A Voronoi-based approach to generating depth-contours for hydrographic charts

Ravi Peters
r.y.peters@tudelft.nl

Hugo Ledoux
h.ledoux@tudelft.nl

Martijn Meijers
b.m.meijers@tudelft.nl

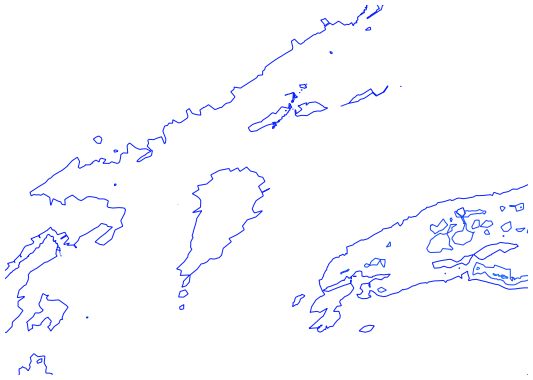
R. Peters, H. Ledoux, and M. Meijers (2014). A Voronoi-based approach to generating depth-contours for hydrographic charts. *Marine Geodesy*, 37(2):145–166. DOI: http://dx.doi.org/10.1080/01490419.2014.902882

The code of the project is available at
https://github.com/Ylannl/Surfonoi

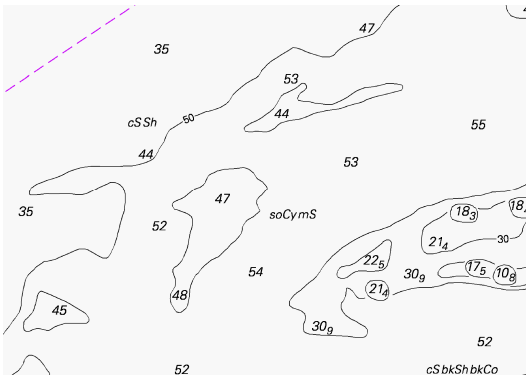
We introduce a new approach for the generation and the generalisation of visually smooth depth-contours for hydrographic charts. Unlike most current approaches, it strictly respects the safety constraint that dictates that the resulting chart may not indicate a depth shallower than originally measured. The main idea is to construct a smooth surface using a Voronoi-based interpolation method. This surface is represented using a triangulation, modified using a series of generalisation operators, and ultimately depth-contours are extracted directly from this surface. We report on experiments made with real-world datasets, and we compare our results with existing approaches.

1 Introduction

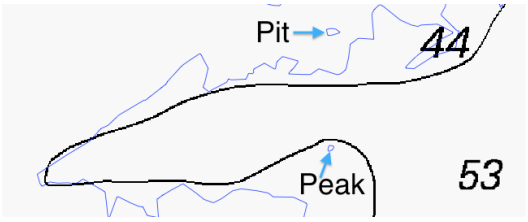
An hydrographic chart is a map of the underwater world specifically intended for the safe navigation of ships. One important element of such a chart are the depth-contours, which have been traditionally drawn by hand by skilled hydrographers. They used a limited set of scattered surveyed depth measurements to deduct and depict the morphology of the seafloor with smooth-looking curves. Nowadays, with technologies such as multibeam echosounders (MBES) offering an almost full coverage of the seafloor, one would expect the contouring process to be fully automatic. It is however in practice still a (semi-)manual process since the new technologies have ironically brought new problems: computers have problems processing the massive amount of data, especially in choosing which data is relevant and which is not. Contours constructed directly from MBES datasets are often not satisfactory for navigational purposes since, as Figure 1a shows, they



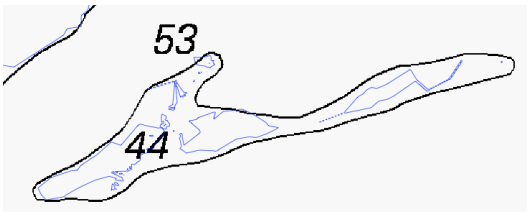
(a) Depth-contours obtained from the raw data



(b) Hydrographic map product of the same area



(c) Pits are removed, while peaks are preserved or integrated with another contour



(d) Groups of nearby contour lines are aggregated

Figure 1: Comparison of raw data and a hydrographic chart from the Royal Australian Navy of the Torres Strait north of Australia. Raw depth contours are blue, generalized depth contours are black.

are zigzagging (the representation of the seafloor contains "waves", i.e.the slope changes abruptly) and they contain many "island" contours (seafloor has several local minima and maxima). These artefacts are the result of measurement noise that is present in MBES datasets, i.e.the variation in depth between two close samples can be larger than in reality, even after the dataset has been statistically cleaned (Calder and Mayer, 2003; Calder and Wells, 2007). Figure 1b illustrates what is

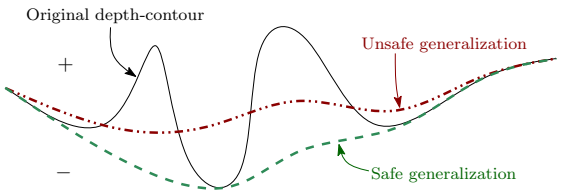


Figure 2: During generalisation, depth-contours can only be moved towards greater depth (indicated by a "–" in the figure).

expected by hydrographers.

Creating good depth-contours requires *generalisation*, i.e.the process of meaningfully reducing information. As Zhang and Guilbert (2011) state, the generalisation of the content of a nautical chart is hindered by the fact that the following four constraints must be respected:

1. The *legibility constraint*. An overdose of information slows down the map reading process for the mariner, thus only the essential information should be depicted on the map in a form that is clearly and efficiently apprehensible.

2. The *safety constraint*. At every location, the indicated depth must not be deeper than the depth that was originally measured at that location; this is to guarantee that a ship never runs aground because of a faulty map. This constraint is a so-called hard constraint, i.e.it can never be broken.

3. The *topology constraint*. The topology of the depicted map elements must be correct, i.e.depth contours may not touch or intersect (also a hard constraint).

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

It should be noted that these four constraints are sometimes incompatible with each other. For instance, the morphology constraint tells us to stay close to the measured shape of the seafloor, while the legibility constraint forces us to deviate from that exact shape by disregarding details. Also, because of the safety constraint, contours can only be modified such that the safety is respected at all times, i.e.contours can only be pushed towards the deeper side during generalisation, as illustrated in Fig 2. It is therefore evident that the end result must be a reasonable compromise between the four constraints, although the hard constraints must not be broken.

The generation of contours, and their generalisation, can be done by several methods. Practitioners usually first interpolate the original MBES samples to create a grid and then directly extract the contours from the grid. If the number of samples is too high to be processed by a computer, they often use a subset, which has the added benefits of creating smoother and simpler depth-contours. We demonstrate in the next section that such workflows can *not* guarantee that the safety constraint is respected, and should therefore not be used. This is one result of this paper. An alternative approach is to construct depth-contours and directly displace the lines. Guilbert and Lin (2007) and Guilbert and Saux (2008) provide the only known methodology to generalise these while respecting

the four constraints. However, these methods require input datasets that are already relatively clean and structured according to a specific format (*b-splines*), and they do not consider how to safely obtain these in the first place.

We propose in this paper, which is an extension of the work of Peters et al. (2013), a unified approach to the generation of safe and smooth depth-contours. It deals with the entire processing chain—from the (statistically cleaned) depth-measurements to generalised contours—and intrinsically respects the four constraints listed above. The main idea is to construct a surface with *all* the original samples, to modify and manipulate this surface to obtain a smoother one, and to extract depth-contours from the smoothed surface only (which ensures that the topology constraint is always respected). This is conceptually similar to the Navigational Surface paradigm (Smith et al., 2002; Smith, 2003) in the sense that a surface is built and maintained from the original samples, and contours are extracted from it when needed. However, instead of using raster coarsening to obtain contours at a given scale (which does not guarantee the safety constraint), we construct a surface using a TIN (triangulated irregular network) and we use a Voronoi interpolant (Sibson, 1981; Gold, 1989; Watson, 1992). This permits us to obtain a smooth surface (i.e.having good slope), which translates to smooth-looking contours; this is key to our approach. We have translated the basic cartographic generalisation operators needed to construct depth-contours (e.g.movement of contours towards greater depth, smoothing of lines, aggregation of local maxima (Figure 1c), enlargement (Figure 1d), etc.) into Voronoi-based operations on the surface. It should also be noticed that, as demonstrated by Gold and Condal (1995), a Voronoi surface deals elegantly with anisotropic sample distributions, which are common in datasets collected with singlebeam echosounders (SBES). We have implemented our approach[1], and we report on experiments made with real-world datasets, including one where MBES and SBES data are mixed. We also compare our results with alternatives.

2 Related work

Figure 3 illustrates the basic processing pipeline commonly used to obtain depth-contours from a set of sample points (MBES and/or SBES). Most methods construct a surface, either a raster or a TIN, by selecting sample points and interpolating, and construct contours from that surface. Some methods do not consider the complete pipeline, and focus only on contours; these could be used in combination with others. We describe below the most common processing operations, taking into account the four hydrographic constraints, and focusing on the most important: the safety constraint. A summary of these methods is shown in Table 1.

2.1 Raster-based methods

The following are methods that use a raster data structure either to select a subset of the input samples or to construct a raster surface.

[1]The implementation is open-source and available at `https://github.com/Ylannl/Surfonoi`

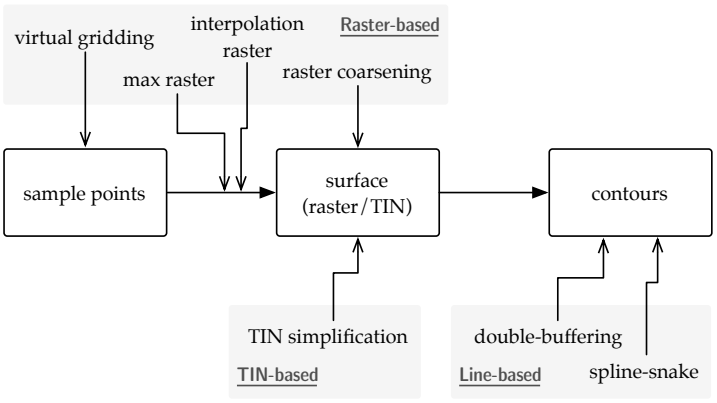
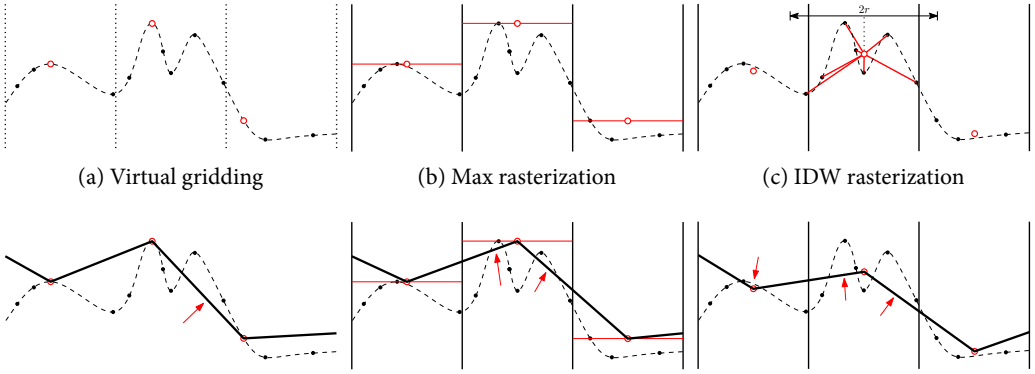


Figure 3: General workflow to construct depth-contours from sample points.

Table 1: Overview of methods to construct depth-contours for nautical charts. All the methods respect the topology constraint.

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

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



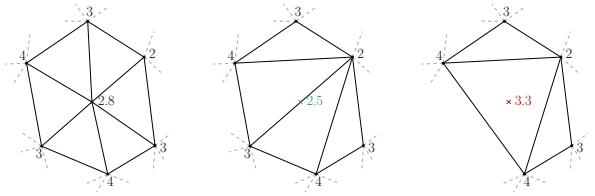
(a) Virtual gridding (b) Max rasterization (c) IDW rasterization

(d) Virtual gridding and TIN-based contour values (e) Max rasterization and contours (f) IDW rasterization and contours

Figure 4: On the top: profile views of different filtering and rasterization methods. On the bottom: the corresponding contours. The arrows indicate where the safety constraint is violated with respect to the original points.

Selection with virtual gridding. This is a point filtering method that aims at reducing the volume of data, in order to create generalised contours and to speed up the computation time, or simply to make the computation possible, in the case the input dataset is several orders of magnitude bigger than the main memory of a computer (Isenburg et al., 2006b). The idea is to overlay a virtual grid on the input points and to keep one point for every grid cell. The selected points can either be used to construct a raster or TIN surface, see below. While different functions can be used to select the point (e.g.deepest, shallowest, average, or median), because of the safety constraint the shallowest point is often chosen by practitioners, see Figure 4a for a one-dimensional equivalent. It should however be stressed that choosing the shallowest point does not guarantee safe contours. The problem is that contour extraction algorithms perform a linear interpolation on the cells of the surface (raster cells or triangles). As can be observed from Figure 4d, this easily results in safety violations at 'secondary' local maxima in a grid cell. The number and severity of these violation is related to the cellsize of the virtual grid: a bigger cellsize will result in more and more severely violated points. Notice that it is not possible to reduce the cellsize such that the safety issue can be guaranteed.

Max rasterization. As Figure 4b shows, it is similar to virtual gridding, the main difference is that a raster (a surface) is created where every cell in the virtual grid becomes a raster cell whose depth is the shallowest of all the samples. This disregards the exact location of the original sample points, and moves the shallowest point in the grid cell to the centre of the pixel. That means that the morphology constraint is not respected. Moreover, as Figure 4e shows, the safety constraint is not guaranteed, for the same reasons as with virtual gridding. Again, the severity of these problems depends on the chosen cellsize.



(a) Initial configuration (b) 1st vertex removal (c) 2nd vertex removal

Figure 5: Due to the re-triangulation after a removal, violations of the safety constraint may occur after a series of points are removed. The first vertex is removed (locally the resulting surface will be shallower). However, the second removal changes the configuration of triangles and at that location the surface is now deeper. A lower number means a shallower point.

Interpolation to a raster. For hydrographic contouring, the raster surface is often constructed with spatial interpolation, particularly with the method of *inverse distance weighting* (IDW), as first proposed by Shepard (1968). Figures 4c and 4f illustrate the process of IDW interpolation, notice that as a result of the averaging that takes place extrema are disregarded and subsequently the safety constraint is also violated.

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

2.2 TIN simplification

Given a set of points $(x, y, depth)$, a TIN is a tessellation of the (x, y) plane into triangles where the vertices of the triangles are the points in (x, y). The triangulated surface can be embedded in three-dimensional space, but since every point has only one depth value, it has the property of being projectable onto the plane (x, y). The objective of TIN simplification is to reduce the number of vertices in a TIN until the surface represented with the triangles deviates by more than a given tolerance to the original TIN. While we are not aware of any method tailored to contours in hydrographic charts, it could be an attractive method since, unlike the raster-based methods, TIN-based methods take into account the geometric configuration of neighbouring points and do not move sample points (Garland and Heckbert (1995) give an overview of the different methods used). However, as Figure 5 shows, the safety constraint is also not guaranteed to be respected when vertices are removed from a TIN. This is due to the fact that the triangulation must be updated (Mostafavi et al., 2003) and the shape of the triangles are usually not controlled by the depth of the vertices, but rather by the Delaunay criterion.

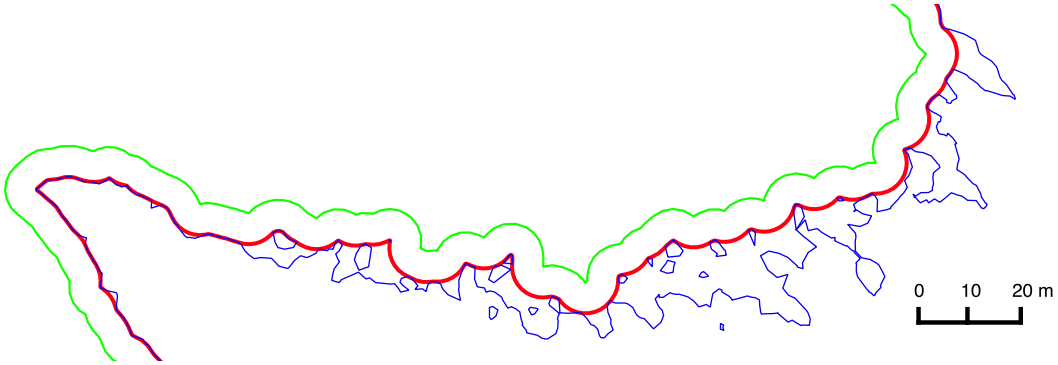


Figure 6: Double buffering. The original blue contour line (bottom) is first buffered to the upper green line, which is subsequently buffered back to the middle red line.

2.3 Line-based methods

Two issues arise with line-based methods. First, moving lines can create intersections and thus break the topology constraint. Methods to ensure that no intersections are created after displacement, or to resolve them, are complex and in most cases do not respect the safety constraint, see for instance Visvalingam and Whyatt (1993), van der Poorten and Jones (2002) and Dyken et al. (2009). Hennau and De Wulf (2006), based on the work of Christensen (2001), propose a method tailored to depth-contours that combines a line-based smoothing technique with a TIN-based patch smoothing technique. Unfortunately, they do not consider the safety constraint. Second, line-based methods require safe and clean contours as input; as described above, obtaining those safe contours is not a trivial task in the first place. The following methods do respect the safety constraint with respect to the input contours.

Double-buffering. It is a popular method employed in major commercial hydrographic packages such as Caris[2]. As illustrated in Figure 6, it works by buffering a set of input contour lines back and forth (by the same distance), effectively taking into account the safety constraint as well as performing a form of aggregation. When a sphere is taken instead of a disk, the method can also be used on a 3D surface (Smith, 2003). The resulting depth-contours are however not smooth (the first derivative is not continuous) because contours are formed by the intersections of circles. The morphology constraint is thus not respected. Another major weakness of this approach is its non-adaptiveness: the buffer distance is strongly dependent on local details in the contours. Also, notice that the safety constraint can only be guaranteed in case of safe input contours. If the input contours are not safe, for example if they are extracted from an IDW interpolated grid, the double buffering operation does not make them safe (nor would it cause extra violations of the safety).

Spline-snake model. A spline is a piecewise polynomial function that is by definition smooth. Guilbert and Lin (2007) and Guilbert and Saux (2008) use splines in combination with a snake

[2] http://www.caris.com

model to perform smoothing of bathymetric contour lines, this is implemented as an iterative optimisation process. Their method respects the safety constraint and achieves smooth contours at the same time. At each iteration during the process, intersections in the contours are checked. If the distance between two line segments is below a given threshold, the conflicting segment is removed. Also the contours are generalised in order of ascending depth, so that deformations resulting from conflicts are propagated towards the deepest contours. The method is designed to be fully automatic, however, in practice, it seems that manual intervention is still required. Indeed, in one of the case studies described a contour line needed to be split because it got stuck between two other contours. Also, it is unclear how well the method performs on raw contours because the input contours in the presented case studies are b-splines already and it is unclear how these can be safely generated in the first place. The authors also note that the computational cost of the algorithm is high for complex lines, where convergence is slow because of the safety constraint.

2.4 Other methods

The Navigational Surface, as introduced by Smith et al. (2002) and Smith (2003), aims at facilitating and supporting the hydrographer in his work, it is not a way to obtain automatically depth-contours. The framework focusses on uncertainty grids and on combining different overlapping surveys, and proposes to keep one high-resolution grid for a given area. Depth-contours are derived with raster coarsening and double-buffer, and thus the results are not guaranteed to be safe.

3 A Voronoi-based surface approach

Part of the problems with existing approaches to generate depth-contours is the fact that the different processes, such as spatial interpolation, generalisation and contouring, are treated as independent processes. We argue in this paper that they are in fact interrelated, and we present in this section an approach where the different processes are integrated into one consistent framework: a Voronoi-based surface.

The key idea behind a surface-based approach is to have one single consistent representation of the seafloor from which contours can be generated on-the-fly (potentially for different map scales, or with varying degrees of generalisation). Instead of performing generalisation by moving lines or using a subset of the original samples, we manipulate the surface directly with operators. Imhof (1965) gives a compelling argument to do so:

> One must never overlook the fact that (geographic) *surfaces* are being depicted with contours. A single line says very little. One line does not define a surface. Everything comes back, eventually, to the formation of the system of lines, that is, the surface.

Figure 7 gives a schematic overview of the different components of our Voronoi-based surface concept. Firstly, all the statistically cleaned input sample points of a given area are used to construct a TIN that respects the Delaunay criterion. Secondly, a number of generalisation operators are used that alter the TIN using Laplace interpolation, which is based on the Voronoi diagram. These

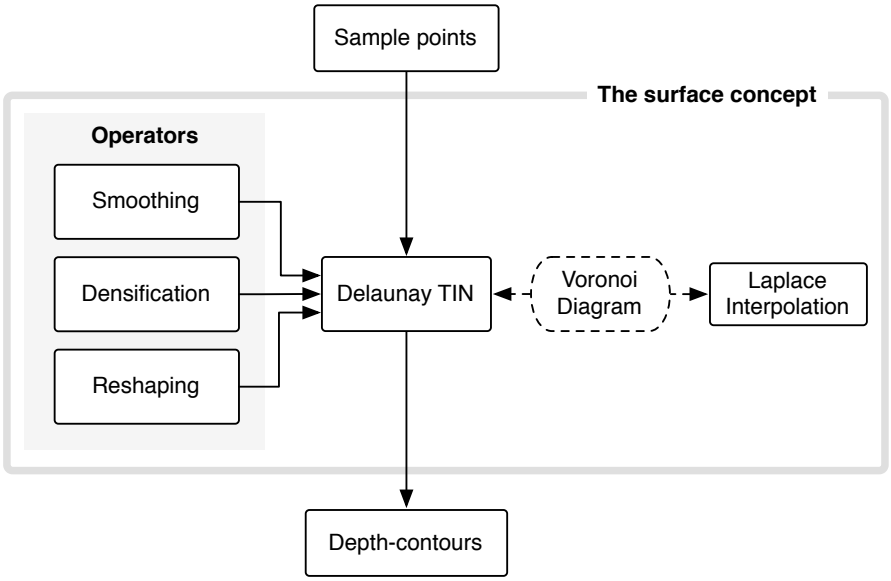


Figure 7: Overview Voronoi- and surface-based approach

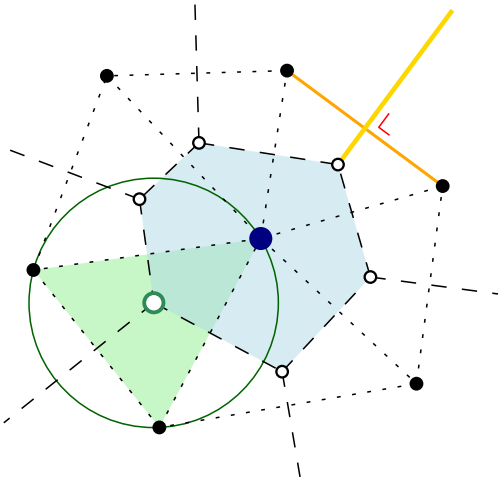
operators aim at improving the slope of the surface, and permit us to generalise the surface. Finally, contour lines are derived from the altered TIN using linear interpolation (Watson, 1992).

3.1 Good slope translates to smooth contours

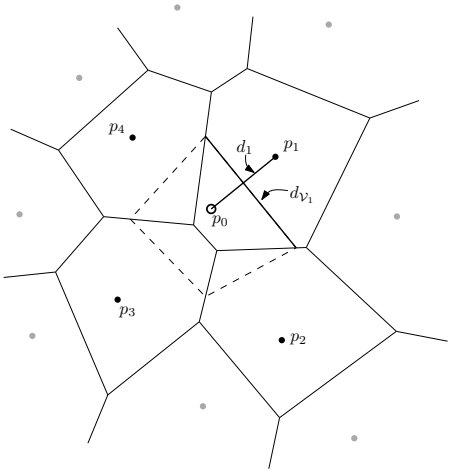
The representation of a continuous phenomenon in space such as the depth of a given area can be done with a *field* (Goodchild, 1992). A field is a model of the spatial variation of a given attribute a, and can be described by function; in our case the function would be $f(x, y) = depth$. A contour line is the set of points in space where $f(x, y) = depth_0$, where $depth_0$ is a constant.

The mathematical concept 'Implicit Function Theorem' states that a contour line extracted from a field f will be no less smooth than f itself (Sibson, 1997). In other words, obtaining smooth contour lines can be achieved by smoothing the field itself. Sibson (1997) goes further in stating that: "The eye is very good at detecting gaps and corners, but very bad at detecting discontinuities in derivatives higher than the first. For contour lines to be accepted by the eye as a description of a function however smooth, they need to have continuously turning tangents, but higher order continuity of the supposed contours is not needed for them to be visually convincing." In brief, in practice we should use functions whose first derivative is continuous (called a C^1 interpolant); C^0 interpolants (function is continuous but its derivatives are not continuous) are not enough, and C^2 ones (first and second derivatives are continuous) are not necessary.

Representing a field in a computer is problematic since computers are discrete machines. We therefore need to *discretise* the field, i.e.partition it into several pieces that cover the whole area (usually either grid cells or triangles). Contours in Figure 1a are not smooth basically because the seabed is represented simply with a TIN of the original samples, which is a C^0 interpolant.



(a) The VD, drawn in dashed lines, uniquely maps to the DT, drawn in dotted lines. A DT triangle maps to a VD vertex that is located at the centre of the circumscribed circle of the triangle. A DT edge corresponds to a VD edge, and is perpendicular to it.



(b) Laplace interpolation. Black points indicate natural neighbours of the point p_0. The dashed region marks the Voronoi cell that would be added with the insertion of p_0 to the Voronoi Diagram

Figure 8

However, as we demonstrate in the next section, we can obtain a smooth looking approximation of the field by densifying the TIN using the Laplace interpolant (which is C^1).

3.2 Delaunay triangulation and Voronoi Diagram

The TIN surface is a Delaunay triangulation (DT) because it permits us to extract on-the-fly the Voronoi diagram (VD) needed for the smooth interpolant. As Figure 8a shows, the DT maximises the minimal angle of its triangles, through the application of the Delaunay criterion that states that the circumscribed circle of any triangle must not contain any other vertex. As a result, triangles are as 'fat', or equilateral, as possible. As with any other TIN the edges in a DT implicitly define neighbour relations between vertices. However, only for the Delaunay triangulation it is guaranteed for every vertex that its neighbouring vertices are closer than any other vertex in the triangulation.

The Voronoi diagram (VD) is a dual graph of the DT, see Figure 8a. It is a subdivision of the plane into 'proximity' regions: every point p of the set S is mapped to a Voronoi cell $\mathcal{V}_p$ defined as the set of points $x \in \mathbb{R}^2$ that are closer to p than to any other point $q \in S$.

It is important to note that the DT and the VD are completely adaptive to the spatial distribution of the points. The neighbouring relations between points are also meaningful because they indicate which points are closest in any given direction. As Gold and Condal (1995) show, this property allows us to handle SBES datasets better than raster-based approach; our experiments corroborate this (see below).

3.3 Laplace interpolant

The Laplace interpolant, or non-Sibsonian interpolation (Belikov et al., 1997; Hiyoshi and Sugihara, 1999b), is a spatial interpolation method that exploits the spatial relationships between vertices in a VD. It is a computationally faster variant of the Sibson, or natural neighbour, interpolation method (Sibson, 1981; Gold, 1989).

Laplace interpolation is completely determined by the configuration of the natural neighbours, i.e.the generators of the adjacent Voronoi cells, of a vertex inserted into the VD at the interpolation location x. The interpolated depth at x, denoted as $\hat{h}_0$, is the average of its natural neighbours' weights:

$$\hat{h}_0 = \begin{cases} h_i & \text{if } p_0 = p_i, \\ \dfrac{1}{\sum_{i=1}^{n} w_i} \cdot \sum_{i=1}^{n} w_i h_i & \text{if } p_0 \neq p_i \end{cases}$$

The weights are defined by:

$$w_i = \frac{d_{\mathcal{V}_i}}{d_i} \tag{1}$$

where h_i is the depth of vertex v_i, d_i is the Euclidean distance in $\mathbb{R}^2$ to the natural neighbour p_i (the distance from p_0 to p_i), and $d_{\mathcal{V}_i}$ is the length of the Voronoi edge incident to $\mathcal{V}_0$ and $\mathcal{V}_i$. Figure 8b illustrates this. Note that the fraction becomes indeterminate when p_0 equals one of the sample points p_i. In this case the Laplace interpolant therefore simply defines that $\hat{h}_0 = h_i$.

Watson (1992) lists the properties of the ideal spatial interpolation method in the context of real-world geographical datasets. A brief review of these properties in the light of Laplace interpolation demonstrates some of the fundamental features of our approach. Firstly the Laplace interpolant is exact: the interpolation method returns the exact value, rather than some estimate, of a sample point when it is queried at that precise location. Note that an inexact interpolation method might violate the hydrographic safety constraint at the locations of sample points, if the interpolated depth is deeper than the original depth. Secondly, it is continuous and continuously differentiable (C^1) everywhere except at sites where finitely many Voronoi circles (as defined in Figure 8a) intersect (Hiyoshi and Sugihara, 1999a). We found that this is not a problem in practice. Thirdly, it is local, i.e.it uses only a local subset of data for the interpolation of a point. This limits the computational cost and supports efficient addition or removal of new data points. Finally, like the VD itself, it is adaptive to the spatial configuration of sample points. Unlike other methods such as IDW interpolation, the Laplace interpolant requires no user-defined parameters.

3.4 Operators on the surface

We introduce in this section three generalisation operators that permit us to obtain a smoother surface from which depth-contours can be extracted: (i) smoothing, (ii) densification and (iii) reshaping. These three operators are based on the Laplace interpolation algorithm (described in

Algorithm 1, and referred to as InterpolateDepth in the following) to interact with the surface. As shown in Algorithm 2 InterpolateCheckDepth, is used to ensure that the safety constraint is never violated (the newly interpolated depth is only returned if it is shallower than the current depth at the vertex). If the interpolated depth is deeper than the current depth, the current depth is returned.

Algorithm 1 InterpolateDepth

Input: a vertex v in a TIN $\mathcal{T}$
Output: the Laplace interpolated depth h for v

1: **for** each v_i adjacent to v **do**
2: $e_1 \leftarrow edge(v, v_i)$
3: $e_2 \leftarrow dual(e_1)$
4: $w_i \leftarrow \frac{length(e_2)}{length(e_1)}$
5: **end for**
6: $h \leftarrow 0$
7: **for all** w_i, h_i from the natural neighbours v_i around v **do**
8: $h \leftarrow h + \frac{w_i}{\sum w_i} * h_i$
9: **end for**

Algorithm 2 InterpolateCheckDepth

Input: a vertex v in a TIN $\mathcal{T}$
Output: depth of v is safely updated

1: $h \leftarrow$ InterpolateDepth (v)
2: **if** h shallower than current depth at v **then**
3: depth of $v \leftarrow h$
4: **else**
5: depth of v stays the same
6: **end if**

3.4.1 The smoothing operator

The operator Smooth (Algorithm 3) is the most trivial application of the InterpolateCheckDepth algorithm: it simply calls it for all input vertices, after which their depths are updated (see Figure 9). Thus, smoothing does not change the planimetric coordinates of vertices, but only lifts the vertices' depths. It can be performed either on a portion of a dataset, or on the whole dataset. Furthermore this operator can be applied any number of times, delivering more generalisation with each pass.

The primary objective of smoothing is to generalise the surface by removing high frequency detail while preserving the overall feature shape. Applying Smooth both reduces the angle between the planes spanned by adjacent triangles and simplifies overall shape. Smooth performs two linear loops over the n vertices of the TIN (the depths are only updated after all the depths have been

Algorithm 3 SMOOTH

Input: a TIN $\mathcal{T}$
Output: a smoothed $\mathcal{T}$

for all vertices $v_i \in \mathcal{T}$ **do**
  $h_i \leftarrow$ INTERPOLATECHECKDEPTH (v_i)
end for

for all tuples v_i, h_i **do**
  update depth of v_i with h_i
end for

Algorithm 4 DENSIFY

Input: a TIN $\mathcal{T}$
Output: a densified $\mathcal{T}$

for all triangles $t \in \mathcal{T}$ **do**
  if $area(t) > maxArea$ **then**
    insert vertex v in $\mathcal{T}$ at $circumcenter(t)$, and update $\mathcal{T}$ for the Delaunay criterion
    set depth $v \leftarrow$ INTERPOLATEDEPTH
  end if
end for

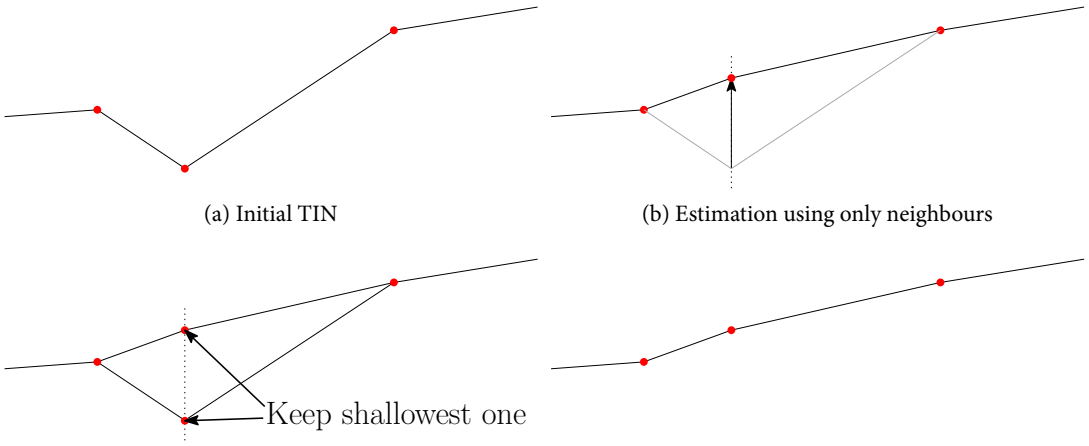
Algorithm 5 RESHAPE

Input: a TIN $\mathcal{T}$; a set V of vertices to be reshaped
Output: depth of vertices in V updated

vertexCache $\leftarrow$ [ ]
for all vertices $v \in V$ **do**
  append v to *vertexCache*
  remove v from $\mathcal{T}$
end for

for all vertices v in *vertexCache* **do**
  insert v in $\mathcal{T}$
  update depth of v with INTERPOLATECHECKDEPTH (v_i)
  remove v from $\mathcal{T}$
end for

for all vertices v in *vertexCache* **do**
  insert v in $\mathcal{T}$
end for



(a) Initial TIN (b) Estimation using only neighbours

(c) Comparison of depths (d) Resulting TIN

Figure 9: Cross-section view of the smoothing of a single vertex in a TIN.

estimated), and since the smoothing of one vertex is performed in expected constant time, the expected time complexity of the algorithm is $O(n)$.

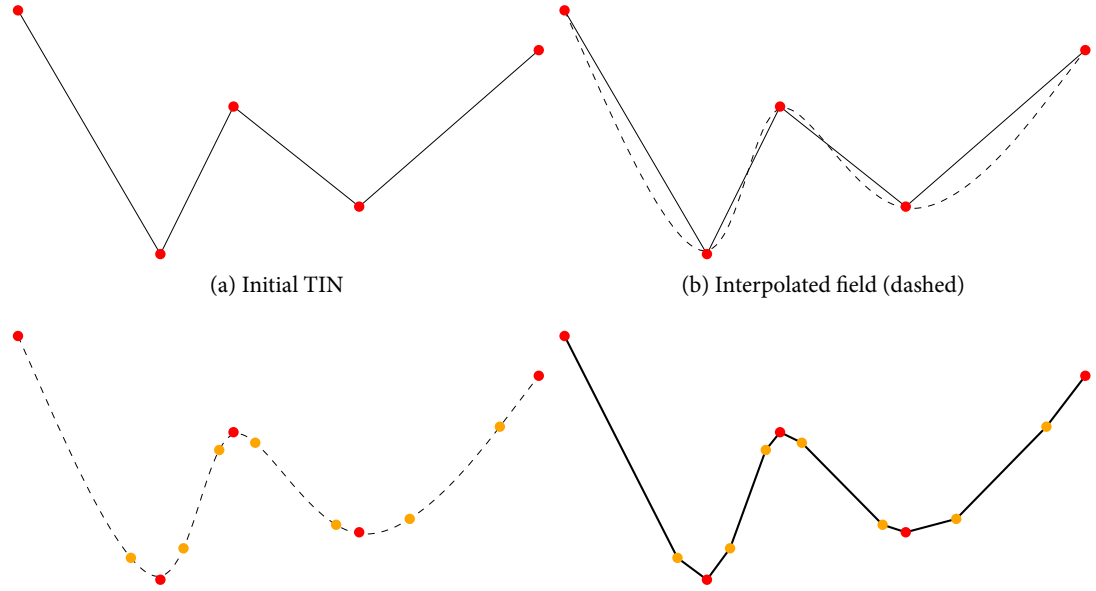
3.4.2 The densification operator

Its objective is primarily to minimise the discretisation error between the Laplace interpolated field and the contours that are extracted from the DT, this is illustrated in Figure 10. By inserting extra vertices in large triangles (to break them into three triangles), the resolution of the DT is improved. As a result also the extracted contour lines have smoother appearance because they now have shorter line-segments. We insert a new vertex at the centre of the circumscribed circle of any triangle that has an area greater than a preset threshold (Algorithm 4); its depth is assigned with InterpolateDepth (Algorithm 1). The circumcenter is chosen here because that location is equidistant to its three closest points, and subsequently results in a very natural point distribution.

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

3.4.3 The reshaping operator

Reshape (Algorithm 5) should be considered as a mechanism to perform generalisation that requires specific alteration of the surface that can not be achieved through Smooth or Densify alone. An example is the aggregation of two distinct peak features, as illustrated in Figure 11. The reshape



(a) Initial TIN

(b) Interpolated field (dashed)

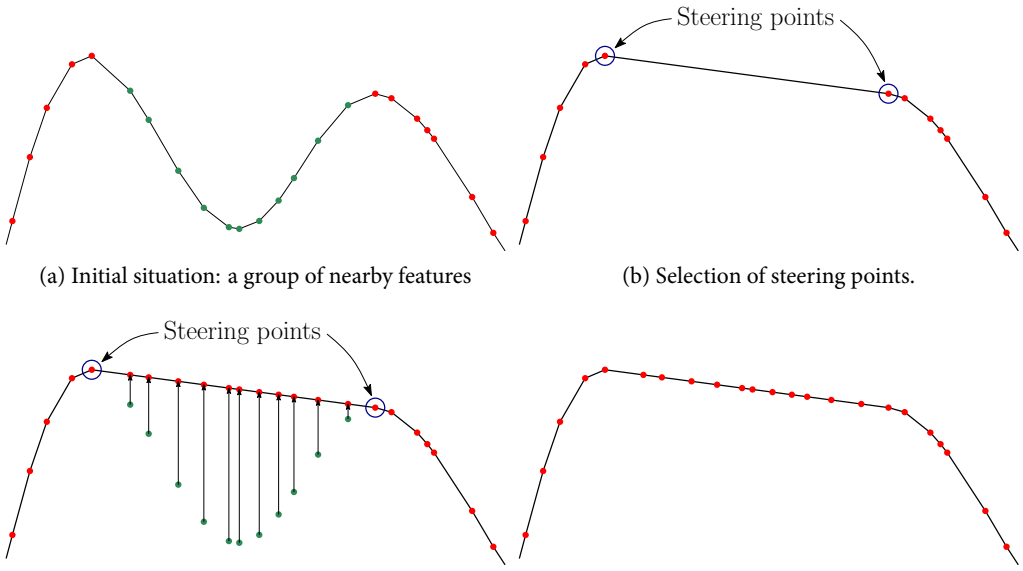
(c) Addition of intermediate points (orange) in accordance with the interpolated field

(d) Resulting TIN

Figure 10: Cross-section view of the densification operator in a TIN.

operator is also based on InterpolateCheckDepth, but unlike Smooth, prior to updating the depth of a vertex the set of input vertices V are temporarily removed from the TIN. Prior to calling Reshape, the relevant feature(s) need to be identified (which is currently done manually) and a set of so-called *steering points* needs to be assigned. Figure 11a depicts these steering points, they are local maxima of the features to be aggregated. The set of input vertices V represents the vertices that are to be reshaped; in Figure 11a these are enclosed by the steering points. These vertices are then temporarily removed from the TIN (see Figure 11b) and then each input vertex $v \in V$ is individually re-added to the triangulation, assigned a depth of InterpolateCheckDepth (v) and removed again (see Figure 11c). After doing this for every input vertex $v \in V$, they are permanently inserted back into the TIN, but now with new depths that can be significantly different, as depicted in Figure 11d.

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



(a) Initial situation: a group of nearby features

(b) Selection of steering points.

(c) Depth-estimation of points enclosed by the steering points.

(d) Reshaped feature.

Figure 11: Cross-section view of the aggregation of 2 local peaks on the surface using the reshaping operator.

Table 2: Details of datasets

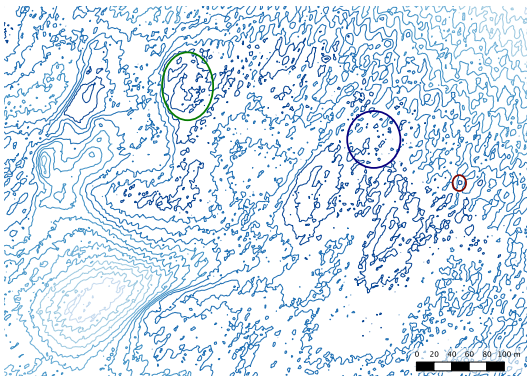
	Antilles	*Australia*	*London*	*Zeeland*
Point count	1081	1613	151,704	102,954
Area	2960.74 km^2	0.27 km^2	1.12 km^2	0.32 km^2
Type	SBES	MBES	MBES+SBES	MBES

4 Experiments

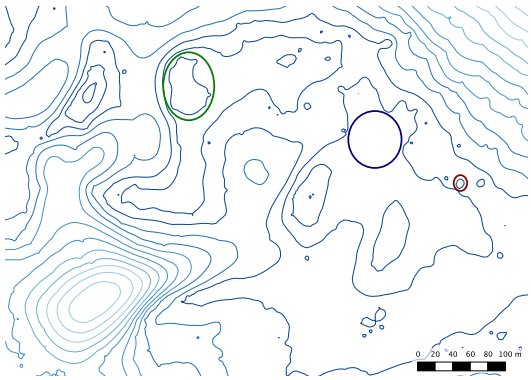
We have implemented the algorithms described in the previous section with the C++ programming language using the CGAL library[3] for its implementation of the DT. A number of experiments were performed to investigate the effectiveness of the proposed operators and to compare results with existing methods. Note that we rasterized some depth fields exclusively for evaluation purposes.

Table 2 gives an overview of the used datasets and their characteristics. Both SBES and MBES datasets were used, as well as a dataset with a mixture of those.

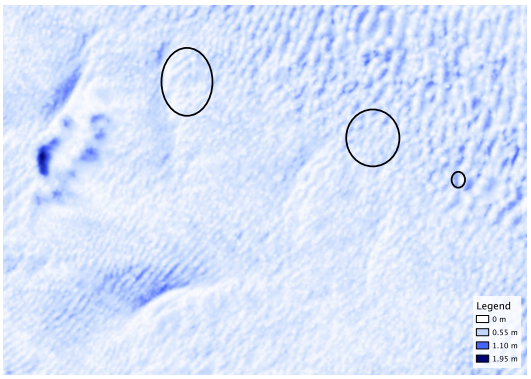
[3] `http://www.cgal.org`



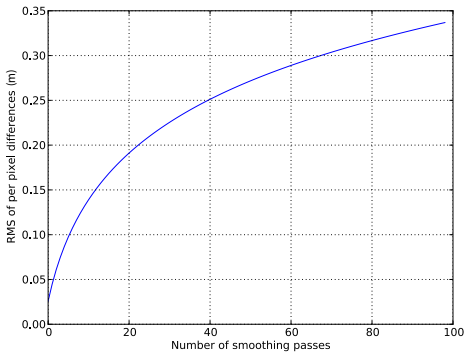
(a) Raw contours extracted at a 50cm depth interval.



(b) Smoothed contours (100 smoothing passes). The ellipses mark areas where aggregation (left), omission (middle) and enlargement (right) take place.



(c) Difference map between the initial and 100x smoothed interpolated and rasterized fields (pixelsize 50 cm).



(d) RMS of differences with respect to the initial interpolated field as a function of the number of smoothing passes

Figure 12: The effect of the smoothing operator in the Zeeland dataset.

4.1 The smoothing operator

As can be observed from Figure 12a, the raw and ungeneralised contours in the Zeeland dataset have a very irregular and cluttered appearance. However, the smoothed contours (100 smoothing passes) from Figure 12b have a much cleaner and less cluttered appearance. Clearly, the number of contour lines has diminished. This is both because pits (local minima) have been lifted upwards by the smoothing operator, and nearby peaks (local maxima) have been aggregated (because the region in-between has been lifted upwards). Thus omission and aggregation take place in the contour lines. Notice also that a third effect of the smoothing operator is the enlargement of certain features as a result of the uplifting of the points surrounding a local maximum.

As to be expected, the overall effect of the smoothing operator on the morphology of the surface is significant. As shown by the plot of the root mean square (RMS) error between the initial and the generalised field in Figure 12d, the smoothing of the surface is most significant in the earlier smoothing passes and results in a per pixel difference of tens of centimetres. That supports the idea that the smoothing of the surface works against the preservation of *all* morphological features. However, from the difference map in Figure 12c, it can be seen that the high frequency features are especially altered, thus preserving the general surface shape (low frequency pattern).

4.2 The densification operator

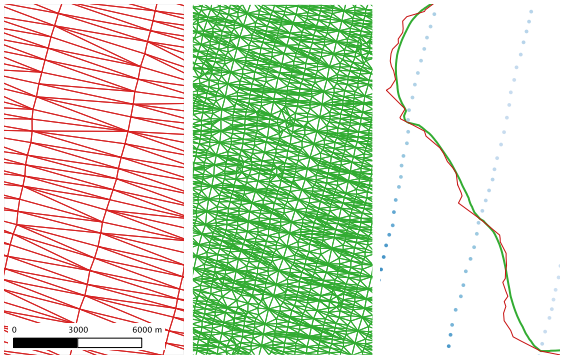
The effects of the densification operator are illustrated in Figure 13a. The sharp edges of the undensified lines are caused by the large triangles in the initial TIN, however after densification these large triangles are subdivided into much smaller ones. The result is a much smoother contour line that still respects the sample points. Figure 13b shows how the approximation error of the TIN with respect to a Laplace interpolation of the input sample points improves with more densification passes.

4.3 The reshaping operator

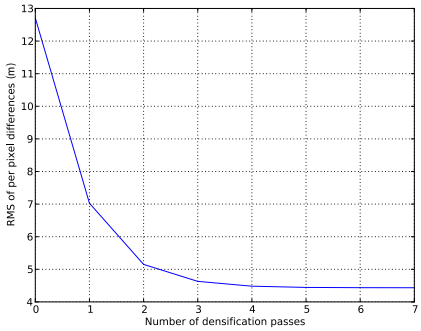
Figure 14 illustrates the use of the reshaping operator in the aggregation of two peaks in the surface. Evidently, the region in between the peaks is significantly raised such that it has the same depth as the original peaks. For this experiment the steering points and input points for Algorithm 5 were manually selected; we are currently working on an approach to automate this step.

4.4 Heterogeneous data

The London dataset, shown in Figure 15, is composed of a mixture of SBES and MBES sample points. The abrupt transition in point density and homogeneity is problematic for interpolation methods in general (Gold and Condal, 1995). It demonstrates the adaptivity of the Voronoi-based interpolation method. Since Laplace interpolation (see Figure 15b) is fully adaptive to the density of input samples there is no distance parameter that needs to be set, and it can easily be applied

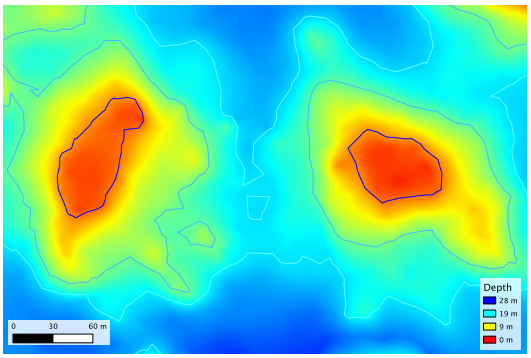


(a) TIN before densification (left), TIN after 3x densification (centre) and comparison of corresponding contour lines (right). Original sample points shown in the background. (detail)

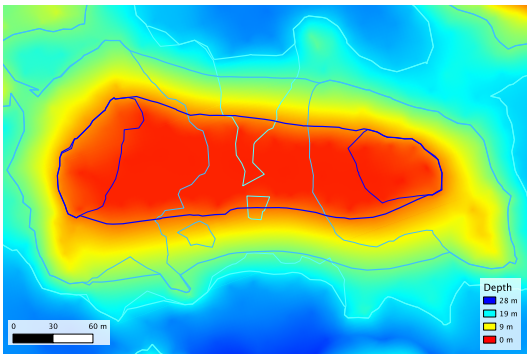


(b) Approximation error of TIN interpolation with respect to the initial Laplace interpolated field as a function of the number of densification passes. Computed for a rasterization of the fields with 10 m pixelsize.

Figure 13: Densification illustrated on the Antilles dataset.



(a) The initial Laplace interpolated field with corresponding contours



(b) The field after reshaping. Both the initial (thin line) as the reshaped contours (fat line) are shown.

Figure 14: The reshaping operator performs aggregation in the Australia dataset.

to datasets with a highly anisotropic distribution of samples. However, we can observe contours that are unrealistically bent toward the inner part of the dataset due to the anisotropic sampling pattern. If IDW is used with a 150 m search radius (so that the whole region can be interpolated), it results in cluttered contours in the bottom of the area and rather disturbing artefacts in the center area (see Figure 15c). Therefore we conclude that, while still not perfect, the Laplace interpolation performs significantly better in this case.

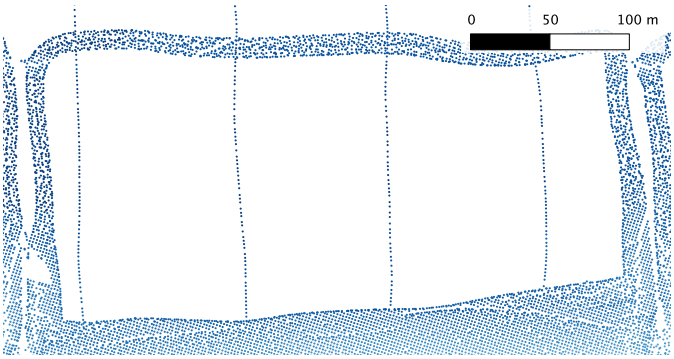
4.5 Comparison with existing methods

Figure 16a shows the distribution of the error between the sample points and the interpolated surfaces constructed with different approaches. The parameters were chosen in such a way that the resulting contours look as similar as possible, thus providing the same overall degree of generalisation. As expected, since it is an average, the raster-based IDW interpolation violates the safety constraint for roughly half the number of points. In contrast, our approach violates no points at all. After applying the smoothing operator the safety constraint is still respected. With the max rasterization method, the safety constraint is also respected in its field representation (note that it does not mean the resulting contours are safe, see Figure 4e). However when compared to the smoothed Laplace interpolated surface it is clear that the variance of the distribution is larger, thus raster coarsening is less accurate than the smoothed conceptual surface at a comparable level of generalisation.

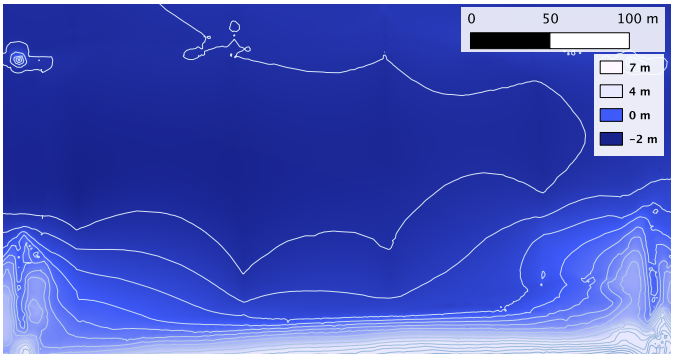
From a visual comparison of the different contours (Figure 16b), it can be seen that the contours of both double-buffering and raster coarsening have sharp corners. In case of double-buffering this is due to the convex features (pointing south and towards the deeper region). Those features are not smoothed by the process of double-buffering unless a very large buffer distance is chosen. In case of concave features in the input line, the doubly buffered line clearly has circular arcs. It is evident that the contours from the coarsened raster are indeed quite coarse and therefore not smooth at all. As expected, since with our approach the conceptual surface is by definition smooth and safe, the resulting depth-contours are smooth and are all moved/generalised in the direction of deeper areas.

5 Conclusion and discussion

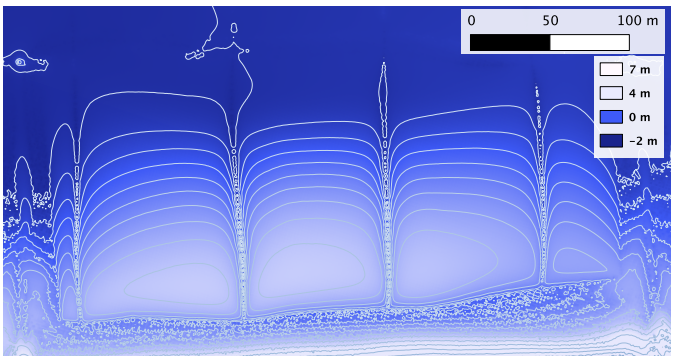
We have shown in this paper that the raster-based approaches, which are commonly used in practice for generating depth-contours, do not necessarily respect the safety constraint, i.e.the depth displayed in a nautical chart could be deeper than in reality. In contrast, we have developed an approach—based on the Voronoi diagram and the Laplace interpolant—that can generate both smooth-looking *and* safe depth-contours. It respects the two hard constraints (safety and topology) and performs well for the two soft constraints (legibility and morphology). Conceptually, this is achieved by modifying with operators the surface that is fitted through the original sample points only in one direction: up. The Voronoi-based lifting improves the smoothness of the surface (its slope) and hence the resulting contour lines also appear smoother, less “island” contours



(a) Sample point distribution

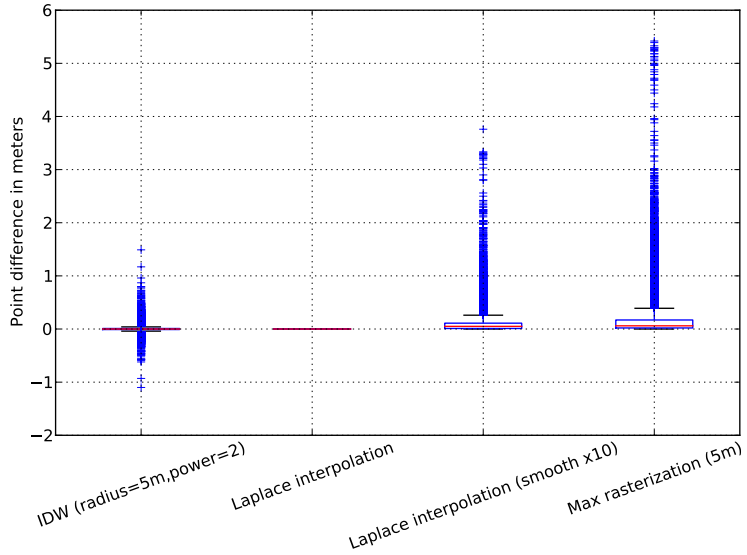


(b) Laplace interpolant, contours densified 5 times

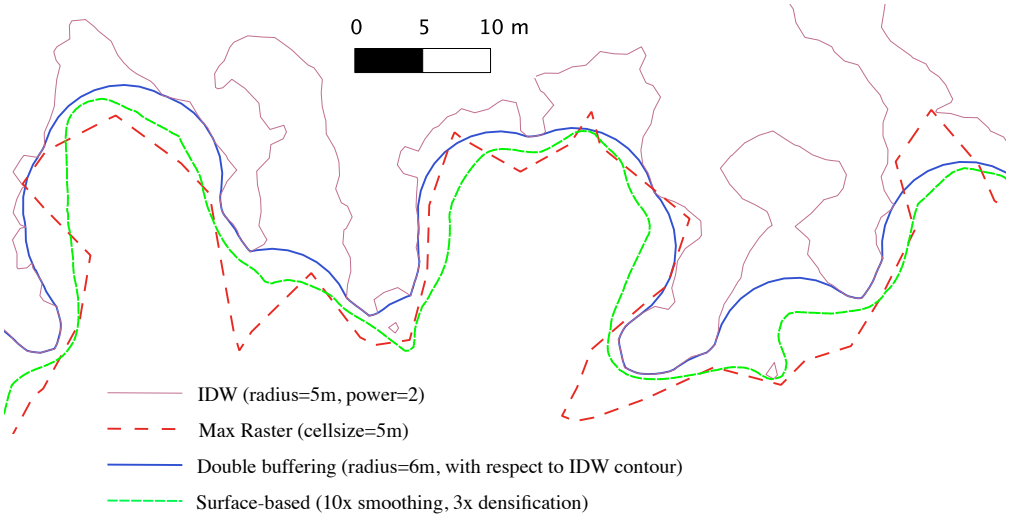


(c) Inverse Distance Weighting (radius=150m, power=2)

Figure 15: Contours for the London dataset. Contours are shown at every 50 cm.



(a) Boxplots of the sample point differences with respect to the interpolated fields



(b) Comparison between hydrographic contouring approaches

Figure 16: Comparison of four methods to generate depth-contours applied to the MBES part of the London dataset.

are present, and pits are removed. Also, more specialised generalisation operations can be realised with our approach: we have shown how a reshaping operation, which aggregates peaks, can help improve the legibility of nautical charts.

Although we have primarily focussed on generalisation operators that modify the surface (which has a direct effect on the resulting contours), we see this work as an initial step in "linking" both representations. Indeed, we envision a stronger and bidirectional link between the two, and that could lead to a powerful interactive system where an operator would have a linked view of both representations and could therefore choose the most appropriate representation for applying a modification (either change the surface or the lines), which is then directly translated into both representations. Such an approach would simplify the implementation of several generalisation operations, while respecting the four constraints for nautical charts.

While our approach deals with the entire processing chain (from the samples to generalised contours), it does not replace nor invalidate previous results. Indeed, we believe that the safe and smooth depth-contours we generate could be used as the input for the work of Guilbert and Lin (2007) and Guilbert and Zhang (2012), as the input needs to be already cleaned. Furthermore, extension of this work could be used to analyse the surface and detect where our operators could be used: Guilbert (2012) introduces a method to extract (bathymetric) terrain features and to store them, and their spatial relationships, in a hierarchical structure.

Finally, our future work includes:

1. The automation of the reshaping operator so that it detects peaks in the surface and applies the operator to these detected locations.
2. The RMS error graph allows us to *quantify* the generalisation error we make with smoothing and densifying. This can possibly be used to locally control the error that is introduced during generalisation (currently there is no guarantee on the horizontal displacement that is introduced in the contours while generalising). An open question is: how we can assess the quality of the resulting contours for the softer constraints (legibility and morphology)?
3. Data management and scalability: How can we scale the method to deal with a huge number of input points? At this moment, we are limited by the memory of the computer used, which is often not sufficient for real-world MBES datasets. We plan to investigate the streaming paradigm of Isenburg et al. (2006a) and see if it can be adapted to our operators, since it would allow us to process much larger datasets.

Acknowledgements

The research was financially supported by: (1) the Dutch Technology Foundation STW, which is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs (project code: 12217); (2) Rijkswaterstaat, the Ministry of Infrastructure and the Environment of the Netherlands. We also would like to thank George Spoelstra for providing the datasets.

References

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

Calder B and Wells D (2007). CUBE user's manual. Technical report, Center for Coastal and Ocean Mapping and NOAA/UNH Joint Hydrographic Center, University of New Hampshire. Version 1.13.

Calder BR and Mayer LA (2003). Automatic processing of high-rate, high-density multibeam echosounder data. *Geochemistry, Geophysics, Geosystems*, 4(6):1048.

Christensen AHJ (2001). Contour smoothing by an eclectic procedure. *Photogrammetric engineering and remote sensing*, 67(4):511–517.

Dyken C, Dhlen M, and Sevaldrud T (2009). Simultaneous curve simplification. *Journal of Geographical Systems*, 11(3):273–289.

Garland M and Heckbert PS (1995). Fast polygonal approximation of terrain and height fields. Technical Report CMU-CS-95-181, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA.

Gold CM (1989). Surface interpolation, spatial adjacency and GIS. In Raper J, editor, *Three Dimensional Applications in Geographic Information Systems*, pages 21–35. Taylor & Francis.

Gold CM and Condal AR (1995). A spatial data structure integrating GIS and simulation in a marine environment. *Marine Geodesy*, 18:213–228.

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

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

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

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. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume I-2. Melbourne, Australia.

Hennau M and De Wulf A (2006). Smoothing contour lines of hydrographical maps: An optimised approach. In *Proceedings 15th International Congress of the International Federation of Hydrographic Societies*, pages 199–201. Antwerp, Belgium.

Hiyoshi H and Sugihara K (1999a). Generalization of an interpolant using voronoi diagrams in two directions. In *Shape Modeling and Applications, 1999. Proceedings. Shape Modeling International'99. International Conference on*, pages 154–161. IEEE.

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

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

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

Isenburg M, Liu Y, Shewchuk JR, Snoeyink J, and Thirion T (2006b). Generating raster DEM from mass points via TIN streaming. In *Geographic Information Science—GIScience 2006*, volume 4197 of *Lecture Notes in Computer Science*, pages 186–198. Münster, Germany.

Mostafavi MA, Gold CM, and Dakowicz M (2003). Delete and insert operations in Voronoi/Delaunay methods and applications. *Computers & Geosciences*, 29(4):523–530.

Peters R, Ledoux H, and Meijers M (2013). Generation and generalization of safe depth-contours for hydrographic charts using a surface-based approach. In *Proceedings of the 16th Workshop of the ICA Commission on Generalisation and Multiple Representation*. Dresden, Germany.

Shepard D (1968). A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings 23rd ACM National Conference*, pages 517—524.

Sibson R (1981). A brief description of natural neighbour interpolation. In Barnett V, editor, *Interpreting Multivariate Data*, pages 21–36. Wiley, New York, USA.

Sibson R (1997). Contour mapping: Principles and practice. Report available at `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.63`.

Smith SM (2003). *The Navigational Surface: A Multipurpose Bathymetric Database*. Master's thesis, University of New Hampshire.

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

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

Visvalingam M and Whyatt JD (1993). Line generalisation by repeated elimination of points. *The Cartographic Journal*, 30(1):46–51.

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

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