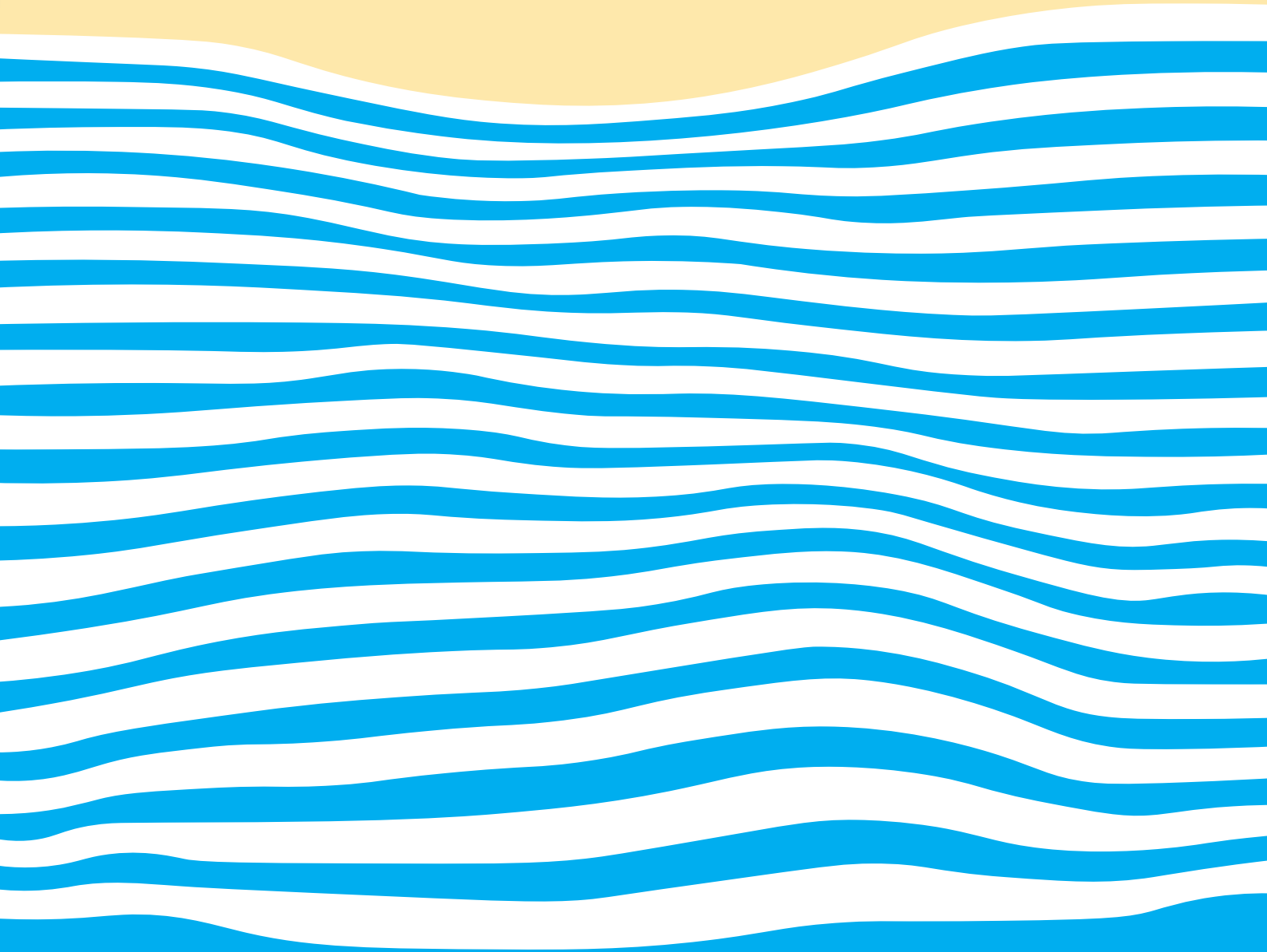
Automatic isobath generalisation for navigational charts

Willem van Opstal



TU Delft

MSc thesis in Geomatics
July 2020

MSc thesis in Geomatics

Automatic isobath generalisation for navigational charts

Willem van Opstal

July 2020

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

An electronic version of this thesis is available at https://repository.tudelft.nl
The accompanying software is available at https://github.com/willemvanopstal/hydropolator

The work in this thesis was carried out at:

MSc Geomatics
Faculty of Architecture and the Built Environment
Delft University of Technology

Author:	Willem van Opstal
	willemvanopstal@home.nl
Supervisors:	dr.ir. Martijn Meijers
	dr.ir. Ravi Peters
Co-reader:	dr. Roderik Lindenbergh

Abstract

Navigational charts play a vital role in a ship's safety while navigating the seas, rivers or lakes. With most of the features and obstructions being out of sight — below sea-level — these charts are more critical than e.g. topographic maps. For routing but also positioning, depth information is a key aspect on these charts. This depth information is available in either depth contours, coloured depth areas and individual soundings. However with the data originating from dense and accurate but usually erratic survey data, a visualisation of raw data is not sufficient for use in a navigational chart directly. It would not clearly convey the information to a human operator in one sight, and thus this visualisation is in need of generalisation: a simplified representation of the same data with irrelevant details being omitted. This thesis gives new insights in the generalisation process for isobaths only and proposes a new framework to deal with those.

In navigational charts the amount of generalisation changes with scale and purpose of the chart as well as local properties of the area, e.g. an anchorage, nature reserve or busy port should be treated differently. Currently the complex task of defining *irrelevant* details in different situations is in need of lots of manual intervention. That is mainly due to incompatible generalisation constraints: simplifying a line without changing morphology is simply impossible. Especially if the *safety* of the chart should always be guaranteed. An automated generalisation approach would bring economic and safety benefits to the maritime community. This research continues on such an automated approach: the Voronoi- and surface-based approach. The method relies on a triangulated surface of the soundings and generalises it by smoothing. However it lacks a connection with the final cartographic product and thus the generalisation constraints. The method just continues generalisation in an iterative approach, for a finite amount of iterations. There are no means for self-evaluation and thus no means of stopping it when results are acceptable.

In this thesis we propose a framework based on a novel auxiliary data structure to link a triangulation to the resulting isobaths: the *triangle region graph*. It links the position of isobaths directly to individual triangles, as well as establishes relations between the isobaths themselves. With this structure we ultimately link the survey data to the final cartographic product. In theory we could thus integrate all information across the generalisation pipeline in one and the same process. We have successfully used this framework with a basic rule-based evaluation model: first we isolate conflicting isobaths, triangles and vertices based on legibility requirements; then we apply targeted generalisation operators only on these conflicts.

With this approach we can successfully maintain more of the morphology while still yielding a finely legible chart, in comparison with the original Voronoi- and surface-based approach. Especially at large scale charts the results are promising: narrow channels, pits and bends remain if legibility permits. With smaller scale charts the challenge now is to generalise *beyond smoothness*. More radical generalisation operators are needed to omit all irrelevant details. However the overall framework using the triangle region graph as integrating mechanism has potential to do so. It is easily extensible due to its modular approach and can incorporate most depth information: from survey accuracy to size of isobaths and even *golden sounding* selection in the future.

Preface

”Navigation is all about knowing where you are, where to go and what to do to get there. But especially about knowing where not *to go.”*

How true this can be for navigation itself, while writing my thesis I felt this may be even more true for a graduation project like this. You sometimes get lost in your own ideas, become insecure about the progress you are making or whether you actually contribute enough to the scientific community. And then there are those moments you think: *now what?* or even worse: *why have I done this?* Just as in navigating a ship at sea — surrendered to the ebb and flow or an occasional grounding — sometimes a thesis is more about planning rather than hard work. No matter how hard you work, sometimes you simply ground.

And while hard work may be completely up to yourself, planning and ideation is certainly not. I was lucky having my mentors Martijn and Ravi supporting me in this. During our meetings ideas were plentiful and feedback was always supportive. But with it, the choices of where to go and where not, grew as well. I am very thankful to both Martijn and Ravi for all the help in these choices, their willingness to share knowledge and the few times I needed software support. I could always walk in on them, approach them personally and professional when appropriate. I am also thankful for the feedback Roderik Lindenbergh supplied in the final stages. It was the last push I needed to end up at the place I wanted to go.

I personally believe a large project like this should have a possible impact onto the real world to justify it. Context is for me of upmost importance. Of course this can be extracted from the web, but I have found it of great inspiration to visit those institutions actually being part of this context. Therefore I would like to thank Rogier Broekman, Jan Schaap and Rajet for having me at the *Dienst der Hydrografie* and guide me through the entire thinking– and compilation-process of navigational charts. At *Rijkswaterstaat*, René Visser was equally receptive for having me. I enjoyed the view over the Maas river, but even more the views and concepts he had on inland navigational charts. Thanks to everyone, without all the knowledge I gained through them this project would not have been as relevant as it hopefully is now.

After knowing where you are and where to go, the time comes of actually making progress: the hard work. Most of this happened at my usual spot, far left at the window in the *GeoLab*, which even sustained after the great move at the faculty. But no-one could have anticipated on what happened next. While working from home I actually appreciated the support of my girlfriend even more. She was there when needed, and gone on the right moments. It was however a difficult time. It was me and the not-so-well-prepared dining table in constant battle with the distraction of the house and upcoming spring. No sports or drinks, but above all not seeing my parents and family as much as before. I would have loved to share more of my work, especially because of their unforgiving support throughout my long educational career. Thank you all.

Of course I sometimes struggled, but above all I enjoyed thinking and ideating about such a complex topic. After all of the concepts which have passed my mind, doodles I have drawn and lines of code I wrote, I believe this project has come to the place I had liked it to end up at. Please have a good read and may it be useful to whoever is interested in this beautiful topic.

Willem van Opstal
Delft, 2020

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

BAG	Bathymetric attributed grid	28
CDT	Constrained Delaunay triangulation	60
CCW	Counter clockwise	20
CRS	Coordinate reference system	60
DT	Delaunay triangulation	18
ECDIS	Electronic Chart Display and Information System	1
ENC	Electronic Navigational Chart	1
ID	Identifier	61
IHO	International Hydrographic Organisation	1
IMO	International Maritime Organisation	121
MAS	Multi-agent system	26
MBES	Multi beam echo sounding	1
NEM	Nautical elevation model	28
NLHO	Netherlands Hydrographic Office	13
NOAA	National Oceanic and Atmospheric Administration	16
POI	Point of interest	44
RMSE	Root mean squared error	57
RNC	Raster Navigational Chart	9
SBES	Single beam echo sounding	1
TIN	Triangulated irregular network	18
TRG	Triangle region graph	38
UKHO	United Kingdom Hydrographic Office	60
VD	Voronoi diagram	18
VSBA	Voronoi-surface based approach	29

1 Introduction

A ships' crew uses navigational charts, whether analogue or digital, to ensure safe navigation throughout their time at sea or inland waters (example in Figure 1). These charts contain among others information about fairways, depths, fixed and floating marks, harbours, obstructions as well as basic information about the topography on land. In conjunction with other nautical publications like tidal information and updated buoy-positions, a navigator uses the chart to determine where it currently is, where to go, and especially where *not* to go.

National hydrographic offices are responsible and obliged[2] to create these charts and publications (IMO, 1974). Such charts contain information of different natures all acquired through different technologies and suppliers. However depth information or morphology of the seabed is of particular interest in routing and accessing ports. In the old days this depth information was acquired through lead lines, but advancing technology supplied single beam echo sounding (SBES) and later multi beam echo sounding (MBES) systems. Especially MBES enables very dense data distributions and can even enable full coverage of the seabed. One might think that dense data per definition enables the best representation of the seabed, but raw survey data is not of particular interest for navigators and can actually be confusing or overwhelming. Survey data is therefore processed into a format which communicates the same information, but now into a visually appealing and clear manner. The International Hydrographic Organisation (IHO) develops standards for this representation and states depth information should be in the form of selected soundings, depth contours (isobaths[3]) and depth areas (IHO, 2018c).

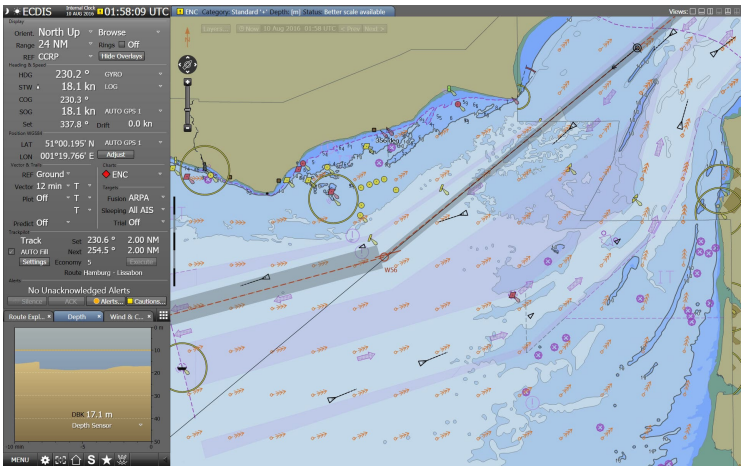


Figure 1.1: Example of an Electronic Chart Display and Information System (ECDIS) with an Electronic Navigational Chart (ENC) of the English Channel. *Taken from Wartsila*[1]

The natural aim of a cartographer is to depict all known details at larger scale[4] charts. However this could create a representation which is cluttered and thus not clearly conveying the information contained. At smaller scale charts this cluttering of detailed features would even become worse. The information should therefore be *generalised* for different scales, only retaining essential information and smooth lines and symbols. A properly generalised chart not only yields a clear representation, it also decreases the amount of maintenance upon updating and may force a deep-draft ship to use a larger scale chart (IHO, 2018c). While generalisation is applicable to all information on the chart, applying it to depth information is particularly challenging. This process still needs significant amounts of manual interventions and the increasing density of survey-data makes it even more complex. Multiple researches aimed at automation of the process, but none of them have reached a fully integrated method yet. Therefore this research aims at proposing a methodology to take it one step further towards automation.

1.1 Generalisation constraints

Generalisation in its simplest form — reducing detail — may sound easy, and in some cases it is. For example, a lamppost will be reduced to a point-feature at relatively large scale maps. Also, a group of buildings will gradually move from a collection of separated polygons; to grouped polygons; to an outline of a city; to a single point and may finally disappear from the map if the city is not relevant at a certain (smaller) scale. What makes generalisation a challenge are the complex choices you should make in this process. For the lamppost: will you draw the point in its center of gravity, the location of the light, or at the point it touches the ground? And for the buildings: which buildings do we take together at which scale and where and when do we place the point resembling the entire city; is it the central square, geometric center or something else like where most people will enter the city?

In a general topographic map the information is usually not very critical. For a pedestrian it is not very critical to have the lamppost resembled in the map exactly on its actual position. When a person is walking by, it will immediately see its actual position and adapt its route to it. However, in navigational charts some of this generalised information *is* very critical, especially sub-surface information. The sub-surface features simply cannot be seen or sensed, and making small mistakes may result in catastrophic failures. Cartographers and agencies responsible for these publications therefore pay a lot of attention to this generalisation process. For isobath generalisation four main constraints are identified and described one should take into account while generalising. Isobaths satisfying these constraints are seen as *good* navigational isobaths. The constraints are clearly described by Zhang and Guilbert (2011) and can be summarised by:

- **Legibility** Individual isobaths should be clearly legible and groups of isobaths should be clearly distinguishable.
- **Functionality** The information extracted or interpreted through the isobaths should be *safe*, i.e. an isobath may never indicate an area being deeper than what was actually measured or may be expected.

- **Morphology** The overall shape and significant features of the seabed should be maintained as much as possible.
- **Topology** Isobaths may not touch or intersect each other.

The legibility constraint makes sure the information can be easily and intuitively interpreted. In one glance, the navigator should be able to establish a general overview of the information contained. Isobaths should therefore be simple — or smooth — enough and they may not visually overlap[5]. Since navigational isobaths are not labeled, their enclosed area should also be large enough to contain at least one sounding symbol.

The functional or safety constraint is of major importance for navigational charts. This enables safe navigation and reduces the risk of grounding due to invalid charts. A safe chart implies a ship sailing on the deeper side of an isobath will never encounter a depth shallower than the isobath-value. This constraint makes the generalisation of navigational isobaths significantly different compared to topographic generalisation.

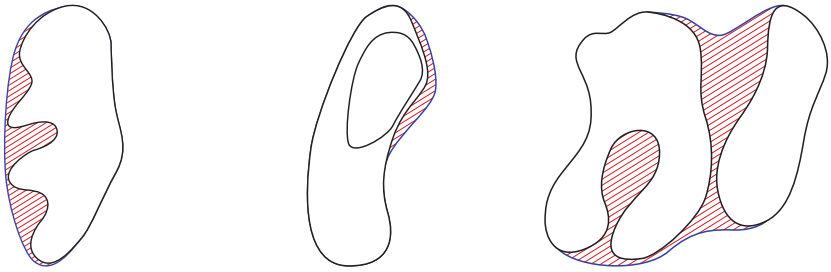
After generalisation the morphology of the seabed should still be representing the actual situation as much as possible. Significant features like slopes, peaks, pits should be preserved as well as the indicated depth being as accurate as possible. A navigator may use these to locate themselves or selecting anchor-grounds.

Related to the morphology is also the topological constraint of maintaining relative distances, e.g. the location of the deepest point in a fairway or separation of two peaks. This may be mainly a morphological constraint, but topology in this case states that e.g. the deepest channel should run eastwards of the peak, and not suddenly to the west of it. Finally, generalised isobaths should also be geometrically valid: no isobaths may intersect or touch and it should be relatively easy to deduce equal-depth areas from them.

1.2 Problem statement

MBES survey-data brings an increasingly dense and complete coverage of the seabed. Still, hydrographic offices have realised bathymetric data is currently not sufficiently represented in ENCs (Moggert-Kageler, 2018). Peters et al. (2014) already posed the generalisation constraints may be incompatible with each other: it will be difficult to perfectly satisfy all of them. While the safety and topological constraints are hard constraints and can be evaluated quantitatively, there is room for compromise between the legibility and morphology constraints. They are however not independent of the prior two, as well as chart scale and isobath value.

The incompatibility between the constraints especially manifests itself in masking deeper waters. On relatively steep slopes (dependent upon scale) horizontal separation between two consecutive isobaths can be very small. Legibility constraint dictates they may not visually overlap and thus the deeper isobath is pushed towards a deeper area (due to the safety constraint) while retaining its original depth-value. At sea this results in a decreased area of deeper areas (Figure 1.2ab), while in narrow approach channels or fjords entire navigable channels may disappear (Figure 1.2c). This can lead to dangerous situations with ships crossing safety-margins or entirely disregarding the actual



(a) Line smoothing (b) Line displacement (c) Aggregation

Figure 1.2: Safe generalised isobath (blue) of the original isobath (black) results in masked waters (red). Prior to generalisation they were navigable, or at least deeper than the current representation.

approach channel (UKHO, 2018; AHO, 2019). There is thus an economic benefit in optimising the generalisation process and finding a *best* compromise in the constraints.

Obtaining *good* isobaths and thus finding a good compromise in these constraints is currently done manually or at least in need of manual interventions. Reducing manual interventions may of course reduce personnel costs, but also increase efficiency of the process (Mellor, 2018). Being able to process large datasets more efficiently also implies the information is reaching the navigators more quickly and thus enabling a safer chart with more up-to-date information.

Multiple efforts are undertaken to automate this process (see Section 2.5). However none of them are yet fully automatic or integrate the generalisation constraints and all of them have their particular implications. This research tries to overcome these defects by directly integrating the four generalisation constraints (resembling chart representation) in the generalisation process which takes place on the survey-data in its purest form. Meaning most of the survey-information can be kept in the process, rather than generalising some in-between representation (like grids).

1.3 Research objectives

In this research we will investigate what is needed for and which methods we may use for a truly automatic isobath generalisation method. An ideal outcome would be a process without any human interference or subjective parameter choices. By achieving this, the generalisation method will in theory always result in the same outcome, no matter when and by whom it is executed. It is almost impossible to end up with a production-ready software package within the timespan of this thesis, but at least we will have established a basic framework, identified promising approaches and documented the choices throughout.

The challenge behind such an automated generalisation process is mainly in the integration of *all* generalisation constraints. Due to their incompatibility we need to find means for a *best* compromise or optimisation of those. The challenge is therefore actually two-fold: quantify the constraints; and relate those quantifications to the survey-data. By

establishing a truly integrated approach we can better steer various generalisation operators, and possibly stop generalisation when it is not needed anymore. We will have less over-generalised features, meaning more of the actual morphology is present in the representation. The seabed shape will therefore be more accurately described, possibly giving us the safety and economic benefits described in Section 1.2. The safety and topology should be satisfied by definition, thus only leaving room for compromise in legibility, morphology and possible human intervention. We can summarise the research objectives and its challenges in:

- reduce human interference,
- quantify and integrate generalisation constraints,
- evaluate metrics with respect to the constraints directly within the process, and
- establish metrics and apply operators locally, rather than on the entire dataset.

Research questions

The objectives above are the ideal outcome of the to-be proposed methodology. However it is at this stage not clear whether this can actually be achieved within the timespan of the graduation. Therefore this research will explore possibilities to reach those objectives and at least answers the main research question:

> *To what extent can we locally steer generalisation operators to account for cartographic constraints, in a surface-based isobath generalisation method?*

The main question can be broken down into sub-questions, each covering different aspects of the generalisation process:

- *What are the minimum legibility requirements for navigational isobaths, cartographic and legally at different chart scales?*
- *How can we quantify the cartographic constraints into local surface metrics?*
- *What is the effect of applying different local operators on the global surface, and how can this be exploited?*
- *What are valid and realistic assumptions on input data in the field of application?*
- *How can the extracted features be validated and does the method perform better than available alternatives?*

The first question addresses the desired outcome of the isobaths for different features at different scales. What is the isobath interval, line thickness and smoothness as well as which features to preserve, amplify or discard? Then by answering the second question, these conceptual properties should be quantified into (local) metrics to be useful in a digital evaluation model. Since it is not efficient to consecutively process, extract isolines and evaluate their metrics it is also tried to establish these metrics directly on the underlying conceptual surface. Then to overcome possible conflicts identified through evaluation, the effects of different operators are explored together with their influence on the metrics. The fourth question addresses assumptions on input data; what are typical spatial distributions, what information do they contain and what is the implication

of different spatial distributions on the expected result? Finally, it is important to validate the results. Firstly against the generalisation constraints, but possibly also against other implementations to explore the differences and say something about its relative performance.

1.4 Scope

Time for this graduation research is limited. It is not possible to cover every aspect of chart compilation or isobath generalisation. The research will therefore be focused on certain aspects within this process. Focus areas are defined using the prioritisation method of MoSCoW. This section elaborates shortly on them, a complete list was established during preparation of this thesis[6].

The field of isobath generalisation is a small one, a niche. There is actually lots of commercial activity and interest — since IHO member states are obliged to chart their waters — but not much available literature on the associated topics. Therefore significant amount of energy must be put in formulating and formalising the cartographic constraints and conceptual framework. Some existing isobath generalisation approaches will be investigated and then the main focus of the research is to design and implement a conceptual framework for a novel integrated approach. Forming and documenting this framework is of prime interest, in which some basic constraints and metrics will be implemented as a proof of concept. We will make sure this framework can later be extended.

The proof of concept will be limited to projected vector data (points) as input and should be able to output simple line features. Input is assumed to be statistically cleaned depth measurements with one simple attribute being a depth value. It should not matter whether it is acquired through SBES or MBES. We will also develop methodology to discuss and validate the overall constraints, as well compare the outcome to existing approaches. Focus is on the actual methodology and a proof of concept. In this stage we will simply disregard computational efficiency.

Once the basics are in place and validated, it is possible to extend the methodology with more functionality later. While the conceptual framework should be able to support these extensions, they are for now not regarded. Think for example of: more complex metrics, breaklines, coastlines, boundary problems, support gridded data, superimpose tidal data, updating of the database, incorporate survey-statistics, etcetera. We will for now only focus on extraction of isobaths and not depth areas or soundings.

1.5 Thesis outline

We have now introduced the field of application and context of this research.

In Chapter 2 we will continue on this in a more elaborate way. First the specifics for navigational charts — its legal framework, compilation and visualisation — are discussed in Section 2.1 and 2.2. From Section 2.3 onwards a broader view permits us to summarise

and describe existing research and methodologies. This comprehends both research in the field of isobath generalisation as well as more general geo-spatial theories like terrain representations, triangulation structures and efficient isobath extraction methods.

Chapter 3 proposes the novel integrated approach for isobath generalisation. Mainly a new data structure is introduced to link survey data to the final cartographic product. This data structure can then be used in various ways for interacting with the data, we elaborate on a possible model to do so. It is this chapter where most of the novel theories are presented and discussed.

The proposed methodology is implemented as a proof of concept and Chapter 4 contains specific information about this implementation. Although most of the theories are presented in Chapter 3, this chapter also explains why some choices were made specifically.

Chapter 5 is all about the results obtained through the proposed methodology and analysis of those results. In Chapter 6 these results are used to conclude this research. The overall process is discussed and we take a view on the overall contribution to the field of application; the methodology's limitations and what the possibilities are to take it one step further in the future.

2 Theoretical background

This chapter presents an overview of available literature and methodology which is later used to develop the new methodology in Chapter 3. The chapter starts with more formal background information on navigational charts and how they are currently being generated and compiled in Section 2.1. This gives us the context of what information should be present. However the complex generalisation choices are presented separately in Section 2.2. This section also gives some examples on desired results for navigational charts and defines what the methodology should work towards. With the full context on navigational charts in place, from Section 2.3 onwards, we take a more theoretical look on the processes. This overview starts with backgrounds on datastructures for surface representations in Section 2.3 and continues with contour extraction in Section 2.4. Section 2.5 presents an overview of the state-of-the-art in (automatic) isobath generalisation.

2.1 Chart compilation

Basically, a navigational charts conveys some information through some visualisation principles. For navigational charts these aspects are formalised and discussed below. However, a navigational chart is not simply one chart, it consists of a set of charts, usually at different scales with different purposes.

2.1.1 Chart scales

Navigational charts are compiled at different scales regarding their intended use. These scales are referred to as compilation scales and may be differently interpreted for Raster Navigational Charts (RNCs) and ENCs (IHO, 2018b). For RNCs or paper charts this is the scale on which the information is actually displayed or printed. A user cannot change this scale anymore after compilation. For ENCs the compilation scale represents the optimal scale for which the information was designed. Since for ENCs the actual displayed scale can be changed by the user, sub-optimalities may be expected and accepted.

Large scale charts contain the most detailed information and at smaller scales details will be generalised based on their significance to the navigator and the general purpose of the chart. Any information on small scale charts should therefore be present in the larger scale equivalent, but not necessarily the other way around. Each stack of charts covering the same area at different compilation scales should thus be vertically consistent and should be achieved by first compiling the large scale charts (IHO, 2018c).

Table 2.1: Standard chart compilation scales and their purpose. *Adapted from IHO (2018c)*

term	name	purpose	min	max
Small-scale	Overview		<	1:2 000 000
Medium-scale	General	passage, landfall	1:2 000 000	1:350 000
	Coastal	coastal navigation	1:350 000	1:75 000
Large-scale	Approach	port approach, congested waters	1:75 000	1:30 000
	Harbour	harbour, anchorage, straits	1:30 000	<
	Berthing		very large scales	

Table 2.2: Standard selectable radar ranges and consistent chart compilation scales. *Adapted from IHO (2018d)*

Selectable radar range	Standard scale
200 NM	1:3 000 000
96 NM	1:1 500 000
48 NM	1:700 000
24 NM	1:350 000
12 NM	1:180 000
6 NM	1:90 000
3 NM	1:45 000
1.5 NM	1:22 000
0.75 NM	1:12 000
0.5 NM	1:8 000
0.25 NM	1:4 000

IHO (2018c) proposes a list of different purpose charts and their compilation scales. These can be found in Table 2.1. IHO does not require charts being compiled at an exact scale, as long as it is a multiple of 1:1 000 or 1:2 500. This gives national hydrographic offices the freedom to adapt their compilation process to already present charts and information. But more importantly, most of the electronic charts are used in an ECDIS in which the user has more control over the visualisation scale. With such control over the charts, some publishers even incorporate `scamin` attributes on particular features. A `scamin` value dictates at which scale a feature is still visible. Such an attribute may yield different visualisation for different scales, even within one chart folio at a particular scale.

Hydrographic offices thus do have lots of freedom in choosing compilation scales. But since an ECDIS is also able to display additional radar information, they do however recommend charts to be compiled in consistency with standard radar ranges (IHO, 2018d). These consistent pairs can be found in Table 2.2 and enable the most efficient visualisation at different selected radar ranges.

2.1.2 Depth information

Depth information must be represented by isobaths, soundings and depth areas (Figure 2.1). Isobaths being lines of equal depth, separating two depth areas and soundings being selected point-features typical for the surrounding seabed. For now, isobaths are

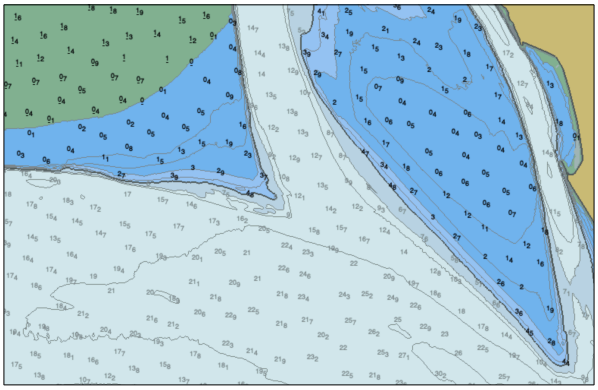


Figure 2.1: Available depth information in an ENC: soundings, depth contours (gray lines) and depth areas (coloured polygons). The safety contour (thick black) is also displayed.

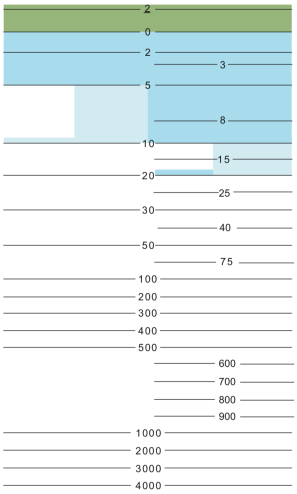


Figure 2.2: Standard IHO isobath series. *Standard series* on the left, complemented with the *complementary series* on the right. *Taken from IHO (2018c)*

of main interest in this research but since these are directly linked to the depth areas we will not forget about those either. IHO (2018c) defines isobath values in three different series (Figure 2.2). The *standard series* comprises the values of 0m, 2m, 5m, 10m, 20m, 30m, 50m, 100m, 200m, 300m, 400m, 500m, 1000m, 2000m, continuing with intervals of 1000m. With 0m only applicable to tidal areas and the 2 and 5m isobaths may be omitted if they do not serve the purpose of the chart. If the data permits, and over large almost horizontal areas, *supplementary* isobaths may be used. Examples are 3m, 8m, 15m, 25m, 40m and 75m. The 2500m isobath has legal relevance related to the continental shelf, but is not of particular interest in shipping. In shallow areas with sufficiently accurate data, also the 4 and 6m isobaths from the *other* series may be shown. These should however be labeled with their value. National or port authorities are free to produce other nautical products like high-density ENCs. These usually have isobaths every meter. However, these kind of products are supplementary to the required international charts and may have totally different users and requirements.

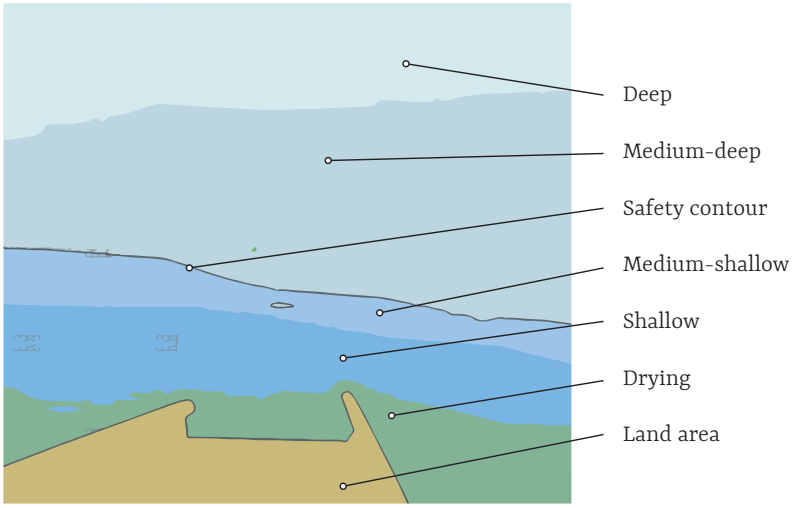


Figure 2.3: Depth zone shading and the safety contour. From bottom to top: land area (yellow), drying (green), shallow (dark-blue), medium-shallow (blue), safety contour, medium-deep (light-blue) and deep (white).

Isobaths gives the mariner a direct view of up to which line it is possible to navigate and a general overview of the seabed morphology. For example, a navigator knows if the ship's draft is 12m it should pay particular attention to the isobaths of 15m and it simply cannot go up to the isobath of 10m. Any area on the deeper side of the 15m isobath is safe to navigate. In an ECDIS it must be possible to select this so-called *safety contour* (IHO, 2015). In the example of the ship mentioned, a logical choice is the 15m isobath. If this isobath is not available in the particular chart folio, automatically the next deeper isobath is selected. Thus guarantying the safety constraint. This safety contour is not only emphasised upon display, it can also be used to differentiate depth zones. If also a shallow- and deep-water value is selected, the ECDIS can differentiate five different depth zones: deep, medium-deep, medium-shallow, shallow and drying (Figure 2.3). These depth zones are all visualised with a different color (green, dark-blue, blue, light-blue and white) and give the navigator a clear overview of the situation.

Isobaths are complemented by soundings. They should be strategically selected and are shoal-based. Shoal-based is being tested by drawing a triangle or line between some selected soundings, and by no means another sounding within this triangle or segment may exceed the selected soundings (Kastrisios and Calder (2018); Kastrisios et al. (2019)). However, not only shoal soundings should be selected. There should be sufficiently enough deep soundings as well to resemble the actual morphology and assist the mariner in passage planning. (IHO, 2018c). Thus they should indicate most of the isolated peaks, but also the pits. The latter especially relevant in fishing or anchoring.

The recently adopted S-100 standard (IHO, 2018a) enables a larger variety of hydrographic data including imagery, gridded bathymetry and irregular terrain representations. The new possibilities however will not exclude the *old-fashioned* way of depth information since the main purpose of a chart is clearly conveying information and extensive data display may counteract that. It however enables a more complete usage of hydrographic data next to navigation if that is needed in particular situations like fishing, anchoring and research.

Table 2.3: Minimum ECDIS display requirements and an example for the display with minimal requirements. *Adapted from IHO (2015)*

	property	required	unit
Minimum requirements	Size	270 x 270	mm
	Resolution	864	lines/s
	Colours	64	
Standard display	Lines	3.2	lines/mm
	Pixelsize	0.3125	mm

2.1.3 Display and symbology

Commercial vessels are required to carry up-to-date official navigational charts. They may be in the form of analogue printed charts or ENCs/RNCs in two independent ECDISs (IMO, 1974). An ECDIS is always an officially approved system with minimal display requirements set by IHO (2015). These requirements can be found in Table 2.3.

The same standard defines minimum requirements for symbology. These are based on being easily interpretable at the operational viewing distance. Usually this is 70cm for planning, but may be larger in other situations. As a rule of thumb IHO states symbols should be approximately 4mm in size, or at least span 12 pixels. Although we will not handle chart symbols directly, they may be important in determining minimum size of a peak since small areas of isobaths should at least contain one sounding symbol. There is yet no standard width for digital isobaths, but for paper charts IHO (2018c) states isobaths should be continuous black lines with a width of approximately 0.1mm. The Netherlands Hydrographic Office (NLHO) uses a width of 0.3mm for all its products. It is discouraged to emphasise important isobaths by making them thicker. Instead depth-zone shading should be used. Depth-zones described above all have different specified colours (Figure 2.3).

The combination of symbology, chart scale and display requirements is always in relation with the legibility constraint on chart generalisation. The combination of on-screen line-width and chart scale results in implications for the covered area in the real world. Take for example a line-width of 0.1mm. At a (large) scale of 1:45 000 this line will span $(0.1mm * 45\,000 =)$ 4.5m in the real-world. Exactly the same line at a (small) scale of 1:700 000 will span $(0.1mm * 700\,000 =)$ 70m. now relating this fact to the legibility constraint, assuming a horizontal separation of 20m between two isobaths with respect to their centerlines. In the large scale chart the two lines could still be distinguished from each other, leaving 15.5m space in the real world or, 0.34mm $(15.5m/45\,000)$ on screen. On the small scale chart they cannot be distinguished anymore, they start to form one thicker line of 0.13mm wide $((2 * 35m + 20m)/700\,000)$.

2.2 Chart generalisation

2.2.1 Generalisation principles

It must be avoided that the mariner is faced with too much or irrelevant information. The *amount* of information mainly depends on the chart scale. That is the property defining how much space is left on a visualised chart. What is *relevant* information depends on the purpose of the chart (Table 2.1). It is simply impossible to display all information at each chart. Therefore generalisation of features is necessary and is different for each scale and purpose. IHO (2018c) states different purposes of generalisation, as well as risks and considerations for this process:

- Generalisation should eliminate the least essential information, by smoothing lines or omitting irrelevant features.
- We can differentiate *full* and *minimal* depiction of detail. Full depiction is usually impossible due to the available space on a display in combination with the symbology-requirements. With minimal depiction some details are omitted while there is actually space on a chart. However, it may be useful in forcing a mariner to use a larger scale chart. This can for example be seen in harbour approaches.
- Omitting too much detail at small scale charts can be inconvenient in planning a route across large areas. A mariner would need lots of adjacent large scale charts.
- Deeper parts are less important in terms of safety. Still they should be reasonably displayed to indicate areas where local traffic can be expected, as well as using the chart in combination with the on-board echo sounder for positioning purposes.
- Generalisation of isobaths is shoal-based; they are always pushed towards deeper parts. This resembles the safety constraint.
- Morphology should be respected as much as possible, e.g. a steeply rising pinnacle may not be generalised by slowly sloping isobaths. It may endanger a ship slowly approaching it and expecting a beach-like structure instead of a rock.
- Smoothing of isobaths is only done if the original (jagged) isobath would confuse the mariner.
- The jagged nature of isobaths may actually reveal something about the accuracy and spatial distribution of the underlying survey.
- While generalised isobaths must be safe, it is also important to keep sufficiently deep navigable channels. A harbour's access-channel disappearing from the chart is not desired. A navigator still knows it is accessible and prepares a route, however now without proper charts.

Also, van Kuijk and Engels (2017) from the NLHO supply similar generalisation guides, having a more practical approach:

- Is detail of any purpose, e.g. is a gully wide enough to be navigated?

- Is the amount of detail resembling the nature of the terrain? Can you suggest a slowly sloping beach while in reality it is a cliff? And the other way around.
- Are all soundings contained in their safe isobath?
- Are all isobaths geometrically valid, e.g. non-intersecting, closed, not intersecting land or piers, etcetera.
- Isobaths around (isolated) shoals should always be sufficiently large. And spurs should always contain a sounding value, especially if it is a single one in a fairway.

All those guidelines seem reasonable. But on closer inspection some of them contradict or are not that *hard* as it seems. And especially the NLHO guidelines are questioning guidelines. They let over control and the actual choices over to the cartographer. That is not a surprise. Generalisation is a subjective art with lots of complex choices and it is therefore very difficult to set a list of hard requirements in each situation. For example near fairways or anchorages we want more detail than in nature reserves. An isobath of 250m depth is not as important as the ship's safety contour, etcetera. The role of the human cartographer cannot be underestimated, and this is also why at the NLHO these isobaths are still manually drawn.

2.2.2 Generalisation process

In Section 1.1 we have already noticed generalisation is not merely one operation, but a complex process. According to Weibel (1997) generalisation can take place at different levels in this process. Being *object-*, *model-* and *cartographic* generalisation. Object generalisation works on an object — e.g. generalising a perfect circle to a polygon —, model generalisation deals with the integrated computer representation of sets of objects, and cartographic generalisation also takes into account the scale and symbology of the entire data. It is this last type of generalisation which makes isobath generalisation a particular challenge. Isobaths may be seen as isolated objects, but actually together they form a field representing the seabed morphology. Thus while we are dealing with isolated objects, we may not forget about their relation with the whole.

Harrie and Weibel (2007) define three basic options for the actual generalisation *process*. These are summarised as follows. In condition-action modelling some conditions are tested, and if appropriate an action is triggered. In human-interaction modelling there is a synergy between computer and user. The computer supporting the human in objective tasks while the human addresses the subjective tasks. The last one is constraint-based modelling, which is very similar to the condition-action modelling. However rules in the conditions can be seen as what has to *happen* and constraints define what should be *obtained*. Also, with constrained-based modelling we do not necessarily follow a fixed, linear path of operations but the integrated synthesis is able to automatically detect what should happen. The current process at NLHO is a human-interaction model. There is always a cartographer doing the *thinking* but being supported by automated information from the the computer. For example the computer supplying a list of peaks and pits with their area, and the human selecting which pits to delete and which peaks to enlarge.

In each approach tasks should be executed, whether these are for identification of conflicts or reparation of those conflicts. McMaster (1992) has identified multiple types of such tasks in the categories *cartometric evaluation* and *spatial transformations*. Peters (2012) actually gives a very nice summary of the spatial transformations applicable or useful particular for navigational isobaths. These are visualised in Figure 2.4. What should be noted from this figure is that all operators are *safe*, meaning the lines are always shifted towards the deeper parts. And most operators work on the object-level, except for the aggregation. The aggregation operation does not only alter its own geometry, it affects others in the set as well. These operators are part of the spatial transformations. The cartometric evaluation tasks are discussed in Chapter 3.

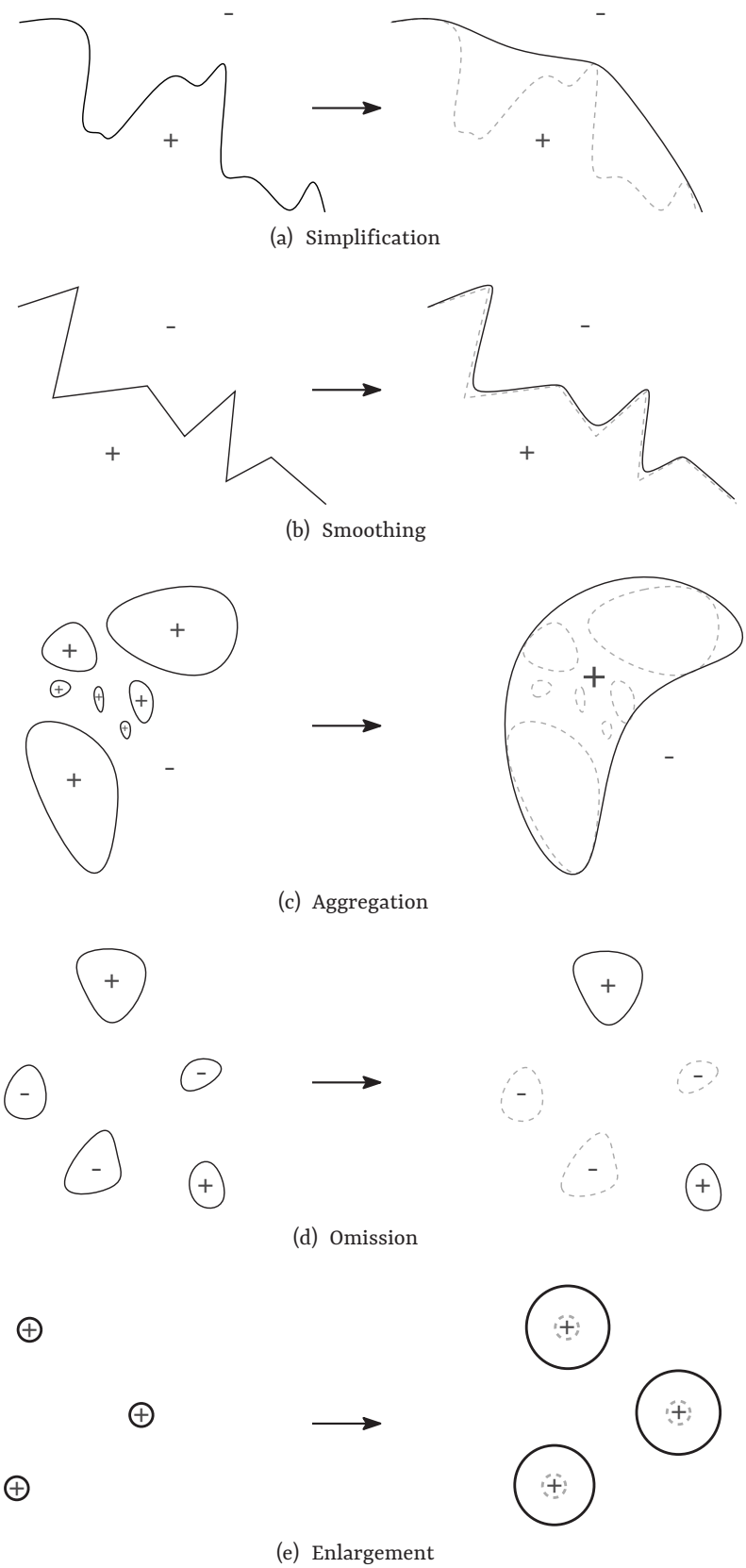
2.2.3 Desired results

Displaying bathymetry may serve different purposes. Already within the field of navigation alone different needs exist between for example overview charts or piloting in hazardous waters. A chart should therefore be generalised with the general purpose of the chart in mind and this can be different for changing scales, areas and type of users. It is difficult to capture the needs in objective computer-rules and the insight and experience of the cartographer cannot be underestimated.

Take for example the two charts of the same area in Figure 2.5, both compiled by official authorities. The NLHO 1800 series[7] is designed to serve a broad range of users while the Schelde ENC is particularly aimed at pilots. The isobaths in the Schelde ENC are much less smoothened or aggregated. They display more morphological detail at the price of being less legible. The 1800 series chart is compiled with lots of manual intervention, but this makes it possible to leave out some detail which is not relevant for the intended users. Isobaths are overall more smooth, aggregated and small indents are removed if they do not allow navigation at all. The drying green regions are enlarged, small indents on banks are removed and shipping direction is more or less taken into account. The Schelde ENC can be automatically generated rather easily, but may violate legibility constraints for general overview charts.

The desired result for this project is to automate the process of going from such representation to the representation in the NLHO 1800 series. From that statement we can already conclude the resulting isobaths should be smooth; indents/gullies too small to be navigated should be removed; features unable to be distinguished from each other due to line-width should be removed; shallow areas could be enlarged; small deep areas (pits) may be removed; peaks separated by a channel/saddle which is not likely to be navigated should be aggregated.

Figure 2.5 shows an example of different chart purposes and their amount of generalisation. In Figure 2.6 the same isobath value (18.2m) is extracted from three different purpose charts (overview, coastal and approach) with different scales. It is real data as extracted from National Oceanic and Atmospheric Administration (NOAA) ENCs. The figure clearly shows more detail in the lines with an increasing scale. Lines are smoothened, gullies are removed, features are aggregated and shoals are enlarged; a nice example of real-world isobath generalisation.



(a) Simplification

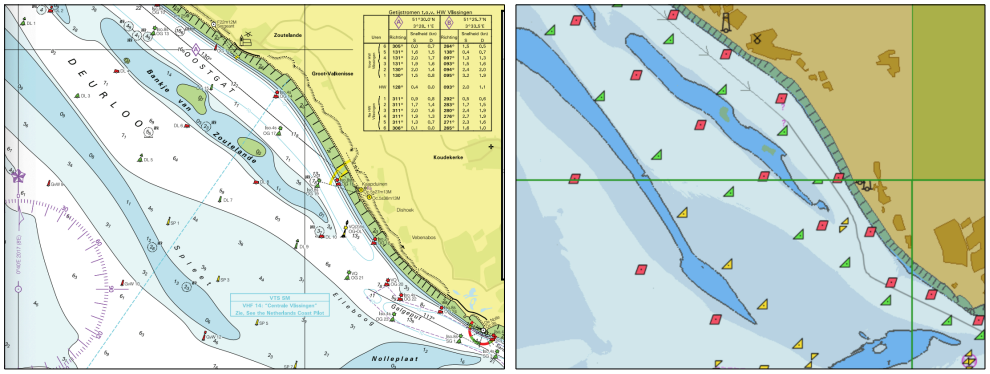
(b) Smoothing

(c) Aggregation

(d) Omission

(e) Enlargement

Figure 2.4: Safe generalisation operators for navigational isobaths. Note that '–' indicate deep waters and '+' indicate shallow waters. *Taken from Peters (2012)*



(a) NLHO 1800 series (b) VTS Schelde ENC

Figure 2.5: Different depth representations in different chart products of the same area. *Taken from NLHO (2019); VTS-Scheldt (2019)*

2.3 Surface representations

Surfaces can be represented by either gridded data, a set of contour lines, triangulations and possibly even point clouds[8] (Figure 2.7). For reasons explained in Chapter 3 we will focus on triangulated representations for now. Also, this project does not aim at the efficient generation and updating of (Delaunay) triangulations, it is however extensively used as a tool within the process. It is assumed there is already basic knowledge on triangulations. Therefore, this small section will give a summary of the tools and definitions used in the rest of this report.

Datasets acquired through echo sounding are usually irregular, they are not captured in a well-defined grid. We will therefore deal with triangulated irregular networks (TINs) which are usually the Delaunay triangulation (DT). In a triangulation we can define *faces*, *edges* and *vertices* (Figure 2.8). The DT is a special triangulation. It is usually unique[9] and it has some nice spatial properties. It tries to minimise its circumcircle by maximising the smallest angle in each face. Thus creating faces which are as equilateral as possible and having the most regular spatial distribution throughout the data. The DT also has a dual in the form of the Voronoi diagram (VD) (Figure 2.8). The VD consists of cells belonging to exactly one vertex in the DT. These cells have a special property, being every point within that cell is closer to its DT vertex than any other in the set.

TIN data structures

A TIN can thus be seen as a collection of related faces, edges and vertices. We can easily store all these features, or only the faces as *simple features*. This is a common technique for visualisation purposes, because it is very simple, has no overload and is sufficient for this purpose. However, if we would like to interact with the triangulation to for example find neighbours of a triangle, incident triangles to a certain vertex or walk across a TIN, we can better store the it with a more appropriate (topological) data structure.

A very common and topological data structure is the triangle-based. This structure stores for each triangle exactly three pointers to its vertices and pointers to its neighbouring triangles (Figure 2.9). Note the order and symmetry of the two pointer tables:

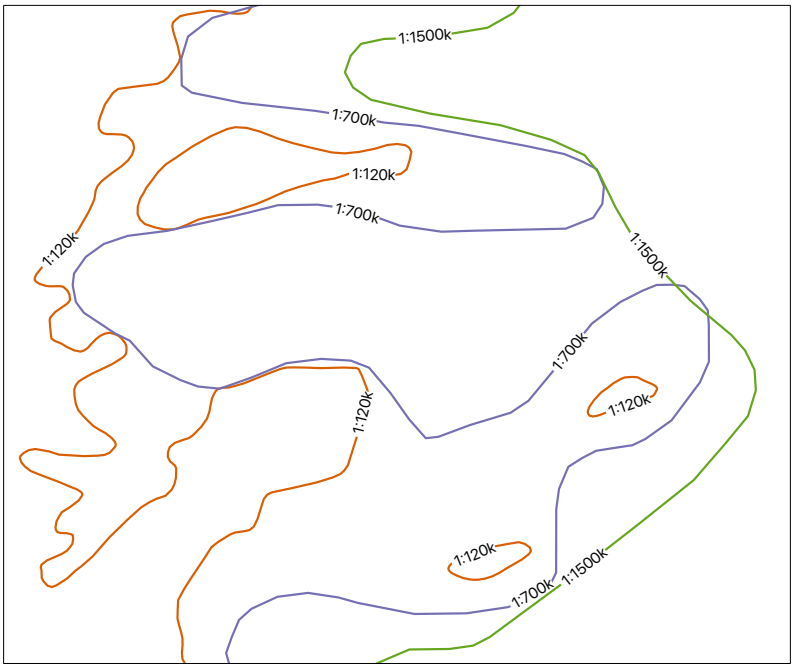


Figure 2.6: The same isobath generalised at different scales. Deeper parts of the seabed are on the right side of the figure. From large scale (orange), medium scale (purple) to small scale (green). *Data from* NOAA

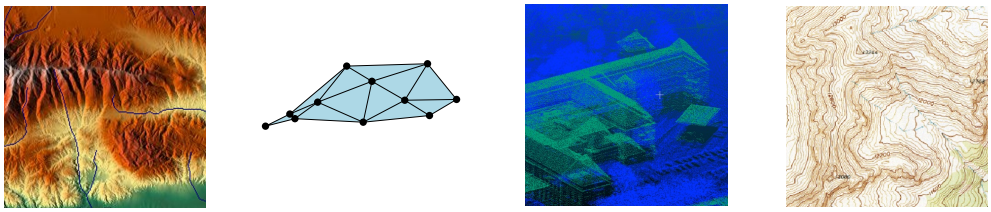


Figure 2.7: Different types of surface/terrain representations. *Taken from Ledoux et al. (2019)*

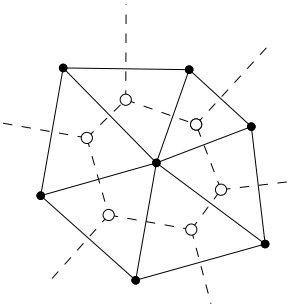
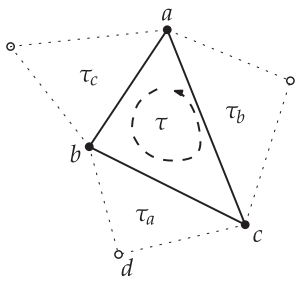


Figure 2.8: The vertices of the TIN (black dots) with its DT in black and the dual VD in dashed lines. Note the triangulation edges (solid black lines) and faces (triangles enclosed by three consecutive edges). *Taken from Ledoux et al. (2019)*



triangle	v_1	v_2	v_3	adj_1	adj_2	adj_3
τ	a	b	c	τ_a	τ_b	τ_c
τ_a	b	d	c	$\tau_{...}$	τ	$\tau_{...}$
...	...	...	...	...	...	...

Figure 2.9: The triangle-based data structure. Each face in the TIN is stored as a triangle with three (ordered) pointers to their vertices and three ordered pointers to their neighbouring triangles. Vertices are stored only once, but in a separate table. *Taken from Ledoux et al. (2019)*

vertex two is directly opposite of neighbouring triangle 2 and all of them are ordered counter clockwise (CCW). It is a more complex structure than *simple features*, however it makes interacting with the TIN significantly easier. Neighbouring faces of a triangle can be directly accessed. There is no need of checking touching relations between every face anymore. Also, shared vertices are not stored redundantly anymore. While this property needs more code to retrieve the actual geometries, it saves disk space but also makes it easy to update an attribute of a vertex. We have to only update one feature in stead of visiting every triangle sharing that particular vertex. Also note that if a triangle has less than three neighbours, it is on the boundary of the triangulation.

Other common data structures rely on the edges in the TIN, rather than the faces: half-edge (Muller and Preparata, 1978), winged-edge (Baumgart, 1975) and quad-edge (Guibas and Stolfi, 1985). Basically these structures store all edges with pointers to their vertices, pointers to the next and previous edges and possibly a pointer to their left and right face. They are more challenging to implement than the simple triangle-based method but can be more convenient if you for example plan to walk over the edges in a terrain.

Another structure does relate to the vertices in the TIN. This *star*-structure is described in Blandford et al. (2005) and Ledoux and Meijers (2013). The star-structure stores for each vertex its geometry and an ordered list (CCW) of incident edges to that vertex. An incident triangle to the vertex is implicitly referenced as two consecutive edges to that vertex. The missing *link* is computed from the two adjacent edges automatically. The beauty of this structure is for our purposes mainly in the direct relation between Delaunay neighbours. It can directly access all natural neighbors in the triangulation and could therefore be useful in Laplace-interpolation which relies on the distance between two neighbouring vertices. Also, it can be used to directly compute a Voronoi cell for a vertex. And again, because the vertices are only stored once, upon updating one vertex we will only have to visit one feature instead of a bunch of incident faces.

2.4 Contouring

Contour lines are widely used to display terrain and can be a convenient way to visualise overall shape of the terrain. However not all details of the morphology are maintained, simply because there is no information displayed in between contours. This can be a problem when one wants to know every small detail, but is usually not needed and then an overall idea is sufficient. Also if more detail is needed, contouring interval may be decreased resulting in more lines.

Contour lines are actually *isolines*, meaning lines of equal height. For underwater features or bathymetry these lines are called *isobaths*, meaning lines of equal depth. Thus, an isobath represents positions of equal depth in a given field $f(x,y) = d$. For example the isobath with value d_i is the line connecting all positions where $f(x,y) = d_i$. However this includes horizontal areas. For an isobath to be valid, also the gradient of that field $\nabla f(x,y)$ may not be equal to zero. This excludes horizontal areas being covered by isobaths. An isobath is therefore every (closed) line with $f(x,y) = d_i$ and $\nabla f(x,y) \neq 0$. Isobaths are typically closed lines, except where they reach the boundary of the surface. Another view on isobaths is to take the 2D intersection at a given value with the surface.

2.4.1 Isobath extraction from a TIN

Extracting isobaths from a 2.5D TIN is conceptually simple: for every face, compute the intersection of the horizontal plane of interest (the isobath value) and the face. Figure 2.10 shows all possible intersections with the isobath plane. From top-left to top-right; no intersection, horizontal triangle at exactly the isobath value and only one vertex intersects. All these three cases may be ignored for isobath extraction. The middle case does have a gradient equal to zero and the third case does simply not result in a line. In the best case, this same vertex is captured by its two incident edges. In the worst case it is an isolated peak. Thus while it is not resulting in an isobath, it may be an important feature to highlight as an obstruction. The bottom three cases are all valid intersections which should be extracted. It should be noted we will only extract the bottom-right situation if the third vertex is deeper than the isobath value. This may be important if the morphology consists of a slope with a terrace at equal height of the isobath value (Figure 2.11). Due to the safety constraint we have to pick the one isobath nearest the deepest part, and not the other (red). As a consequence we may miss some very small valleys with a width of only one edge. On the opposite, we will never miss a small ridge which for navigational isobaths is much more relevant.

The simplest implementation of isobath extraction from a TIN is a brute-force approach. Simply visiting every face and find every intersection with a given value. This approach results in separate line segments which should be merged afterwards. However every face has to be visited and after that the connected line segments should still be found by traversing those again. Another common method is by traversing or walking the TIN. This approach directly creates connected isobaths. It starts from a face intersecting the isobath value, and then follows the line through triangle-neighbours. Obviously it helps to have the triangulation stored topologically in this approach. Following the line is

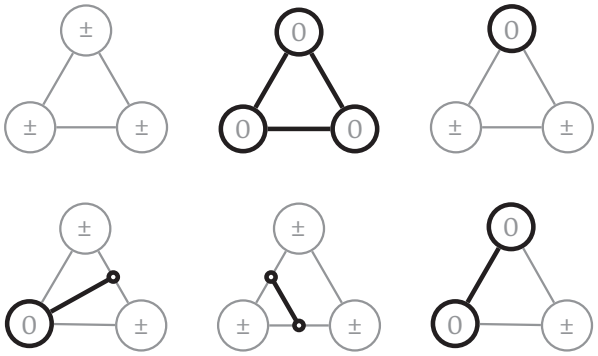


Figure 2.10: Possible isobath intersections with different 3D triangles. *Adapted from Peters (2012)*

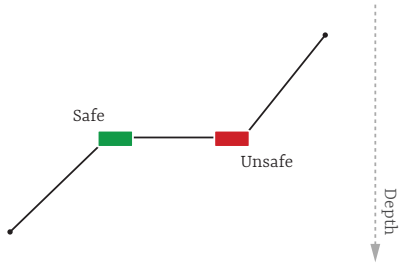


Figure 2.11: Terrace in the morphology with safe (green) and unsafe (red) isobaths. An ordinary contouring algorithm would generate both isobaths.

(a) Valley (b) Ridge

Figure 2.12: Consequence of the 'terrace-rule' on valleys (a) and ridges (b). Some very small valleys will not be captured (red line), while all ridges are still captured (green line).

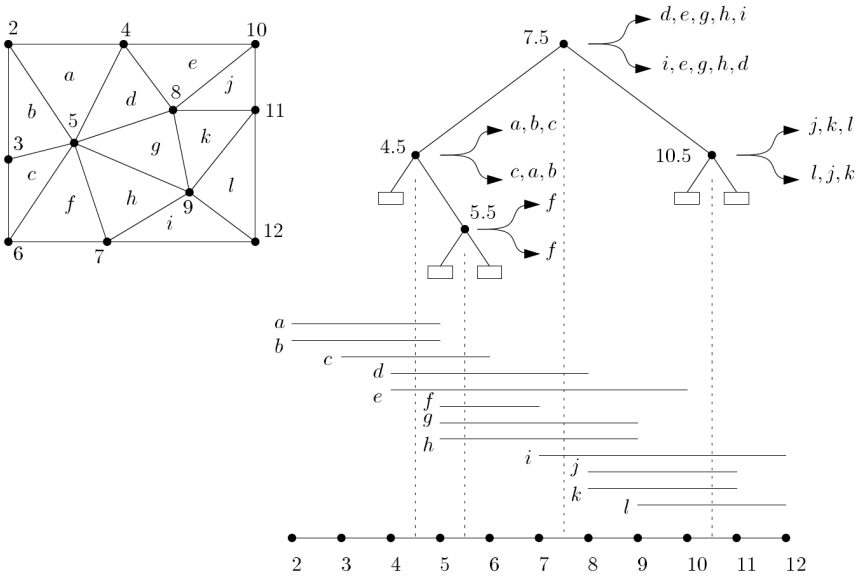


Figure 2.13: Triangulated terrain and its accompanying interval-tree. This tree is generated for the isobath values of 4.5m, 5.5m, 7.5m and 10.5m. From the three faces intersecting a specific isobath value can efficiently be extracted. *Taken from van Kreveld (1996b)*

done until the isobath is closed again or there is no neighbour detected at the end (at the boundaries). Both of these methods are clearly described by van Kreveld (1996a).

Both methods are quite expensive in terms of computational resources. In the brute-force approach every face is visited, also in the worst case that no intersection or very few are present. To overcome this, van Kreveld (1996b) describes a way to structure the faces based on their minimum and maximum values in an *interval tree* (Figure 2.13). Once established, the tree can be traversed to find all faces which actually intersect the given isobath value. This can significantly increase computational efficiency, especially when there are relatively few triangles intersecting the isobath. However, to initially construct the interval-tree each face has to be tested, limiting the increased efficiency when relatively more isobath values are computed. Also, only faces which intersect predefined isobath values are present in the tree. All other triangles which are in between two intersection are not stored and thus you need to predefine the values of interest.

Another approach which automatically select so-called *seed sets* is described in Bajaj et al. (1998). This approach is especially useful in a walking approach described above. With seed set selection a minimal amount of faces is selected from which the walking algorithm can start. It is therefore not needed anymore to visit every face upon starting a new isobath.

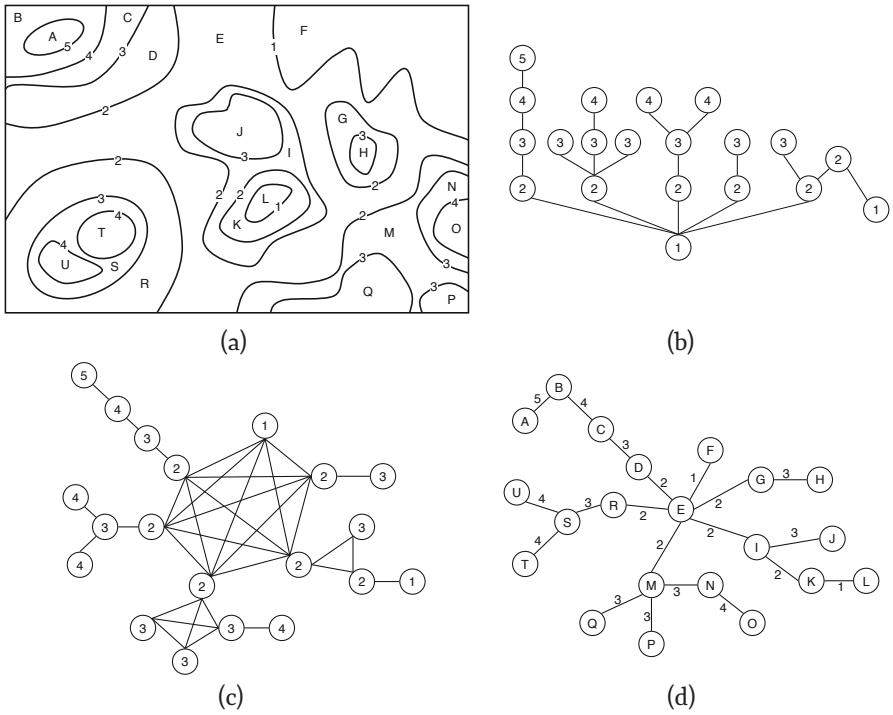


Figure 2.14: Contour map (a) with its corresponding contour tree (b), contour graph (c) and inter-contour graph (d). *Taken from Guilbert and Zhang (2012)*

2.4.2 Structuring sets of isolines

Existing isobath extraction approaches result in isolated isobaths as simple features. Obviously they have their elevation as an attribute, but their relationship as a whole is usually not available. A human may use its perception to identify topographical features, but in order to use that same information in a computer system we have to explicitly structure the set of isobaths as well. It is important we can handle these relationships in isobath generalisation, because as we have seen in Section 2.3, one isobath does not tell us much about the morphology. It is the complete set of isobaths that does.

We have three common structures to capture these relationships: a contour tree, contour graph and inter-contour graph (Figure 2.14). In the context of navigational isobaths, Guilbert (2012) gives a perfect summary of those three.

The contour tree (Figure 2.14, b) depends on containment relations (Roubal and Poiker, 1985). A root node — representing an isobath — is selected and from there each isobath contained by its parent is added to the three. Thus, every node is contained by its parent node. Note that most implementations of this structure are quit simple representations with one distinct feature, or relatively easy separable features on the map (Cronin, 1995; Kweon and Kanade, 1994). In such cases we can easily identify the root node (Figure 2.15). In Figure 2.14 an arbitrary choice was made to select the shallowest isobath as root. All other open isobaths in the set are assumed to be contained by the root. Also we do not have enough information to generate a unique contour tree in the case of non-closed isobaths, as illustrated in Figure 2.16. The figure already shows two different options

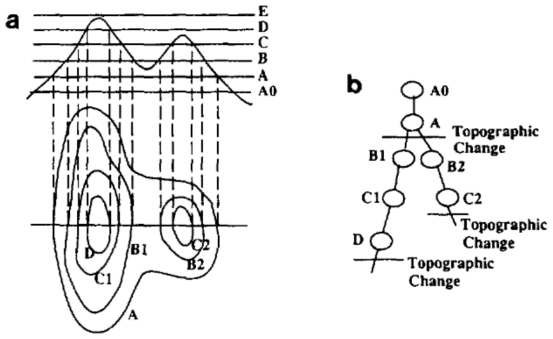


Figure 2.15: Simplistic topography with its corresponding contour tree. The root node is in this case simple to detect. *Taken from Kweon and Kanade (1994)*

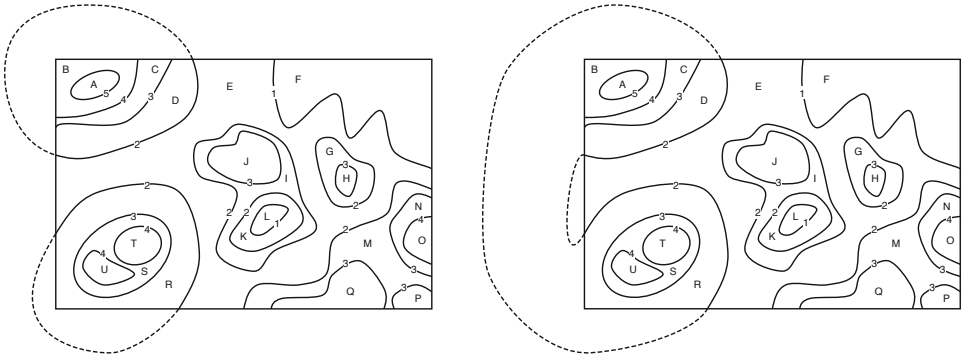


Figure 2.16: Two possible containment situations and thus different contour trees for the same dataset. Figure 2.14b assumes the situation to be as on the left, while the situation on the right is as valid. *Adapted from Guilbert (2015)*

from the finitely many in this example. The contour tree can be a good option if the root node is clearly detectable and if you are interested in containment relationships. However if you are just interested in neighbours of a certain isobath, complex traversal may be needed. A contour graph (Figure 2.14, c) is then a better option. In such graph isobaths are represented as nodes and edges represent adjacency relations. It can handle non-closed isobaths but not all containment relations can possibly be extracted. It is however unique for the dataset and no arbitrary decisions has to be taken. The inter-contour graph — or region graph — is the dual of this contour graph (Figure 2.14d). In the region graph edges represent the isobaths and nodes the region between a set of isobaths. In an abstract way, this region is thus contained by its edges or the isobaths (and possibly the boundary). Since the contour and inter-contour graph are the dual of each other, they share the same properties.

As far as we know, contour trees and graphs are currently only constructed from the contours (or the inter-contour regions) themselves as starting points, e.g. as in Guilbert (2012). It is assumed the geometry or topology of the isobaths will not change anymore and thus usually the link with the underlying terrain can be forgotten about. To establish a tree or graph, usually the extracted isobaths are tested for containment or adjacency using a region-growing algorithm. This can be a tedious task to repeat after each change in terrain, but may also cause problems for non-closed isobaths.

2.5 Generalisation approaches

All official navigational information in The Netherlands is compiled or at least checked manually. In the official NLHO charts almost every isobath is being checked by at least one cartographer. These institutions believe this is still necessary due to its liability but mainly because generalising those lines is still seen as an *art*. Very complex choices from which it is believed it cannot be achieved yet in a computer system alone. Slowly the field is accepting or experimenting with a more automated way of generalisation. For example cartographers do not draw in the first raw isobaths anymore. In the past this was maybe still possible with soundings every few kilometers, but with MBES data this would simply be a very time-consuming task. So at the NLHO they use for example TIN or grid based contour extraction, as well as a first (safe) smoothing step to support them in their work (a human-interaction model). The remainder of the generalisation process is still done manually. It results in differences between cartographers, but mainly is very time-consuming. This has its financial consequences but also results in the raw survey data later being included in the published products.

There is however some research and commercial interest in automating this process more. Currently available approaches can be discriminated in basically two types: line-based and surface-based. These are described separately below.

2.5.1 Line-based approaches

The common denominator of all line-based approaches is it directly and only works on the extracted isobaths. From the survey data an intermediate structure is generated like a TIN or a grid from which the raw isobaths are extracted. This isobath extraction is only done once and the generalisation process only relates to the raw isobaths (Figure 2.17).

Line based approaches are also common in topographic generalisation and for example used in Matuk et al. (2006), Gökgöz (2005) and double buffering. They are not safe by definition but could be adapted to be so. Of the three examples, double buffering is implemented in some commercial packages probably because it is rather easy to implement. While it smoothens the lines overall and can get rid of high frequency *noise* in an isobath, it could still result in sharp edges, circular artefacts and results are dependent upon a user-specified buffer amount (Peters, 2012). Also, for all these examples to be safe, the algorithm needs be aware of where the safe side is in relation to the isobath.

An approach especially designed for bathymetric generalisation is in Guilbert (2015). It is designed as a multi-agent system (MAS) with a set of rules for the generalisation constraints and the system itself decides upon which operators to apply (Guilbert and Zhang, 2012). An example of constrained based generalisation (Section 2.2.2). Most operators to deform the lines are based on the (complex) snake model which can handle the safety constraint by itself (Guilbert and Saux, 2008). Examples of operators they have implemented are smoothing, displacement, deletion, enlargement and merging. By structuring the isobaths in a contour tree, in most cases the safe side of an isobath can be automatically detected. However while the method is aware of general features like pits and peaks, it may need user input to specify the safe (deeper) areas in more diffi-

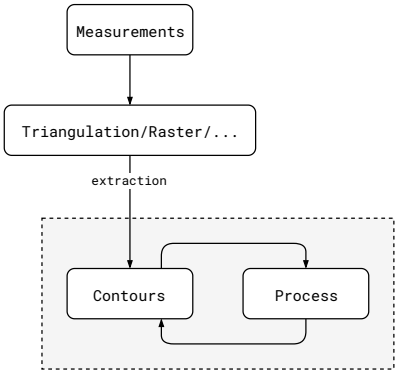
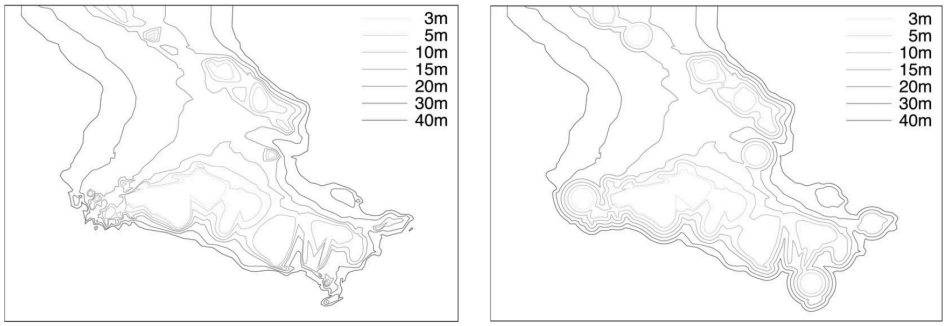


Figure 2.17: Conceptual overview of a line based generalisation approach. Measurements are being processed into some intermediate data structure from which isobaths are extracted only once. Generalisation then takes place only on these extracted lines.



(a) Before generalisation (b) After generalisation

Figure 2.18: Results of feature-driven isobath generalisation with the multi-agent system approach. *Taken from Guilbert (2015)*

cult areas. For example where an emerging plateau contains a pit. In order to implement these challenging features a nautical ontology was established and partly implemented in the MAS approach (Yan et al., 2014, 2016).

Implementation of this approach seems complex, but results are promising. Figure 2.18 shows the results taken from their research and shows how sharp corners are removed, isobaths are clearly separated, small peaks are enlarged and some parts are aggregated. It could use some improvements; especially in maintaining morphology on enlargement and aggregating peaks more naturally. Also, this approach needs already safely extracted splines from the raw data. It is in their research unclear how those are achieved.

2.5.2 Surface-based approaches

Contrary to line-based, surface-based approaches only process within the intermediate structure between raw survey data and extracted isobaths. These approaches all generalise the surface to a certain extent. Once satisfied with the results or some threshold is reached, the (generalised) isobaths are extracted only once (Figure 2.19).

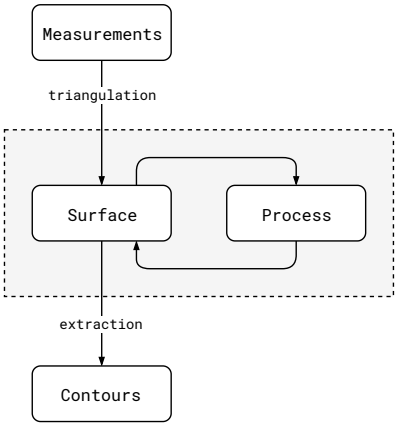


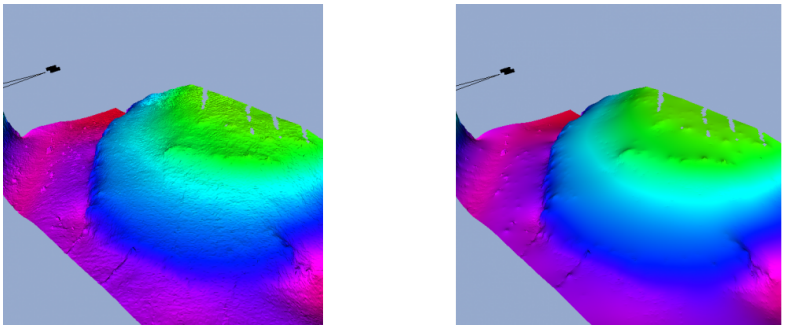
Figure 2.19: Conceptual overview of a surface based generalisation approach. In this approach generalisation is done on to the intermediate data structure and isobaths are only extracted once afterwards.

As we have seen in Section 2.3, surfaces may be represented by (variable-resolution) grids or TINs. Grids are conceptually simple and can be implemented rather easily. Especially for visualisation purposes. Also, because of their regular nature processing with them — e.g. neighbour selection — is simple, straightforward and computationally efficient. Grids are the current standard for storing and sharing bathymetric survey data in the form of bathymetric attributed grids (BAGs). However, isobaths extracted from grids may violate the safety-constraint (Peters, 2012) and there is an ongoing discussion how to establish the grid from the input vector data in order to yield a safe surface and at what resolution it should be established. These properties are all related because a grid is always some aggregated value of multiple values at an arbitrary point (the cell center), or a value is missing and interpolation is needed. Also, because some cells may represent multiple measurement points, direct relation with the survey attribution is not available anymore.

On the other hand, TINs can include *all* survey data points[10]. It results in a larger amount of data but also allows accounting for the information stored in these points directly. They automatically have a finer resolution in areas where survey density is high. TINs do not rely on user specified parameters and have data on every location within the convex hull of the data. Since survey data is directly part of a TIN it is possibly more convenient for updating (partial) regions. Because of the more versatile nature and the ability to access full survey information in such a representation, we will continue using this approach. While some commercial contouring implementations also handle surface representations, in publicly available literature only two methods are described which handle TINs with their full resolution.

Navigational surface

Smith et al. (2002) proposes the idea of establishing the so-called *navigational surface*, also known as a nautical elevation model (NEM). The concept of a navigational surface is maintaining the survey data in the form of a surface. This surface may be altered, but to ensure the safety it may only be brought upwards. For generalisation purposes



(a) Before generalisation (b) After generalisation

Figure 2.20: Example of a generalised navigational surface. *Taken from Moggert-Kageler (2018)*

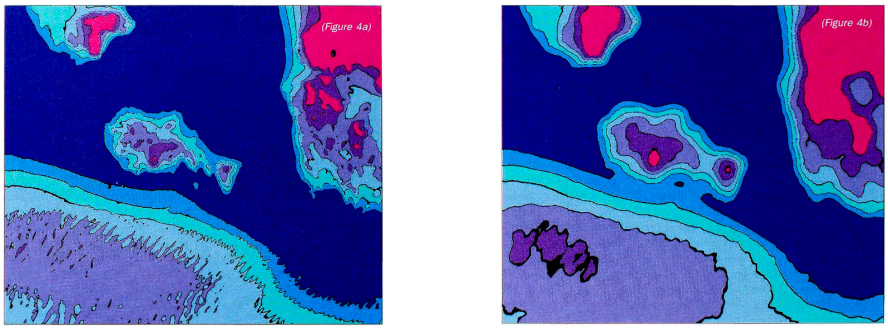
this surface can for example be smoothened and by only moving data points upwards the safety is guaranteed by definition. Once the surface satisfies some objective rules it is accepted and (safely generalised) isobaths can be extracted relatively easy. In their approach, they put emphasis on incorporating data quality within this process, but also state the advantages of handling the full resolution of the survey data. It can produce more products than only navigational isobaths and it is easy to update with new data (Smith et al., 2002). The navigational surface concept is now widely used in commercial implementations (Mellor, 2018; Moggert-Kageler, 2018; University of New Hampshire, 2019). An example of such a generalised navigational surface is given in Figure 2.20.

Smith et al. (2002) also propose a surface adjustment in the form of double buffering. It is a common GIS operation for lines but they extended the idea to 2.5D surfaces. It is more complex on a surface because of the added dimension, but since the safe side is known (up) it may also be easier to implement. There is no choice anymore of left or right, only up. There is still a specified buffer-radius needed as input, but can be dependent upon the amount of generalisation. Example of the results with this approach can be found in Figure 2.21. While the overall line is simplified towards the safe side by definition, it may still result in an artificially looking morphology and the result is not always smooth because the lines are the result of intersecting spheres. Also, on steep but perfectly smooth surfaces double buffering has no means of displacing consecutive isobaths if they are visually overlapping.

Voronoi-based approach

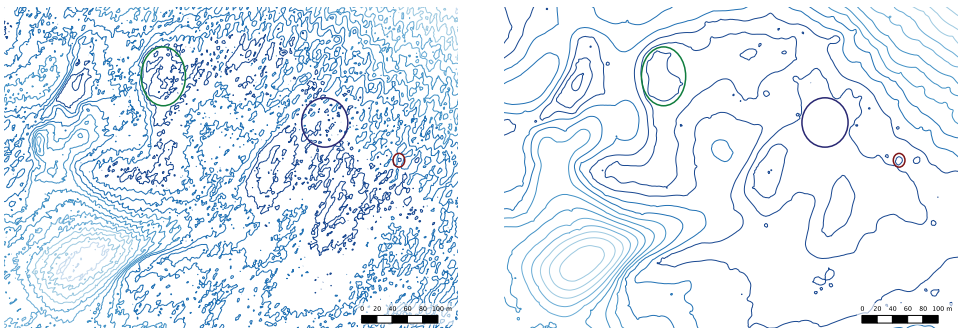
The Voronoi-surface based approach (VSBA) is another method which is based on a NEM (Peters, 2012; Peters et al., 2014). It establishes the surface as a 2.5D Delaunay triangulation — the dual of the Voronoi diagram — and extracts isobaths directly from it. Instead of lifting the entire surface, adjustments are now done by spatially interpolating the vertices in the TIN. Results with this approach can be found in Figure 2.22.

Its concept is simple: all vertices of the TIN are visited, disregarded and a new estimate for every vertex is computed using the Laplace interpolant. After such a cycle, again all vertices are visited and it is tested if the estimate is less deep than its previous value. If this is true, the vertex height is updated. This test ensures the surface is only lifted



(a) Before generalisation (b) After generalisation

Figure 2.21: Example of generalised isobaths with 3D double-buffering. *Taken from Smith et al. (2002)*



(a) Before generalisation (b) After generalisation

Figure 2.22: Example of generalised isobaths with the Voronoi-surface based approach. In this case only the smoothing operator is applied. *Taken from Peters et al. (2014)*

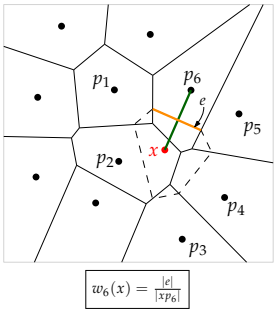


Figure 2.23: Voronoi diagram of all black vertices (solid black) and the Voronoi cell of the red vertex (dashed). Illustrated is the weight computation for point x with respect to p_6. With e the Voronoi edge and xp_6 the Delaunay edge. *Taken from Ledoux et al. (2019)*

upwards and thus guarantees safety. The Laplace interpolation relies on the Voronoi diagram and the DT. Figure 2.23 illustrates the geometry of this interpolation technique for the interpolation of vertex x. For every natural neighbour p_i of x we can compute a weight as in Equation 2.1. Now considering every vertex p_i also has a height value h_i we can take the weighted average to obtain an estimate for the height at x (Equation 2.2).

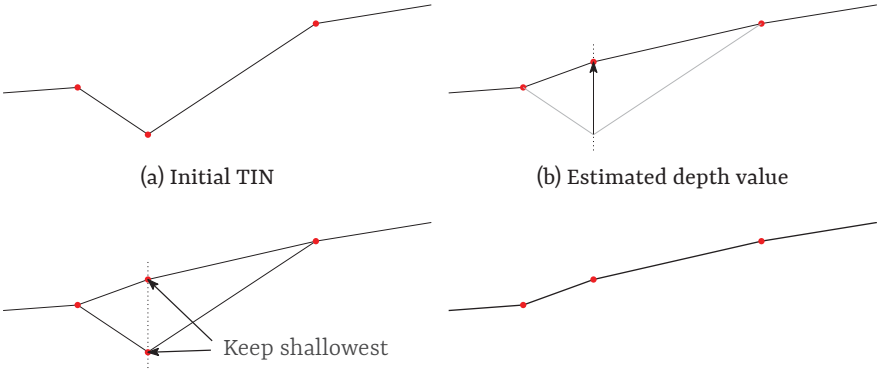
$$w_i = \frac{|e_i|}{|xp_i|} \tag{2.1}$$

$$\hat{h}_x = \frac{\sum w_i h_i}{\sum w_i} \tag{2.2}$$

Laplace interpolation is a convenient choice because it generates a smooth-looking field; the Voronoi diagram is simply the dual of the already established Delaunay triangulation; it is independent of user-specified parameters and acts local. Above all it handles anisotropic data distributions beautifully since it relies on the Voronoi edges in weight computation. Now because the interpolation would result in a smooth surface — if the data distribution is dense enough — the extracted isobaths would share the same properties due to the *implicit function theorem* (Sibson, 1997).

The interpolation process described above is defined as the *smoothing* operator (Figure 2.24). It iterates over all vertices in the surface and effectively smoothens the entire surface. While repeating this operation a certain degree of generalisation is carried out on the surface — and thus the extracted isobaths — in every iteration. It smoothens the lines, removes small pits and aggregates peaks (Figure 2.22).

However, since a TIN is a discrete representation of the field, and interpolation is only executed at the vertices of it, we can never reach an exactly smooth (C^1) field. There is a discretisation error at the edges of every face. These errors will reappear in the extracted isobaths as sharp corners and can actually never be avoided. However they may be decreased in magnitude with the *densification* operator (Figure 2.25). The densification operator is applied after a number of smoothing iterations and inserts new vertices in the surface, preferably where a high discretisation error is expected — such as very



(a) Initial TIN (b) Estimated depth value

(c) Comparison of estimate and original (d) Resulting TIN

Figure 2.24: Safe smoothing operator through estimation using the Laplace interpolant and comparison with previously known value. *Taken from Peters (2012)*

large faces or faces with high aspect ratios. The height value of an inserted vertex is again computed through the Laplace interpolant.

While the smoothing and densification operators itself are completely free of user-defined parameters, the entire process is not. Unlike the multi-agent system (MAS) approach in Guilbert (2015) the process is not yet able to decide on where to apply such operators and where not. In stead the process is iterative with a user-specified iteration count for the smoothing and densification process. In every iteration, every single vertex is smoothened and thus generalised a bit more. Some erratic features in an otherwise smooth area can thus alter the result significantly, e.g. a small erratic area in need of more generalisation also implies the generalisation of distant but already accepted areas. The method therefore typically over-generalises some areas on the surface, unnecessarily masking safe waters. An example of this problem is given in Figure 2.26. In this case, the large bay was already smoothened enough after the first iteration while the erratic peninsula on the bottom could have used some more iterations. The current approach is not able to distinguish these areas, or handle them separately.

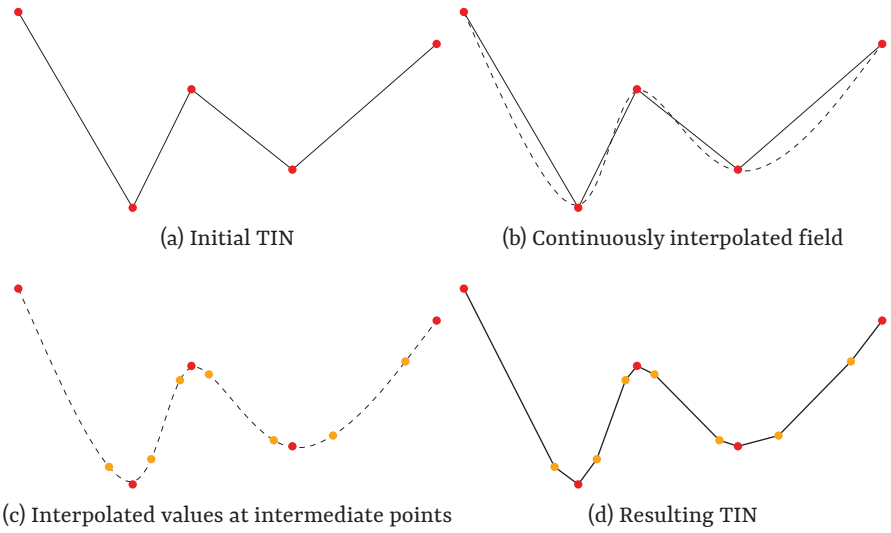


Figure 2.25: Densification operator. The operator adds new points in between known vertices and estimates their depth value through interpolation. *Taken from Peters (2012)*

Figure 2.26: Example of problems in a dataset-wide iterative approach. Light-gray line is the original, the blacker the line, the more iterations were carried out. Left shallow area, right deeper area. The large bay is already acceptably legible after the first iterations, while the peninsula on the bottom is still too erratic. Smoothing the entire dataset more — to overcome the erraticness of the peninsula — means we unnecessary apply more generalisation in the bay as well. We mask more safe waters while it was not needed.

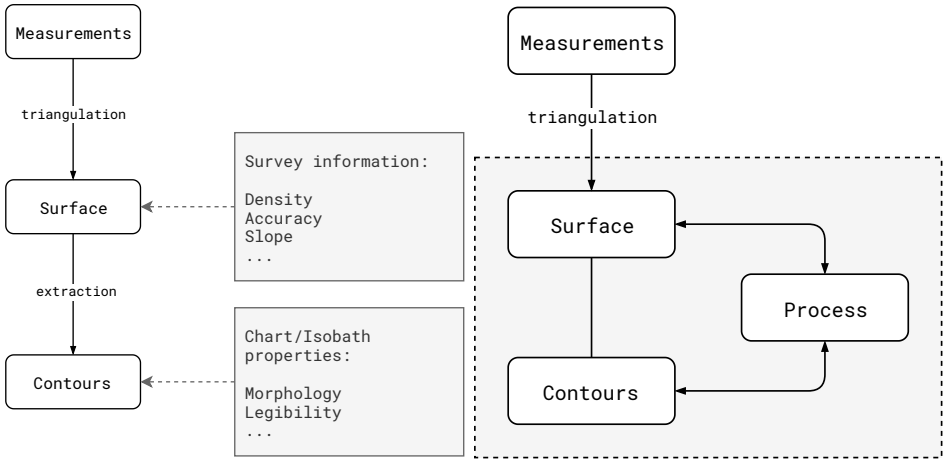
3 Methodology

This chapter proposes a novel conceptual framework to interact with survey data and generate safe navigational isobaths. First, in Section 3.1 the project objectives are summarised and some extended with the gained knowledge from the previous chapter. In Section 3.2 the weak and strong points of both generalisation approaches are discussed and we propose the use of an integrated approach. Section 3.3 is used to highlight some important definitions used throughout the rest of this thesis. In Section 3.4 we present the actual novel data structure to yield the integration of both survey data, isobaths and the relations between isobaths themselves. We elaborate on the general concept, generalisation operators to use and a basic model of evaluation based on defined legibility requirements. With Section 3.6 we show means of actually extracting depth features like isobaths, depth areas and soundings and in Section 3.7 we conclude this chapter with some remarks on the validation on those features and the process in general with respect to the four generalisation constraints.

After reading this chapter it should be clear what is lacking — or what the missed opportunities are — in currently available approaches and why we have come up with a new integrated approach. The conceptual basics of the framework are clear: what does the novel data structure look like geometry-wise, how it can be interacted with and how it can be used in a simple evaluation model.

3.1 Objectives

Already in Section 1.3 we have identified the overall objectives in an automatic generalisation process. Basically what we would like to achieve with automation is reducing human interference where possible. So basically get rid of human interaction like clicks and selections, but also user-defined parameters throughout the process. Then within the framework of generalisation we have the input data on one side and the cartographic product — isobaths — on the other. The four generalisation constraints can give us guidance on the final product, but to be able to efficiently work towards an optimal presentation, these constraints should be *interpretable* or used within the actual process. This is for example what Guilbert (2015) partially achieves with the MAS. What we also learned from that approach, is that some sort of feature detection or at least isobath relations can be very useful. Basically what this approach does — and the new method should always be able to do — is detecting where the cartographic constraints are not yet satisfied and generalise only that local region of the data. On the other side of the process, input data also has its consequences. By integrating it, we may for example take measurement accuracy or density into account. Also, since it unlikely hy-



(a) Available information (b) Integrated approach

Figure 3.1: The different types of information available in each stage of isobath extraction (a) and an integrated generalisation approach to integrate all those information (b).

drographic measurements are taken in a regular interval or at regular areas it would be nice to be able to handle (partial) updates of the underlying measurements. At the end of the pipeline we should not forget the information is finally presented in an ECDIS. While we focus on isobaths, we must not forget depth areas and soundings are also prime information. In the future even intermediate results like surfaces may be useful in IHO's S-100.

3.2 An integrated approach

In Section 2.5 we have seen line- and surface-based are basically the two approaches known in literature. Recalling Figure 2.17 and Figure 2.19 they both operate on different levels in the pipeline without a relation to any other level. Both of them can therefore only use information available in one of those levels (Figure 3.1a). In a triangulated surface *survey information* like accuracy, density and date is available. Through the extracted isobaths we may say something about the *chart properties*, directly related to the four generalisation constraints. Now because *all* of this information is important throughout the generalisation process, we should look for an integrated approach (Figure 3.1b) in which we can account for all of these properties (survey information as well as chart properties) directly. The link between cartographic product and the underlying measurements is therefore to be made.

Line- versus surface-based

We can integrate all the available information taking either the line- or surface-based approach as a starting point. We take the Voronoi-surface based approach (VSBA). Firstly because it was the original motivation for this thesis overall. But also because it has some beautiful properties and characteristics in relation to the multi-agent system (MAS) line based approach:

- The basic concept is free of user-defined parameters. The DT and VD are unique for a particular dataset and the Laplace smoothing operator does not rely on arbitrary choices.
- Isobaths can relatively easily be extracted from a 2.5D TIN and will always be topologically valid — e.g. non-intersecting.
- The concept of being *safe* is simple in a surface: only move it upwards. In line-based approaches the choice of moving left or right is still to be made.
- The contour tree in the MAS approach is able to make these choices. However with non-closed isobaths this tree is not unique and in other situations generating one needs human intervention.
- The full processing framework — from measurements to conceptual surface to isobaths — of the surface-based approach is known. For the MAS approach it is not. For example it relies on already safely extracted splines.
- In the future, the conceptual surface and other intermediate products may also be visualised through IHO's S-100[11].
- The surface-based approach has potential to also include depth areas and soundings.
- Safe polylines are extracted from the surface. These can in theory be used for further (line-based) generalisation. The reverse case is not possible.

Missing link

Although the VSBA is promising, it also has its flaws. It currently is an iterative approach with no means of automatically detecting where to generalise. As a consequence, the entire dataset is generalised and only evaluated afterwards by a human operator. The choices of where to generalise basically lead back to a link with the generalisation constraints on the cartographic level. Guilbert (2015) establish such a link between the constraints and isobaths using the contour tree and the classification of topographic features like pits and peaks. The tree-structure also enables the definition of containment relationships between isobaths. Through these relations quantified metrics such as minimum size of peaks or minimum distances between isobaths are easily established. Apart from this tree not always being unique, it can only be generated with the isobaths already extracted. In the VSBA we would like to deepen this link from constraints to isobaths to eventually the surface itself. With such a link we can establish constraints or rules based on the geometry of isobaths and the relation between those isobaths. Moreover, with such a deepened link these relationships can be used on the underlying surface directly, and thus lead specific isobaths and sets of isobaths back to individual soundings. With a deep linkage between survey data, surface, isobaths and sets of isobaths, a conceptual processing pipeline can then be as follows:

1. Establish navigational surface from measurements.
2. Establish linking data structure between surface and isobaths — and relating isobaths to each other.

3. Define quantified metrics based on generalisation constraints, either on the surface or isobaths.
4. Evaluate metrics and identify conflicts.
5. Link back these conflicts to a particular region on the surface.
6. Select generalisation operators.
7. Apply generalisation operators on this surface — locally.
8. Extract features of interest and validate them.

Step 4 – 7 may be repeating steps in a rule-based system or can be part of an optimisation process evaluating different scenarios. For the linking data structure we propose a novel structure based on the interval tree (Figure 2.13) and inter-contour graph (Figure 2.14d). It is termed the *triangle region graph* (TRG) and more information is in Section 3.4. More details on the process itself and what we can do with the link follows in Section 3.5.

3.3 Definitions

From here on we will use some specific terminology to clarify the methodology.

Isobath A 2D polyline representing equal depth in a bathymetric dataset[12].

Triangle A 2.5D face in the TIN.

Region graph The conceptual graph with contour lines and the regions between them, establishing adjacency relations. The graph consists of *nodes* and *edges* only (see Section 2.4.2). In literature also referenced to as the inter-contour graph.

Triangle region graph (TRG) The auxiliary data structure linking the TIN and its triangles to isobaths; and establishes relations between individual isobaths. Contrary to the conceptual *region graph* where nodes point to areas and edges to lines, both nodes and edges now point at least to a set of triangles (Section 3.4).

Node A node in the region graph, it represents an area between two adjacent isobaths. Possibly with (multiple) incident edges. A node may contain pointers to other information, like sets of triangles (node triangles) or attributes.

Node triangles The set of adjacent triangles belonging to the same interval and pointed to through a TRG node.

Edge An edge in the region graph. An edge is always incident to two nodes and may contain pointers to other information, like sets of triangles or attributes.

Edge triangles The set of adjacent triangles resulting from the intersection of two adjacent nodes. It is pointed to from an edge in the TRG.

Interval The vertical interval between two consecutive isobath values. For example assuming the isobath values $\{0m; 2m; 5m\}$, we have four intervals: $(-\infty, 0m]$; $(0m, 2m]$; $(2m, 5m]$ and $(5m, \infty]$.

Triangle interval The vertical range between and including minimum and maximum depth of a 2.5D triangle.

Region A set of adjacent triangles with triangle intervals all belonging to a certain interval. Regions are usually enclosed by other regions or touch the convex hull of the TIN. Note that regions may overlap each other. In a TRG nodes are always *regions*.

Smoothing operator Original process of estimating a new depth value (through Laplace interpolation), comparing it to the current value and possibly updating the depth value at a vertex.

Densificiation operator Original process of inserting new vertices in the TIN and assigning an estimate of the depth value through the Laplace interpolant.

3.4 Triangle region graph

The TRG's function is twofold: link the surface to isobaths; and relate sets of isobaths with each other. The second is what Guilbert (2015) achieved with a contour-tree. However, this contour tree is not unique and in some cases needs manual intervention. Instead, we propose usage of the inter-contour — or region — graph (Figure 2.14d). This can be generated completely automatic and is always unique for the given dataset. While containment relations are not directly available, we can still select simple peaks and pits as well have information about neighbouring isobaths. The other advantage of the region graph is we do not need to extract isobaths first. It relies on the regions between those and a basic idea of their position is already sufficient. These regions also play a part in the first function: linking surface and isobaths. As said, we can establish a region graph based on regions and their relations alone. By defining these regions as sets of faces of the TIN we integrate the triangulation into our region graph yielding the so-called TRG. Doing so, we thus have ultimately linked the isobaths to the surface. For example by selecting two neighbouring isobaths (edges in the region graph), the node connecting them represents their inter-contour region. In the TRG this node or region directly points to a set of TIN faces which is in between those isobaths. Similarly selecting an edge (an isobath) in the TRG also points to a set of triangles.

3.4.1 Geometry of the TRG

Triangle intervals

Isobath values are usually known on beforehand, for example the IHO *standard series* or a value every meter (Section 2.1.2). With this information we can establish different *intervals*, each representing a region between two isobath values. For example assuming the isobath values $\{0m; 2m; 5m\}$, we have four intervals: $(-\infty, 0m]$; $(0m, 2m]$; $(2m, 5m]$ and $(5m, \infty]$. We can now test each triangle to which interval it belongs. In Figure 3.2 we do this for two triangles. The first triangle interval — that is the minimum and maximum depth value — only intersects a single interval. It completely falls within the interval $(5m, \infty]$. The right triangle intersects two intervals: $(2m, 5m]$ and $(5m, \infty]$. Therefore we can already conclude the $5m$ isobath intersects this triangle, without actually computing it. Note that a triangle always belongs to at least one interval.

Node triangles

A region in the TRG consists of adjacent triangles all belonging to the same interval. Since a triangle may intersect multiple intervals, it may also belong to multiple regions. Figure 3.3 shows how we can identify these triangle regions using each triangle interval. Triangle regions are represented by nodes in the TRG. Each node then contains a pointer to a set of triangles: the *node triangles*. In Figure 3.3 region A points to the set $\{a, b, c, d, e, f, h\}$ and region B to the set $\{d, e, f, g, h, i, j\}$.

Edge triangles

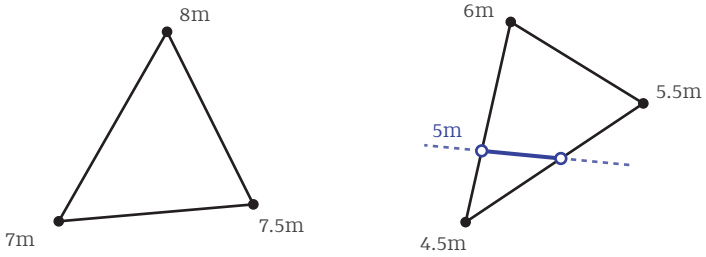
Edges in the TRG are established between intersecting regions (represented as nodes in TRG). As can be noted from Figure 3.3 region A and B intersect. From $A \cap B$ follows the intersection is the triangle set $\{d, e, f, h\}$ (Figure 3.4a). All of these triangles have in common they are part of multiple intervals. If such an intersection exists between two nodes in the graph, they are adjacent through an edge. The resulting TRG is pictured in Figure 3.4c. The edge in the graph contains a pointer to this triangle set, being termed the *edge triangles*. Again, a triangle may be part of multiple edges. Two nodes can only be adjacent if they point to regions which have consecutive intervals, i.e. a region with interval $(-\infty, 0m]$ can only be adjacent to the interval $(0m, 2m]$ and not directly to $(5m, 8m]$.

The beauty of this triangle region graph is we have now — solely based on the triangle intervals and adjacency relations in those triangles — found the approximate location of all isobaths. As can be seen from Figure 3.4b the $5m$ isobath perfectly follows the identified edge triangles. The TRG thus generates some sort of seed-set selection (Section 2.4.1). Also, with a proper definition of the intervals we can already exclude unsafe isobaths on e.g. terraces (Figure 2.11). In Figure 3.5 two different definitions of the intervals are used. In the left situation the correct and safe way, by including every triangle intersecting the isobath value in the shallow region. The intersection and thus the edge triangles will therefore move towards the deeper edge of the terrace. No edge triangles are present on the *unsafe* side anymore and thus no isobath is extracted there. That is contrary to common contouring algorithms where every equal-depth line is considered.

3.4.2 Properties of the TRG

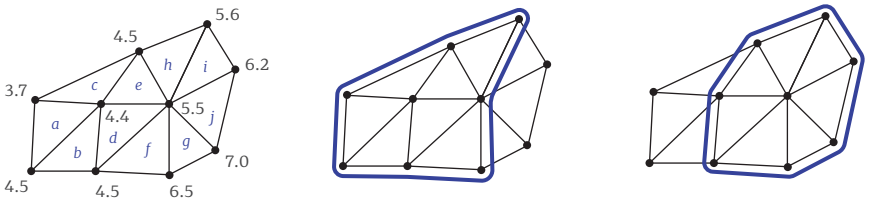
The main properties of the TRG enabling or hindering it as a viable tool for isobath generalisation are as follows:

- The most basic but also most important properties of the TRG is that we know for each isobath its relation to the set of triangles it is directly influenced by. We know for example which triangles the isobath traverses, but also know all triangles located in between a set of (adjacent) isobaths.
- Conceptually, isobaths are represented by edges in the TRG; and inter-contour areas are represented by nodes in the TRG. Both the edges and nodes in the graph are pointing to sets of triangles — which is clearly different from a traditional region graph.



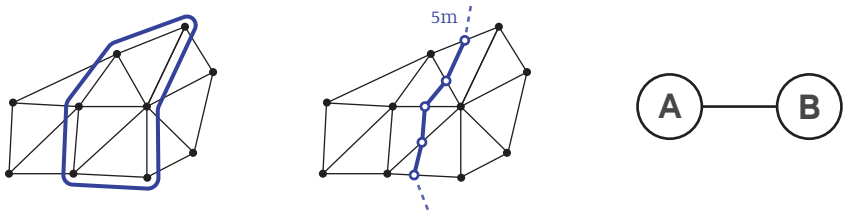
(a) One interval $(5, 8]$ (b) Two intervals $(2, 5]$ and $(5, 8]$

Figure 3.2: Two triangles with different triangle intervals. The left triangle has a minimum and maximum of $7m$ and $8m$ respectively. The right triangle $4.5m$ and $6m$. Since the right triangle intersects two intervals, an isobath intersects this triangle.



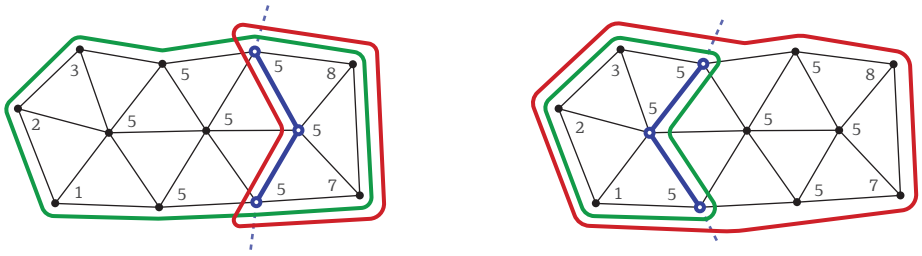
(a) Input TIN (b) Region A $(2, 5]$ (c) Region B $(5, 8]$

Figure 3.3: For each triangle we can identify to which interval they belong. If triangles belong to the same interval and are adjacent, they form a triangle region (blue). This example consists of two regions, based on the isobath values of $2, 5$ and $8m$.



(a) Intersecting regions (b) Isobath (c) Resulting TRG

Figure 3.4: The intersection between two triangle regions defines the *edge triangles* (a), the $5m$ isobath perfectly traverses these edge triangles (b) and we can visualise a conceptual region-graph (c) establishing relationships between all these triangle sets. Note that both nodes and edges in this graph have pointers to the blue outlined triangle sets.



(a) Safe isobaths with intervals as $(2,5]$, $(5,8]$

(b) Unsafe isobaths with intervals as $[2,5)$, $[5,8)$

Figure 3.5: Two definitions for the intervals on node level. The difference is in how triangles touching an isobath value are handled. In a safe way (a), these triangles are only included in the shallow node (green). In an unsafe way (b), these are included in the deeper node (red). As a consequence the edge triangles and thus the isobath itself (blue) will move to a deeper or shallower position respectively.

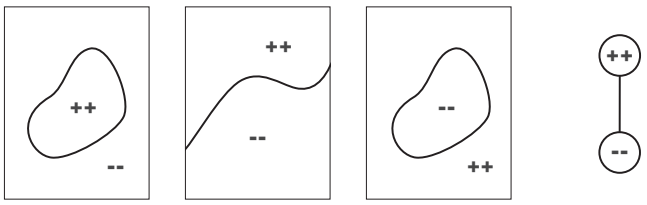


Figure 3.6: Three geometrically different datasets result in the same TRG.

- Isobaths as polylines are not directly part of the TRG. It is known which triangles they traverse — making extraction of them more efficient — but are not explicitly needed in construction of the graph. However, after extraction they can of course be pointed to from a graph edge.
- The TRG is an auxiliary data structure. It does not replace data structures for the efficient storage and retrieval of triangulations (Section 2.3). It can better be seen as an indexing structure to index triangles alone.
- The TRG is uniquely generated for a given dataset and a set of isobath values. However, with an established TRG we cannot directly go back to a unique geometrical representation of the data. For example in Figure 3.6 three different datasets generate the same TRG.
- The TRG can — contrary to the contour-tree — handle non-closed isobaths. In such case, the two incident regions to that isobath are bounded by at least the convex hull of the TIN.

3.4.3 Interacting with the TRG

The TRG is only an auxiliary data structure supporting us in the generalisation process. However with this structure no isobaths are yet extracted or features classified. This section elaborates on some of the possibilities to do so and why that is important.

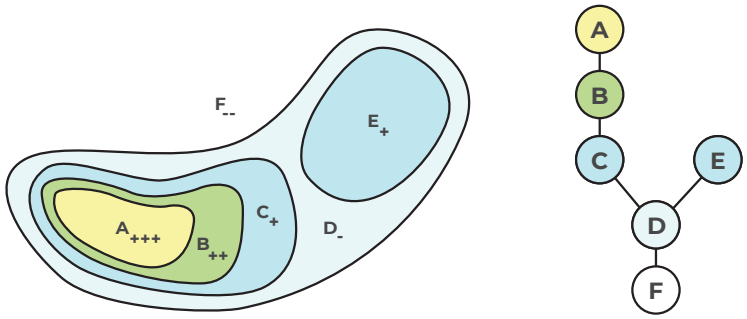


Figure 3.7: A very small example of a set of isobaths with its TRG. Note the local minima *A* and *E* and local maximum *F*. Also, *C* and *E* form a saddle through *D*.

Topographic feature classification

The region graph does not capture containment relationships directly like the contour tree. We can thus not immediately classify entire parts of the graph as e.g. a peak. Figure 3.6 captures this in a very simple manner. In all situations, a node with no shallower adjacent nodes can be classified as a local minimum. But Figure 3.6c shows this does not necessarily mean it being a peak, it can also be part of a slope. The same can be true for pits. If we would switch the shallow and deep parts on Figure 3.6a we will still have the same TRG, but now the deeper node being a pit.

The inability to fully classify features is not a big problem in relation to the generalisation constraints. With respect to the the generalisation constraints and display requirements (Section 2.1.3) we can identify three relevant feature types and their influence. 1) peaks should have a minimum size to be seen as well should contain at least one sounding symbol; 2) pits should have a minimum size to contain at least one sounding symbol and 3) narrow saddles should be aggregated. For pits and peaks, it implies that if the upper level of such feature (the local minimum) is large enough, also the consecutive parts of the feature will be large enough. Figure 3.7 illustrates this. Node *A* is a local minimum since it does not have any shallower adjacent nodes. Now if the region where *A* is pointing to is large enough to contain a sounding symbol, also *B*, *C* and *D* will be large enough. We can also find possible saddles in the TRG. It can be deduced from node *D* having two same-interval shallower nodes *C* and *E*, it may form a saddle or at least is a candidate for aggregation.

Isobath extraction

The TRG indexes only the triangles, no isobaths are yet extracted. However since the isobaths are guaranteed to follow the edge triangles (Figure 3.8) we have greatly simplified the extraction of isobaths from the entire TIN. Each edge in the TRG represents an isobath, whether closed or non-closed. To actually extract isobaths as simple features we can still use a brute-force approach or a walking algorithm. In both situations the TRG makes this process more efficient. In a brute-force approach we simply have a limited selection of triangles to be intersected by the isobath value. And in a walking-approach we can simply select one of the edge triangles as the starting point or the *seed*.

Extraction of isobaths is not needed to achieve a working TRG. Relations between edge

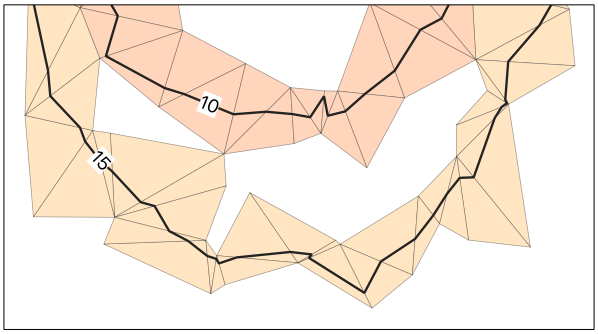


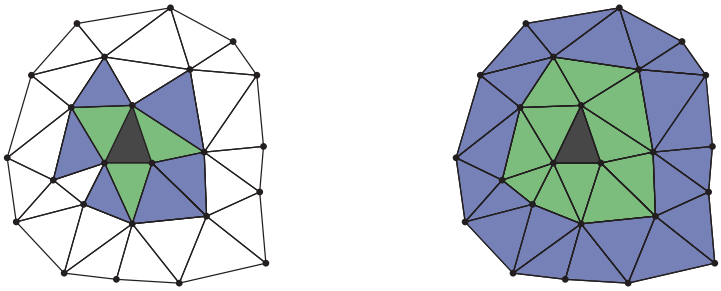
Figure 3.8: Two sets of edge triangles (saturated) with their extracted isobaths.

triangles — for example two edges incident to the same node meaning they are neighbouring — are the same for extracted isobaths as lines. However it is this phase that really links the triangulation to the isobaths, completing the deep missing link in current generalisation approaches.

Area selection

In a classical TIN some form of region expansion is already available. Starting from one identified face, we can expand this selection in two ways: by including each neighbouring triangle of the selection; or by including each incident triangle of each of the selected vertices (Figure 3.9). The second approach is more aggressive and really forms closed *rings* around the initial selection. It is also this approach we will use in the remainder of the project. Mainly because of its behaviour around isobaths. Consider Figure 3.10. A point of interest (POI) is near a vertex of a triangle. In such case, one could argue which triangles are more of interest regarding this POI. I would say it are the triangles enclosed by the green region, more than the neighbouring triangle directly opposite to this nearby vertex. Due to the vertex-adjacent triangle expansion being more aggressive, these triangles are already selected in one iteration in stead of four. And by doing so, we can actually limit the range of affected triangles because the region is not expanded towards the unnecessary other side.

With the presence of the TRG we can also select more complex regions. The two basic selections are on the node and edge level itself. Nodes and edges contain pointers to node triangles and edge triangles respectively. These queries thus result in the selection of an inter-contour area or local area of an isobath respectively. For example, if we would like to process a complete isobath, we can select an edge in the graph and extract the area around such an isobath (recall Figure 3.8). Now if we are also interested in the area on the deeper side of this isobath, we can simply use the TRG to visit the deeper incident node to this edge and append it to the selection. This concept can be applied on both edges and nodes. Considering Figure 3.7, assume we are interested in all areas deeper than node E. Then we can simply traverse the graph starting from node E, visiting only its deeper adjacent nodes (D) and continue this. Thus we find all deeper nodes D and F which both point to different triangle sets. Unioning those sets brings us to a certain polygonal area. Note that these regions always result in some sort of triangle set. These can always be expanded again using the ring expansion.



(a) Neighbouring triangles through edge-adjacency

(b) Neighbouring triangles through vertex-adjacency

Figure 3.9: Region expansion in a TIN through rings. (a) shows the original selection (black) with the first (green) and second (blue) expansion based on only the selections' edge-adjacent neighbours. (b) shows the same rings but now based on all incident triangles to every vertex of the selection.

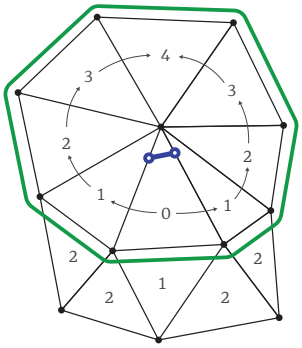


Figure 3.10: The center triangle is assumed to have a point of interest (e.g. an isobath intersection) in blue. The triangles enclosed in green are then triangles of interest, interpreted by a human. Using the edge-adjacent expansion it would take four iterations to include all of them. While with the vertex-adjacent approach these are included in one iteration.

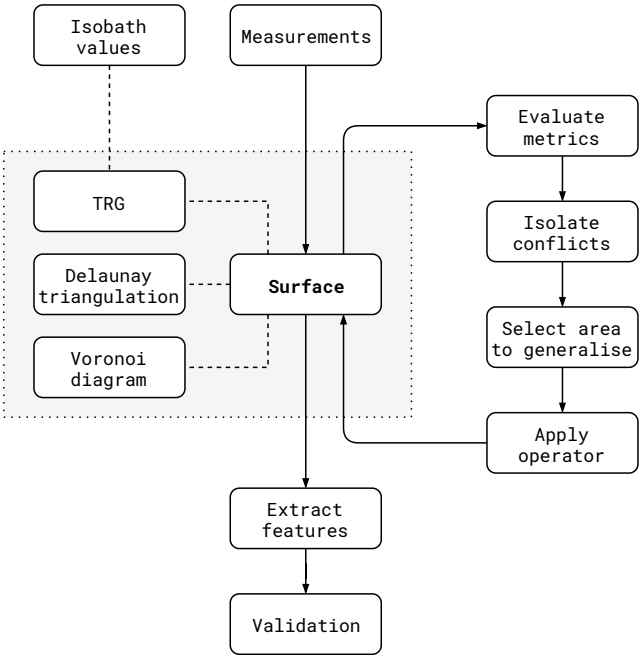


Figure 3.11: Overview of the integrated surface-based approach with a simple proposal for an evaluation and generalisation process.

3.5 Processing framework

With the triangle region graph (TRG) we have established the link between a TIN and isobaths. Still the Voronoi-surface based approach (VSBA) is unable to make choices of where and where not to generalise. This section presents the general principle of how these choices can be made in conjunction with the available operators and the TRG. It will basically complete the proposed pipeline in Section 3.2.

3.5.1 Overview

An overview of the proposed method is in Figure 3.11. At the core is the conceptual surface generated from all survey measurements. This surface comprises a Delaunay triangulation and Voronoi diagram. Relations with the isobaths are made through the TRG. Isobath values should be known on beforehand for the generation of the TRG. Now basically, isobaths and other features can be directly extracted from this (navigational) surface. We can also generalise this surface through a secondary process.

The generalisation process is in this first proposal a rule-based model. In a linear process, the surface and isobaths are evaluated through different metrics (Section 3.5.3). If conflicts are found they will be isolated, and their area possibly extended. Then on this extended area a generalisation operator is applied (Section 3.5.2). Ideally, after some of these iterations no conflicts will be found anymore. That is the moment we can extract features of interest from the surface (Section 3.6).

Modular and local

We propose a *conceptual* framework, not an exhaustive implementation. Therefore it is designed as a modular system around the conceptual surface. Some metrics, operators and extraction mechanisms are suggested, but the same framework can be used in different ways. As mentioned, we take a simple rule-based approach, but in the future the same framework may be used in a human-interaction or constrained-based (optimisation) model.

To make sure this is possible every operator and metric is designed with certain fixed input and output types, consistent with the possibilities of the conceptual surface — including TRG. For example, all generalisation operators are adapted to work on specific triangles or vertices of choice rather than on the full dataset. These triangles can be found from either a specific node or edge in the TRG, but can also be outputted by a conflicting metric. The TIN already enables the conversion between triangles and vertices, making it possible to either use vertices or entire triangles as input for generalisation operators.

3.5.2 Generalisation operators

As mentioned in Section 3.2 we will continue on the VSBA which concretely means reusing its generalisation operators (Peters et al., 2014). There are currently three implemented: smoothing, densification and aggregation. The combined use of these operators have proved itself able to generalise isobaths in a variety of situations. They for example get rid of high frequency *noise* in lines, they can smoothen lines overall, remove small indents in lines and sometimes aggregate different isobaths. Above all its implementation is *safe* by definition. Due to these properties our operators are almost identical to the ones Peters et al. (2014) proposed. The differences are highlighted below. The main difference for all operators is they now act local, on a specific set of triangles or vertices. Also, they return information on what is updated. If nothing was updated, it is also not needed to update the TRG.

Interpolation

Most operations are based on the Laplace interpolant (Section 2.5.2). This interpolation technique supplies us with a depth estimate at any — already inserted — vertex in the TIN. The algorithm is described in Algorithm 1 and from now on referenced to as `EstimateDepth()`. Note that only the neighbours are visited in this algorithm, not the vertex itself.

Smoothing operator

The smoothing operator (Algorithm 2, `SmoothVertices()`) is at the core of the VSBA. The operator computes for every vertex in a set of input vertices a new depth value through `EstimateDepth()` and is able to update this vertex in the TIN. The important part of this operator is that first, for every vertex a new value is computed without updating any of the vertices. If not done like this, a newly updated neighbouring vertex would influence the estimate of a vertex. Concretely put: then the order of smoothing within one and

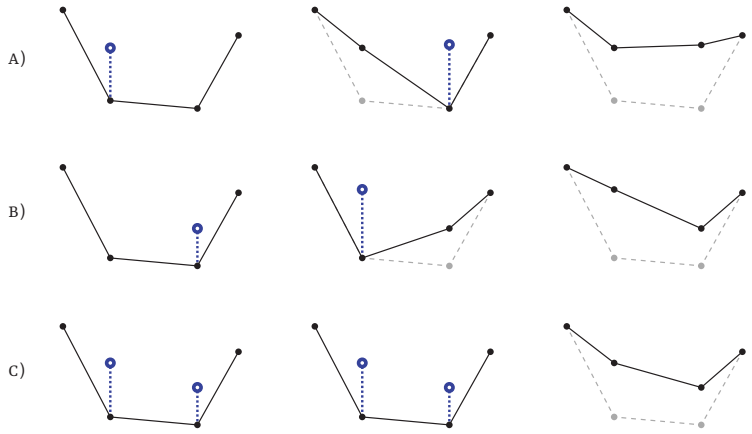


Figure 3.12: Three different orders of smoothing iterations which all yield different results.

the same iteration would matter which is not a desired result. Also, we only add the vertices to an updating queue if the estimated depth is shallower than the original. This ensures the surface is only moved upwards. The difference with the original operator is that we also check if a vertex is on or near the convex hull. We preferably do not want to interpolate here, because in some situation half of the natural neighbours can be missing. The operator returns a list of updated vertices, which is useful in tracking changes later on.

Densification operator

The purpose of the densification operator is to decrease the discretisation error on linearly interpolating the isobaths from a TIN (Peters et al. (2014), Section 2.5.2). The operation basically adds new vertices in faces where a high discretisation error is expected. The adapted operator is described in Algorithm 3 and referenced to as `DensifyFaces()`. The main difference with the original operator is we do not directly insert vertices in the TIN, but again first add it to an update queue for the same reasons as for the smoothing operator. Also we first check if a newly inserted vertex (at the face's circumcenter) is not outside of the convex hull. That would result in an invalid point since we do not have any information about the data outside of that. The operator is called with a set of already identified faces, in stead of doing the checks within. We now have simply more control over the entire process.

Displacement

Originally the VSBA comprises the *reshaping* operator. However this operator needs extensive definition of steering and secondary steering points without this being possible to easily automate. The operator is mainly used to aggregate (isobath) features but would also be capable of omission and enlargement. Almost the same results can be achieved by a simple displacement of vertices in the vertical direction. Now in stead of using the steering points' depth values and interpolate those, we can simply assign some fixed value. The simple algorithm is described in Algorithm 4 and referenced to as `DisplaceVertices()`.

Algorithm 1: EstimateDepth $(\mathcal{T}, v)$

Input: A 2.5D Delaunay triangulation $\mathcal{T}$, and a vertex v for which to estimate a depth value
Output: An estimated depth value $\hat{d}$

$\sum w * d \leftarrow 0$
$\sum w \leftarrow 0$
forall *natural neighbours* v_i *around* v **do**
    $e_{dt} \leftarrow edge(v, v_i)$ // Delaunay edge
    $e_{vd} \leftarrow dual(e_{dt})$ // Voronoi edge
    $w \leftarrow \frac{length(e_{vd})}{length(e_{dt})}$
    $\sum w * d \leftarrow w * d_i$ // with d_i the depth value of v_i
    $\sum w \leftarrow w$
$\hat{d} \leftarrow \frac{\sum w*d}{\sum w}$
return $\hat{d}$

Algorithm 2: SmoothVertices $(\mathcal{T}, \mathcal{V})$

Input: A 2.5D Delaunay triangulation $\mathcal{T}$, and a set of vertices $\mathcal{V}$ to smooth
Output: An updated $\mathcal{T}$, and the set of vertices $\mathcal{U}$ which was updated

foreach *vertex* v *in* $\mathcal{V}$ **do**
    if v *on convex hull* **then** // skip vertices on convex hull, too few information
        continue
    $\hat{d} \leftarrow$ EstimateDepth $(\mathcal{T}, v)$
    if $\hat{d} < d$ **then** // only move vertices upwards
        $\mathcal{U} \leftarrow v, \hat{d}$ // add to set to be updated
foreach *tuple* $(v, \hat{d})$ *in* $\mathcal{U}$ **do**
    update depth d of v with $\hat{d}$
return $\mathcal{U}$

Algorithm 3: DensifyFaces $(\mathcal{T}, \mathcal{F})$

Input: A 2.5D Delaunay triangulation $\mathcal{T}$, and a set of faces/triangles $\mathcal{F}$ to densify
Output: An updated $\mathcal{T}$

foreach *face* f *in* $\mathcal{F}$ **do**
    $cc \leftarrow$ CircumCenter(f) // 2D point
    if *cc outside convex hull* **then**
        continue
    Insert cc in $\mathcal{T}$
    $d \leftarrow$ EstimateDepth $(\mathcal{T}, cc)$
    Remove cc from $\mathcal{T}$
    $\mathcal{U} \leftarrow (cc, d)$ // thus adding the 2D point with a depth value
foreach (cc, d) *in* $\mathcal{U}$ **do**
    Insert (cc, d) in $\mathcal{T}$

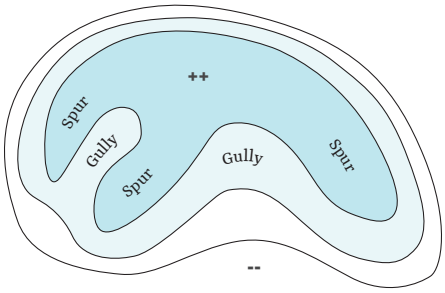


Figure 3.13: Indication of what spurs and gullies are.

As we shall see in Chapter 5 also without the (manual) selection of steering points we can achieve the same level of aggregation using the TRG. We will not omit any features (pits) from the surface on the vertex-level, but rather not extract isobaths in those cases. This makes sure we still have a complete surface from which for example soundings can easily be extracted. But by not extracting isobaths we still achieve a similar result.

Algorithm 4: DISPLACEVERTICES ($\mathcal{T}$, $\mathcal{V}$, h)

Input: A 2.5D Delaunay triangulation $\mathcal{T}$, a set of vertices $\mathcal{V}$ to displace, and a new value h

Output: An updated $\mathcal{T}$, and the set of vertices $\mathcal{U}$ which was updated

```
foreach vertex v in V do
    if h < v_d then
        depth v_d of v ← h
        U ← v
return U
```

3.5.3 Evaluation

The evaluation process (on the right side in Figure 3.11) can be designed as a rule-based, human-interaction or constrained-based model. As a proof of concept we will first define a rule-based model using quantifiable metrics. These metrics relate to the four generalisation constraints: legibility, morphology, safety and topology. Safety and topological correctness (the hard constraints) are valid by definition in the VSBA. As mentioned in Section 1.2 the main problem is in a compromise between legibility and morphology. The assumption now is, that if some minimum legibility requirements are satisfied the generalisation process can stop. By stopping this process exactly if these are minimally met, morphology will be represented as good — thus as original — as possible.

Minimum legibility requirements

From Section 2.2.1 we can deduce the basic legibility requirements for a navigational chart. Recall that the objective of generalisation is to avoid displaying too much and/or irrelevant information in conjunction with the scale and purpose of the chart. We can describe the following legibility requirements. Note that these are non-exhaustive and may be extended in the future.

Plotted lines may not overlap or touch. In conjunction with chart scale and line thickness this requirement makes it possible to distinguish different line features on any chart. Consider Figure 3.14 and the fact a plotted line is not infinitely thin like in the real-world. When line thickness $t[mm]$ and chart scale $s[-]$ is given, we can compute the minimum distance in real-world units $D[m]$ to ensure a desired on-screen spacing $d[mm]$ like in Equation 3.1.

$$D = \frac{(t+d)}{s} * \frac{1}{1000} \tag{3.1}$$

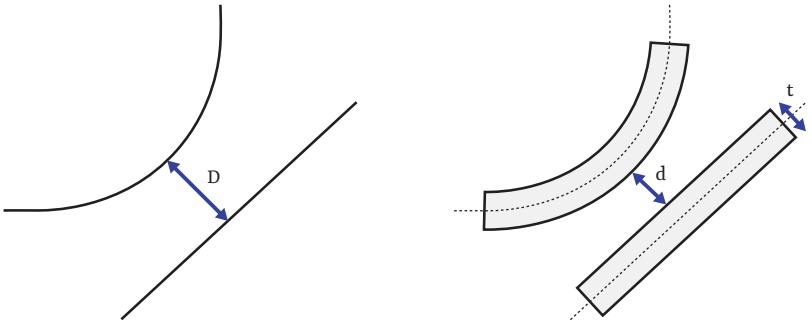
Isobaths are large enough to at least contain one sounding symbol. This requirement more or less approaches the generalisation principle from the other side. While irrelevant features should be removed, important features are emphasised. In this case through extra information in the form of a symbol. This requirements especially manifests itself on peaks, but also spurs running out in e.g. a fairway (Figure 3.13). Again, it relates to chart scale and symbol size. Any isobath should be able to enclose a circle with diameter $D[m]$ with specified line thickness $t[mm]$, chart scale $s[-]$ and symbol size $p[mm]$ (usually 4mm) like in Equation 3.2.

$$D = \frac{(t+p)}{s} * \frac{1}{1000} \tag{3.2}$$

Irrelevant pits should be removed. Here, *irrelevant* is difficult to interpret and dependent upon the situation. Pits are for example more important within an anchorage and less if they are only slightly below the local average. However, for now we assume a pit to be irrelevant if it is *small*. Again, *small* is a difficult property. Small in terms of area is different from small in circumference. It is a difficult requirement to quantify and preferably handled in a human-interaction model. However we can to an extent use the previous requirement. If a pit is too small to contain a symbol, it definitely is too small. But if it is only slightly larger, we leave the choice to the cartographer.

Channels irrelevant for navigation should be aggregated. If a channel — being a linear channel in between two peaks, a small saddle or even a dead-end gully — is too narrow to be safely navigated, it should be removed. The same holds for the space between a collection of peaks. For saddles and channels this can be done through aggregating the neighbouring isobaths. A gully can be closed by simplifying the isobath. A navigable area differs from ship to ship and time to time. However an arbitrary choice in terms of distance can be made by the chart compiler.

Isobaths should be smooth. This ensures a visually appealing presentation, but more importantly decreases the chance of misinterpretation. In mathematics a smooth function is formally defined as a continuous function (C^0) from which its first (C^1) and second derivative (C^2) can be computed anywhere in its range. However Sibson (1997) stated that for a human eye, a C^1 function is already enough to be interpreted as smooth. With isobaths as polylines[13] the problem arises they can never be a C^1 function since the formal derivative at any vertex of the line is unknown. We could approximate a smooth function by decreasing the angle of propagation between two consecutive line segments. Although not perfectly smooth, with small angles it may be perceived so. Note that the densification operator tries to achieve this by decreasing the discretisation error. Also, a formally smooth line does not always achieve the



(a) Real-world line geometry (b) Incident triangles

Figure 3.14: Geometry of line spacing (see Equation 3.1). In (a) the real-world distance (D) with infinitely small line, in (b) scaled and displayed with line thickness (t) leaving a smaller visual distance (d) on the chart.

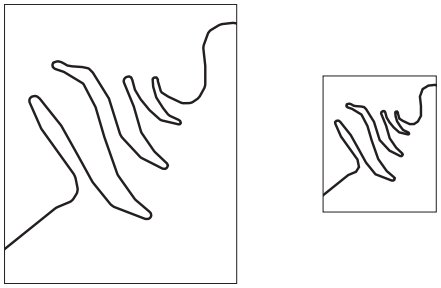


Figure 3.15: A formally smooth line at large (left) and small (right) scale. At smaller scales, the same line becomes more erratic.

assumed objectives of such line. Consider Figure 3.15 with in both bases the same formally smooth line. In a small scale chart, the same line can actually be cluttered and perceived as not smooth. The same principle holds for the representation of lines on a display. Line vertices collapsing under the resolution of one pixel are represented differently than the original definition. In such cases (erratic lines) line simplification is maybe a better option to achieve the same generalisation requirement.

Metric requirements

We use metrics (defined in Section 4.4.1) to test if the minimum legibility requirements are satisfied. By obeying some rules in the design phase of these metrics, we ensure they fit within the conceptual framework. We also pose some general remarks:

- A metric relates to one of the four generalisation constraints, chart display and/or scale.
- A metric is properly quantified so it can directly be interpreted by a computer.
- If possible, a metric is available locally, rather than on the entire dataset. This enables the generalisation operators to also act local, and only generalise in areas where needed.
- A metric may be based on the full TIN, individual faces, sets of (neighbouring)

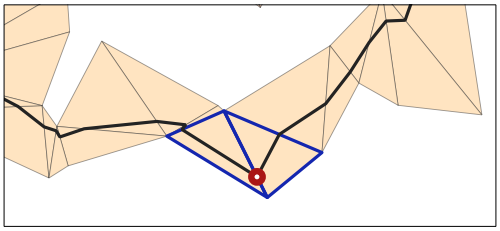


Figure 3.16: The two incident edge triangles (blue outlines) to a specific isobath vertex (red). In this case it could for example be this vertex is too sharp.

faces, (sets of) vertices and on node or edge triangles. It may also act on intermediately extracted isobaths. In the process it can also use any property of the available structures like the DT, VD or TRG (isobath relations).

- A metric is usually based on some form of threshold. Thus being able to differentiate between valid and invalid outcomes. This threshold can be computed, predefined or changed in the case of a human-interaction model.
- The output of a metric should be in the form of conflicting (sets of) faces or vertices in the TIN, (sets of) nodes or edges in the TRG, (sets of) complete isobaths or isobath vertices.

Conflict isolation and area selection

As required, conflicting metrics always result in either TIN faces or vertices; or isobath vertices. Generalisation operators either operate on vertices or faces. To convert an identified face to vertices, we simply extract all vertices of the face: a basic operation in TINs. To convert vertices to faces we take all incident faces to those vertices. Furthermore, it is possible to extend this region through either the TRG (e.g. select all triangles in the node) or ring-expansion (Figure 3.9b). Area selection through the TRG can especially be useful in generalising an entire isobath. One could simply visit the edge pointing to that isobath and extract the complete set of edge triangles.

It would become more difficult to isolate TIN faces if the result of an invalid metric is an isobath vertex. This vertex is not likely to be an existing TIN vertex, but usually is at the edges of two adjacent edge triangles. In that case it is possible to find the edge triangles incident to the isobath vertex by either a spatial query or using a smart indexing approach. As we will see in Section 4.3.2 we have implemented it with the latter. Again this would result in having faces as an isolated conflict (Figure 3.16).

3.5.4 Structure maintenance

Upon application of an operator, the conceptual surface changes and should thus be maintained to stay valid. With the given operators we have the following possibilities:

A vertex depth value changes. Now we have two possibilities: either the triangle interval in each incident triangle to this vertex changes, or does not change. Figure 3.17 illustrates this. In (a) we have the original triangulation with a 5m isobath in

blue. Now the center vertex is updated and brought upwards. And while the isobath actually changes in geometry, it still intersects the same triangles. Also, each triangle still has the same triangle intervals as before (in this case two: $(2m, 5m]$ and $(5m, 8m]$). In such cases, the DT, VD and TRG all stay the same. No nodes or edges are added or removed. Now consider the same vertex being lifted to $9.4m$ (c). This operation causes the triangle intervals being changed, with the addition of the interval $(8m, \infty]$. The $5m$ isobath is shifted even more, but more importantly we introduce a new triangle region (node), edge and thus the $8m$ isobath. In this case the DT and VD stay the same, but the TRG needs to be updated with a new node and edge. The effect is local, and only affects the triangles directly incident to the vertex. However in one generalisation iteration it is likely also neighbouring vertices are updated. In that case both operations affect each other and we cannot assume a local change due to one single update anymore.

A new vertex is inserted in the TIN In this case both the DT and TRG need to be updated (Figure 3.18). Again it only has a local effect, being slightly larger. Due to the change in one face of the DT, its neighbours may also be affected due to edge flipping. However the effect is never large and in practice we would only update a very small neighbourhood around the changed face. Obviously new faces are added to the TIN, but it is also possible new nodes and edges are introduced in the TRG.

Note that for every updated vertex an isobath can only move to an incident triangle with respect to its original (Figure 3.19). Due to this locality, at first sight it seems easy to handle and update the TRG. However since it is possible — and likely — that some of the neighbouring vertices also change, it can get complex very fast. Due to this complexities we do not update the TRG locally, but choose to rebuild an entire node and its affected neighbouring nodes. Not optimal, but safe for now (Section 6.4).

Also note that while the geometry of the VD changes upon a vertex insertion, we do not actively maintain the diagram. It is only used within `EstimateDepth()` and can be extracted — locally and computationally efficient — when needed using the natural neighbours of each vertex.

3.6 Feature extraction

Currently supported ENC depth features are soundings, isobaths and depth areas but not a triangulation yet (Section 2.1.2). These features are still to be extracted from the conceptual surface. This happens when the legibility requirements on the surface-level are met, and the generalisation process is therefore exited. While after extraction of the three types of features the data the TRG structure is not likely to change anymore, that does not exclude further generalisation of these features themselves. Also, it does not exclude further generalisation of the surface itself, e.g. for increasingly smaller scales.

Isobaths

Section 2.4.1 already posed the possibilities to extract isobaths from a TIN and in Section 3.4.3 we have shown the TRG enables a more efficient extraction of those. We make some special notes in this section.

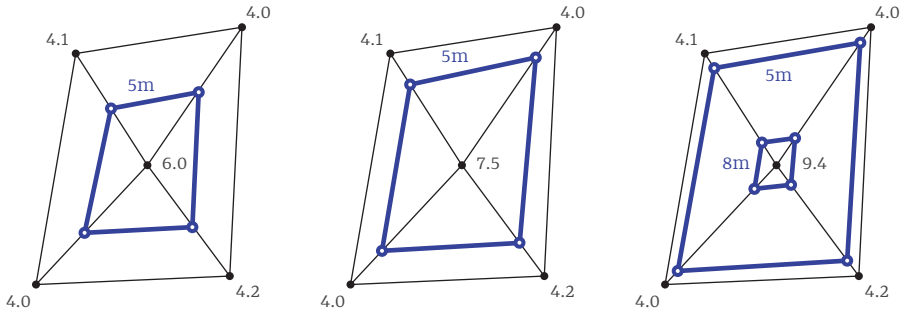


Figure 3.17: A change in depth value at one vertex only affects the incident triangles. The triangle intervals of these triangles may remain the same (middle) or change (right). In the last case, an isobath may be introduced or removed. Isobaths (blue) are shown for clarity.

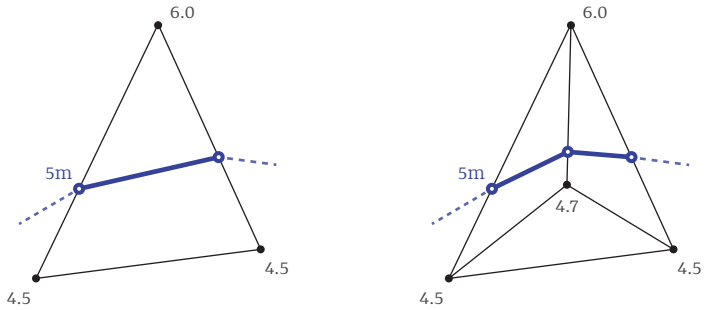


Figure 3.18: A TIN and isobath before and after the insertion of new vertex through densification.

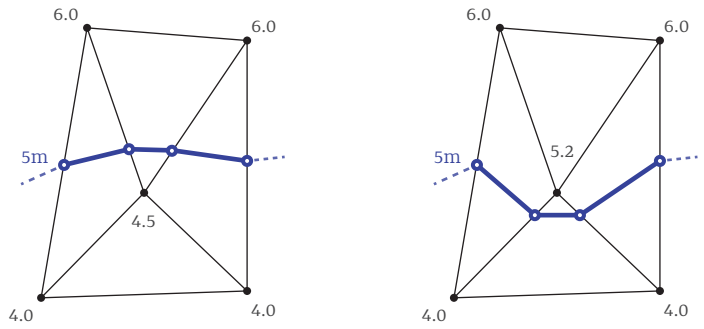


Figure 3.19: An isobath may move to another incident triangle due to an update of a vertex. However, it can only move across these incident triangles, not any further without updating of another vertex. The 5*m* isobath can simply not move beyond the 4*m* vertices without those being updated as well.

- An isobath always follows a collection of edge triangles in the TRG. And every set of edge triangles contains exactly one isobath.
- Isobaths are extracted as polylines.
- Isobaths may be closed or non-closed lines. Non-closed isobaths start and terminate at the boundary of the convex hull.
- We propose to extract isobaths with a constant direction, having deeper water on its left hand side. To do so, we only have to check isobath segment and its intersected triangle, and then continue walking from this segment only in one direction.
- Isobaths can have — next to their depth value — multiple attributes based on their relations in the TRG. We can for example add pointers to adjacent isobaths or a hint on their topographic typification.
- If an isobath is closed, we may compute its enclosed area.
- Isobaths are always topologically correct since they are extracted from a 2.5D TIN. They will never intersect each other or themselves.

Depth areas

Although not part of the original scope of this project, we propose an option to extract these features as well. Depth areas are already conceptually available in our structure: they are represented by nodes in the TRG. However these nodes translate to a collection of triangles, not a polygon. One possibility is to dissolve these triangles together and clip off the overshoot using their incident isobaths. The other option is using all incident edges to a node. By computing the area of each incident isobath to a node, we can identify the outer-boundary of the depth area — having the largest area. The remainder of the incident isobaths can then be assumed to be holes. In the second approach we directly relate the depth areas to the isobaths and topological errors (slivers and overlaps) due to accuracy are decreased. Also note that again we may attach attributes to these features based on their relations in the TRG. We can for example attach pointers to all incident isobaths of the depth area or identify if it is a local minimum or maximum.

Soundings

Again not part of the original scope, but we see potential to integrate this in the future. Since all original survey data is still present in the conceptual surface, also soundings may be generalised and extracted in an integrated approach. This is however a topic on its own since the same problems hold as in isobath generalisation (Kastrisios et al., 2019). Visualising *all* soundings in a chart would be confusing. Extending the proposed methodology to also automatically deal with soundings and isobaths integrated is like the holy grail in navigational charts. However the framework supplies already some relations which may be of help in such process. For example we know which vertices are between two isobaths or within a depth area. An established DT of soundings is already available and the smoothing operator may yield some information on *golden soundings*[14]. For example, vertices never being smoothened are usually the shallowest in a local region, making it a potential dangerous part of the surface.

3.7 Validation

If the designed process is implemented correctly, the defined metrics should all lie within the given thresholds by definition. Otherwise the process would not have been exited as it was supposed to be. However in some cases we may have to preliminary exit the process. Also in a (future) optimisation approach it can happen some metrics are not perfectly adhering the requirements. In all cases we need to be able to say something — either qualitative or quantitative — about the resulting navigational chart and its individual features. While validation is preferably resulting in objective metrics, a cartographic product is not likely to let it capture in some measurement. In below validation principles we propose quantified metrics to support a final human judgement on the end-product.

Legibility A legible map really is *in the eye of the beholder*. Of course we can test if every contour can contain at least one sounding symbol or is not overlapping with another line. But especially judgement on the smoothness or erraticness and the overall amount of detail is hard to quantify. It is mainly this constraint where a cartographers eye is not to be underestimated. Therefore we have to judge the overall legibility by ourselves. However we will carry out a simple smoothness test based on the average angularity distribution in the isobaths, because we assume large angularities are never good.

Morphology The goal still is to achieve a legible chart with as less changes in morphology as possible. With every update of a vertex, morphology is thus changed and should ideally be limited. However, most of the vertices will never be displayed to the end-user. What communicates the morphology to them are the extracted isobaths, depth areas and some selected soundings. In general we can quantify the amount of changes by the distribution of errors in soundings, e.g. a root mean squared error (RMSE). But to actually test what is the result for the chart we will compute for each closed depth area its share in the total dataset area. Due to the safety constraint, deeper depth areas will slowly become shallower areas and by this ratio we can test the result on the chart. Also see Section 4.6.

Safety If implemented correctly, safety will be guaranteed since vertices are only moved upwards. This can also be easily checked afterwards. However we only have ground-truth at exact vertices, not in the middle of a TIN face. The interpolated value at these locations can actually move down due to geometry of the TIN. However since we did not have data on beforehand, we can also not say it was not safe. A final check on safety can be to check each depth area only contains soundings below its safe threshold. Thus, the depth area $(2m, 5m]$ can only contain soundings deeper than 2m.

Topology Simple topological violations like (self-) intersecting lines would not occur by definition in the VSBA. In some situations isobaths can be touching (e.g. Figure 2.12) but with properly defined metrics these situations should have been properly generalised before extraction. More complex topology — like relative distances between peaks and the location of a fairway with respect to these peaks: left right or right through the middle of those; or the distance of an obstacle from the side of a fairway and thus a recommended passageway— is for now out of scope and should be validated by the human operator.

4 Implementation

In this chapter we build upon the previous one by proposing a first implementation of the conceptual methodology. Section 4.1 clarifies some of the tools and specific mechanisms used in our prototype. Section 4.2 gives a very brief overview of the data we have tested and because of which properties we have selected those. In Section 4.3 we list basic properties of the TRG and how it is specifically structured in the prototype. We also show how to generate it from a TIN alone and how it can be maintained upon changes in the geometry. Section 4.4 proposes metrics to quantify the set legibility requirements, as well as means of isolating conflicts resulting from their evaluation. The actual evaluation phase of those conflicts are described in Section 4.5. In this section we describe the need for parameters, their values and which generalisation operator should be applied to overcome the isolated conflicts. We conclude this chapter in Section 4.6 with a view on how we have assessed the compromise between legibility and morphology.

After gone through this chapter it should be known in what system the prototype is implemented and why we have tested specific datasets. We have shown in-depth algorithms and considerations on the generation and maintenance of the TRG. It should be clear how our basic rule-based evaluation model works and what the role is of the TRG, but also what are specific aspects of the overall generalisation process which hinder or enable a successful generalisation.

4.1 Prototype

A software prototype has been built to test the methodology. Its source code is available under an open-source license through *GitHub*[15]. The prototype is implemented in *Python*. While the original VSBA is implemented in *C++*[16], it would personally take me too much time to handle this new language to justify its use. A *C++* implementation may have eventually been *faster*, but I believe *Python* is better human-readable and thus perfectly fitting a conceptual implementation where it is expected the software will change heavily in the future.

The prototype does not rely on large dependencies rather than some basic input/output libraries. At the core is a triangulation and since we mainly operate on this structure it should be able to do at least the following operations: generate a DT; input new vertices; reference to vertex attributes and preferably have some means of efficiently interact with the data like finding neighbours and updating of vertex depths. The *Python* library *StarTIN*[17] is capable of these operations. Also, because the TIN is stored as a star-based structure we have direct access to (natural) neighbouring vertices (Ledoux and Meijers, 2013), Voronoi cells per vertex and it even supports a Laplace interpolation at any given

point within the convex hull. For a constrained Delaunay triangulation (CDT) it is harder to find a suitable *Python* library. The *C++* software *Triangle* offers lots of control over the input and output of such a triangulation and is used for this purpose (Shewchuk, 2005). It is not ideal, however available *Python* packages were not available with the amount of control we need. Finally, the graph structure is implemented without the help of any packages. Again allowing much more control this was done using *Python* dictionaries. More information on that is in Section 4.3. More information on the actual implementation — including its paramaters and methods — is on *GitHub*.

4.2 Data

We have assumed that all data can be transformed into perfectly projected points in a cartesian coordinate system. It is out of scope for this project to handle these transformations separately. Only (x, y, z) point tuples are accepted. With z being the minimum depth value at that location. Note that depth is defined as the minimum expected watercolumn. Thus a point with a larger height has smaller depth, and at locations where is water the depth is a positive number. For now, measurement uncertainty is not accounted for in the implementation nor the methodology. More information on the used datasets is in Appendix A, but we give a small introduction below:

Simulations For overall testing and development we have created different simulated datasets. An example of such dataset is in Figure A.1. They are simulated with testing in mind. Therefore they contain lots of different features like pits, peaks, slopes, saddles etcetera. They are also composed of (almost) evenly distributed points mimicking MBES data as well as linear distributions to mimic SBES data. However there is no connection with the real-world. They are placed on a grid of 500x500 units and are arbitrarily drawn.

Admirality Data The United Kingdom Hydrographic Office (UKHO) supplies its survey data as open data on their Admirality Data Portal[18]. We have selected both a high density MBES dataset: Dover Strait and an older low density SBES dataset: Margate Head to Margate Road (Figure A.5 and A.6). Both datasets are in `csv` format in the `WGS84` coordinate reference system (CRS). Note that while isobaths are also displayed, this does not necessarily mean these are generated using solely these specific datasets. The UKHO data is mainly used used for comparison between two different distributions (MBES and SBES) of measurements at different scales.

NOAA Bathymetry NOAA supplies its bathymetric surveys as open data through their data portals[19]. The survey data is similar to that of the Admirality, but NOAA also supplies the generalised isobaths for different scales. An example was already in Figure 2.6. This additional resource makes it possible to compare our results to the official chart. We have selected dataset New York because it is the most recent data for the area, has full seabed coverage and has some interesting morphology. The selected soundings are in Figure A.9 and its corresponding ENC isobaths in Figure 5.14.

Isobath values

Beside the survey data we also need to input the isobath values on beforehand. For general purposes we either use the IHO standard series (Figure 2.2) or isobaths every meter. NOAA currently uses different values recalculated from feet. In these cases we will also extract those values, just for a convenient comparison. Isobath values are a simple list of values in meters.

4.3 Triangle region graph

Preliminaries

The triangle region graph (TRG) is auxiliary to the DT. In this implementation we build it upon this existing triangulation. The triangulation can be seen as a collection of triangles. Triangles have pointers to their three vertices. The vertices are maintained in a secondary structure to enable convenient updating of their depths.

- Triangles are CCW ordered lists of vertices. E.g. `[34,879,56]`. They do not have a unique identifier (ID) other than this (unique) list of vertices.
- Triangles can be queried for their neighbouring triangles which results in a list of triangles, e.g. `adjacent_triangles([34,879,56]) = [[879,251,56],[56,18,34],[34,82,879]]`
- If a triangle only has two adjacent triangles, it is touching the convex hull.
- Vertices do have unique IDs and contain more information. All vertices are stored in a dictionary. Once inserted in the TIN a vertex ID will never change. `vertices[vertex_id] = {x,y,depth_current,depth_original,depth_queue}`
- For each vertex we can query the *star* of incident neighbouring vertices, resulting in a CCW ordered list of vertex IDs, e.g. `incident_vertices(34) = [56,18,5,82,879]`. From those we can also calculate incident triangles and the Voronoi cell of that vertex.

Graph structure

The TRG structure is a simple *Python* dictionary with at its roots again two dictionaries of nodes and edges. Each node and edge is referenced through a unique identifier and points to the properties described in the code block below.

The current conceptual implementation does contain some duplicate pointers. For example, it is theoretically not needed to point to the edges a node belongs to from within the node itself. However these pointers make a conceptual implementation far more easier to interact with using less complex queries. The basic structure points from nodes or edges to triangles. Again for convenience we maintain a duplicate dictionary of triangles with to which nodes and edges they belong. Both these types of duplicate pointers increase computational efficiency and simplicity on simple queries, at the price of storage and complexity upon updating. Somewhere in between there should be an optimum, but investigation of that was outside of the scope of this thesis. Some things to keep in mind:

- Both nodes and edges have unique identifier.
- Both nodes and edges point to a list of triangles.
- Nodes have an interval identifier (which shows to which vertical interval it belongs) and edges have a single isobath value.
- Edges point to two node IDs: first the shallow node, second the deeper node.

```
tr_graph = {nodes, edges}
nodes[node_id] = {
    'interval': int(),
    'triangles': set(),
    'edges': [edge_id, ... ] }
edges[edge_id] = {
    'edge': [node_id, node_id],
    'value': float(),
    'closed': bool(),
    'triangles': set(),
    'isobath': geometry }
```

Triangle interval

Since isobath values are already known at the initialisation of the process, we can split these up in the various intervals. For convenience these intervals are then mapped onto integer labels. 0 being the shallowest interval, increasing up to the deepest. With this we can for example move to one shallower interval by simply subtracting 1.

Triangle intervals are computed per triangle. Algorithm 5 illustrates this process. In the implemenentation we actually use a simpler `bisect` operation. Recalling Figure 3.2 triangles may be part of multiple intervals. These intervals will always be sequential. It is not possible for a triangle to be part of e.g. interval 1, 2 and 4. It always must be 1, 2, 3 and 4.

Algorithm 5: TriangleIntervals $(t, \mathcal{I})$

Input: A triangle t, and a list of intervals $\mathcal{I}$
Output: A list of intervals Λ the triangle is intersecting

$min_d \leftarrow t$.MinimumDepth()
$max_d \leftarrow t$.MaximumDepth()
foreach *interval i, index(i)* **do**
    if $min_d \leq i_{max}$ **and** $max_d \geq i_{min}$ **then**
        $\Lambda \leftarrow$ index(i)
    if $max_d \leq i_{min}$ **then** // Intervals are ordered from shallow to deep
        break
return Λ

4.3.1 Generation and maintenance

Generation

Two approaches for the generation of the TRG have been implemented. Firstly born out of practical issues, but both also have different characteristics.

The first and simpler approach is illustrated in Algorithm 6. It basically visits every triangle, computes its triangle interval, puts triangles in separate containers per interval, and then forms adjacent regions within those interval containers. This last step generates the actual nodes. Afterwards it iteratively finds edges between those nodes. It is an approach which can be simple to implement and especially useful if not much nodes are present in the dataset. The computational complexity of the first steps in this algorithm scales linearly with the amount of faces. However it gets costly very fast in the last part where it for every node, all of its potential neighbouring nodes — those nodes only one interval deeper or shallower — are tested for intersections with each other. The amount of these combinations grow with both the number of isobath values as well as the vertical variation in the data. However a highly variable dataset does not necessarily lead to this by definition, e.g. if the variations do not intersect the isobath values back and forth and thus do not introduce more and more nodes, nothing happens with complexity because no edges are introduced. It will be a complex relationship between these three properties: number of faces, isobath values and vertical variation. However as a guideline we may say the complexity increases linearly with the number of faces (n), and then slightly less than quadratic with the number of expected isobaths (k). In a worst case it thus scales with $O(n + k^2)$, with the note k is thus already an estimate based on the variation and size of the data as well as the number of selected isobath values. A highly variable dataset and a high number of input depth values, but none of these values is intersecting the dataset still scales linearly.

The second TRG generation algorithm bypasses the last computational expensive step by including the establishment of edges directly in the region growing algorithm. It is described in Algorithm 7[20]. A node is grown from a single triangle and all neighbouring triangles in the same interval are added to a queue within that node. This process is continued until no same-interval adjacent triangles can be found anymore. If one of the neighbouring triangles encounters a triangle with an interval-index one less or higher (shallower or deeper) it is added to either a shallower or deeper queue, again within the node itself. If the current region growing queue is empty, either a deeper or shallower triangle queue is selected. A random face from this queue is selected and added to a new node. An edge is directly established between this new and the previous node. Now, the process of region growing this new node repeats as above. However if it now encounters a triangle in the interval equal to the *previous* node it was just formed from, it is also checked if this triangle is present in the queue for this *previous* node. If true, it will be deleted from both queues. An edge was already established from another intersected triangle in both nodes, and this newly found triangle can thus be disregarded from the edge for now. This effectively empties all queues iteratively. Now if all queues are empty the process is finished and the TRG generated completely. This process is more efficient since we do not need to visit all possible combinations of edges afterwards. It has the complexity of the region growing algorithm itself, probably linearly with the amount of faces (n) to visit. However triangles may be part of multiple intervals and thus nodes. If

a triangle belongs to multiple nodes, it is also visited multiple times in the region growing. Again the amount of complexity this property adds is dependent upon the vertical variation and number of isobath values. Again leading them back to the number of expected edges — an edge triangle belongs to two nodes — and specifically the expected percentage of *edge triangles* of total triangles (r) we may expect the overall complexity approximately in the form of $O((1+r)n)$, almost linearly. Thus note it does not completely depends upon number of isobaths, but also density of the survey data. With denser data, ratio of edge triangles will drop.

The general region growing algorithm to establish the sets of adjacent triangles is fairly simple and only based on the already computed triangle intervals. However, as in Figure 4.1 it can happen two adjacent triangles belong to an identical interval but cross an edge. The algorithm therefore also checks if the common edge in two adjacent triangles is also completely in the same interval. This is *not* the case in Figure 4.1 if the algorithm is initialised on the interval up to 5m. The common edge is deeper than this. Still, the triangles *are* adjacent when growing from the interval deeper than 5m.

Algorithm 6: GenerateGraph $(\mathcal{T}, \mathcal{I})$

Input: A triangulation $\mathcal{T}$, and a list of intervals $\mathcal{I}$
Output: A triangle region graph $\mathcal{G}$

```
   /* Compute triangle interval for each triangle                    */
   /* and inventorize them in a dictionary                           */
1  IntervalTriangleDict L
2  foreach triangle t in T do
3      Λ_t ← TriangleIntervals(t, I)
4      foreach λ in Λ_t do
5          L[λ] ← t

   /* Establish adjacent regions within each interval                */
6  Node collection G[nodes]
7  foreach interval λ in L do
8      AdjacentTriangles ← RegionGrowing(L[λ])      // based on adjacency alone
9      foreach set of triangles s in AdjacentTriangles do
10         G[nodes] ← s

   /* Establish edges between nodes                                  */
11 Edge collection G[edges]
12 foreach node N, node interval λ in G[nodes] do
13     shallowerNodes N_s ← G[nodes] with λ = (λ − 1)
14     deeperNodes N_d ← G[nodes] with λ = (λ + 1)
15     foreach n_s in N_s do
16         if N ∩ N_s then
17             G[edges] ← [N_s, N]
18     foreach n_d in N_d do
19         if N ∩ N_d then
20             G[edges] ← [N, N_d]
```

Algorithm 7: GenerateGraph $(\mathcal{T}, \mathcal{I})$

Input: A triangulation $\mathcal{T}$, and a list of intervals $\mathcal{I}$
Output: A triangle region graph $\mathcal{G}$

```
1  select triangle t from T                    // preferably a triangle with only one interval
2  λ ← TriangleIIntervals(t, I)
3  current node n_c ← CreateNode(t, λ)
4  G[nodes] ← n_c
5  current queue Q_c(n_c) ← t
6  deeper queue Q_d(n_c)
7  shallower queue Q_s(n_c)
8  indexed triangles J
9  while J < T do
10     while Q_c(n_c) do
11         foreach t in Q_c(n_c) do
12             t_n ← AdjacentTrianglesChecked(t)
13             foreach t_n do
14                 if λ_t in TriangleIIntervals(t_n, I) then
15                     n_c ← t_n
16                     J ← t_n
17                     Q_c(n_c) ← t_n
18                 if λ_t + 1 in TriangleIIntervals(t_n, I) then
19                     if t_n in Q_s(n_deeper) then
20                         add edge between nodes
21                         remove t from both queues
22                     else
23                         Q_d(n_c) ← t_n
24                 if λ_t − 1 in TriangleIIntervals(t_n, I) then
25                     if t_n in Q_d(n_shallower) then
26                         add edge between nodes
27                         remove t from both queues
28                     else
29                         Q_s(n_c) ← t_n
30             remove t from Q_c(n_c)
31     select new triangle t from either Q_d(n_c) or Q_s(n_c)
32     current node n_c ← CreateNode(t, λ)
33     G[nodes] ← n_c
34     current queue Q_c ← t
35     establish edge between n_c and n_previous
```

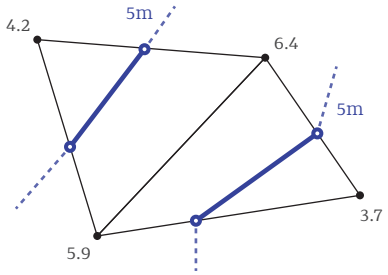


Figure 4.1: Two adjacent triangles with identical intervals. However they do not belong to the same node with interval up to $5m$ since their common edge (5.9, 6.4) is outside this interval.

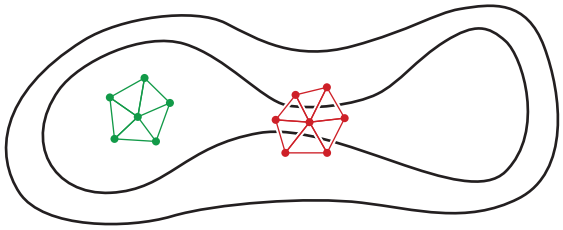


Figure 4.2: Green updated triangles will result in a local (node-level) change of the TRG. Red updated triangles will split an entire node into two separate nodes, it is not a local operation anymore but spreads across multiple nodes.

Maintenance

In Section 3.5.4 we have already identified the two possible changes in a TIN due to updates. To check whether a triangle interval changes due to an operation, we also maintain a separate structure with triangle intervals and pointers for each triangle to which node it belonged: `triangle: {'intervals': list(integers), 'nodes': list(node_ids)}`. If a triangle interval changes due to an operation these triangles are dropped from the TRG and again inserted through Algorithm 7. In most situations an updated triangle (its interval is changed) results in a new (shallower) node or the removal of a small deep node. However as can be seen in Figure 4.2 if the changed triangle will bridge a narrow passage, a node may be split up in two different nodes. It is therefore not a local operation anymore. To be consistent, for every changed triangle we drop the entire node and all its adjacent nodes from the TRG and rebuilt all of those. This is more costly especially when nodes are large, but always works. It is of course also possible a vertex value is updated, but its incident triangles all stay within the same triangle intervals as before. In those cases no new nodes or edges are introduced, neither are they removed. In those cases we simply update the depth values only. Still, if those triangles are part of edge triangles, isobath geometry may have actually changed.

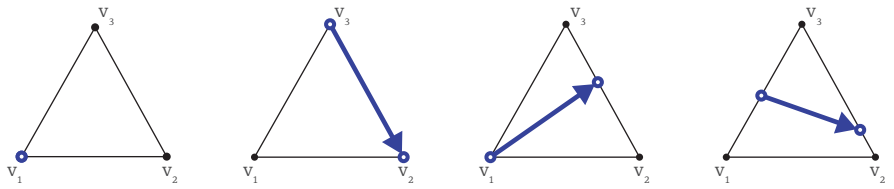


Figure 4.3: Different variations of isobath intersection with a triangle and their resulting `intersections` object. This array can be stored per triangle.

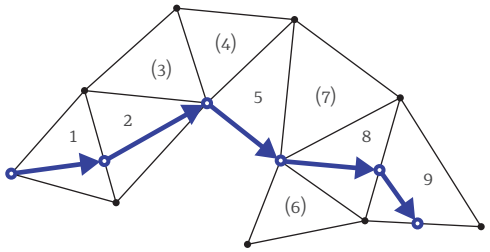


Figure 4.4: Ordered edge triangles for one specific edge. If the transition is from one triangle to one another triangle they receive consecutive identifiers. If transition is through a single vertex and other triangles are incident, these are assigned randomly in-between identifiers.

4.3.2 Feature extraction

Isobaths

Isobaths can be extracted in all stages of the process so they can be analysed by the metrics. This is done with a walking algorithm starting from a randomly selected triangle from the edge triangles. At the first triangle we directly make sure the line is properly directed, with the deeper side on its left. We keep track of the triangle order we intersect with an auxiliary dictionary in each edge: `edges[edge_id][ordered_triangles] = {1: {triangle, intersections, iso_geom}}`. `Intersections` is a list of the abstract intersections with the triangle. Recalling Figure 2.10 these can be various combinations. In case the isobath is touching only one point it is in the form of `[vID]`. In case it is on the edge of a triangle `[vID, vID]`, in the case of one edge and one vertex: `[[vID, vID], vID]` and in case of a full intersection on two edges of the triangle `[[vID,vID],[vID,vID]]` (Figure 4.3). From these vertices we can eventually deduce the actual geometry of the line segments quite easily. The ordering of the triangles seems at first hand a bit too much. However with this ordering we can retrieve for example all edge triangles between two isobath vertices for further analysis. But more importantly we can query back from a given isobath vertex (ID) to a specific triangle (Section 4.4.2). The situations in which an isobath segment terminates or follows exactly at vertices is also handled by first assigning the last ordered triangle, and then randomly assigning IDs in between for the rest of the triangles (Figure 4.4). After extraction of an isobath the full geometry of the line is added to the edge object as well as an attribute indicating it is closed or not.

Depth areas

Consider Figure 3.7 and the wish to extract the depth area of node D. We can then visit each edge incident to D: [E,D], [C, D] and [D, F]. Each edge is a closed isobath from which we can calculate its area. The isobath with the largest area is assumed to be the outer boundary of the depth area, with the rest of the isobaths all being holes. Note that currently no non-closed isobaths are supported. That needs some more complex interaction with the convex hull.

Grids

Since the BAG is the current standard for the exchange of bathymetric data and navigational surfaces, we can also generate a raster of the data. For each gridcell we then compute a value at the center of the cell using Laplace interpolation. This does not always result in a *safe* surface, however it demonstrates the idea. This is anyways not in the further scope of the project.

4.4 Evaluation

Based on Section 3.5.3 we propose a set of quantified metrics for the prototype. We also propose methods to isolate and extend conflict regions on which the generalisation operators can be applied.

4.4.1 Metrics

Triangle area

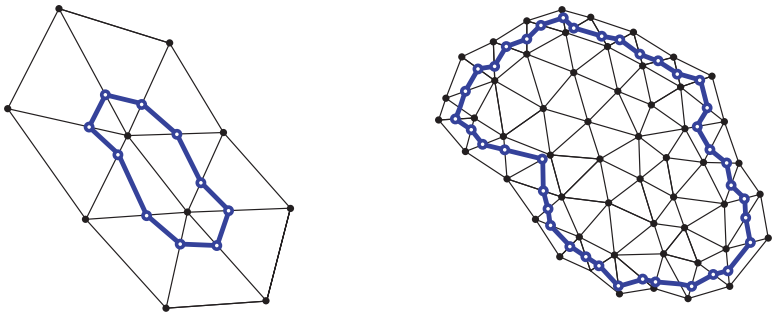
Given a triangle $v_1v_2v_3$ its area is defined as $A = 1/2 * |\vec{v_1v_2} \times \vec{v_1v_3}|$. A conflict results in a triangle if it is lower than a defined minimum area.

Node area

We can compute the area of a node in two ways. Either sum the areas of all node triangles, or actually compute the area of the outer boundary. The latter is more difficult to compute, especially is the region touches the convex hull. In some cases the triangle area can be a good approximation, but one should be aware of the differences (Figure 4.5). Triangle area will always be larger, and the difference becomes more apparent if the total area is smaller, or individual triangle area decreases (denser data). As a metric, node area is used in both peaks and pits. Both should have a minimum area to be valid, and if not they result in a complete node indicated as a conflict.

Triangle aspect ratio

The aspect ratio of a triangle is not formally defined in literature as far as I could find. However, the property of a Delaunay triangulation trying to maximise the minimum angle in each triangle is related to the triangle's aspect ratio. One usually defines aspect ratio of a triangle as the ratio between the circumscribed and inscribed circle as $AR = (e_1 * e_2 * e_3)/(8(s - e_1)(s - e_2)(s - e_3))$ with $s = 1/2 * (e_1 + e_2 + e_3)$ and e_i being edges of the triangle. For an equilateral triangle — which the Delaunay triangulation aims for



$A_\Delta \gg A_{iso}$ $A_\Delta \approx A_{iso}$

Figure 4.5: Ratio between summed area of all node triangles and the actual isobath area may differ per situation. With large triangles and a small isobath area (left) differences are high. With smaller triangles and larger isobath areas differences decrease and may become insignificant. Also the actual position of the isobath on each triangle plays a role.

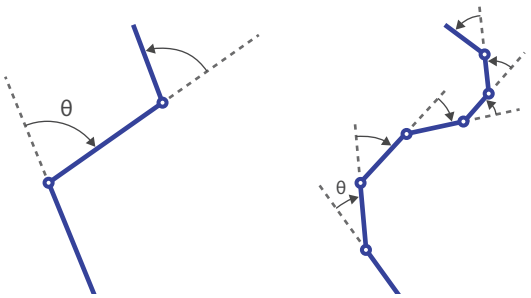
— this evaluates to 1. A conflict results in a triangle if the aspect ratio is larger than a defined value.

Triangle slope

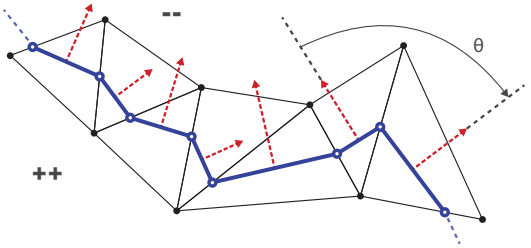
The orientation of a triangle $v_1v_2v_3$ can be fully captured by its normal vector $N = v_1\vec{v}_2 \times v_1\vec{v}_3$. The gradient of the triangle is then the angle of the normal vector with the horizontal plane. The aspect or direction of the triangle can be obtained by dropping the z-direction from the normal vector. The normal vector of a triangle is directed. Problems may arise when a triangle is badly formed (ordered) from its edges. However a NEM is always a 2.5D surface and normal vectors can never point downwards. If the computed normal is thus directed downwards, one can invert it to obtain the correct one. Slopes cannot be computed directly on edges and vertices. In practice the average is then taken from all incident triangles.

Smoothness

Section 3.5.3 already posed the difficulties in quantitively assessing the perception of smoothness. However we have also seen that if the angles between two consecutive isobath segments are decreasing, it approaches a C^1 function more and more. Therefore we use a very basic smoothness metric of the angularity between each segment: the deviation angle from collinear propagation (Figure 4.6). Considering the two consecutive line segments p_1p_2 and p_2p_3 the angularity θ at p_2 is defined as $\arccos\left(\vec{A}\cdot\vec{B}/|\vec{A}||\vec{B}|\right)$. With $A = (x_2{-}x_1 \; y_2{-}y_1)$ and $B = (x_3{-}x_2 \; y_3{-}y_2)$. Since we can order the edge triangles properly we can actually also use the aspect of each of those triangles and use the angularities between those instead. An isobath intersection with a triangle is always orthogonal to its aspect. This saves us from extracting the actual geometries each time. A conflict results in an isobath vertex — note that this is not directly linked to the TIN — if the angularity is exceeding a defined threshold. The result of this metric is illustrated in Figure 4.7.



(a) Angularities in a polyline. Not *smooth* on the left, *smoother* on the right.



(b) Normal vectors (red) of ordered edge triangles directly relate to the angularities of the isobath. Although with a difference of 90°.

Figure 4.6: Angularity in isobath as polylines (a) and directly derived from the (ordered) edge triangles they intersect (b).

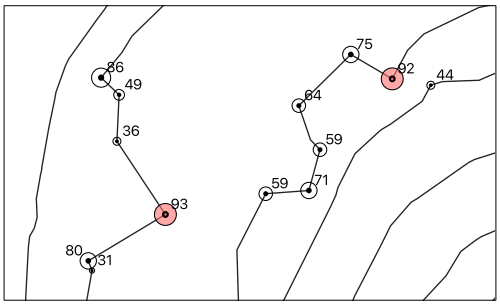


Figure 4.7: Example of angularities in degrees at isobath vertices. Only shown for angles larger than 30°.

Line spacing

If two isobaths are intersecting the same triangle, line spacing can easily be computed using the slope of that triangle. The vertical distance of the isobaths is known and implies we can compute the horizontal real-world distance (Equation 3.1) using the relation $dy/dx = slope_{triangle}$, of which the slope can be computed using the normal vector. A steeper seabed (triangle) implies isobath line spacing decreases. Simply using this relation only is possible in an ideal situation, but usually isobaths are not intersecting the same triangle but are separated over multiple. This hinders a direct relation between isobath separation and the triangulation directly. However in the prototype we do not directly account for this kind of conflicts. From a personal meeting with the NLHO followed this is not a frequent situation nor a big problem. In some cases the conflicting isobath is simply left out from the chart, depending on the purpose of the chart. In other cases the cartographer makes a substantiated decision based on its surroundings and significance for navigation to decide how to deform the isobath. An example of such case compromising legibility is for example pointed at with the arrows in Figure 5.16b.

Spurs, gullies and narrows

While we do not account for line spacing between two different isobaths, we do test if spurs and gullies are not too narrow (Figure 3.13). Still both of these features should be wide enough to be able to distinguish the individual line segments. Spurs should also be able to contain at least a sounding symbol (4mm circumference). If that is not possible, it should be widened. Gullies should on the other side be wide enough to be navigated. If that is not the case the gully may be removed or simplified. In both situations the input threshold can be separately defined. Also, the concept of minimal width of a spur can also be used in a narrow part of a ridge, or applied to a peak-feature. With this, we can directly test if a peak is large enough to contain a sounding symbol.

Spurs and gullies can always be detected on a single isobath. We do not need relations with other features. To detect these features, initially the link was made with rolling-coin simplification algorithms. Ai et al. (2016) proposes a method to detect both inward and outward *bends* using a constrained Delaunay triangulation (CDT). However the method needs to have selected an initial triangle and is directed. The principle of using the CDT is borrowed to test distances between isobath segments:

1. Isolate an edge from the TRG.
2. Generate the full CDT of this isobath (Figure 4.8a).
3. A CDT element can either be inside or outside the isobath. Since the isobath is ordered with its deep side on the left, we detect inside and outside elements by comparing the order of one constrained edge to the actual isobath. Since all elements are CCW oriented, they are one the deep side (gullies) of the isobath if the constrained edge follows the order of the isobath vertices.
4. For every element in the CDT we can distinguish three situations (Figure 4.8b):
    - With two constrained edges, the length of the third open edge directly dictates the threshold value.

- With one constrained edge, we project the third opposite point onto this constrained edge. The length of this projection dictates the threshold value. If the projected point is not within the limits of the constrained edge, the length of the shortest open edge dictates the threshold value.
- With no constrained edges, the element is skipped. The same distances are handled by neighbouring elements.

5. Examples of invalid spur and gully elements are in Figure 4.9.
6. If an element is set as invalid, all the (isobath) vertices of that element are set to invalid.

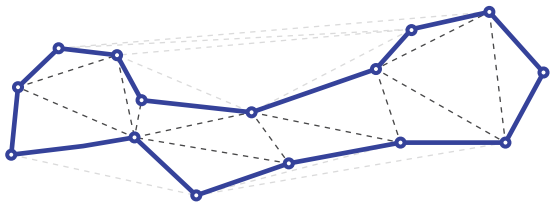
Aggregation

Where the spur and gully detection operates on a single isobath, we can also use the same methodology for detecting gaps between multiple sets of isobaths. We do this for detecting narrow saddles and narrow spaces in between peaks. Considering Figure 3.7 we can detect all possible saddle regions by visiting each node and check whether they have at least two shallower nodes. In this case it is only the node combination [C,D,E]. Both edges between the 'root'–node D and shallower nodes C and E are then triangulated as in spur/gully detection. Now, while checking each element of this triangulation, we only compute distances between two different isobaths and not the elements or edges connecting vertices from one and the same isobath. An example of the resulting conflicting constrained triangles is in Figure 4.10. There is no need to isolate the isobath vertices first, we are in this case only interested in the actual TIN vertices contained in these conflicting triangles. We select those with a spatial query based on only the vertices in root-node D. This metric thus directly results in a set of vertices of the original DT.

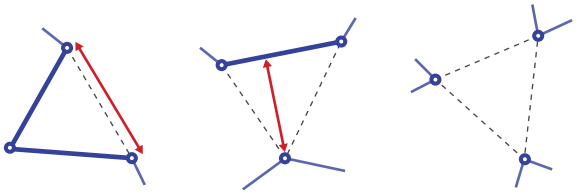
4.4.2 Area selection

Both the angularity and spur/gully detection metrics will result in conflicting isobath vertices as an outcome. The aggregation or saddle-test results in TIN vertices. And the check on node area results in entire nodes and thus their node triangles. The generalisation operator needs isolated triangles or vertices as input. In Section 3.5.3 we showed the transformation between triangles and vertices is quite easy and in Figure 3.9b we showed a method of expanding selected triangles and vertices. However it takes more effort to transform an isobath vertex in either a triangle or vertex. To do this efficient we use the ordered triangles present in each edge. The first vertex of the isobath is at the start of the first ordered edge triangle. With this property we can directly go from the i^{th} isobath vertex, to edge triangle i and $(i-1)$. From those incident triangles we can further expand the region. Aggregation tests result in triangles not directly linked to the navigational surface. Therefore we make use of a spatial query to retrieve all points in the 'root'-node, contained by those triangles.

While the Laplace operator has multiple advantages, for example it being local, this can also be counterproductive. The smoothing operation (Algorithm 2) only smoothens its direct natural neighbours. However the critical points in a gully or spur may cross mul-



(a) An isobath (blue) with its CDT (dashed).



(b) Three variations of CDT faces. Constrained edges (blue) with one, two and three inserted edges (dashed). The threshold distances are shown in red.

Figure 4.8: Spur and gully detection using a CDT.

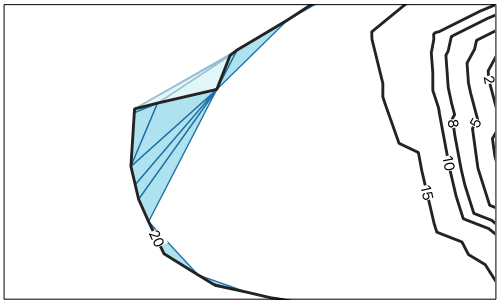


Figure 4.9: Example of invalid spur (dark blue) and gully (light blue) triangles. Only shown for the 20*m* isobath and the shallow region is on the right.

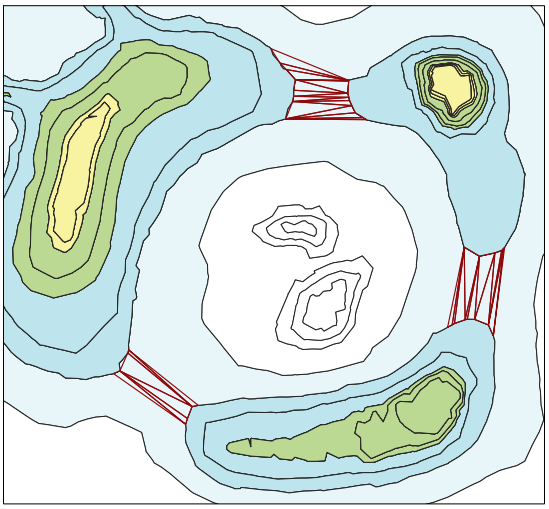


Figure 4.10: Example outcome of the aggregation metric. The red triangles are invalid elements detected from the CDT. They always *bridge* depth areas with equal depth.

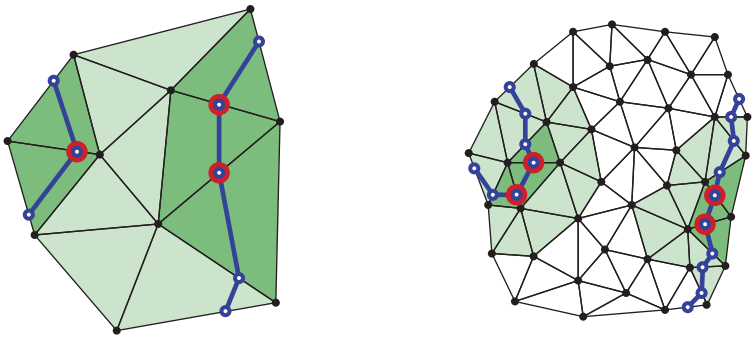


Figure 4.11: Denser survey data implies also more vertices/triangles are needed to cover the same distance. The direct incident triangles (dark green) to conflicting isobath vertices (red) do not necessarily span the entire space in between. Expanding those with only one ring (light green) may be enough in less dense data (left) but may not span the entire critical region in denser data (right).

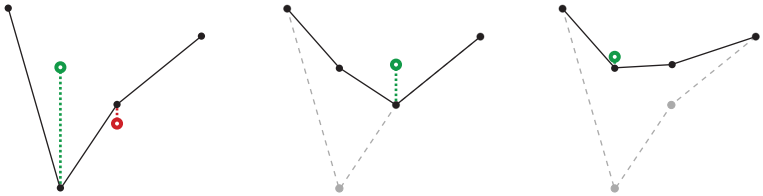


Figure 4.12: Current vertices (black) with their safely estimated depths (green). In iteration one, only the left vertex is safely updated; the right vertex is not updated due to safety reasons. However, the change of the left vertex allows the right vertex to be safely updated in a following iteration. It is even possible the left vertex can be safely updated again after that iteration.

tiple triangles. Only smoothing the direct incident triangles to the conflicting isobath vertices may thus not cover this entire critical region and is dependent upon survey density (Figure 4.11). We may want to expand the isolated triangle regions appropriate for the local survey density. Expanding to more triangles if survey density is high.

4.5 Processing

A single smoothing operation on the isolated conflict areas does not necessarily neutralise each conflict directly. The geometry of the triangulation may change due to such an operation, effectively moving the isobath to deeper waters (Figure 3.19). Also in some cases the estimated depth at a vertex cannot be moved safely upwards in one iteration, while it may happen an iteration later after smoothing the region around the vertex (Figure 4.12). Effectively this means it is possible we introduce new conflicts in each iteration.

Due to the fact all conflicts will not be solved in one go, we take an iterative rule-based approach trying to solve most of the conflicts. We first define some parameters. In Section 4.5.2 we describe the basics of this iterative approach itself.

4.5.1 Parameters

1. **Chart scale** In relation to the compilation scale and purpose of the chart. NOAA uses the scales listed in Table 4.1. These differ from the standard IHO scales, but are a convenient choice for comparison of results.
2. **Angularity threshold** Threshold dictating the maximum allowed angle [°] in between consecutive isobath segments. It is for now an arbitrary choice based on experiments.
3. **Spur threshold** The minimum real-world distance $[m]$ spurs should allow. It should have a minimum value computed through Equation 3.1 with a minimum on-screen spacing d to allow for a clear representation, and in some cases should allow the display of a sounding symbol with radius $d = p$, usually 4mm (Equation 3.2).
4. **Gully threshold** Same as for spur threshold. However a cartographer here can make a better educated guess based on the navigability of such a gully. Thus, in stead of only basing this on line spacing and symbol size, it may decide on an arbitrary value being *safe for navigation*, e.g. being at least 100 or 200m wide.
5. **Aggregation threshold** The same principle holds as for gullies. It should have a minimal width $[m]$ for the line spacing, but is again a cartographers choice to decide on *safely navigable water*. This can also change per chart scale or purpose.
6. **Peak and pit area** The minimum area $[m^2]$ of both peaks and pits. While these are not automatically generalised in our process, they are highlighted afterwards. This gives the cartographer the possibility to quickly select problematic features.
7. **Minimum and maximum ring expansion** As illustrated in Figure 4.11 it is sometimes needed to smoothen more triangles around a conflicting area. This parameter may account for the survey density or scale. It is proposed this relates to both survey density as the smallest threshold chosen. For example if survey spacing is typically 2m and the smallest threshold (e.g. spur threshold) is 50m, the amount of triangles covering the critical region is approximately 25. Expanding conflicts from both sides we at least need approximately 12 rings.
8. **Iteration limit** Ideally, the process would converge towards an actual quantified acceptable solution. However it is likely to be possible this would not happen within reasonable time. For those cases we also define an iteration limit on which the process terminates although not having satisfied its generalisation requirements.

4.5.2 Generalisation process

We have access to three generalisation operators: `SmoothVertices()`, `DisplaceVertices()` and `DensifyFaces()`. The densification operator is always applied last since it does not effectively change the conceptual surface but also limits the effectiveness of the smoothing operation due to an increased density (Peters (2012) and Figure 4.11).

Table 4.1: NOAA standard ENC chart scales per purpose of the chart.

Chart purpose	NOAA scale
Overview	1:1 500 000
General	1:1 500 000
Coastal	1:700 000
Approach	1:260 000
	1:120 000
Harbor	1:45 000
	1:22 000

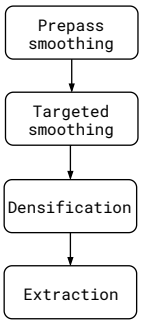


Figure 4.13: Generalisation phases.

Due to the effects of previous iterations of smoothing and smoothing in the neighbourhood (Figure 4.12) the overall process is again set up iteratively. The generalisation phases are displayed in Figure 4.13. We can first apply smoothing to all vertices if we want to get rid of the initial noise in the so-called *prepass* phase. This smoothens all vertices in the dataset and can best be compared to the original VSBA.

In the phase of *targeted smoothing* we use the established metrics to target the smoothing using the process in Figure 4.14. In this process smoothing is done until no conflicts are found anymore. Conflicting vertices are isolated and only those are smoothened. If none of the vertices is safely updated anymore, but conflicts are still present, we temporarily add one ring to the expansion. We may repeat this until at least one vertex is updated again. Then in the next iteration, the rings are again reset to the initial value. But since the neighbourhood is smoothened, we may expect different depth estimates due to that and smoothing may continue (Figure 4.12). The process stops if an iteration limit is reached, maximum ring expansion is reached or no conflicts are found anymore.

After all smoothing phases we apply densification. In this phase evaluation takes place similar to that of the the targeted smoothing. However now we will only use the angularity and aspect ratio metrics to find the conflicting edge triangles and densify those to decrease discretisation errors on extraction of isobaths.

At the end of the entire process we also calculate the area of pits and peaks. They are also exported as attributes to the isobaths so a cartographer can manually decide what to do with it.

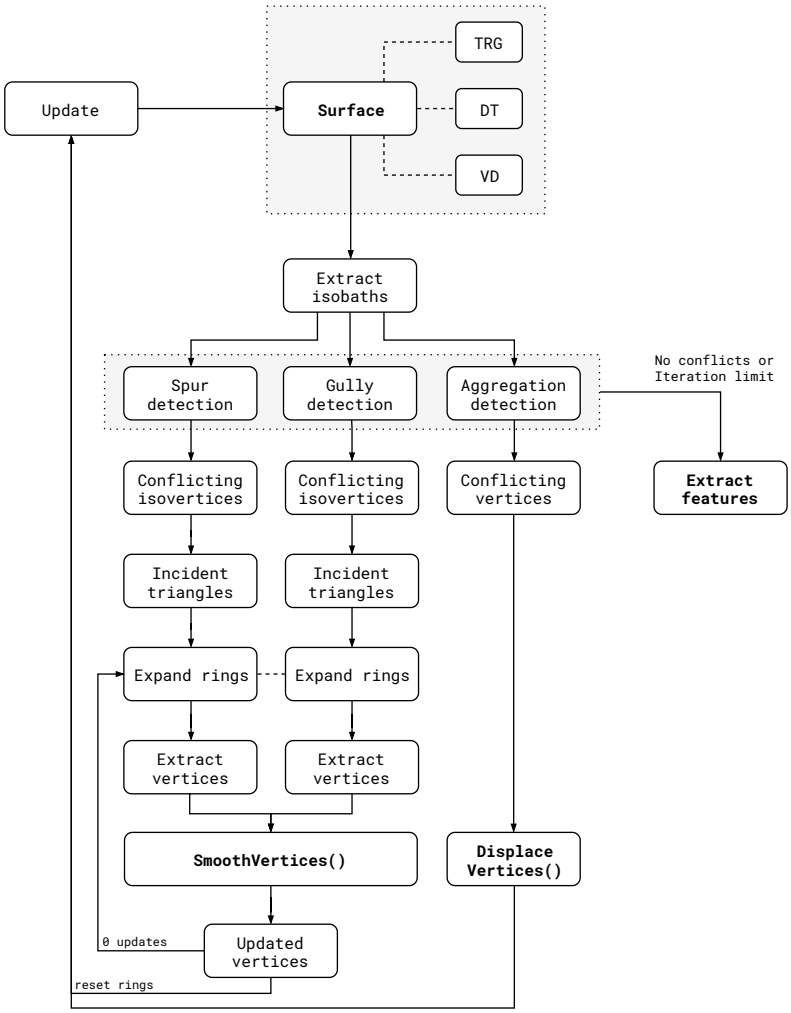


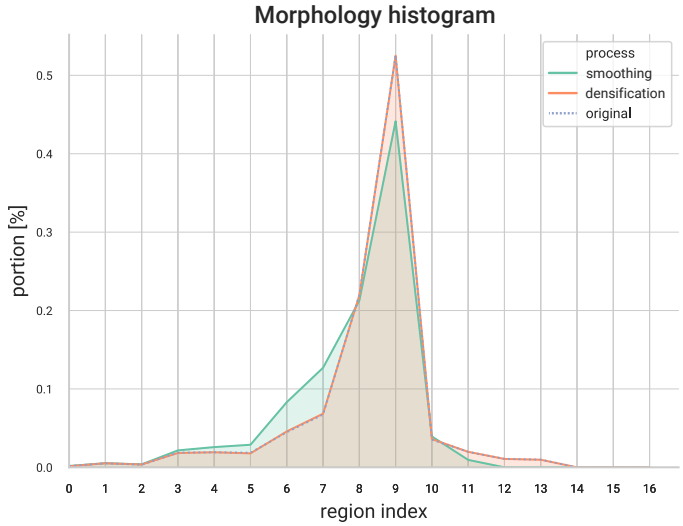
Figure 4.14: Overview of the targeted smoothing process.

4.6 Validation statistics

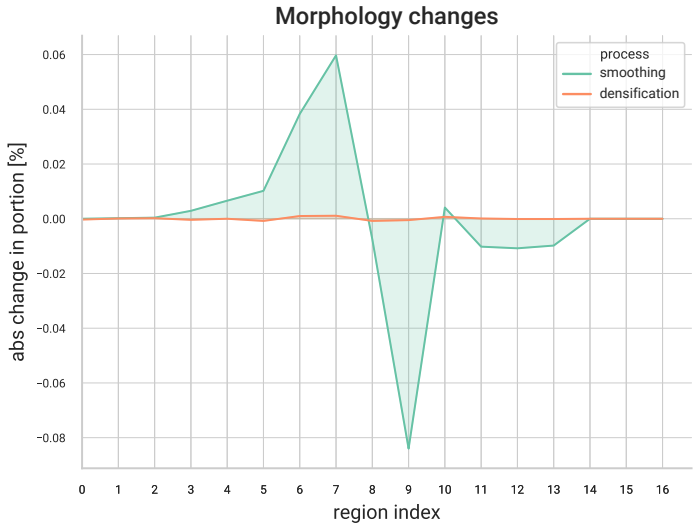
In the VSBA the topological and safety requirements are met by definition. We are looking for a compromise between legibility and morphology. Legibility is very hard to quantify. However for morphology — or better the change in morphology — that can be simple through for example an RMSE. With an RMSE the overall change of depth at each vertex is perfectly quantified. However it does not immediately tell the most important information on changes actually visualised to a user. A user only sees the isobaths and some selected soundings. RMSE is therefore not the best statistic. For example, a very deep pit with some measurements changing hundreds of meters does affect the RMSE significantly, but are not seen at all in the chart. Simply because the changes between two isobaths are not seen, only their effects on the changing geometry of an isobath.

We can better capture the changing morphology through a histogram of depth regions (Figure 4.15a). In such histogram for every interval (represented in this case by their indexes) the portion of the total depth area is visualised. In the example we see most depth regions are around indexes 8 and 9: almost 70 percent of the chart is covered with depth areas of those intervals. For the green line in the graph (after smoothing), the total depth area stayed the same, but moved somewhat shallower (because of the safety constraint). It gained some area on the 6 and 7 regions and took that away from region 9. These regions thus directly link to the cartographic end product. Also, if the critical depth of the chart (e.g. 20m isobath for large vessels) in this case is between region 7 and 8, any shifts of regions below that value does not directly influence the navigability of the area: it was not of interest at all because it already was too shallow. However if the morphology changes significantly crossing this threshold, it *does* matter and decreases the amount of navigable waters. It it thus unwanted this graph shifts it balancing point across the critical depth. These changes are also separately visualised in Figure 4.15b. It shows per region the percentual change in increase or decrease. It does not show however from which region to which regions it changed. Again it becomes critical if the visual *balance* is around the critical depth. Figure 4.15b also shows how densification does not affect morphology significantly in comparison with smoothing.

Measuring legibility is difficult and requirements — or even better: preferences — differ from user to user. We can however quantify some of the line properties. In Figure 5.11a and 5.11b we capture for example the overall line segments length and the overall angularities in each isobath respectively. Next to those two metrics we can also isolate closed isobath areas smaller than some threshold. The idea behind that is that if we have very small peaks or pits they cannot be readable.



(a) Distribution of depth areas in the chart. In this case, more than half of the chart display is covered with depth areas of region interval 9, corresponding to a depth between 25*m* and 30*m*. Smoothing the data introduces more shallows (shift to the left), while densification does not affect it much.



(b) Absolute change per region index. We introduce more — relative — shallows (6 and 7) and decrease the amount of deeps (9 and above).

Figure 4.15: Example of operator effects on the morphology in a chart. The region indices all indicate a certain interval between two isobaths. Zero is shallow, increasing with depth.

5 Experiments and analysis

This chapter presents a selection of experiments with the newly proposed methodology. We will also discuss the effectivity of its operators and process on itself, as well in relation with the original VSBA and official ENC products. Throughout the chapter, we use the datasets described in Appendix A.

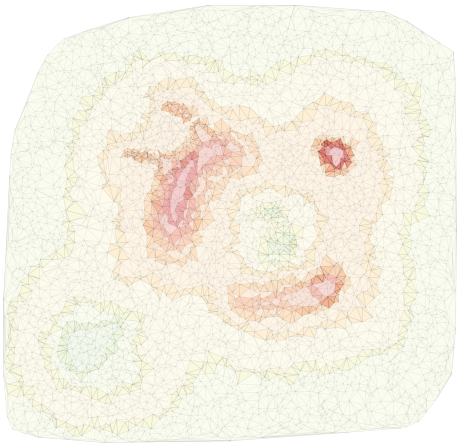
We first start with a full scale demonstration of the triangle region graph in Section 5.1. Next we show some effects and shortcomings of the generalisation operators in Section 5.2. Hand in hand with those, we check a little on the generalisation process itself in Section 5.3. What we can do with it more and how that affects the results. Lastly we use some real-world datasets to test the overall process in comparison with other approaches in Section 5.4.

5.1 TRG overview

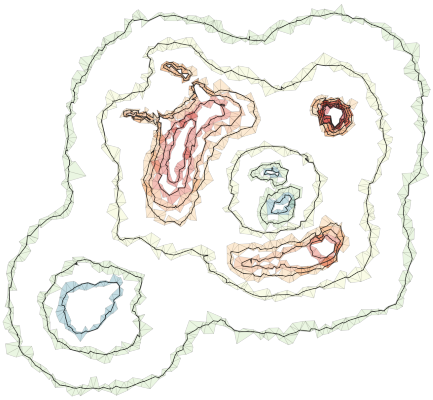
Figure 5.1 shows a full example of a generated TRG structure. In (a) the full triangulation of all measurements is shown and then coloured based on the triangle intervals. These adjacent coloured areas are the actual nodes. The nodes are coloured transparently, and it can thus be clearly seen neighbouring nodes overlap (more saturated colour). In (b) these overlapping nodes are extracted and inserted as edges. It only shows the edge triangles — coloured on their isobath value — with the raw extracted isobaths following them. Note that in some (steep) areas edge triangles may overlap and thus contain multiple parallel isobaths. In (c) the TRG structure of nodes and edges is visualised. Nodes are here displayed with their interval (top in node) and their ID (bottom in node). The graph is ordered from shallow (top red) to deep (bottom blue). Each horizontal slice therefore represents nodes of equal depth. The edges represent the isobaths from (b). Note that some nodes have either a red or blue outline. These are the local minima and maxima respectively. At first it looks simple to propagate e.g. the *peak* at node 26 up to node 6, which in some cases it is. But it becomes more difficult to classify entire features in the case of node 4. In (d) we have used the structure to generate closed depth areas (DEPARE) and coloured them accordingly to the IHO standard scheme. It directly makes it possible to make a distinction between shallow and deep (white) waters.

5.2 Effects of operators

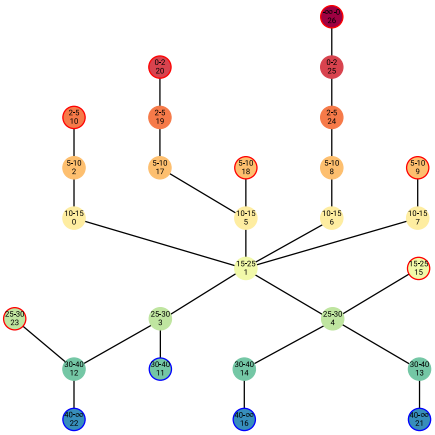
Peters (2012) already did an extensive review on the effects of both the smoothing and densification operator. We will only highlight the overall nature of those and focus on the differences in our approach and the influence of some parameters. We also pay attention to the balance with morphology.



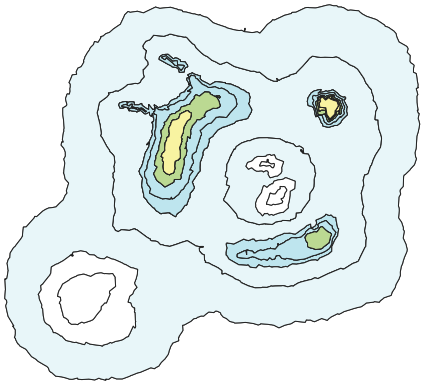
(a) Triangulation coloured on interval (nodes).



(b) TRG edges with their isobaths.



(c) Triangle region graph.



(d) IHO coloured depth areas.

Figure 5.1: A collection of available information in the TRG data structure. *Dataset Simulations*

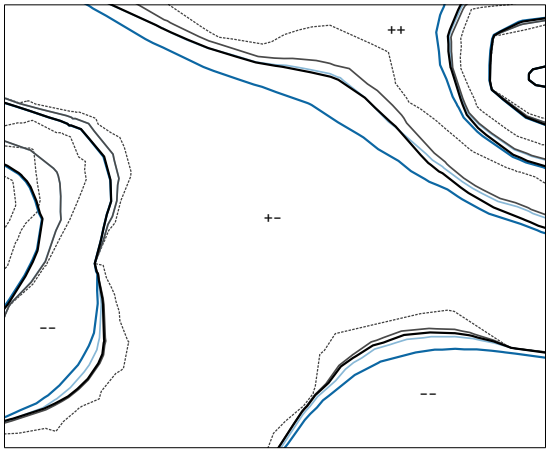


Figure 5.2: Example of smoothened isobaths. Blue by smoothing *all* vertices 50x, black by targeted smoothing 50x. Dotted line is the original isobath extracted from TIN and lighter colors only 25x smoothing.

5.2.1 Smoothing

Smoothing basically smoothens the conceptual surface, and due to the implicit function theorem isobaths share the same amount of smoothness of that surface. Figure 5.2 shows the extracted isobaths from different smoothened surfaces. It is a close-up of the data in Figure 5.1. The figure shows the originally extracted isobath (dotted) and some smoothened relatives. Note that all smoothened isobaths (black and blue) are smoother than the original, but also moved towards the deeper, safer part of the seabed. This is due to the safety constraint. The amount of smoothing therefore also has a direct relation to the morphology. In Figure 5.3 the morphology histograms are shown for all the lines in Figure 5.2. We can distinct targeted and non-targeted smoothing. With targeted smoothing we choose where to generalise and in the non-targeted we simply smoothen every vertex. But, since smoothing is still an iterative operation we can also distinct between 25 (light) and 50 (saturated) iterations. From both the figure as well as the bottom diagram in Figure 5.3 we can see more iterations results in less deep water. But also, non-targeted smoothing makes it even worse. Keeping in mind all isobaths in Figure 5.2 are considered legible, we can say morphology is better represented with targeted smoothing or applying less smoothing steps. The targeting of operators is also made clear in Figure 5.4. This figure shows every vertex which has changed, together with its vertical shift. Of course, in steep regions[21] lots of change is expected. What this figure moreover shows is that irrelevant areas — areas without isobaths, or no conflicts — are not generalised at all. Conflicts can only be found around isobaths (or saddles) and thus also only generalisation takes place in those places.

Boundary problems

We have slightly adapted the original smoothing operator to not operate near the convex hull or boundary of the dataset. The current operator will not smoothen a vertex part of this convex hull and it is optional to specify even a distance to this hull. Figure 5.5 shows why this is important. The figure shows a (deeper) channel entering the dataset

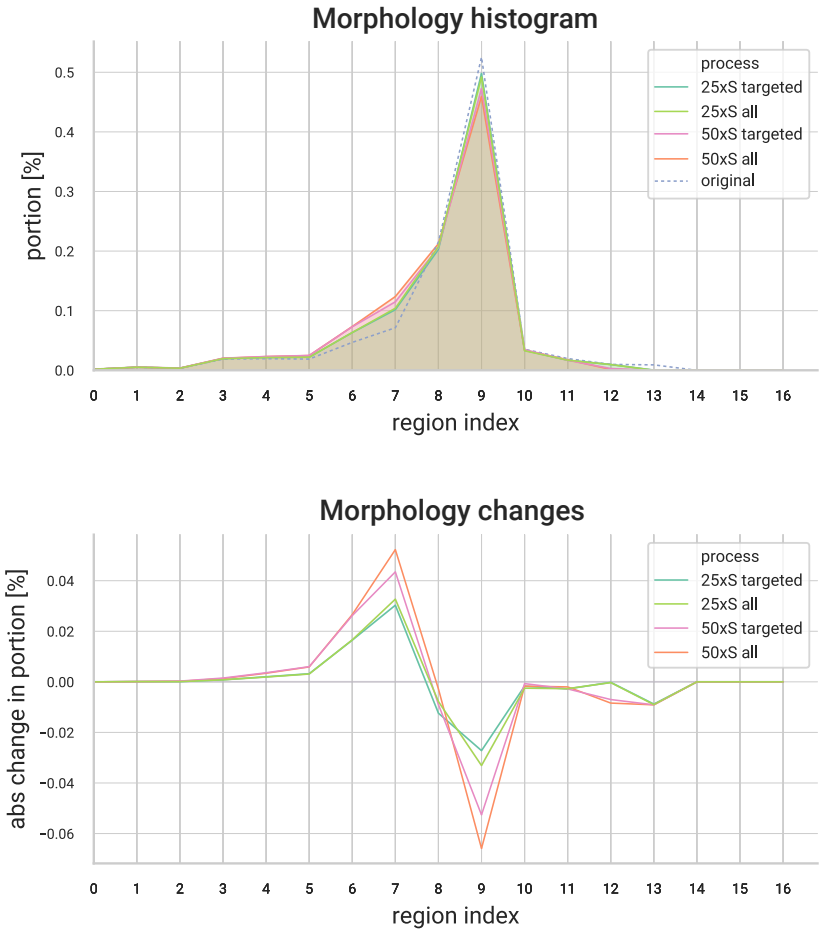


Figure 5.3: Effect of smoothing on morphology. More smoothing affects the morphology more, but targeted smoothing relatively decreases the changes.

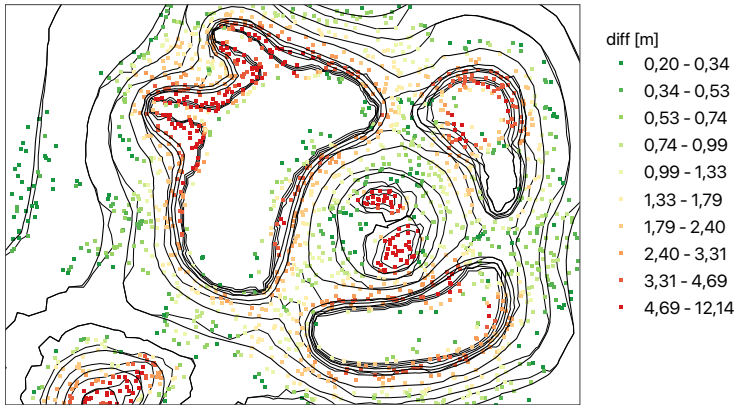


Figure 5.4: Depth changes per vertex after 50x targeted smoothing. Note that due to this targeting, only the vertices around isobaths are affected. Some areas are not smoothened at all.

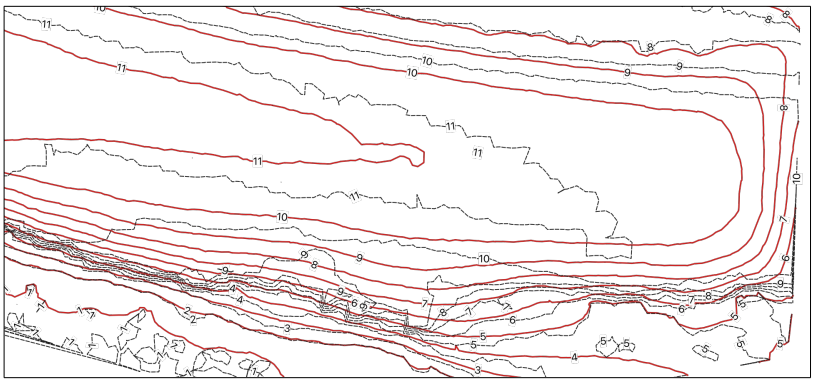
from the right. Both top and bottom parts of the data are shallower. In black the original isobaths, red the original VSBA operator and in green the adapted form. Due to the lack of information on the boundary in such a narrow gully, the original smoothing operator gives preference to a shallower depth estimate on the convex hull. To illustrate: imagine a vertex having two shallower neighbours (top and bottom) and only one (in stead of two) deeper or equal vertex (on the left in this case). Obviously, smoothing will move this vertex upwards, unlawfully. The new operator performs better in this case, but is still affected by other boundary problems like in this case a sliver polygon; an artefact of the convex hull construction.

This figure also shows the propagation of smoothing over different iterations. If it was a perfectly local operation, only small parts around the boundary would be closed. However with every iteration this red 10m isobath slowly moves to the left.

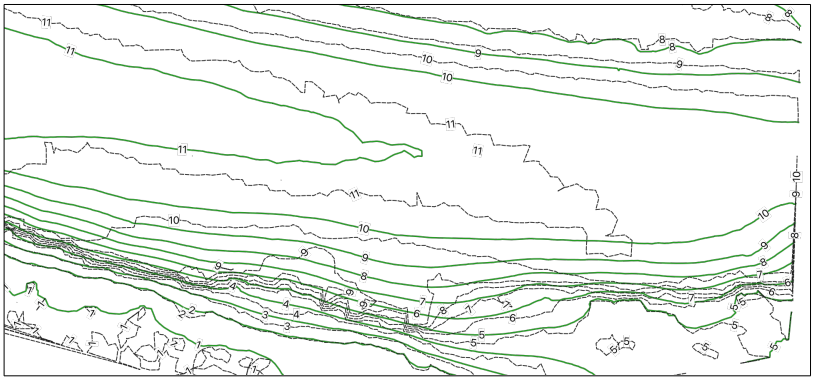
Effects of parameters

In the implemented targeted generalisation process (Figure 4.14) we basically have three main parameter groups: iteration count, thresholds, and the smoothing radius. Ideally, we would only have the thresholds group dictating how much generalisation should be applied at a certain scale. However we have seen not always an optimum solution can be found, neither the generalisation is perfectly local. Figure 5.6 shows a somehow ideal situation. It shows the original TIN isobaths (dotted) and two generalised isobaths. The black with large thresholds (spur, gully, aggregation and angularity) and green with smaller thresholds. In this situation it performs as expected: less generalisation with smaller thresholds and thus maintaining more of the morphology. For the pit on the left of the figure, even the 45m isobath is maintained, as well in other parts it can clearly be seen the green isobaths are sufficiently legible and maintain large parts of the morphology.

Both an increase in iteration count, threshold values or smoothing radius result in more generalisation. Figures 5.7, 5.8 and 5.9 illustrate this for the example dataset. More iterations simply allows the evaluation process to solve more conflicts. Larger thresholds



(a) Interpolating on boundary *closes* the channel.



(b) Skipping interpolation on boundary keeps more of the morphology.

Figure 5.5: Dataset with large gully crossing the convex hull on the right with original isobaths (dashed). Red isobaths are smoothened 50x using the original VSBA. Green lines are also smoothened 50x but now by *not* smoothing the vertices on the convex hull. *Dataset Rijkswaterstaat*

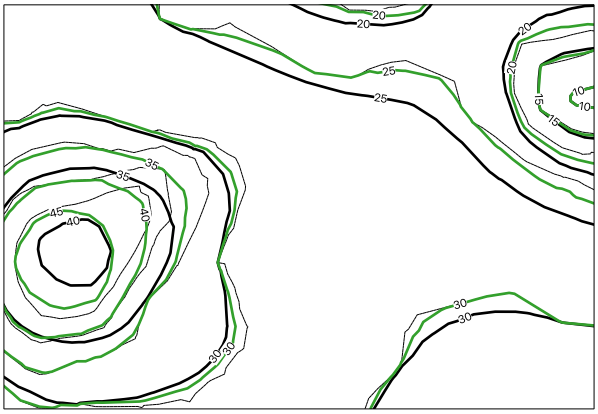


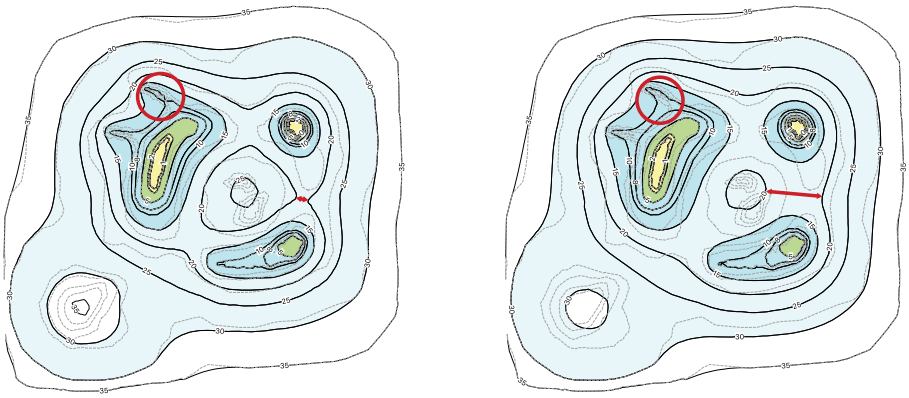
Figure 5.6: Effect on isobaths of the evaluation process. The black line is 50xS with large thresholds on spurs and gullies. The green line is smoothened more (100x), but with smaller spur/gully thresholds. Therefore, on some places generalisation was not needed anymore. More morphology is maintained of the original.

result in more conflict areas, thus an earlier smoothing of different regions. And a larger smoothing radius[22] — in the same dataset, at the same survey density — has a more aggressive approach. One could imagine if this smoothing radius was to extended to include all triangles, it would behave the same as the original VSBA.

5.2.2 Densification

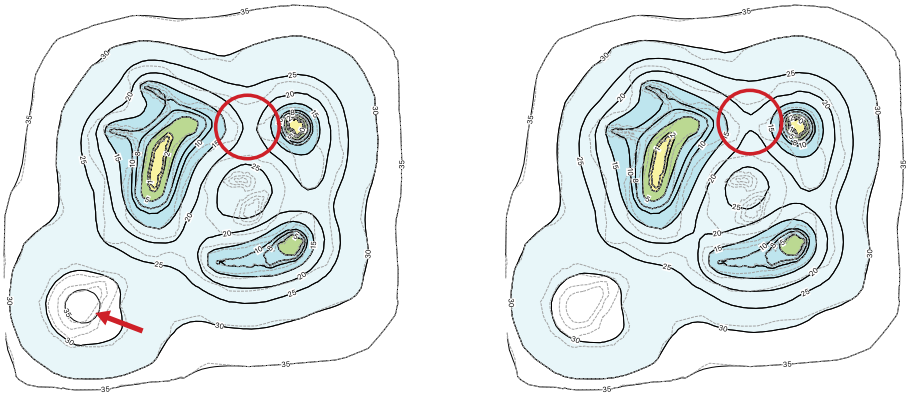
The main difference for the densification operator it now does not add vertices outside of the convex hull. The original operator simply inserted new vertices everywhere if that was the circumcenter of the face. Now, especially on the convex hull we expect some triangles with large aspect ratios. New vertices can therefore be added far away from the original convex hull resulting in strange behaviour. For the rest, the effects of the densification operator is unchanged. We do have more control over which triangles to densify. First, we only regard the edge triangles, because this is where isobaths are present and the discretisation error counts (Figure 5.10a). We can also first check where the isobaths contains sharp corners and only densify those triangles incident to this sharp vertex (Figure 5.10b). Usually we only use the angularity metric as a threshold. For example, only densify those triangles incident to a vertex sharper than 60 degrees. With every densification step angularities are then decreased until no conflict is found anymore. Where the original VSBA densified everywhere, also if there was no isobaths to be found, the new approach only densifies the triangles of interest: at the isobaths. Figure 5.10c and d illustrate this targeted densification. It can clearly be seen how vertices are only added at the edge triangles, while leaving triangles in between untouched.

The implications of a densification iteration is twofold. Firstly, if we use angularity as a threshold, total angularity will of course decrease. With it, we assume also smoothness of the line increases. Figure 5.11b shows the distribution of angularities in the original, non-targeted and targeted densification. Densification of all edge triangles shifts this distribution significantly to the left. With targeted densification we clearly see only



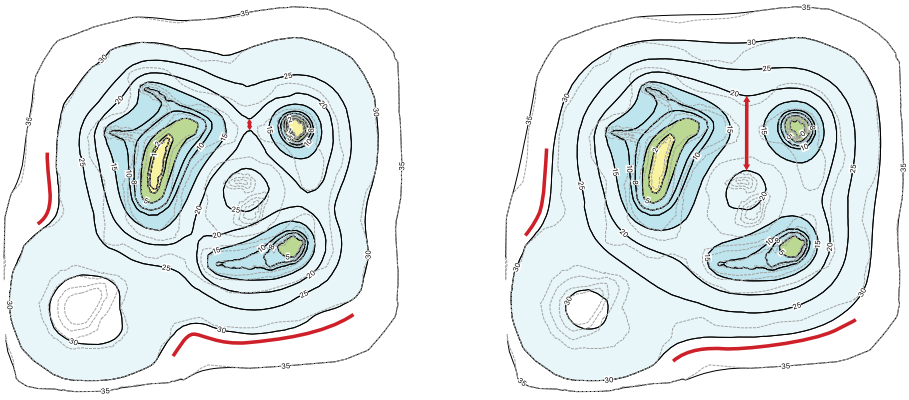
(a) 100xS 3xD (b) 200xS 3xD

Figure 5.7: Effects of the number of iterations.



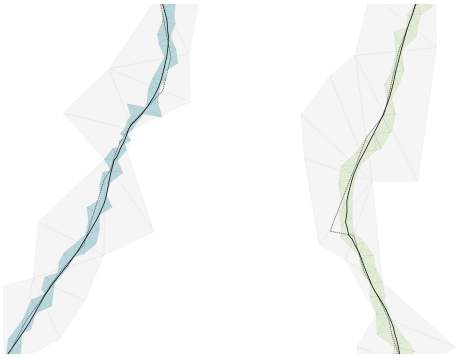
(a) Small thresholds (b) Large thresholds

Figure 5.8: Effects of choosing larger thresholds.

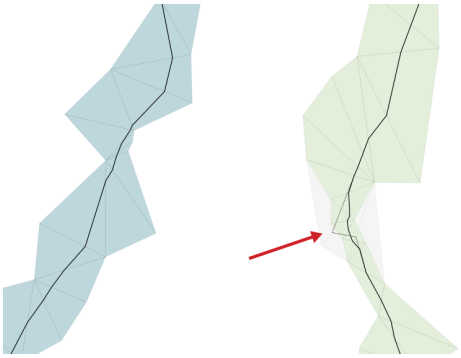


(a) Small radius R1 (b) Large radius R5

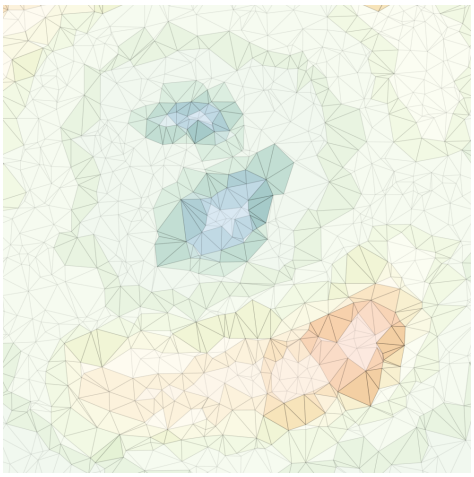
Figure 5.9: Effects of larger initial smoothing radius.



(a) All edge triangles densified 3x.



(b) Only incident edge triangles to sharp corners densified (max 3x).



(c) Original triangulation

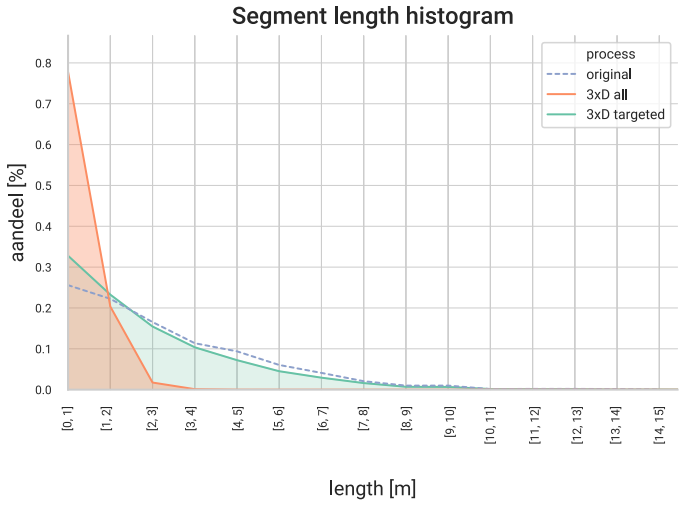
(d) Triangulation after 3x densification

Figure 5.10: Densification effects on edge triangles and isobaths. Grey are the original edge triangles, coloured the densified edge triangles. Grey dashed is the original isobath. The red arrow indicates a conflict. In (c) and (d) it is illustrated how only edge triangles are affected with respect to node triangles.

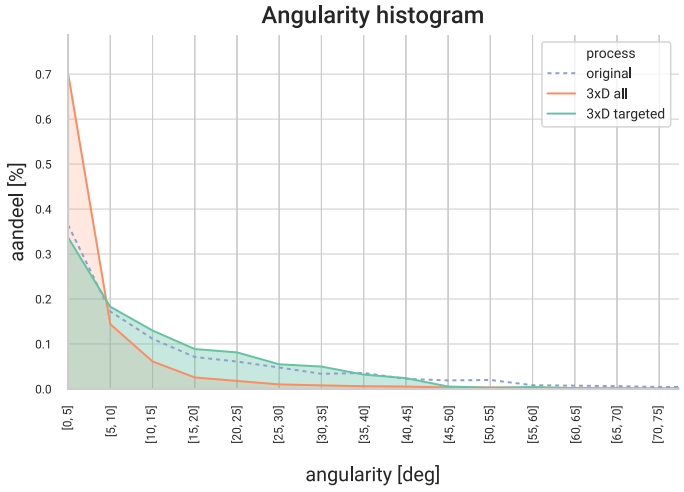
those angles above the threshold (45°) are diminished. It is up to a cartographer to decide on a maximum allowed angle and may also differ for different scales and survey densities. However from our experiments seems somewhere below 60 degrees results in acceptable results. By densifying the triangles, we effectively decrease the segment lengths of the isobaths (Figure 5.11a). While retaining an almost identical length of the complete isobath, this implies we introduce more and more isobath vertices. And thus a computational load upon portrayal. Especially in conjunction with the maximum allowed file size of ENC cells, this may become problematic and thus one should find a balance on the maximum allowed angle.

5.2.3 Aggregation

Figure 5.12 shows an example of the aggregation operator. Of course, only shallower areas are aggregated due to the safety constraint. The operator queries all points in the

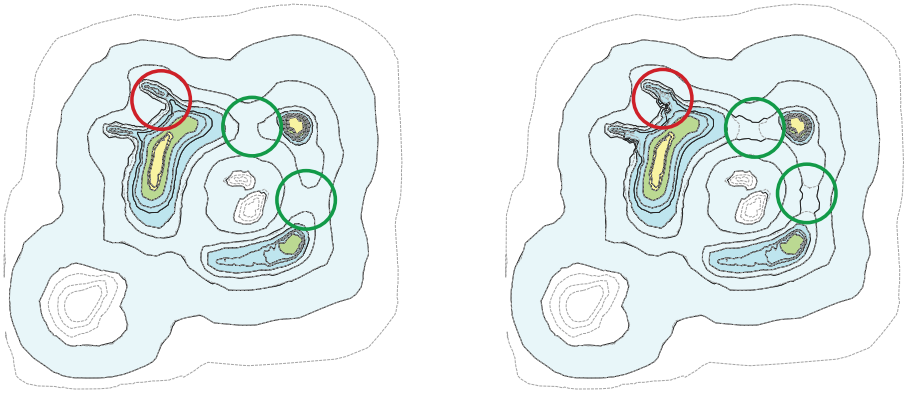


(a) Segment lengths decreases.



(b) Large angles will be removed. Note that for the targeted densification (at 45deg), the threshold is clearly visisble.

Figure 5.11: Densification effects on the validation metrics.



(a) Before (b) After

Figure 5.12: Example of aggregation. Note the two larger areas being aggregated quite nice, but the two smaller piers on the top-left are still very erratic. This is due to the harsh displacement of the vertices to the current level. Smoothing afterwards is needed.

root-node (in Figure 5.1 this is node 1) within the aggregation triangles (Figure 4.10). It extends these identified vertices with their incident triangles and then displaces those. The result is not directly smoothened, that takes place one iteration later. It therefore results in a somewhat artificial appearance. Especially in the two small piers, top left in the figure this is clearly visible.

5.3 Generalisation process

Feedback and progress

The implemented generalisation process is iterative. To illustrate what is actually happening regard the following code snippet from the feedback system. Such list is generated for each smoothing iteration. Densification and aggregation have similar reports. The first two lines indicate how many conflicting triangles are immediately incident to conflicting spurs or gullies. Sharp points indicate how much isobath vertices are conflicting due to angularity. The spur, gully and immediate incident triangles of sharp points, together sum up to the `conflictingTriangles`. As it is not simply summed, we can already conclude most of them overlap with each other. The `extendedConflictingTriangles` are all triangles from which we will extract `verticesToSmooth`. The amount of extended triangles is dependent upon the smoothing radius. From these `extendedConflictingTriangles`, we extract at least as many `verticesToSmooth`, but not all of them are safely updated. Lastly, we check if the updated vertices lead to a change of the TRG and possibly update those nodes.

```
>>> iteration 1 <<<
spur triangles:  2157
gully triangles:  1770
sharp points:  39
> isolating...
```

```
conflictingTriangles  2696
extendedConflictingTriangles  7072
> generalising...
updated vertices: 2488
changed nodes:  14
> inserting triangles again...
triangles to insert:  6945
> setting process back to zero
```

Feature classification

In Figure 5.1 we have already seen we can successfully classify local minima and maxima. This information is also reported within the process and exported to the resulting isobath file format (see code block below). With the help of the TRG we can conveniently point to either nodes or isobaths part of those features. We do not yet handle these automatically within the process, but we hope this enables the cartographer make better choices. It is at least made as convenient as possible now to handle these features afterwards.

```
PEAKS (minimum 100 m2)
  nodeId  closed      fullArea     bArea  conflict      contours
--------  --------  ----------  --------  ----------  ----------
       9  True         385.421    68.237  True                 8
      10  True         883.712    450.38  False                9
      15  True          45.171     2.728  True                14
      18  True         245.971    44.936  True                17
      20  True         1621.14   1002.57  False               19
      23  True           5.086     2.034  True                22
      26  True         577.557   266.939  False               25

PITS (minimum 100 m2)
  nodeId  closed      fullArea     bArea  conflict      contours
--------  --------  ----------  --------  ----------  ----------
      11  True          86727.9    134299  False               10
      16  True         165.758    58.679  True                15
      21  True         551.669   244.023  False               20
      22  True         3182.96   2431.68  False               21
```

Radius and locality of the smoothing operator

We have already seen (Figures 3.12, 4.12 and 5.5) it *does* matter in what order and how much neighbours we smoothen in each iteration. Increasing the smoothing radius (across triangles) enables a more aggressive smoothing and thus probably solves legibility conflicts quicker. However, due to its balance with morphology we would also like to solve conflicts as local as possible. We should make a well funded decision on the minimum radius in relation to the survey density and minimum thresholds (Figure 4.11). But, only smoothing very local does not solve all conflicts. In Figure 5.13a a typical process is shown in terms of conflicting vertices and the amount of successfully (safely) updated vertices. After some iterations, conflicts cannot be solved anymore by a local update. It is that moment we add one triangle ring to the smoothing radius: the spikes in the

diagram. This temporarily enables the ability to smoothen again some conflicts. As can be seen in Figure 5.13b, conflicts will first increase due to a moved isobath (and thus introducing new conflicts), but slowly decreases over time. While the Laplace operator is formally a *local* estimation principle, its implications thus reach way further than its original location in a generalisation process.

5.4 Use cases

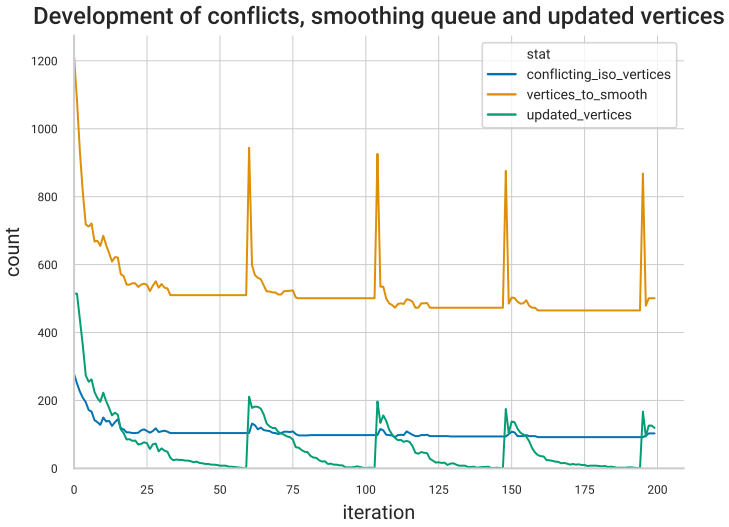
We have prepared three experiments with real-world bathymetric data. This enables a comparison on real-scales with the existing VSBA as well a comparison with officially produced ENC isobaths. All generalisation parameters are in Table 5.1.

5.4.1 New York: towards smaller scales

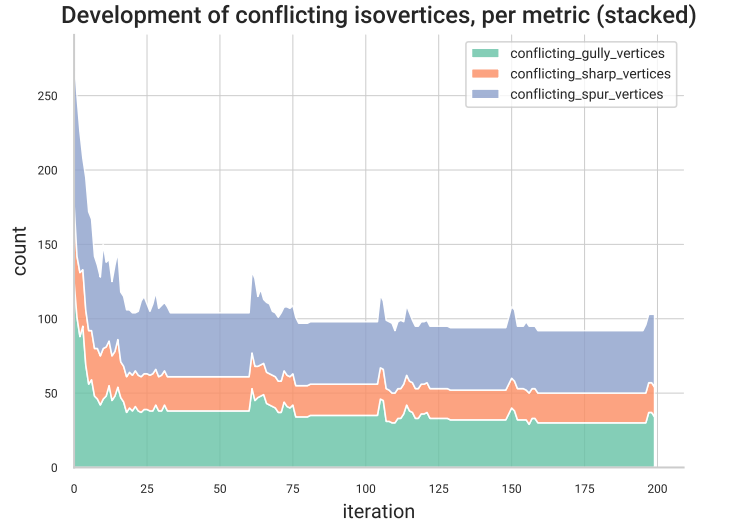
The New York dataset is used to test performance on smaller scales, and thus the need to do relatively more generalisation. For al US territorial waters the official ENC isobaths are also available as `shapefiles` which makes it convenient to compare our results with the choices a proper cartographer would make. The ENC isobaths are compiled at different scales. They are displayed together with the raw extracted isobath in Figure 5.14. All isobaths have the same value, displayed at different scales. The deep waters are at the bottom. The official isobaths clearly shows simplification, aggregation and enlargement of isobaths throughout the different scales: the desired result.

It is assumed that with an increase in generalisation parameters, we achieve more generalisation. We first tried to apply the targeted process with parameters based on the 1:120 000 scale (orange) using only Equation 3.1. However, we soon experienced this was not enough. Parameters were therefore significantly increased and aggregation threshold were based on measurements in the offical ENC isobaths themselves. The results can be found in Figure 5.15a. Both VSBA with 250 times smoothened as well as our approach with set parameters (Table 5.1) do a reasonable job. The targeted approach generalises a bit less where not needed, resulting in a more natural portrayal (red arrows). they are overall more similar to the official ENCs. However, the official isobaths are way more enlarged — e.g. on the left or the small peak at the bottom. This is something we cannot yet achieve with our automated process.

Going towards smaller scales — 1:700 000 — the problems with a smoothing approach arise. At this scale, NOAA enlarges the isobaths even more, but also aggregates further features together. We have tried to increase iteration count to yield more generalisation, but as can be seen in Figure 5.15a even 1000 times smoothing does not really adds to a solution. Our new integrated approach is also not yet able to generalise more than the original VSBA. In stead, it only limits the amount of generalisation where possible. In Figure 5.15b we have tried our only means of more generalisation than the original VSBA: a large aggregation threshold. As can be seen, it results in significant jagged artefacts, and again there is not enough enlargement of the features to achieve the official isobaths at such small scales. It would be better if features were aggregated as in the dashed blue line in a more natural way.



(a) In blue, the identified conflicts on the isobaths. Yellow are the isolated vertices, incident to those conflicts and expanded by some region. Green shows the updated vertices after applying the smoothing operator.



(b) Summation of identified conflicts on the isobath vertex level. Thus, they are not expanded by a rings. Conflicts are slowly decreasing, even with the temporary increase in newly introduced conflicts.

Figure 5.13: Effect of temporarily expanding the smoothing radius on the number of conflicts and updated vertices.

5.4.2 Dover Strait: high density large scale

Dataset Dover Strait is a high density MBES survey. No official ENC isobaths are available at different scales, apart from those on the Admirality data portal with unknown scale (Figure 5.16b). We have adapted our parameters (Table 5.1) to work with it on a 1:45 000 scale[23] and successfully generated safe isobaths of this area in Figure 5.16c. It is different from the the available Admirality isobaths, but we should not directly compare to those since we do not have an official reference scale and do not know if this is the only data used. In comparison with the original VSBA — also 100x smoothened — red in Figure 5.16d, we see large similarities. However large differences lie in the removal of the 30m pits in the VSBA. This is also reflected in the change of morphology in Figure 5.17. The TRG approach significantly maintains more of the morphology while still resulting in a finely legible chart.

We can also note from the ENC isobaths, line spacing between two consecutive value isobaths is clearly handled differently. The two small peak features in the middle of the ridge are enlarged, and actually intersect a deeper isobath (red arrows). This is an operator really difficult to achieve with the surface based approach since it violates some of the topological validness. However in the future we can use the peak classification to identify these features.

5.4.3 Margate Road: single beam data

Where the previous dataset was a high density MBES, dataset Margate Head to Margate Road is an older and less dense SBES. On the other side it has more vertical variation in its measurements, quickly resulting in a jagged representation of the isobaths (Figure 5.18a). It also contains small channels and is in general more challenging to generalise. For this dataset and portrayal, we have chosen for the parameters at a scale of 1:100 000. Compared to a same iteration count of smoothing for the VSBA, we maintain much more of the morphology and small channels (Figure 5.18c and Figure 5.19). Note the isobath values in both situations. Sometimes entire channels disappear and shallower areas are unnecessarily aggregated in the VSBA if compared to the outcome of Figure 5.18b. The green arrows for example indicate channels available with the new approach, but removed with the VSBA. And in general more gullies, pits and details are maintained (green ellipse). While retaining much of the morphology, we stil maintain a clearly legible portrayal of information, especially compared with the reference of the official ENC isobaths. At first, the generalised isobaths sometimes look too jagged (bottom-right Figure 5.18d), but compared to the ENCs (Figure 5.18b) that is actually not the case.

Table 5.1: Generalisation parameters

	New York				Dover		Margate	
	red	green	120k	700k	red	45k	red	100k
Smoothing	250	1000			100		100	
Densification	3	3			3		3	
Prepass			3	3		1		1
Densification			3	3		3		3
Spurs			200 (72)	550 (420)		30		50
Gully			200 (72)	550 (420)		30		50
Aggregation			150	650		30		144
Aspect								
Size								
Angle			60	60		60		60
Ring min			20	50		2		1
Ring max			25	80		5		4
Max iter			250	250		100		100
Scale			1:120000	1:700000	1:45000		1:100000	
d				0.3				
t				0.3				

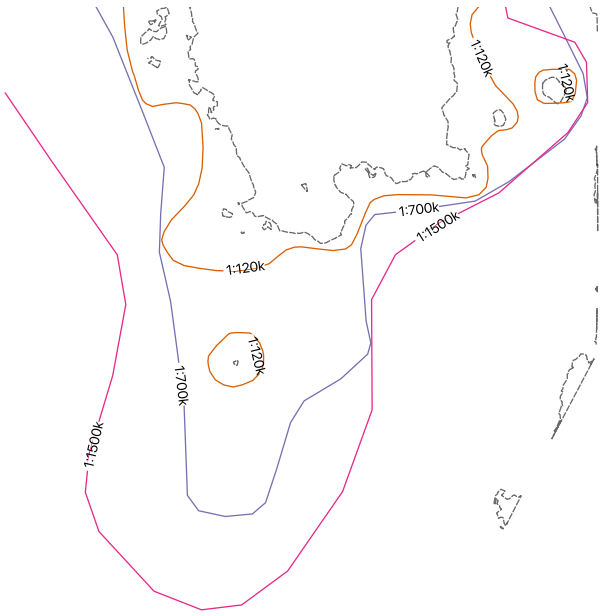
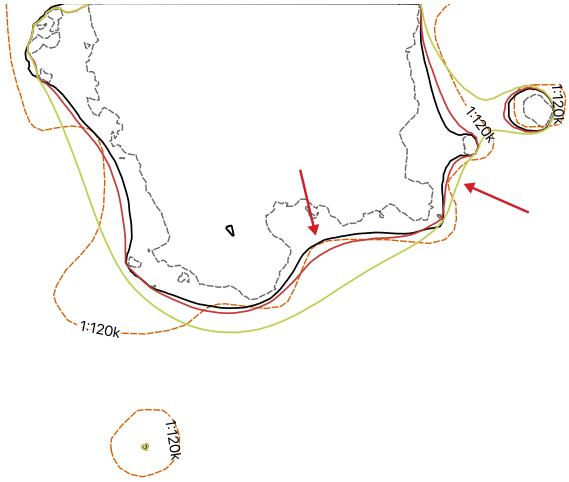
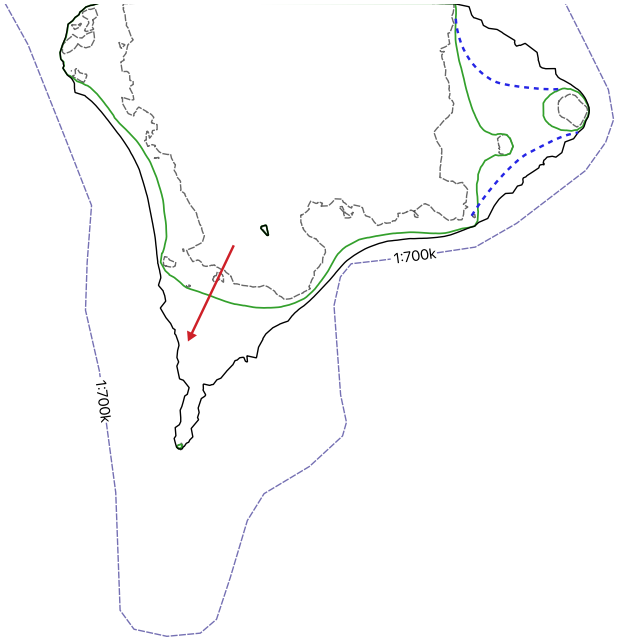


Figure 5.14: NOAA official ENC isobaths for different scales. All isobaths are the same (18.2m) with the deep water on the bottom. Dotted line is the TIN extracted isobath.

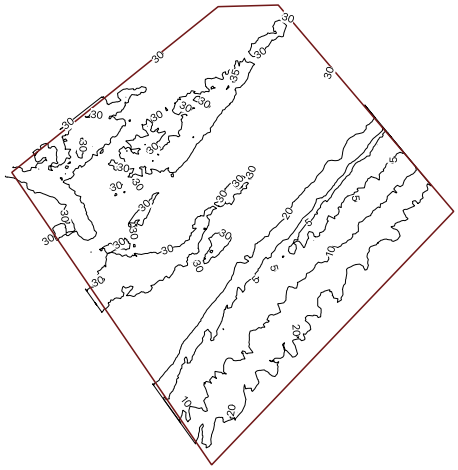


(a) Generalisation for the 1:120k scale. Orange dashed the ENC, black dotted the TIN, red VSBA 250xS, green 1000xS, black with agg4 parameters.

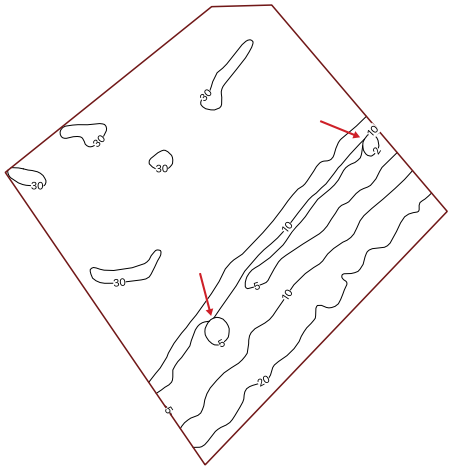


(b) Generalisation for the 1:700k scale. Purple ENC, black dotted TIN, black dashed 120k agg4, black 700kagg

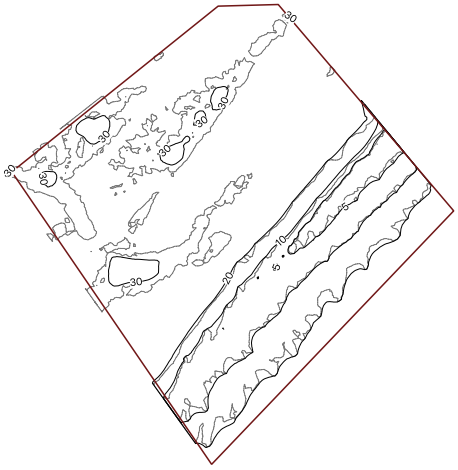
Figure 5.15: Comparison of generalisation for different small-scales. With relation to official NOAAENCs.



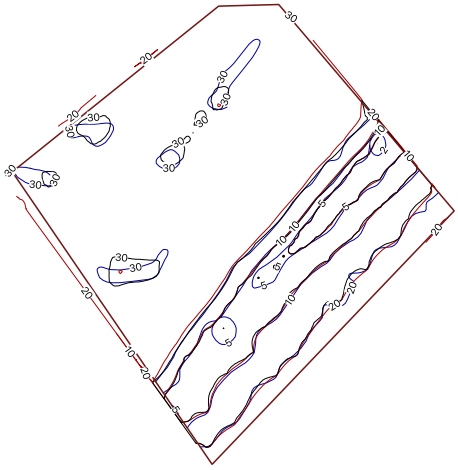
(a) Originally extracted isobaths from TIN.



(b) Official ENC isobaths. *However not directly related to this dataset, just as reference*



(c) TRG generated isobaths with defined parameters. Max 100x smoothing.



(d) Original ENC (blue), VSBA (red) and TRG approach (black).

Figure 5.16: Generalisation of dataset Dover Strait at a scale of 1:45k.

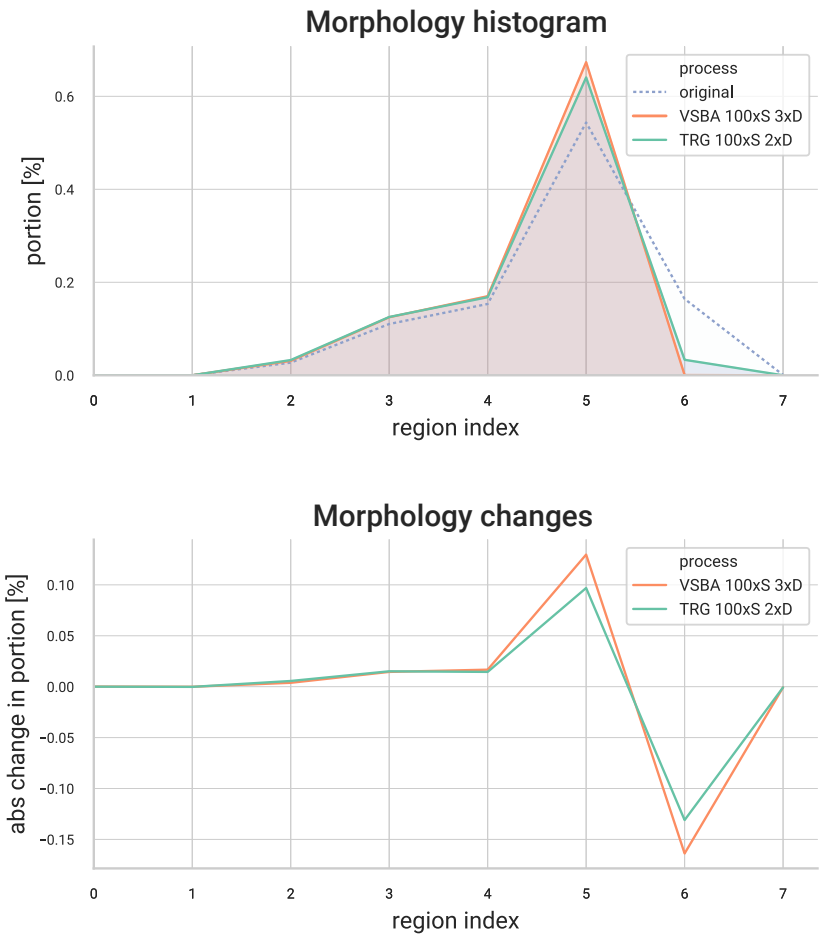
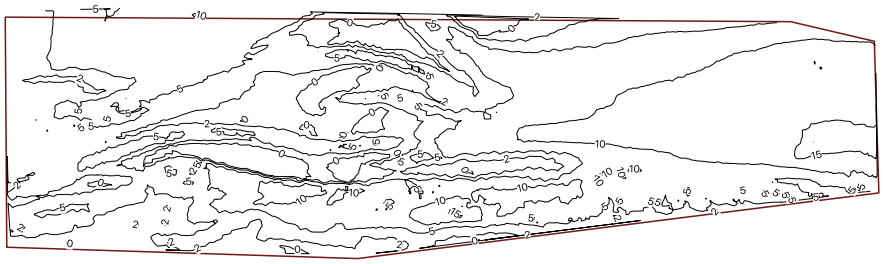
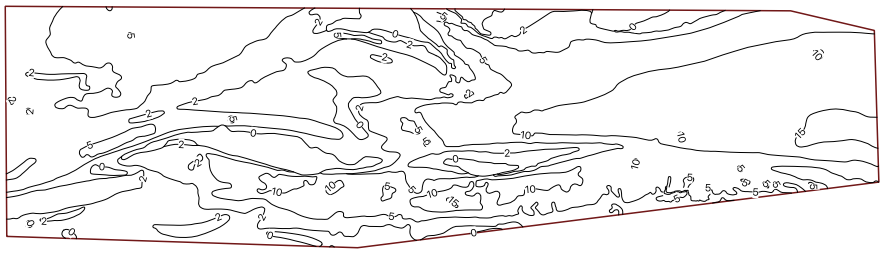


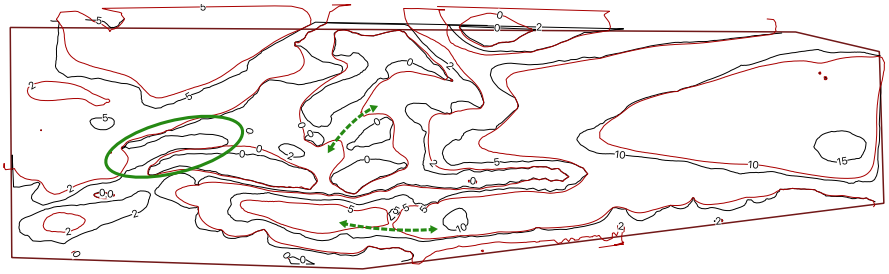
Figure 5.17: Change in morphology for both the original VSBA and TRG approaches. The targeted approach still maintains more of the morphology. *Dataset Dover Strait*



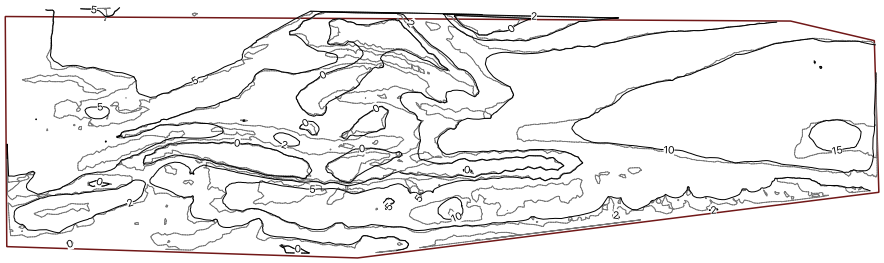
(a) Originally extracted isobaths from TIN.



(b) Original ENC isobaths. *However not directly related to this dataset, just as reference*



(c) Difference between original VSBA 100x (red) and TRG approach (black). The targeted approach maintains more of the morphology.



(d) Generalised isobaths for 1:100k scale using the TRG approach.

Figure 5.18: Generalisation of dataset Margate Head to Margate Road at a scale of 1:100k.

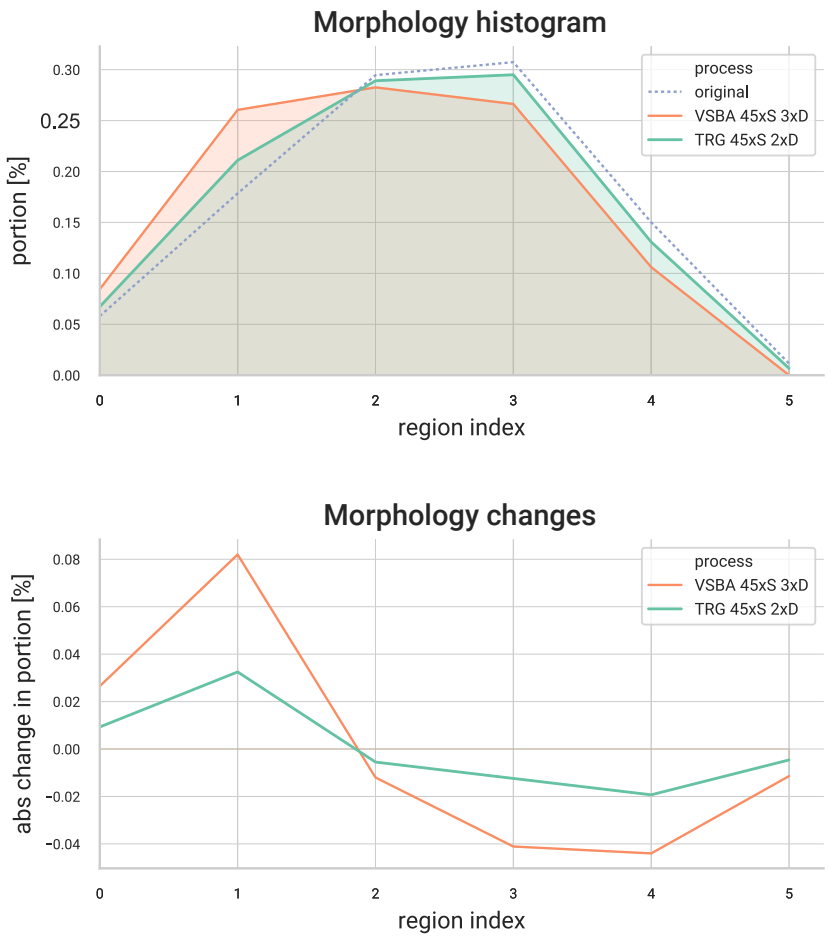


Figure 5.19: Change in morphology for both the original VSBA and TRG approaches. The targeted approach still maintains more of the morphology. *Dataset Margate Head to Margate Road*

6 Conclusions

This chapter summarises the research with first a discussion in to what extent our research objectives and questions are accomplished in Section 6.1. We summarise the properties of the methodology which have contributed significantly to the field of application in Section 6.2 and its limitations in Section 6.3. In Section 6.4 we propose some possibilities to extend this methodology further in the future.

6.1 Discussion

This section concludes the research by summarising the answers on our defined research sub questions, and finally our main research question. We have also integrated a discussion on the overall research objectives within those answers.

– *What are the minimum legibility requirements for navigational isobaths, cartographic and legally at different chart scales?*

  Legibility requirements for navigational charts are often very susceptible to the interpretation of the user. Moreover, they are different from user to user based on their intended application and personal preferences. The cartographer's role can therefore not be underestimated in the generalisation process. However, we can identify some basic requirements for the isobaths being: non-overlapping at all scales; large enough to contain a symbol; small channels irrelevant for navigation being removed; and in overall the irrelevant features should be removed. Irrelevant in this case is mainly in conjunction with the chart compilation scale and purpose of the chart. Thus whether used for berthing in a harbour or just gaining an initial overview of the area. The IHO sets requirements for display and compilation of those information, as well as guidelines how this information *must* be available. Besides the hard requirements and softer guidelines of IHO on generalisation, generalisation and specifically legibility is still seen as an *art*.

– *How can we quantify the cartographic constraints into local surface metrics?*

  Safety and topology are hard constraints: they are either valid or not. In our surface-based generalisation approach these two are valid by definition. By only moving the surface upwards, safety is guaranteed and by extracting isobaths from a 2.5D surface only guarantees topological correctness. The softer constraints — legibility and morphology — cannot be seen as simply passing a defined threshold. They are actually in constant *battle* with each other, i.e. increasing legibility by definition decreases morphology. It is always a compromise to be found and it is dependent upon purpose of the chart which situation should be accepted. In our approach we quantified basic

legibility requirements and worked towards those by compromising morphology, i.e. altering the surface. The assumption is then with a minimal acceptance of the legibility requirements, morphology is represented as good as possible. However one could also define it the other way around: requirements for morphology and compromising legibility. Such approaches can for example be useful in archeology or hydrography. It again depends on the user and its intended use.

We have tried to properly quantify legibility requirements. They do as intended as in they are able to target generalisation operators. However not all metrics can currently be solved by the proposed generalisation process and operators. Especially smoothness of a line is a very hard requirement to properly quantify for poly lines at different scales. Also, preferably these metrics would have been established on the surface directly instead of on the intermediately extracted isobaths. This is however not possible due to the irregular nature of a TIN and the multiple possibilities of slopes and number of triangles in between sets of isobaths. Only for angularity metrics it is possible using the edge triangles and triangle normals, but for the rest of the metrics we have to rely on a constant back-and-forth between triangulation and isobaths.

– *What is the effect of applying different local operators on the global surface, and how can this be exploited?*

  The densification operator successfully decreases the discretisation error prior to the extraction of isobaths. It acts completely local and it can be applied to gain a more *smoothened* and legible line. The displacement operation translates local vertices to a given depth value. It can be used to *bridge* gaps and saddles between two equal-level isobath. It raises selected vertices to a fixed level and thus the outcome is not smooth directly. Introduced conflicts should be handled afterwards. The smoothing operator at the core of the generalisation process effectively smoothens the conceptual surface. With a smooth surface also the extracted isobaths would become smooth through the implicit function theorem. While the operator itself acts perfectly local without any given user parameters, within the generalisation process it may have implications far away from that point. In general it is performing well in smoothing the surface and with it effectively smoothing lines, aggregating peaks and simplifying features. The difficulty in the overall process of applying operators, is that they introduce new conflicts and continuously push isobaths to a new deeper location. Possibly ending up in a vicious circle.

– *What are valid and realistic assumptions on input data in the field of application?*

  Bathymetric survey data comes in a lot of varieties. There is principal difference between SBES, MBES and side scanning sonar. But also variation in density and accuracy due to specific measurements technology, the ship, sea state or properties of the water. Surveys also come in a variety of geometries: small but dense harbour surveys, rectangular at open ocean or linear at a river. Afterwards this data becomes available in either vector or raster formats. The proposed methodology currently only deals with simple (non-attributed) vector data from which the convex hull is assumed to be the survey limit. Still, in theory it is able to deal with survey attributes directly and with the incorporation of breaklines we can in the future also deal with more complex bounding shapes. Difficulties arise when survey density increases, effectively decreasing the influence of the smoothing operator and the need to apply more

iterations. We can partly account for this by extending the smoothing region. The approach has acceptable results for large scale charts. We retrieve legible charts without offering too much on morphology. However difficulties arise for smaller scale charts in need of more generalisation.

– *How can the extracted features be validated and does the method perform better than available alternatives?*

  Since in this surface-based approach all measurements are processed, we can simply check for safety violations by comparing the original and current depth value at each point. Also, the process already guarantees this by definition. Topological checks like line intersections should have been eliminated by definition through extracting the isobaths from a proper 2.5D TIN. These requirements can also be validated relatively simple with basic tools available in e.g. QGIS. As said, there is always a compromise between morphology and legibility. Morphology is assumed best if the measurements are changed as few as possible. However the proposed diagrams of changing depth areas in stead of a simple measure like RMSE can supply more detail. For example it is more critical if large areas changed from navigable area to non-navigable area. If they stay non-navigable it is not that critical. We believe validating the overall chart product — isobaths and its depth areas — in stead of every single measurement, makes more sense in the perception of the user. With regard to available alternatives, the proposed method decreases the amount of change in comparison to the original VSBA, while still achieving a legible chart. However in comparison with an official ENC product like those from NOAA, the outcomes still differ significantly. Maybe this can perform better in the future by adopting different metrics and operators. Especially displacement of lines in the horizontal direction — enlargement — seems an option for further research.

– ***To what extent can we locally steer generalisation operators to account for cartographic constraints, in a surface-based isobath generalisation method?***

  We can successfully link the conceptual surface to the cartographic representation of that surface through the triangle region graph. We ultimately link the original survey data to location of isobaths and depth areas directly. We can identify and isolate conflicts based on quantified measurements related to the four generalisation constraints in one integrated process. With these conflicts we can take targeted generalisation action upon the surface directly. Through this targeted generalisation, we effectively smoothen less of the details out of the navigational surface — retaining morphology —while still ending up with a legible chart. However not all of those conflicts can be automatically resolved, or at least not within reasonable time for now. There is potential to further develop metrics and generalisation requirements around our proposed conceptual framework. Difficulties arise on smaller scale charts needing more extreme generalisation. We can simply not smooth beyond smooth for now. In the future this needs more research: either in altering the underlying surface first, or the development of other operators. The TRG can be of significant help in identifying features and neighbouring isobaths, making it possible to use this information directly in a surface-based evaluation process.

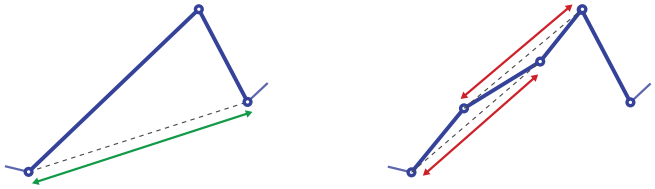
All together we have gained more control of the generalisation process in the VSBA through linking it with the cartographic presentation directly. It is not fully automatic, and strictly speaking it needs even more human-defined parameters. However with those parameters and slight modification of the operators we have achieved a more local approach and with that, retain more of the original morphology where possible.

One of the main problems in the proposed methodology is now the definition of the metrics and the solvability of those with the proposed generalisation operators. This seriously hinders the way to a truly automated approach without any errors left at the end. For example, the spur/gully metric may result in more conflicts — or even conflicts by definition — if survey density increases. Currently, isobaths should almost be perfect straight lines to not be selected as conflicts. If survey density is large (small spacing) almost every constrained triangulation element is smaller than the threshold, resulting in conflicts (Figure 6.1). A solution may for example be to also account for the length of the 'cut-off' (the path-length over the isobath to reach start and end of the inserted CDT element) isobath by the conflicting threshold. The sorted edge triangles can be of help in this operation. Also, if we have a mathematical smooth surface — thus the Laplace interpolation does not propose a new depth estimate — we can identify as much conflicts we can think of, but the smoothing of those will never result in a cancellation of those. We can best illustrate this through a very smooth but small peak. The small peak will be identified as a conflict due to its size, but smoothing of the vertices around that will not result in a new estimate since it is already smooth. In these cases, maybe the smoothing operator can best be replaced with the displacement operator. However it is still difficult to choose when and where to apply displacement in stead of smoothing. We do not want to introduce artefacts due to displacement if that is not needed. This is actually a big problem, especially with regard to the original VSBA. We can always smoothen less than the original approach by isolating conflict areas, but if the surface is smooth already we can by no means smoothen it more — within the same amount of iterations.

Additional problems may arise through boundary problems. As for example seen in Figure 5.5, the boundary of the data can have significant effects on the results. Mainly because smoothing of that boundary propagates to other locations. One should be very careful in handling small parts of larger datasets and this makes parallelization of the evaluation process quite difficult. However with the use of the TRG one could think of other models to evaluate large datasets. We tend to think in tiles, but we can for example also start at shallower nodes and slowly work our way below. Also process-wise, generalisation for different chart scales is now completely restarted from the source data. However to guarantee vertical consistency through scales it may be an option to start the generalisation process for a small scale chart from its larger scale equivalent. This can actually be performed out-of-the-box with our implementation, but simply has not been tested. Results may differ by this different order of generalisation, so it needs some more research what the actual implications are of such approaches.

6.2 Contribution

There is not much research on hydrographic generalisation publicly available. Especially not focussed on navigational charts. It is a niche and expensive field and probably there-



(a) Large segments, no conflict

(b) Small segments, conflicts

Figure 6.1: Influence of line segment length on the spur/gully conflicts. For clarity, only the left isobath segment has been split to simulate larger survey density.

fore reserved for the commercial market. I believe this research has given valuable new insights in the difficulty of achieving navigational legibility, as well as the differences in line- and surface-based approaches. Also, the proposed methodology could in theory be used to link all aspects in navigational isobath (and depth information in general) generalisation. The novel TRG data structure has proven its use in establishing a link between the TIN and to-be-extracted isobaths; as well as between isobaths themselves. Furthermore it may in the future be used in other fields of research as well.

Concretely, the VSBA which already was under the attention of commercial activity has been improved. It now has means of stopping the generalisation if not needed anymore and also gained minor improvements on the implementation level. While not automatic yet, we have moved one step ahead. The conceptual and modular approach of implementation together with the TRG also makes it in my opinion a valuable resource for further development.

6.3 Limitations

While we got a few steps closer to a truly automatic isobath generalisation method, we also found some mayor bottlenecks hindering the current process significantly. Some *can* be overcome in theory, from others it is just good to keep in mind while generalising.

We cannot smoothen already smooth surfaces The VSBA mainly relies on smoothing of the surface. However, if the surface is already smooth we cannot increase that with the available smoothing operator. This hinders the further and more radical generalisation on smaller scales significantly. We should in the future think of more appropriate operators in those situations, like the displacement or reshaping of vertices and isobaths. The aggregation operator already shows an increase in (automatic) generalisation in some situations. However the displacement introduces unwanted artefacts and reshaping has not been in the interest of this thesis since it is based on heavy human interference.

Unsolved conflicts In most of our experiments the process terminated on the maximum allowed iterations. This means not all conflicts were solved, or it needed much

more iterations to do so. Either the metrics are not suitable, i.e. they can never be accepted all at once; or the generalisation cannot overcome the conflicts; or we cannot properly quantify the subjective legibility requirements at all.

Do not underestimate a cartographer's eye The experience and interpretation of a well-trained hydrographer or cartographer cannot be underestimated. In some cases this methodology still makes just totally different *choices* than a human cartographer would.

Local smoothing While usually an interpolation technique is considered *good* if it is local and free of user defined parameters, in some cases it is counter productive. Our smoothing operator only affects its local Voronoi region. This region differs with the survey density and thus depending on this density the effective smoothing region may differ (Figure 4.11). Also, because only local vertices are updated in one smoothing pass, it may take some iterations to fully smoothen and solve conflicts in a particular (triangle-) region of interest (Figure 5.13a).

Isobath vertices Isobaths extracted from high density TINs would have a lot of vertices. Especially if we compare it to lines in an officially compiled ENC. There are no means yet to simplify this afterwards, but is not seen as an important part in this conceptual phase. It becomes more problematic with higher density survey data. Line segments of maximum 2m (survey density) in a 1:100k chart are simply not needed. In stead they have unnecessary impact on computational resources.

6.4 Future work

We have now put a promising conceptual framework in place with some basic processing capabilities. I believe the framework has potential but it certainly needs more research or the addition of concrete tools. We present a non-exhaustive list here. These can be overcome relatively easy and may truly enable the framework in its full potential.

Additional operators The approach needs more radical generalisation operators for smaller scale charts. This goes together with the limitations on *smoothing beyond smooth* and the unsolved conflicts. One could think for example of an automatic reshaping operator, which is similar to aggregation but in stead of lifting vertices to a fixed value, it directly assigns a (smoothly) interpolated value. The framework as is does allow for modules or plug-ins for these kind of extensions be implemented with relative ease.

Other evaluation models Currently a very simple iterative rule-based model is implemented. The framework however allows for different approaches. It would for example be nice to see how a user-interaction model with a human operator would work or try to pursue a true optimisation strategy, i.e. a constrained-based model. However the difficulty for optimisation is the sheer amount of possible operations. Results are dependent upon the number and location of vertices smoothened in each iteration but also the order of which it was applied. Furthermore we can also look into morphology requirements in stead of legibility for different purposes than navigational charts. And finally it would be interesting to see what happens if we do not

take a tiled approach, but start generalising based on the node-hierarchy, e.g. start with the shallowest nodes. The problem with this latter however is that currently not all conflicts may be solved, ending up with only generalised shallow nodes. Relating to the solving of conflicts, we can also investigate an approach in which not all identified conflicts are tried to overcome at once but in a hierarchical model. So first solve the most critical ones. Each update influences the next, and with such a hierarchical model we can possibly achieve yet a more local approach.

Feature classification A region graph does not natively support containment relationships where a contour-tree does. These containment relationships in the contour-tree can for example be used for feature classification as in Guilbert (2015) or Yan et al. (2016). Containment relationships can be captured if we make the graph a *directed* one. On the other hand, Guilbert (2015) also proposes a way to create a *feature-tree* from the region-graph directly. With topographic feature classification in place, we can possibly use more complex operators relying on more complex relationships of isobaths.

Survey attributes One of the advantages of a surface-based generalisation approach is the integration of survey data directly. In theory we could therefore use the survey attributes like accuracy and lineage directly. This is however not implemented yet. Models may be developed to account for this in the future. For example by shifting the depth values upwards by its vertical accuracy, or randomly transforming it within its positional accuracy and take the shallowest value in the region. Also, the interpolation technique may in the future account for direction as well. In a fairway or on a river one would usually apply more generalisation orthogonal to the direction of travel. By assigning directed weights within the interpolation this may be achieved.

Maintenance of the TRG Updates in the TRG are currently one of the bottlenecks in the (computational) process. We delete and update entire nodes if only one triangle has changed. At first sight, an updated vertex only yields a local change (Figure 3.17) but complexity comes in if also a set of neighbours is updated. Research is needed to develop methodology to handle these semi-local updates of triangles.

Computational efficiency Now, we have implemented and tested a proof of concept. It is not very fast, actually rather slow and time consuming. There are however lots of possibilities to increase the efficiency on e.g. the generation of the TRG or computation of metrics. At the core improvements should be made in the code itself, it currently has lots of redundancy and debugging information which is actually not needed for its functionality. Next to those formal improvements, most of the evaluation process could also be parallelised, possibly boosting performance (Tsidaev, 2016; Liang and Hale, 2010).

Breaklines Currently, we only handle datasets properly bounded by its convex hull. By introducing breaklines in the TIN we may also handle more complex datasets like meandering rivers or complex harbour basins. Breaklines should then for example be added at coastlines, piers/dams, drying surfaces etcetera. Otherwise the triangulation unlawfully connects unrelated vertices with each other. An optional workflow is for example to disregard every triangulation face intersecting such a breakline or completely being outside the boundary.

Feature extraction We can successfully extract both isobaths and depth areas. The last part of the depth information — soundings — also has a relation with the generalisation surface. We have not looked into existing approaches of *golden sounding* selection, but we do believe it may be possible to link some concepts directly to this surface as well. For example, we can identify small peaks rather easily by finding the soundings which are the least generalised. Next to soundings, there needs some more implementation to be finished on the generation of non-closed depth areas — incident to the boundary of the data.

Gridded surveys For the original VSBA there is already methodology developed to be applied on regular gridded survey data. We believe survey data in vector format yields more advantages, however since most of the data is also published as BAGs it would be nice to handle these in less critical situations or where grids are the only possibility.

Line-based approach All extracted isobaths are structured consistently. They have a deeper region always on its left of the line. They are also related to eachother through the TRG which makes it possible to be handled in an additional line based generalisation process afterwards. For example to decrease the amount of vertices in an isobath, or smooth them towards the deeper side.

A Datasets

A.1 Simulations

An overview of this dataset is given in Figure A.1. Its original TRG is also given in Figure A.2. This dataset is available with the software implementation[24].

It is a fully synthetic dataset and created to test overall methodology and implementation. It therefore has multiple types of topographic features like pits, peaks, saddles, small piers, inlets and slopes. The soundings are generated by manually drawing isobaths, assigning them depth values, meshing a surface in between them and finally dropping random point on this surface. To give it some erraticness in the vertical direction we have also translated all depth values randomly. Its vertical span is between 0m and 50m. However, we have extensively (randomly) altered the data upon loading. This was to *stress*-test the implementation on for example very small nodes (single vertices). The dataset is in a cartesian coordinate system spanning 500 units in both x and y and comprises approximately 5 000 points.

A.2 Rijkswaterstaat

An overview of this dataset is given in Figure A.3. Its original TRG is also given in Figure A.4. The *blob* of shallow red nodes in the upper-left corner are due to the erraticness close to shore. These are groups of all tiny nodes, usually only a few triangles large.

This data is downloaded from the shared bathymetric service *RBB* (*Representatief Bathymetrisch Bestand*) between NLHO and Rijkswaterstaat. It is coarsely gridded data with on average a measurement every 5m, however not in a perfectly regular grid. It also contains SBES data, especially near shorelines. This dataset is again used for overall testing. Especially on boundary effects (open gullies) and testing the performance on (semi-) gridded data. The data is already projected in EPSG:28992 and comprises around 12 000 points.

A.3 Admirality

The UKHO makes both its survey data and ENC contours available through data portals[25]. We have selected two different datasets in nature as well as location and time.

A.3.1 Margate Head to Margate Road

An overview of this dataset is given in Figure A.5. Its original TRG is also given in Figure A.7. The data is available at the UKHO Bathymetric Data Portal under number HI195 and HI259 at name *Margate Hook to Margate Road*. It is completely SBES data and originates

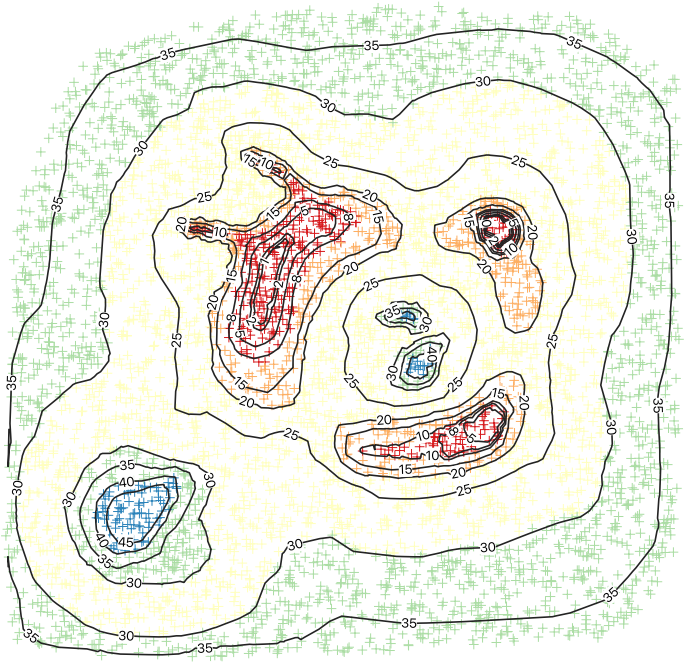


Figure A.1: Overview for the simulated dataset, including individual measurement and TIN isobaths.

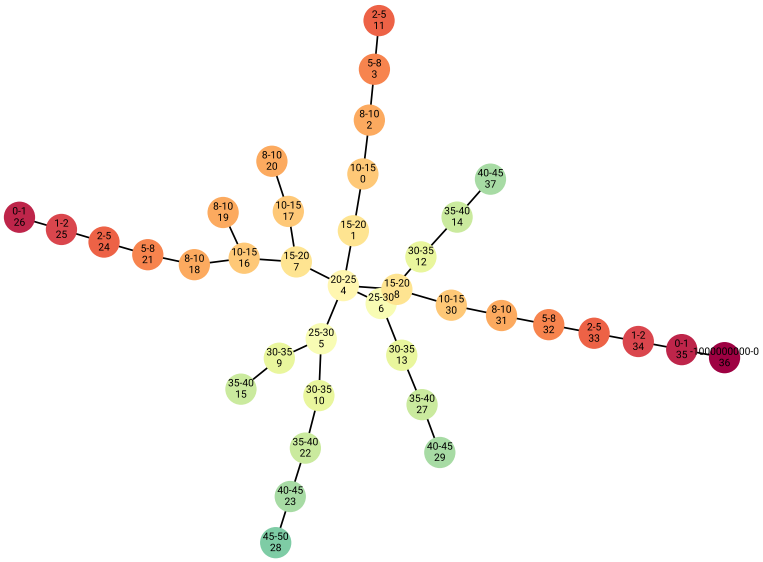


Figure A.2: TRG for the Simulation dataset

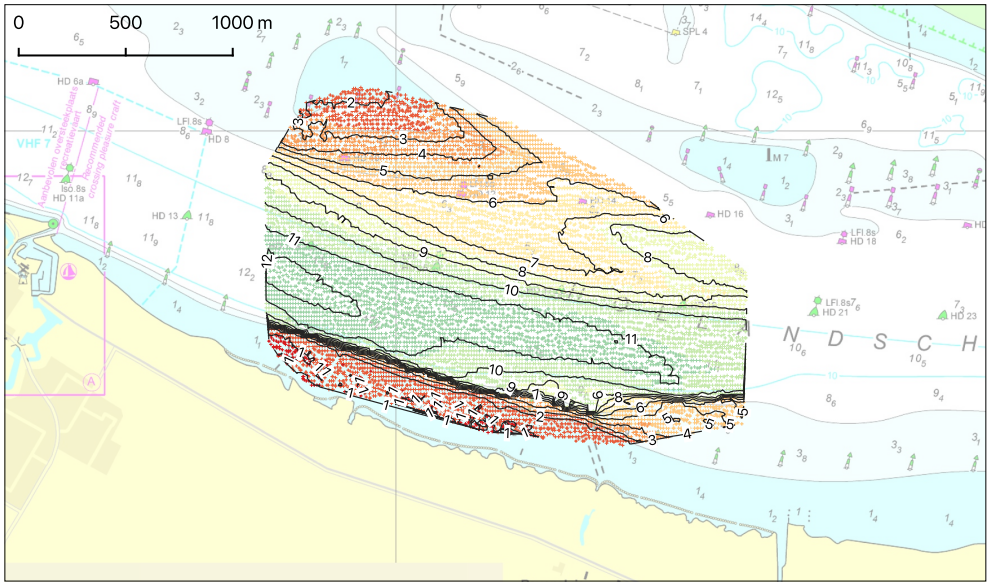


Figure A.3: Overview for the Rijkswaterstaat dataset, including individual measurement and TIN isobaths.

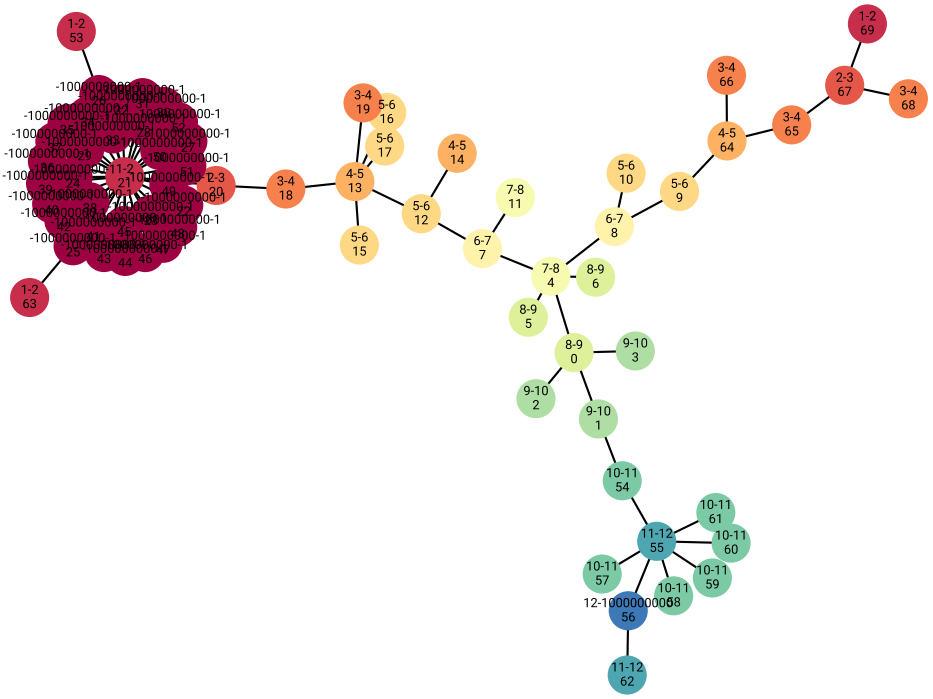


Figure A.4: TRG for the Rijkswaterstaat dataset

from 1986. It comprises around 14 000 points. Due to its nature it was especially tested for the effects of SBES in a relatively erratic area. The area is known for its fast-moving channels and sandbanks.

A.3.2 Dover Strait

An overview of this dataset is given in Figure A.6. Its original TRG is also given in Figure A.8. The data is available at the UKHO Bathymetric Data Portal under number `HI1159` at name *Dover Strait Blk7*. It originates from 2007 and is fully MBES data in `csv` format. The selected area with the most interesting features — a central shallow with some pits — is in the middle of Dover Strait. One of the most important shipping areas in the world. The selection originally contained 100 000 points. However we have decreased this through random selection to 15 000 points. This dataset is used to test more dense data from MBES surveys. It has less isobaths than the Margate Head to Margate Road dataset but still has some critical peaks and pits.

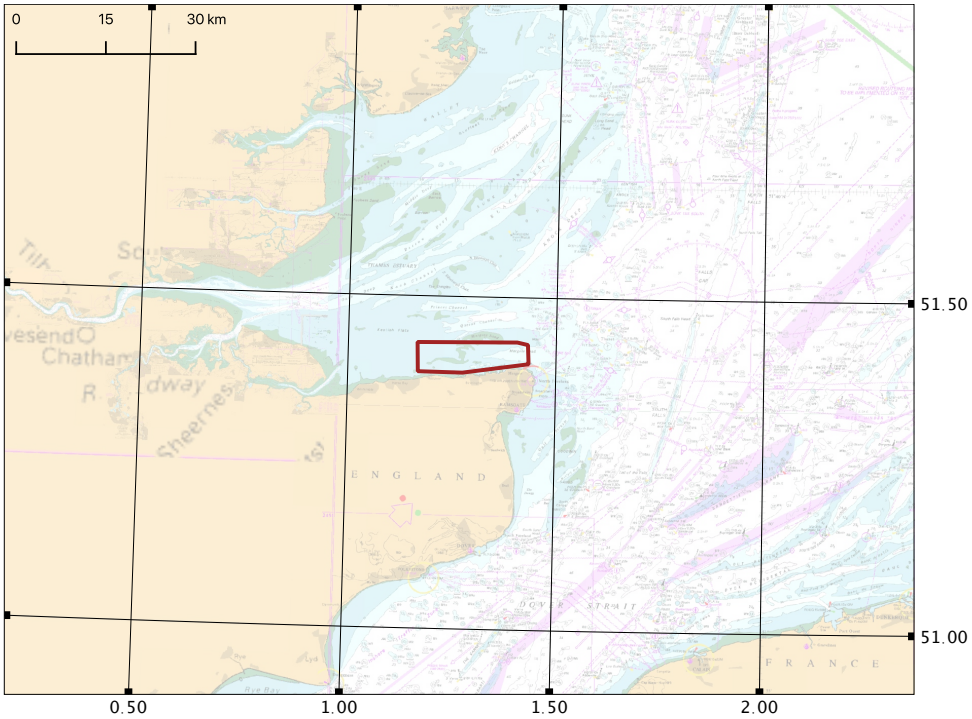
A.4 NOAA

A.4.1 New York

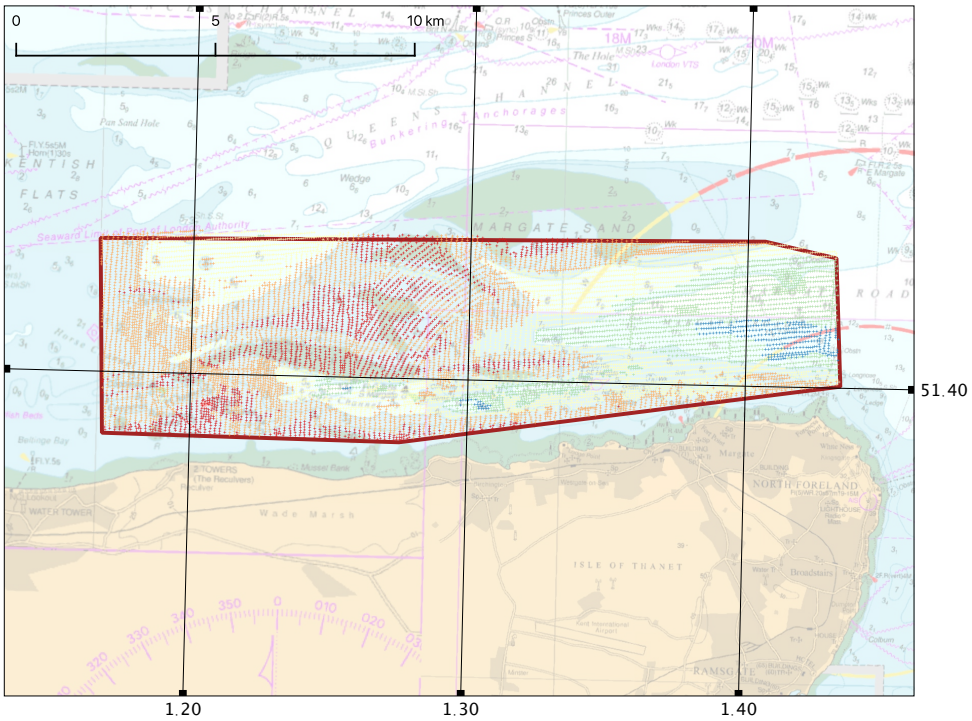
An overview of this dataset is given in Figure A.9. Its original TRG is also given in Figure A.10.

The 20 000 points are extracted from NOAA BAGs. These BAGs were originally downloaded from the NOAA bathymetric data portal[26]. The BAGs with identifiers `H12610` and `H12608` were first merged together at a resolution of 2m. From that merged BAG we extracted the area of interest and extracted points at the center of each grid cell. The main reason for working with this data is the availability of the official isobaths as well, and especially *not* the data in BAG format. We thus have randomly transformed each point within the limits of its grid cell (2m). Still we assume this to be a *true* situation, since it is also unknown but possible the shallowest point in each grid cell can be found randomly within its limits.

The true use of this dataset is comparison to its official ENC isobaths. Both datasets are recent and modern MBES surveys from 2015 and 2014 respectively. No newer surveys are known and thus it is assumed *this* is the data the official ENCs are based upon.

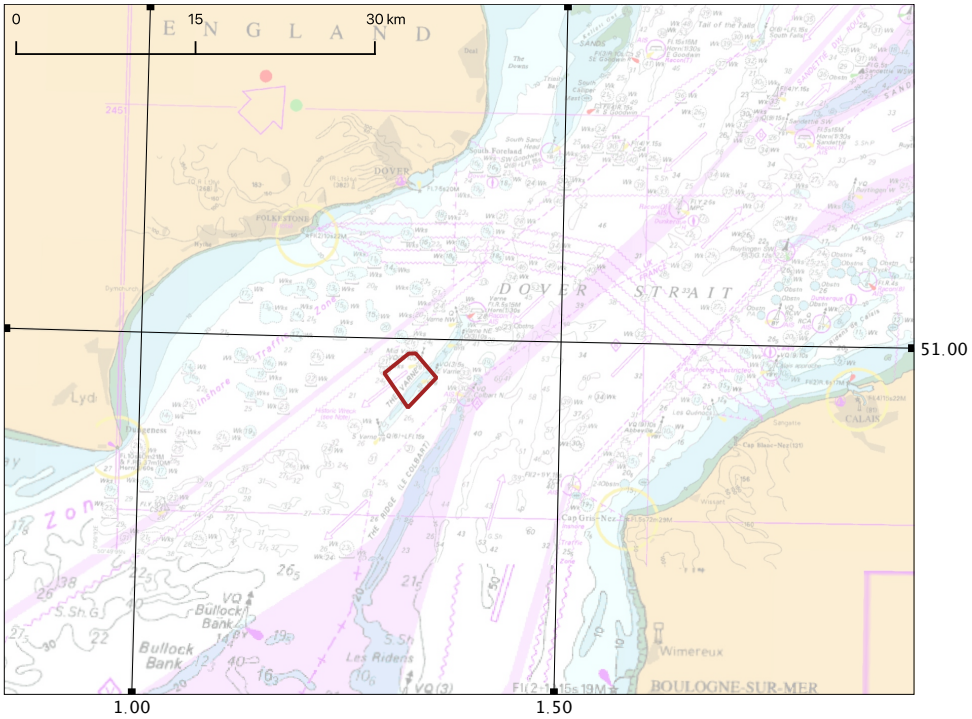


(a) Approximate location in relation to the North Sea and Dover Strait.

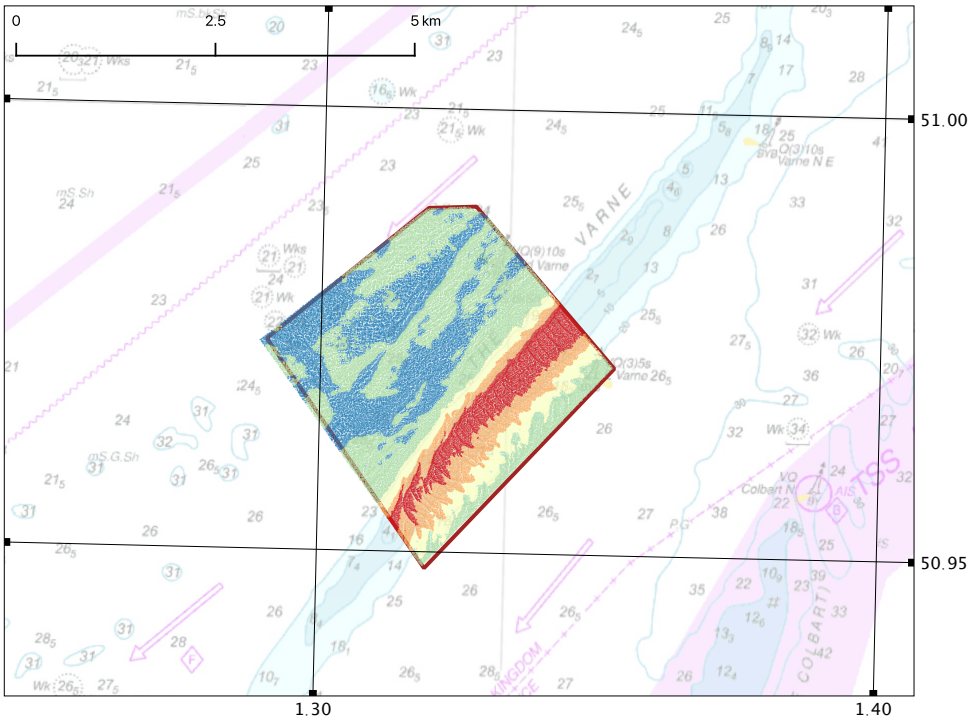


(b) Location relative to the coast and other significant seabed features. Individual measurements are also shown.

Figure A.5: Overview for the Margate dataset.



(a) Approximate location in the Dover Strait.



(b) Location relative to the banks and other significant seabed features. Individual measurements are also shown.

Figure A.6: Overview for the Dover Strait dataset.

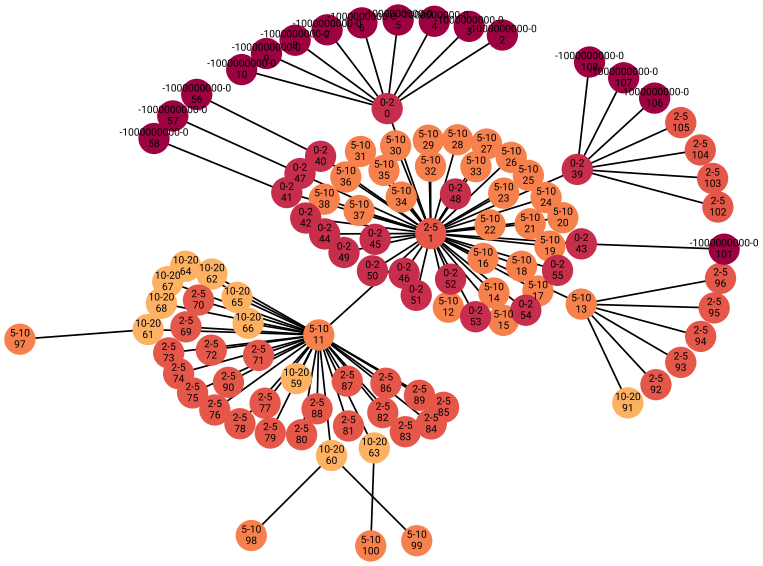


Figure A.7: TRG for the Margate Road dataset

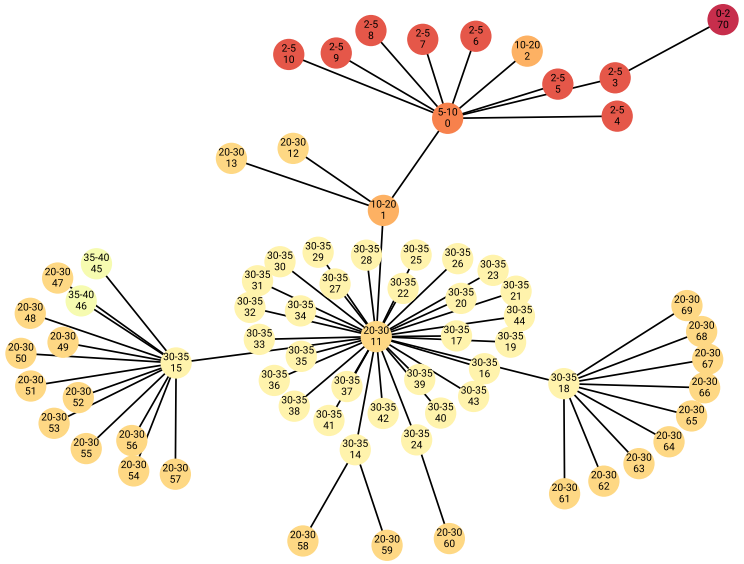
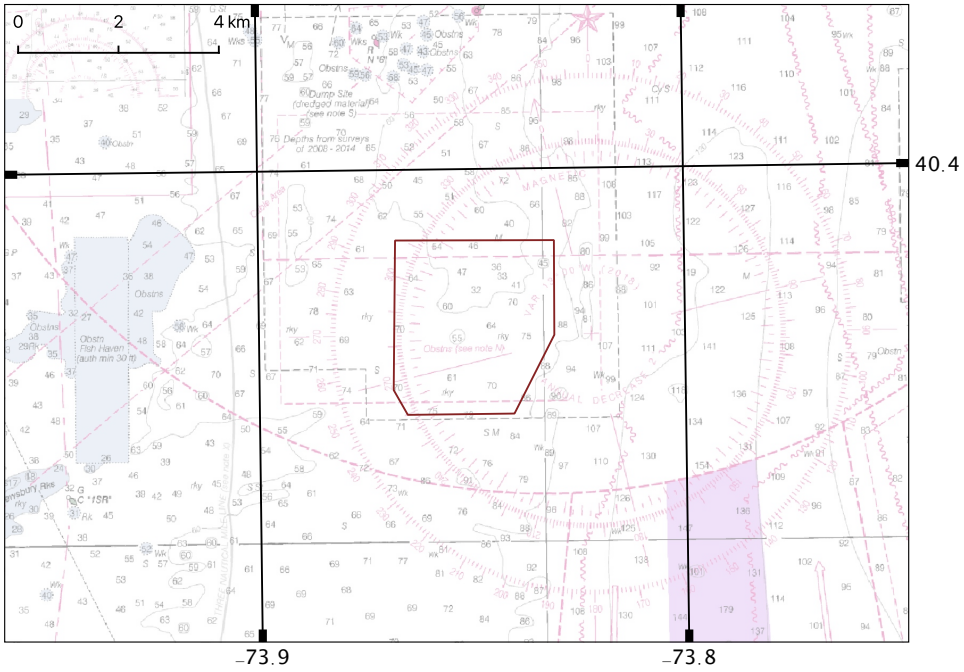
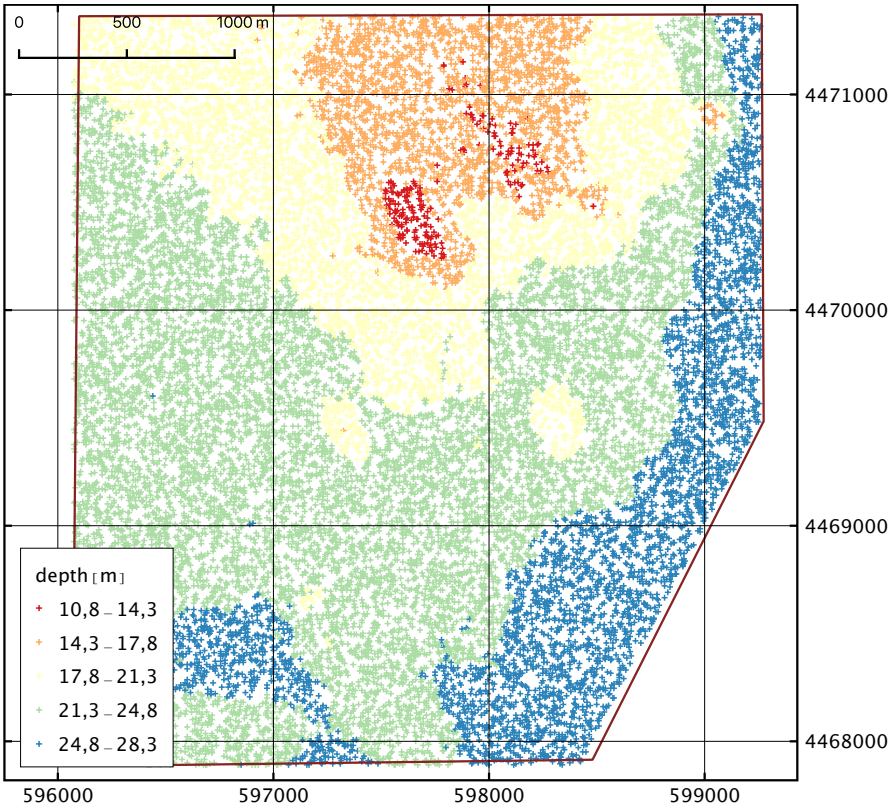


Figure A.8: TRG for the Dover Strait dataset



(a) Approximate location with respect to seabed features.



(b) Used soundings for analysis.

Figure A.9: Overview for the New York dataset.

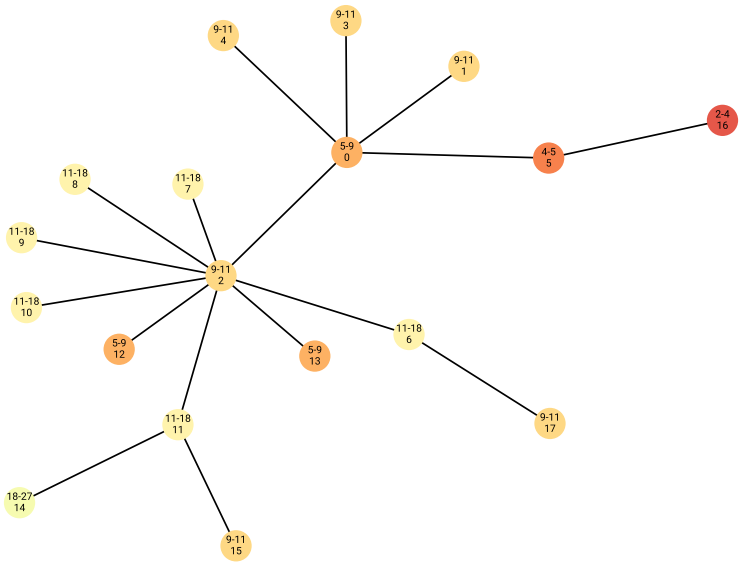


Figure A.10: TRG for the New york dataset

Notes

1. https://www.wartsila.com

2. International Maritime Organisation (IMO) member states are obliged to create navigational charts in parts of their international waters through IMO regulation. However, the form in which these charts should be available in is not dictated by IMO. Instead, IHO regulates the standard content, visualisation and availability of the actual charts.

3. Isobaths are a form of *isolines*. Isolines actually meaning 'lines of equal height or value'. These isolines are better known as contour lines and can be seen on e.g. topographic maps. Underwater isolines are better known as isobaths or depth contours.

4. Large scale charts typically covering smaller areas of land with lots of details. Small scale charts may cover larger areas, but typically with less detail. A small scale chart has been 'zoomed out' more than a large scale chart.

5. While lines in a computer representation are usually of infinitely small thickness, upon visualisation on a screen they do get thickness assigned. *Visual overlap* happens when this visual representation starts to overlap and the user may not distinguish two separate lines anymore. This does not mean the exact lines are intersecting.

6. The graduation plan can be found in the same repository as this thesis at https://repository.tudelft.nl.

7. The NLHO 1800 series chart folio is especially designed for consumers and small (commercial) vessels. It is usually used by recreational sailors, fishermen or divers. The NLHO also compiles the *national* and *international charts*, which are the official product to be used by commercial vessels.

8. While a point cloud is a very abstract definition of a surface, it can actually yield a good overview of a certain terrain. To do so it should have a sufficiently dense spatial distribution. A terrain representation through a point cloud is actually not that different from a contour map. As in a contour map we do not have the full surface representation, but our perception is able to create one from it.

9. The Delaunay triangulation is non-unique in some special cases. For example for perfectly co-linear points or points distributed in a perfectly regular grid. For now, these properties are assumed not very critical and thus disregarded. Also, enough precision (within the limits of the machine) sort of makes it impossible to have *perfectly* aligned points.

10. While in theory every point can be taken into account, complications arise for duplicate points — with or without changed attributes. For 2D navigational charts this problem is minor, since these should be safe and thus the shallowest point has to be selected. 2.75D TINs are not yet handled.

11. S-100 will support more types of depth information and bathymetric data visualisations directly available in ENCs. Both (irregular) triangulations and gridded data will be supported for direct visualisation and minor transformations.

12. *See note 3.*

13. A polyline is a collection of connected straight line segments. An isobath extracted from a linear TIN usually is such a polyline.

14. *Golden soundings* are the most important soundings in a local region. They are either typical for a larger region or indicate specific obstructions and opportunities. They can for example be selected as the deepest point in an anchorage, the shallowest point of a spur or simply indicate the overall depth in the region.

15. The software *Hydropolator* is available at https://github.com/willemvanopstal/hydropolator under an open-source license. Details and documentation can also be found there.

16. The original VSBA software *Surfonoi* is available at `https://github.com/Ylannl/Surfonoi` under an open-source license.

17. The software *StarTIN* is available at `https://github.com/hugoledoux/startin_python` under an open-source license.

18. Bathymetric source data and basic depth contours of the UKHO are available at `https://data.admirality.co.uk`.

19. NOAA bathymetric source data is available at `https://maps.ngdc.noaa.gov/viewers/bathymetry` and official ENC depth contours at `https://encdirect.noaa.gov`.

20. The complete implementation of TRG generation is more complex. For reference; in the (current) prototype this is handled by the functions `build_graph_new2()` and `expand_current_node()` in `Hydropolator.py`. However the same results can probably be acquired through different methods.

21. In a contour map steep slopes are areas where consecutive contour lines are very near to each other.

22. The smoothing radius is the the number of neighbouring triangles in terms of triangle rings which are incident to the identified conflict.

23. The choice of this scale was merely a practical one: it is the scale which resulted in a proper display of all details in QGIS.

24. *See note 15.*

25. *See note 18.*

26. *See note 19.*

Bibliography

AHO (2019). AHO's experience producing High Density (HD) bathymetric ENCs. IHO ENCWG3-5.2.1.

Ai, T., Ke, S., Yang, M., and Li, J. (2016). Envelope generation and simplification of polylines using delaunay triangulation. *International Journal of Geographical Information Science*, 31(2):297–319.

Bajaj, C., van Kreveld, M., and van Oostrum, R. (1998). *Contour Trees and Small Seed Sets for Isosurface Traversal*.

Baumgart, B. G. (1975). A polyhedron representation for computer vision. In *Proceedings of the May 19-22, 1975, national computer conference and exposition on - AFIPS '75*. ACM Press.

Blandford, D. K., Blelloch, G. E., Cardoze, D. E., and Kadow, C. (2005). Compact representations of simplicial meshes in two and three dimensions. *International Journal of Computational Geometry & Applications*, 15(01):3–24.

Cronin, T. (1995). Automated reasoning with contour maps. *Computers & Geosciences*, 21(5):609–618.

Guibas, L. and Stolfi, J. (1985). Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Transactions on Graphics (TOG)*, 4(2):74–123.

Guilbert, E. (2012). Multi-level representation of terrain features on a contour map. *GeoInformatica*, 17(2):301–324.

Guilbert, E. (2015). Feature-driven generalization of isobaths on nautical charts: A multi-agent system approach. *Transactions in GIS*, 20(1):126–143.

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

Guilbert, E. and Zhang, X. (2012). Generalisation of submarine features on nautical charts. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, I-2:13–18.

Gökgöz, T. (2005). Generalization of contours using deviation angles and error bands. *The Cartographic Journal*, 42(2):145–156.

Harrie, L. and Weibel, R. (2007). Modelling the overall process of generalisation. In Mackaness, W., Ruas, A., and Sarjakoski, T., editors, *Generalisation of Geographic Information: Cartographic Modelling and Applications*, pages 67–88. Elsevier.

IHO (2015). S-52: Specifications for chart content and display aspects of ECDIS.

IHO (2018a). S-100: Universal hydrographic data model.

IHO (2018b). S-11: Guidance for the preparation and maintenance of international (INT) chart and ENC schemes and catalogue of international charts.

IHO (2018c). S-4: Regulations of the IHO for international (INT) charts and chart specifications of the IHP.

IHO (2018d). S-66: Facts about electronic charts and carriage requirements.

IMO (1974). International convention for the safety of life at sea (SOLAS).

Kastrisios, C. and Calder, B. (2018). Algorithmic implementation of the triangle test for the validation of charted soundings.

Kastrisios, C., Calder, B., Masetti, G., and Holmberg, P. (2019). On the effective validation of charted soundings and depth curves.

Kweon, I. and Kanade, T. (1994). Extracting topographic terrain features from elevation maps. *CVGIP: Image Understanding*, 59(2):171–182.

Ledoux, H. and Meijers, M. (2013). A star-based data structure to store efficiently 3D topography in a database. *Geo-spatial Information Science*, 16(4):256–266.

Ledoux, H., Ohori, K. A., and Peters, R. (2019). *Computational modelling of terrain*. TU Delft 3DGeoInfo.

Liang, L. and Hale, D. (2010). A stable and fast implementation of natural neighbor interpolation. Center for Wave Phenomena, Colorado School of Mines.

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

McMaster, R. (1992). *Generalization in Digital Cartography Resource Publications in Geography*. Association of America Geographers.

Mellor, T. (2018). Automated contouring algorithm. Presented at ENCWG 3.

Moggert-Kageler, F. (2018). New challenges for digital chart production. *Hydro International*.

Muller, D. and Preparata, F. (1978). Finding the intersection of two convex polyhedra. *Theoretical Computer Science*, 7(2):217–236.

NLHO (2019). Berichten aan zeevarenden. [Online; accessed 12-December-2019].

Peters, R. (2012). A voronoi- and surface-based approach for the automatic generation of depth-contours for hydrographic charts. Master's thesis, TU Delft.

Peters, R., Ledoux, H., and Meijers, M. (2014). A voronoi-based approach to generating depth-contours for hydrographic charts. *Marine Geodesy*, 37(2):145–166.

Roubal, J. and Poiker, T. K. (1985). Automated contour labelling and the contour tree.

Shewchuk, J. R. (2005). Triangle: A two-dimensional quality mesh generator and Delaunay triangulator. https://www.cs.cmu.edu/ quake/triangle.html.

Sibson, R. (1997). Contour mapping: Principles and practice.

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

Tsidaev, A. G. (2016). Parallel algorithm for natural neighbor interpolation.

UKHO (2018). UKHO produces ENC with one-metre depth contours to support navigation in the Bristol Channel.

University of New Hampshire (2019). Open navigation surface project. [Online; accessed 12-December-2019].

van Kreveld, M. (1996a). Digital elevation models: Overview and selected TIN algorithms. Course Notes.

van Kreveld, M. (1996b). Efficient methods for isoline extraction from a TIN. *International journal of geographical information systems*, 10(5):523–540.

van Kuijk, D. and Engels, E. (2017). Handboek werkwijze: Generaliseren.

VTS-Scheldt (2019). Schelde IENC Maps. [Online; accessed 12-December-2019].

Weibel, R. (1997). *Generalization of spatial data: Principles and selected algorithms*, pages 99–152. Springer Berlin Heidelberg, Berlin, Heidelberg.

Yan, J., Guilbert, E., and Saux, E. (2014). An ontology for the generalisation of the bathymetry on nautical charts. In ISPRS, editor, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume II.

Yan, J., Guilbert, E., and Saux, E. (2016). An ontology-driven multi-agent system for nautical chart generalization. *Cartography and Geographic Information Science*, 44(3):201–215.

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

Colophon

This document was typeset using LaTeX, using a modified version of the KOMA-Script class `scrbook`. The source code is available upon request.

Fonts used are *Yrsa* from Rosetta Type, combined with *Montserrat* from Julieta Ulanovsky and *Merriweather* from Sorkin Type. Most plots were created through `matplotlib` using `Seaborn` styling. Figures were mostly done in `Adobe Illustrator` or drawn by hand.