIN}(\mathcal{D})$; $d < maxError$ do
15:         remove $d$ from $\mathcal{D}$
16:         store the adjacent vertices to $v$, $w_1, ..., w_j$
17:         remove corresponding vertex $v$ from $\mathcal{T}$
18:         for all vertices $w_i \in \{w_1, ..., w_j\}$ do
19:             remove corresponding error from $\mathcal{D}$
20:             $d \leftarrow$ VerticalError($w_i$)
21:             if $d > 0$ then
22:                 insert the new $d$ in $\mathcal{D}$
23:             end if
24:         end for
25:     end while
26: end function
```

B SOFTWARE IMPLEMENTATION

B.1 VORONOI- AND SURFACE-BASED APPROACH

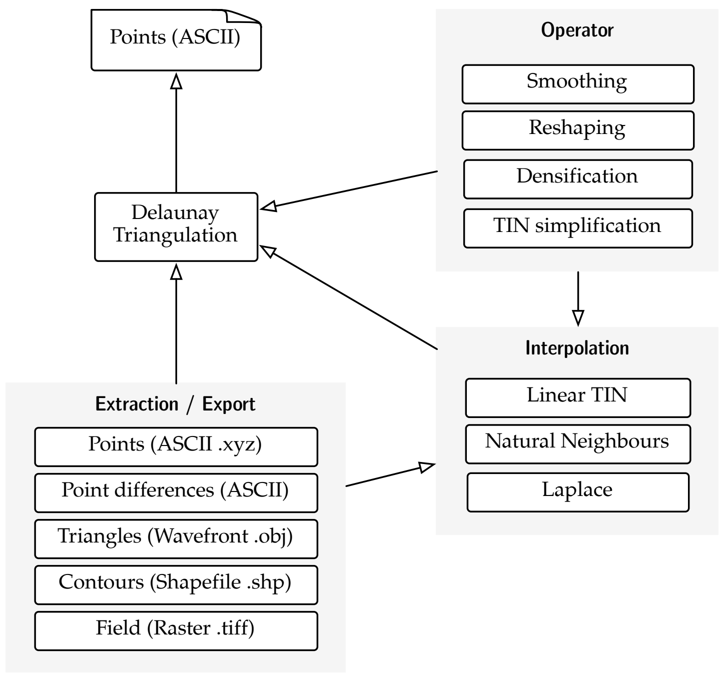


Figure 53: Dependency graph of the prototype software implementation

Figure 53 shows a simplified schema of how the implemented prototype is constructed. It was written in the C++ programming language. At the core of the prototype lies a topological Delaunay Triangulation (DT) data-structure, that is implemented using the Computational Geometry Algorithms Library (CGAL)[1]. It is initially created from an ASCII file. The operators are implemented as indicated in Chapter 4 and can be used to modify the triangulation. The interpolation methods are used by these operators and also by some of the export function. For example, for exporting contour lines, linear TIN interpolation is used (as in § 2.4). The contour lines are processed (line merging and orientation) using the Geometry Engine Open Source (GEOS)[2] and exported to a Shapefile using the Geospatial Data Abstraction Library[3], which is also used for raster export.

1 http://www.cgal.org/
2 http://trac.osgeo.org/geos/
3 http://gdal.org/

The prototype was developed and run on laptop computer with a 2.4 GHz Intel Core 2 Duo processor and 8 GB of RAM running Mac OS X.

B.2 ADDITIONAL TOOLS

In addition to the implementation of the proposed Voronoi- and surface-based approach a number of other programs were also developed:

1. A program to convert the sample data sets into a consistent representation. This program also uses PROJ.4[4], which is a geographic projection library.
2. A program that performs raster coarsening (see Section 3.2.3).
3. An implementation of the double buffering algorithm (Section 3.4.1), using the GEOS library.
4. An OpenGL-based 3D viewer application, see Figure 54.

Other tools that were used for analysis and comparison:

QUANTUM GIS AND GRASS: Both GIS software used for elementary GIS operations and visualization.

GDAL UTILITIES: For its IDW interpolation.

PYTHON: For data conversions, automation and visualization using Matplotlib.

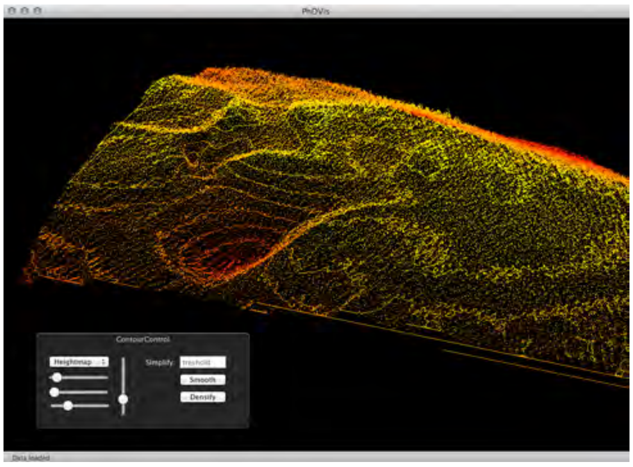


Figure 54: OpenGL-based viewer application

4 http://trac.osgeo.org/proj/

C DATASETS

C.1 OVERVIEW

This chapter details the different datasets that were used for testing and analysis in this thesis. The datasets were given to me by George Spoelstra from the former company Atlis. They originate from various geographical locations and all have different characteristics. Figure 55 shows the approximate locations of the datasets on a world map. Table 2 lists the general (known) characteristics per dataset. All datasets were delivered as an ASCII with for each point: latitude, longitude and depth. Information on point quality and vertical datum was not available. Furthermore, for the London dataset, which represent a part of the river Thames, also the projection is unknown. No form of downsampling or point filtering was applied.

Figure 56 shows for each dataset a histogram of triangle aspect ratios. The triangle aspect ratio is defined as the longest triangle edge divided by the shortest altitude and is calculated for the Delaunay Triangulation (DT). The shape of these histograms is thus related to the spatial distribution of points. Most notable is Figure 56a, the histogram for the Antilles, that clearly indicates a relatively high aspect ratio which can be explained by the elongated triangles (see Figure 46a, p. 77) caused by its single-beam samples. The other datasets have a more isotropic distribution, which results in a lower triangle aspect ratio.

The following four sections give an image for every dataset. These images indicate the different sub-datasets that were used in Chapter 5 and give an impression of the spatial distribution of points. Depths are indicated with a colormap that runs from green to yellow to blue in order of decreasing depth.

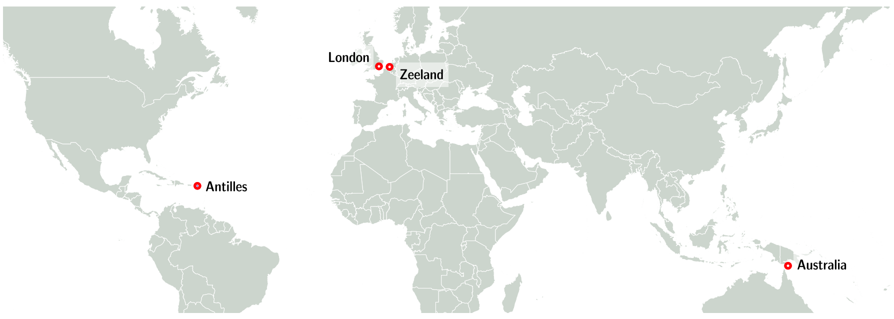
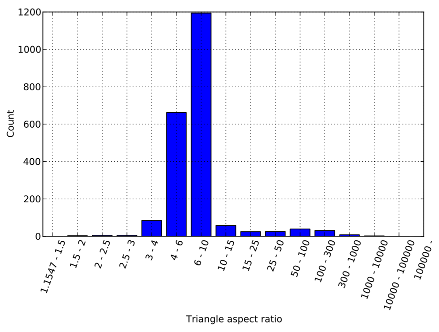


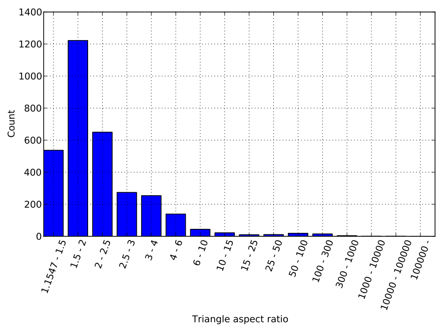
Figure 55: Geographical locations of datasets

	Antilles	*Australia*	*London*	*Zeeland*
Min WGS84 lon	63.3352W	143.780417E	unknown	3.80297E
Max WGS84 lon	63.8660W	143.785766E	unknown	3.81422E
Min WGS84 lat	17.8016N	9.728203S	unknown	51.37805N
Max WGS84 lat	18.2788N	9.723904S	unknown	51.38410N
Depth range	-2034 to -50 m	-28 to -1 m	-15.90 to 4.48 m	-28 to -18 m
Point count	1081	1613	151704	102954
Type	SBES	MBES	MBES+SBES	MBES
Projection	UTM 19 south	UTM 53 south	unknown	UTM 30 north

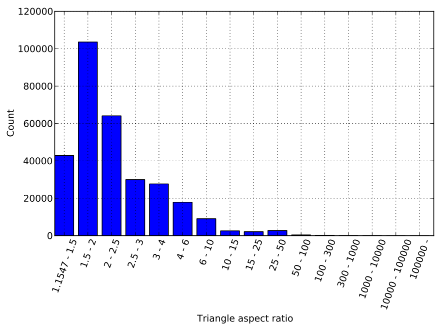
Table 2: Details of datasets



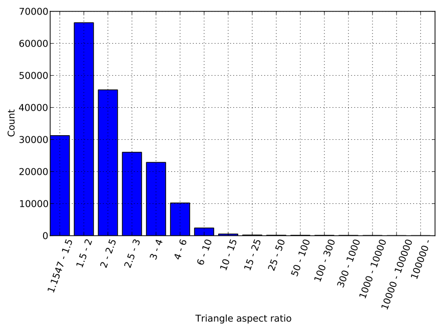
(a) Antilles



(b) Australia



(c) London



(d) Zeeland

Figure 56: Histograms of triangle aspect ratios for all datasets

C.2 ANTILLES

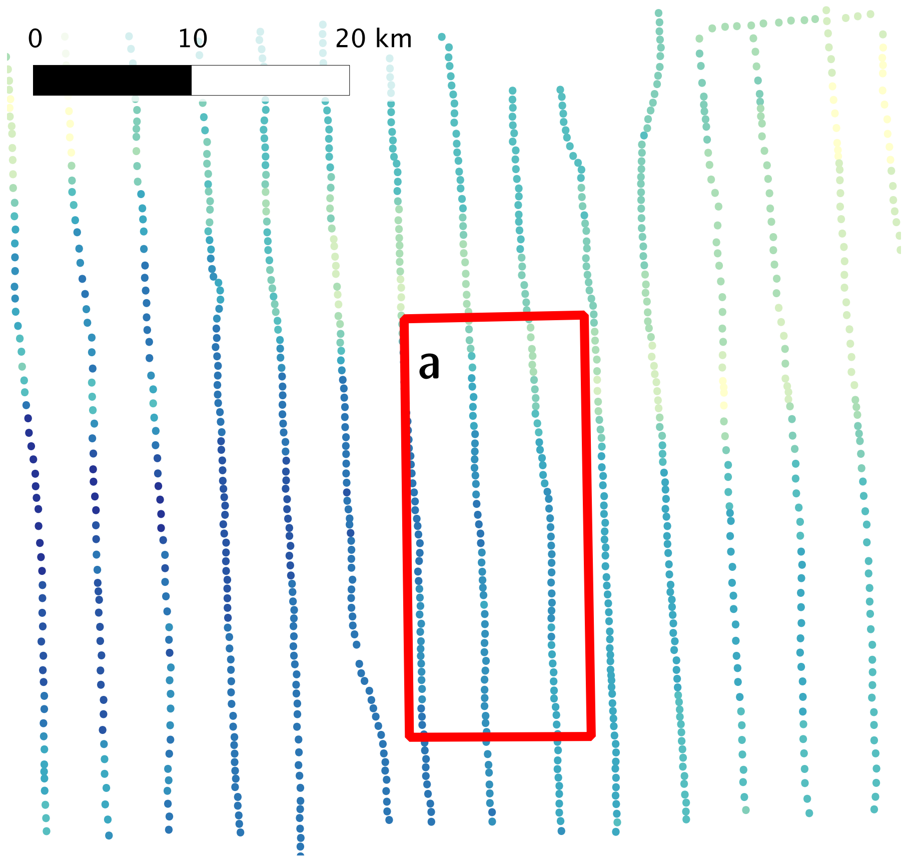


Figure 57: Dataset Antillen

C.3 AUSTRALIA

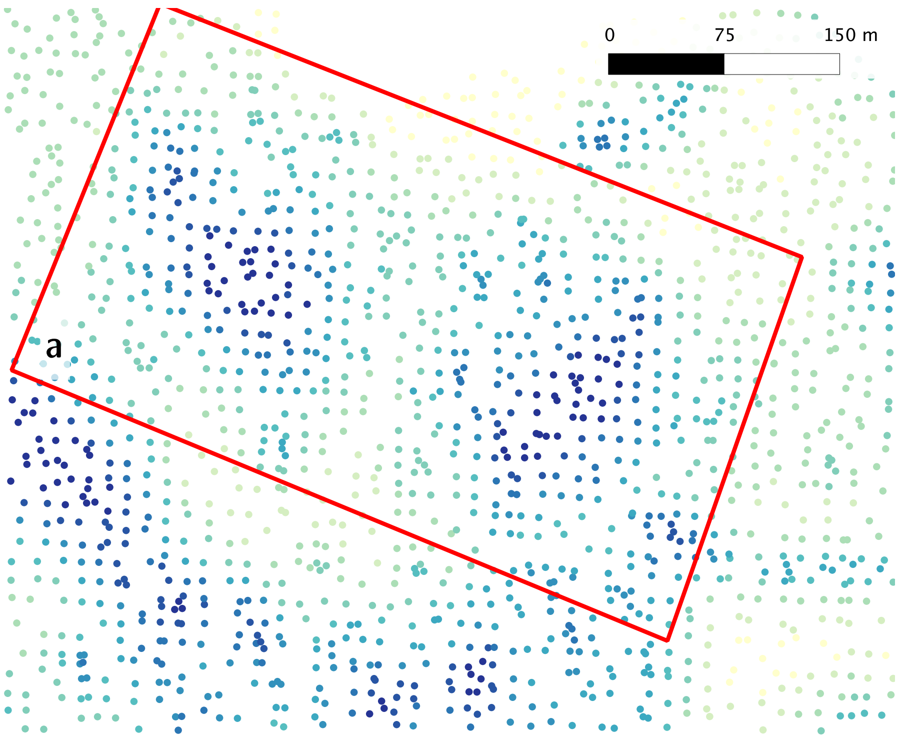


Figure 58: Dataset Australia

C.4 LONDON

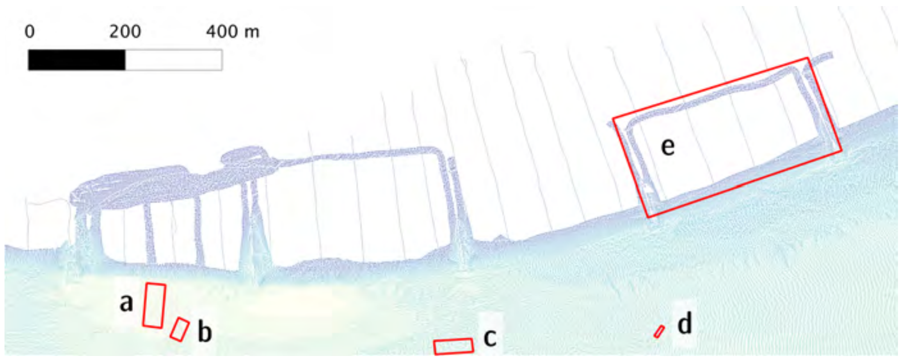


Figure 59: Dataset London

C.5 ZEELAND

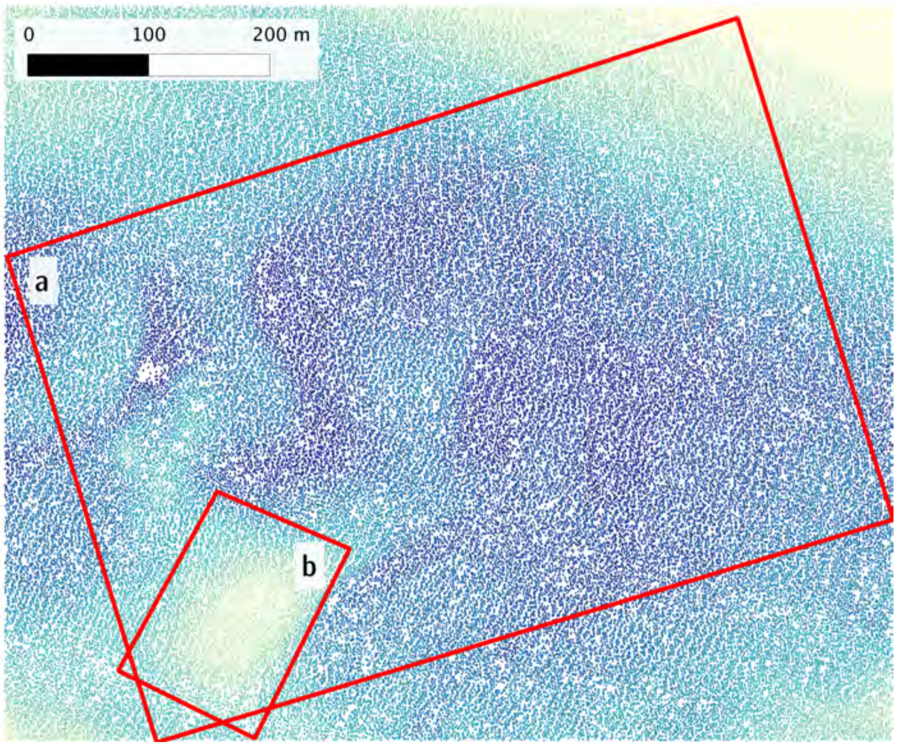
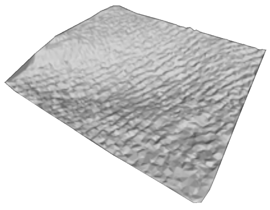
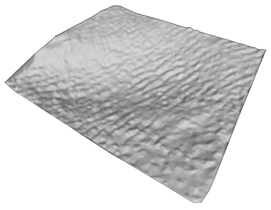
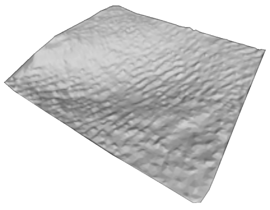


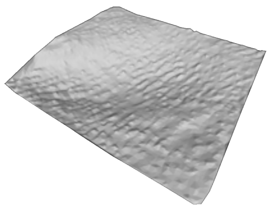
Figure 60: Dataset Zeeland

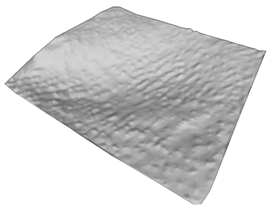
D A 3D VIEW OF SMOOTHING

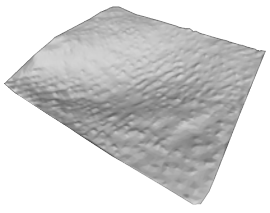

(a) 0x

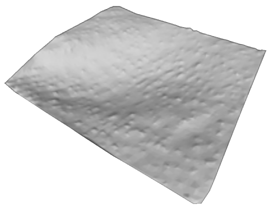

(b) 1x

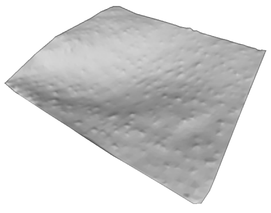

(c) 2x

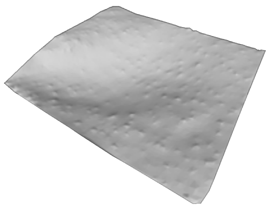

(d) 3x

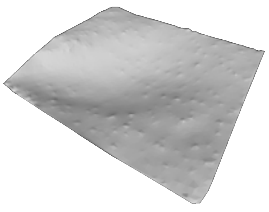

(e) 4x


(f) 5x


(g) 10x


(h) 15x


(i) 20x


(j) 30x

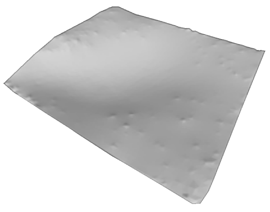

(k) 100x

Figure 61: A 3D view on the effect of the smoothing operator on the surface

BIBLIOGRAPHY

IHO Standards for Hydrographic Surveys, Special Publication N° 44. Technical report, International Hydrographic Organization, February 2008.

L. Arge, K. G. Larsen, T. Mølhave, and F. van Walderveen. Cleaning massive sonar point clouds. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 152–161, New York, NY, USA, 2010. ACM.

V. Belikov, V. Ivanov, V. Kontorovich, S. Korytnik, and A. Semenov. The non-sibsonian interpolation: a new method of interpolation of the values of a function on an arbitrary set of points. *Computational mathematics and mathematical physics*, 37(1):9–15, 1997.

N. Christ, R. Friedberg, and T. Lee. Weights of links and plaquettes in a random lattice. *Nuclear Physics B*, 210(3):337–346, 1982.

M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational geometry: Algorithms and applications*. Springer-Verlag, Berlin, second edition, 2000.

P. Fisher. The pixel: a snare and a delusion. *International Journal of Remote Sensing*, 18(3):679–685, 1997.

L. Floriani, P. Magillo, and E. Puppo. Variant: a system for terrain modeling at variable resolution. *GeoInformatica*, 4(3):287–315, 2000.

M. Garland and P. Heckbert. *Fast polygonal approximation of terrains and height fields*. School of Computer Science, Carnegie Mellon University, 1995.

M. Garland and P. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

C. Gold. Surface interpolation, spatial adjacency and gis. *Three dimensional applications in geographic information systems*, pages 21–35, 1989.

M. Goodchild. The validity and usefulness of laws in geographic information science and geography. *Annals of the Association of American Geographers*, 94(2):300–303, 2004.

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

D. Gruenreich. ATKIS-a topographic information system as a basis for GIS and digital cartography in Germany. *From Digital Map Series to Geo-Information Systems, Geologisches Jarhrbuch Series A. Hannover, Germany: Federal Institute of Geosciences and Resources*, 1992.

E. Guilbert. Multi-level representation of terrain features on a contour map. *GeoInformatica*, pages 1–24, 2012.

E. Guilbert and H. Lin. Isobathymetric line simplification with conflict removal based on a b-spline snake model. *Marine Geodesy*, 30 (1-2):169–195, 2007.

E. Guilbert and E. Saux. Cartographic generalisation of lines based on a b-spline snake model. *International Journal of Geographical Information Science*, 22(8):847–870, 2008.

E. Guilbert and X. Zhang. Generalisation of submarine features on nautical charts. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume I-2, Melbourne, Australia, 2012.

P. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Technical report, DTIC Document, 1997.

H. Hiyoshi and K. Sugihara. Two generalizations of an interpolant based on voronoi diagrams. *International journal of shape modeling*, 5 (2):219–232, 1999.

E. Imhof. *Cartographic relief presentation*. ESRI Press (2007), 1965. ISBN 9781589480261.

M. Isenburg, Y. Liu, J. R. Shewchuk, and J. Snoeyink. Streaming computation of Delaunay triangulations. *ACM Transactions on Graphics*, 25(3):1049–1056, 2006a.

M. Isenburg, Y. Liu, and J. Snoeyink. Streaming extraction of elevation contours from LIDAR points. Not published, available at `http://www.cs.unc.edu/~isenburg/papers/ils-lidar2iso-06.pdf`, 2006b.

A. J. Kimerling and P. Muehrcke. *Map use: reading and analysis*. ESRI Press Academic, 2009.

M. Kraak and F. Ormeling. *Cartography: visualization of geospatial data*. Prentice Hall, 2003.

M. P. Kumler. An intensive comparison of triangulated irregular networks (TINs) and digital elevation models (DEMs). *Cartographica*, 31(2), 1994.

H. Ledoux. *Modelling three-dimensional fields in geoscience with the Voronoi diagram and its dual*. PhD thesis, GIS Research Centre, School of Computing, University of Glamorgan, Wales, UK, 2006.

H. Ledoux. Generation of bathymetric contours: Investigation of the current problems and a proposition of solutions. Unpublished report, Delft University of technology, 2009.

J. Lee. *A drop heuristic conversion method for extracting irregular network for digital elevation models*. ASPRS/ACSM, 1989.

Z. Li and S. Openshaw. Algorithms for automated line generalization1 based on a natural principle of objective generalization. *International Journal of Geographical Information Systems*, 6(5):373–389, 1992.

R. Lindenbergh and A. Hooper. Course in MSc Geomatics: Multivariate Data Analysis. Presentation slides, TU Delft, 2010.

W. Mackaness, A. Ruas, and L. Sarjakoski, editors. *Generalisation of geographic information: cartographic modelling and applications*. Elsevier Science, 2007.

D. Maguire, M. Goodchild, and D. Rhind. An overview and definition of gis. *Geographical information systems: Principles and applications*, 1: 9–20, 1991.

K. Matuk, C. Gold, and Z. Li. Skeleton based contour line generalization. *Progress in Spatial Data Handling*, pages 643–658, 2006.

M. Meijers. Simultaneous & topologically-safe line simplification for a variable-scale planar partition. In S. Geertman, W. Reinhardt, and F. Toppen, editors, *Advancing Geoinformation Science for a Changing World*, Lecture Notes in Geoinformation and Cartography, pages 337–358. Springer Berlin Heidelberg, April 2011.

T. Meyer. The discontinuous nature of kriging interpolation for digital terrain modeling. *Cartography and Geographic Information Science*, 31 (4):209–216, 2004.

M. Molenaar. *An Introduction To The Theory Of Spatial Object Modelling For GIS*. Research monographs in Geographic Information Systems. Taylor & Francis, 1998. ISBN 9780748407743.

M. Oliver and R. Webster. Kriging: a method of interpolation for geographical information systems. *International Journal of Geographical Information System*, 4(3):313–332, 1990.

S. Rippa. Minimal roughness property of the Delaunay triangulation. *Computer Aided Geometric Design*, 7:489–497, 1990.

M. Sambridge, J. Braun, and H. McQueen. Geophysical parametrization and interpolation of irregular data using natural neighbours. *Geophysical Journal International*, 122(3):837–857, 1995.

R. Shewchuk. Star splaying: an algorithm for repairing delaunay triangulations and convex hulls. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 237–246. ACM, 2005.

Y. Shokin and K. Afanas'ev. Application of the natural element method for solution of problems of fluid dynamics with free boundaries. *Russian Journal of Numerical Analysis and Mathematical Modelling*, 26(5):515–531, 2011.

R. Sibson. A brief description of natural neighbour interpolation. *Interpreting multivariate data*, 21, 1981.

R. Sibson. Contour mapping: Principles. Not published, available at http://citeseer.ist.psu.edu/112339.html, 1997.

S. M. Smith, L. Alexander, and A. A. Armstrong. The navigation surface: A new database approach to creating multiple products from high-density surveys. *International Hydrographic Review*, 3(2): 12–26, 2002.

S. M. Smith. The navigational surface: A multipurpose bathymetric database. Master's thesis, University of New Hampshire, 2003.

J. Stoter, D. Burghardt, C. Duchêne, B. Baella, N. Bakker, C. Blok, M. Pla, N. Regnauld, G. Touya, and S. Schmid. Methodology for evaluating automated map generalization in commercial software. *Computers, Environment and Urban Systems*, 33(5):311–324, 2009.

W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46:234–240, 1970.

D. Tsichritzis and F. Lochovsky. *Data base management systems*, volume 460. Academic Press, 1977.

P. van der Poorten and C. Jones. Characterisation and generalisation of cartographic lines using delaunay triangulation. *International Journal of Geographical Information Science*, 16(8):773–794, 2002.

M. van Kreveld. Digital elevation models and tin algorithms. *Algorithmic foundations of Geographic Information Systems*, pages 37–78, 1997.

M. Van Kreveld, R. Van Oostrum, C. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface traversal. In *Proceedings of the thirteenth annual symposium on Computational geometry*, pages 212–220. ACM, 1997.

M. Van Kreveld. Efficient methods for isoline extraction from a digital elevation model based on triangulated irregular networks. Technical Report 1994-21, Universiteit Utrecht, 1994.

E. Verbree and P. van Oosterom. The STIN Method: 3D-surface reconstruction by observation lines and delaunay TENs. ISPRS working group III/3 workshop '3D-reconstruction from airborne laserscanner and InSAR data', Dresden, October 2003.

K. Wang, C.-p. Lo, G. A. Brook, and H. R. Arabnia. Comparison of existing triangulation methods for regularly and irregularly spaced height fields. *International Journal of Geographical Information Science*, 15(8):743–762, 2001.

D. F. Watson. *Contouring: A guide to the analysis and display of spatial data*. Pergamon Press, Oxford, UK, 1992.

R. Weibel. Amplified intelligence and rule-based systems. *Map generalization: Making rules for knowledge representation*, pages 172–186, 1991.

R. Weibel. Generalization of spatial data: Principles and selected algorithms. *Algorithmic foundations of geographic information systems*, pages 99–152, 1997.

D. Wright and D. Bartlett, editors. *Marine and Coastal Geographical Information Systems*. Research Monographs in Geographic Information Systems. Taylor & Francis, 2000. ISBN 9780748408627.

M. Yanalak. Sibson (natural neighbour) and non-sibsonian interpolation for digital elevation model (dem). *Survey Review*, 37(291): 360–376, 2004.

S. Zelinka and M. Garland. Permission grids: practical, error-bounded simplification. *ACM Transactions on Graphics (TOG)*, 21 (2):207–229, 2002.

L. Zhang, Y. Liu, Q. Zhu, and F. Xiao. A solution to the ambiguity problem in depth contouring. *The International hydrographic review*, 9(2):59, 2008.

X. Zhang and E. Guilbert. A multi-agent system approach for feature-driven generalization of isobathymetric line. *Advances in Cartography and GIScience. Volume 1*, pages 477–495, 2011.

COLOPHON

This document was typeset using LaTeX on Mac OS X.

Document layout was generated using the ArsClassica package by Lorenzo Pantieri, which is an adaption of the original ClassicThesis package from André Miede.

Figures and diagrams were mostly drawn using IPE, PGF/Ti*k*z and Omnigraffle. Plots were generated using Matplotlib and maps were created with QGIS. BibTeXwas used to generate the bibliography.

ISBN 978-94-6186-100-9