Geographical Information Management and Applications

DETECTING AND RESOLVING NARROW CONFLICTS FOR VARIO-SCALE MAPS

A comparative study into the efficacy of different approaches for the detection and resolution of narrow conflicts in the context of a vario-scale data structure.

By

Willem van de Corput

Name
Willem van de Corput

Student number
6257984

Student e-mail address
w.vandecorput@students.uu.nl

Supervisor
Martijn Meijers (Delft University of Technology)

Responsible Professor
Peter van Oosterom (Delft University of Technology)

Program
Geographical Information Management and Applications

University
Delft University of Technology

Issue date
31st of May, 2023

Preface

This research was conducted from September 2022 until May 2023 as a thesis for the Master's program Geographical Information Management and Applications (GIMA). This project symbolises the end of my almost two year exploration into the world of Geo-Information and GIS. Over the course of this program, I have gained an tremendous amount of knowledge regarding different aspects of GIS as well as a number of practical skills. However, as a result of my Human Geography background, there was one particular skill that I had not been able to dive into, prior to starting this thesis. This skill was writing code, specifically in Python. As a consequence of wanting to learn Python, combined with a desire to tackle a more technical topic than I had previously done, I landed on the topic of 'vario-scale maps'. The start of my research process was characterised by a steep learning curve, as I had to familiarise myself with some technical details regarding the topic, but also with the basics of python programming. Looking back on the past 9 months, I can gladly say that I have achieved what I set out to achieve, which was getting the hang of python programming and researching a technical topic.

Despite the fact that this process involved a large amount of self-study, I would not have been able to successfully complete it had it not been for the following people. My first token of appreciation goes out to Martijn Meijers, who was kind enough to supervise my thesis. Not only did he provide highly useful feedback, he also encouraged and supported me to tackle issues that I might have previously considered to exceed my technical reach.

Although Martijn was closely involved with my thesis, I also received a tremendous amount of support from my friends. They stimulated me to put in my best effort and even helped me in getting the hang of python. I am thoroughly convinced that I would not have been able to accomplish the results I have now if it was not for the support of my friends.

Last but certainly not least, I want to express my gratitude towards my family for encouraging me to throw myself into a topic that was way out of my comfort zone. They were supportive of this decision right out of the gate and have delighted in seeing me progress over the course of the past months.

To conclude, I hope that those reading my thesis will find it as interesting and enjoyable as it was for me to write it.

Abstract

Zooming operations are fundamental to digital multi-layer maps which are classically comprised of a given number of layers that each represent the area in question at a specific scale level. As a result of this structure, the data from which the layers are built is also structured in layers, meaning that information regarding the same geographical location is stored multiple times. In an effort to resolve this redundancy in data storage and abrupt map changes during zooming operations, the concept of vario-scale maps has been developed. In the corresponding data structure, features are stored as 3D objects that represent their geometry across all scale levels at once. One of the fundamental problems faced by this concept, relates to the question of when features (or parts of them) become too small or narrow to be displayed during zooming operations. By comparing three main types of approaches, this research aims to shed light on the issues of detecting and resolving such conflict locations. The results of both a theoretical and an empirical exploration of these approaches indicate that they can be used effectively to address different aspects of these processes. On top of that, by combining the approaches in a complementary fashion, their strengths can be harnessed in a way that minimises their shortcomings.

Key words: Vario-scale, tGAP, narrow conflicts, Delaunay triangulation, polygonal skeleton, computational geometry

Table of Contents

Table of Figures

1. Introduction

1.1. Background

With the dawn of the digital era, zooming in and out has become an integral part of online maps. More and more maps are available at different scales, from large to small and much in between. To grasp how this zooming process works from a technical perspective, one might think of the map as a stack of pancakes that is made up of different layers. The layers on the bottom display the world in a high level of detail (e.g. streets and houses), while the upper layers only show larger features (cities, regions or even countries). If one were to zoom in or out, for instance on Google Maps, abrupt switching from one layer to another can be seen at a certain scale level. The existence of such separate layers means that the map that is observed is actually made up of different maps that are stacked on top of each other. All these maps have information, i.e. data behind them, that indicates what should be displayed and where (geometry attribute). With the increase of technology over the past few decades, the amount of available data has increased. As a consequence, the different layers of online maps have gotten bigger and bigger from a data perspective. Moreover, the number of layers that constitute the map have increased. In an effort to minimize the amount of data necessary for interactive multi-layer maps and to prevent abrupt changes in the map as a consequence of zooming operations, the concept of vario-scale maps has been developed. The data structure behind the map that is displayed is fundamentally different from the ones behind classical multilayer maps (pancake model). In the scientific literature around vario-scale maps, this data structure is generally referred to as topological Generalized Area Partition structure (tGAP) (Haunert, Dilo & Van Oosterom, 2009; Suba, Meijers & Van Oosterom, 2013; Van Oosterom et al., 2014). The innovation that this concept brings is that, rather than saving information multiple times over multiple scale levels, it stores the information only once. This is done at the largest scale level (bottom pancake), since this layer contains all the information that the layers above do and more. When one then zooms out on the map, the information of the lowest layer gets simplified and aggregated to a less detailed version with a smaller scale (Van Oosterom & Meijers, 2011). In a classical tGAP structure, features that are become irrelevant for display as a consequence of a decrease in scale are added to surrounding features. Although this way of structuring map data results in a map that is in some sense layerless, slightly abrupt changes can still be seen when features suddenly disappear as a consequence of zooming out. In an effort to further smoothen zooming operations, Van

Oosterom and Meijers (2011) developed the concept of a smooth tGAP structure, where even the splitting, removing and merging of features happens gradually. Such a data structure allows for true smooth zooming operations that do not produce any abrupt changes in the visualised map. In order to get a visual grasp of this idea, one could say that the pancake stack (classical model) has now been turned into a solid cake. This 'cake' can not only be sliced at fixed scales but at any scale level (Figure 1). Consequently, vario-scale maps have, for all intents and purposes, an infinite number of layers.

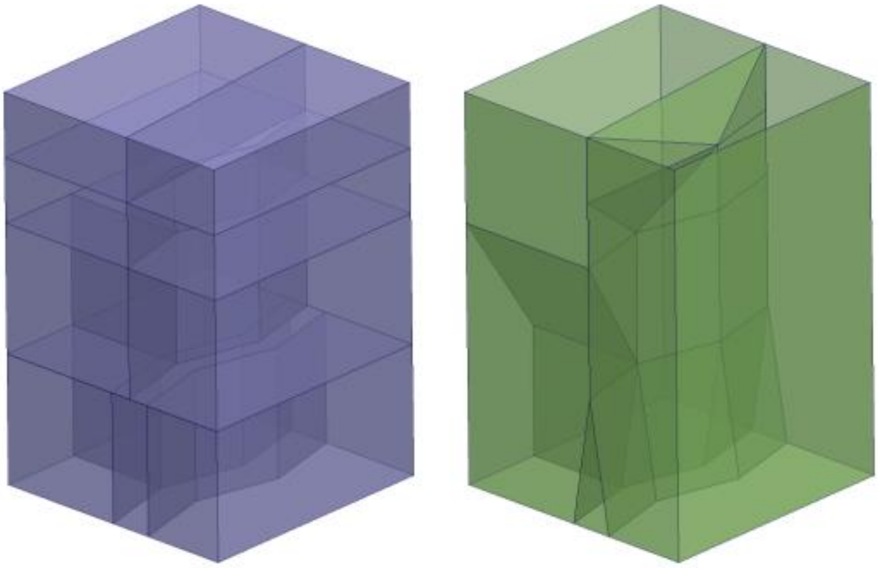


Figure 1 - Classical multi-layer map structure vs Vario-scale structure (source: Van Oosterom & Meijers, 2011)

1.2. Problem introduction

With all new technologies come obstacles and challenges, and so does the concept of vario-scale mapping. A major issue with these types of maps is deciding at what scale level a feature starts to or stops to become relevant for display. As an example, it becomes futile to display a polygon representing a neighbourhood when displaying a map at global scale level. A problem like this occurs when converting a map into a smaller scale map, which means that the problem occurs an infinite number of times in a vario-scale map. Namely, a feature does not become irrelevant abruptly, as is the case in traditional maps. Rather it gradually becomes less and less relevant until, finally, it is no longer displayed. This entails that, as the scale level of the map decreases, its features are continually subject to geometric changes. Within this issue, narrow parts form a particularly challenging geometric phenomenon. Mostly because they might be too narrow to be displayed, but also part of a polygon that is too large or important to be removed. In order to avoid the presence of narrow parts, while still maintaining the approximate geometry of a certain feature, it generally undergoes some type of generalisation. To be precise, narrow parts pose two main challenges in the context of generalisation, which is essentially an ongoing process in a vario-scale map. The first issue is determining which parts of a certain polygon are too narrow. This can be done using different mathematical approaches. The second issue relates to how these ‘narrow conflicts’, as they are referred to within the relevant literature (Bader & Weibel, 1997; Gao et al., 2012; Højholt, 2000), might be tackled. This, again, can be done in a number of different ways.

1.3. Problem definition

Currently, there are a number of mathematical approaches that have proven useful in tackling one or both of these issues. A commonly used one relies on the Delaunay triangulation (Bader & Weibel, 1997; Tran & Fan, 2013), which has been used for years in different fields of study that have to do with the construction and manipulation of shapes. The basic idea behind this approach is that the vertices that make up a polygon are connected in a network of triangles. Based on these triangles, decisions can be made as to the manipulation or modification of the initial polygon.

Delaunay triangulation-based frameworks have been used since the beginning of the development of GIS at the end of the 20th century. At the time, they were already used in mathematics but also turned out to be a viable tool for measuring and manipulating shapes, for instance on maps. Since then, many scientific publications have demonstrated their validity as a method for the detection and resolution of geometric conflicts (Bader & Weibel, 1997; Haunert, 2008; Jones et al., 1995; Van der Poorten & Jones, 2002). Such papers have firmly established Delaunay triangulation-based frameworks within the world of map generalisation and automatic map generalisation subsequently (Galanda, 2003). For this reason, Delaunay triangulation-based frameworks will act as a benchmark in this research.

More recently, however, alternatives or additions to Delaunay triangulation-based frameworks have emerged. One category of approaches relies on polygonal skeletons, which are essentially centre lines derived from polygons. Such skeletons are constructed by finding central points in between the boundaries of the polygon and connecting those into a line segment. As is the case with Delaunay triangulation-based frameworks, this method can be used to detect narrow polygons. In short, this is done by measuring the distance from the skeleton to the closest polygon edge (Du et al., 2021; Haunert & Sester, 2008). Moreover, skeleton-based approaches have also been used to manipulate the geometry of polygonal features, which relates directly to the issue of resolving narrow conflicts (Haunert & Sester, 2008).

Furthermore, there are a number of buffer-based approaches (Gao et al., 2012; Peter, 2001) that rely on the creation of buffers in and around features in order to obtain information regarding size and width. Based on this information, it can be ascertained whether or not a given location is too narrow.

These categories rely on different geometric operations which yield different results. These might be more or less suitable depending on map/feature characteristics or the intended application. The world of vario-scale mapping is still in its infancy. Therefore, no optimal method regarding how narrow conflicts ought to be handled has been definitively determined. This problem, broadly speaking, is the one that will be central to this research. To be more precise, the central question to this research is as follows:

"To what extent can buffer-, skeleton- and Delaunay triangulation-based approaches be effective for handling narrow conficts in the context of vario-scale maps?"

In view of answering this question in a complete manner, the following sub-questions will break down this central question. By using them as stepping stones towards answering the central question, the findings of this research can be logically framed with regard to their relevance to the central question.

- What exactly is referred to by the term narrow conflicts?
- What falls under the umbrella of 'handling' narrow conflicts?
- What additional requirements for 'handling' narrow conflicts are brought on by the principles of vario-scale maps?
- To what extent can the different approaches be used to effectively fulfil these requirements?

Although these sub-questions provide some further sense of clarification regarding how this research is structured, for a more visual overview of how the central concepts relate to one another, Figure 2 can be consulted.

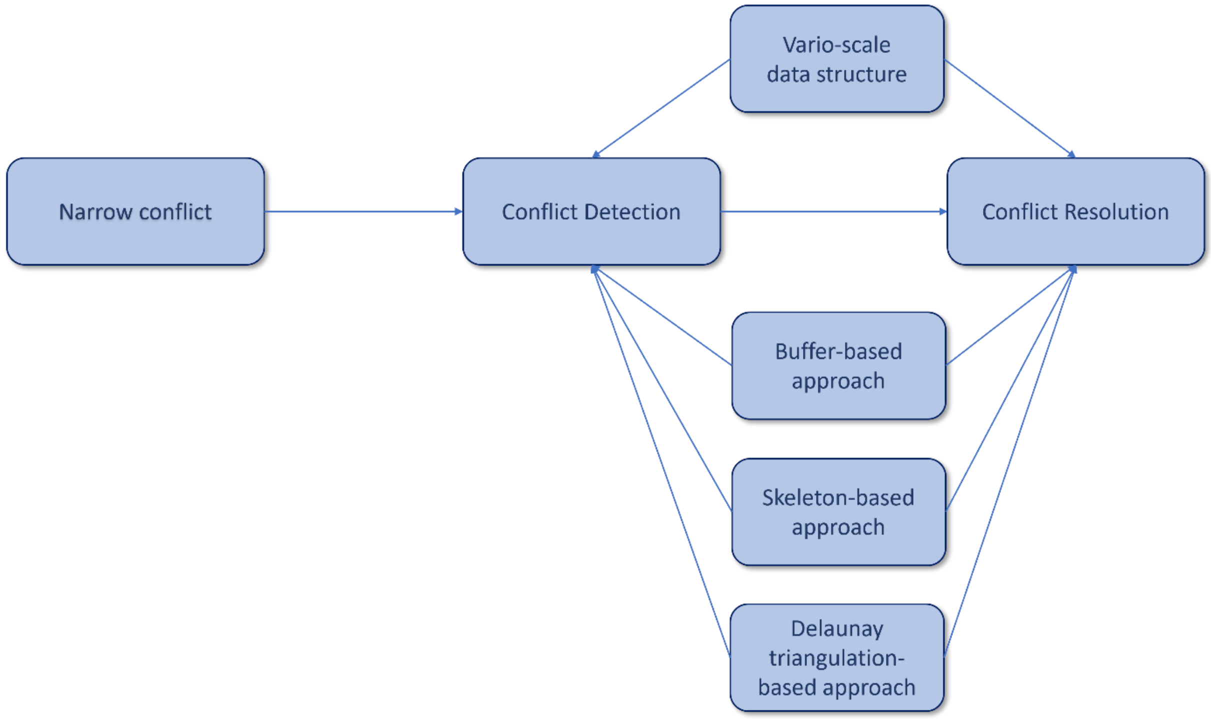


Figure 2 - Conceptual Model (source: own work)

1.4. Research objectives

The desire to address the central question and the sub-questions associated with it, entails some objectives that this research will set out to achieve. Some of these relate to establishing a theoretical foundation or justification for any potential conclusions. Others are concerned with the actual creation or successful implementation of a particular concept. Although some of these objectives are not strictly necessary for answering the central question, achieving them would further strengthen the results and findings of this research. For the sake of overview, the objectives of this research will be listed in the order in which they will be addressed.

1. **Determine the relative strengths and weaknesses of the different approaches in light of handling narrow conflicts**

As the brief introduction of the three categories of approaches illustrates, they rely on different operations. Also, they appear in a large number of related scientific studies, which suggests that none of them has definitively been deemed optimal. These observations indicate that the different approaches are likely to each have their own advantages and disadvantages when it comes to how they deal with narrow conflicts. Therefore, establishing a theoretical understanding of these characteristics might provide useful information, both in view of any empirical experimentation with the different approaches and for the sake of addressing the central question.

2. **Determine what are the fundamental principles behind a tGAP data structure**

Since the focus of this research is to address the different approaches for their application in a vario-scale context, it is directly relevant to understand the data structure behind this innovative type of map. Furthermore, such an understanding is required in order to put the approaches to the test empirically.

3. **Implement the different approaches to the issues of detecting and resolving narrow conflicts in a vario-scale context**

Once objectives 1 and 2 have been achieved, a solid foundation for the actual implementation of the different approaches has been established. Therefore, the 3rd objective of this research is to implement them in order to empirically determine to what extent they can be used to effectively detect and resolve narrow conflicts. This will further demonstrate their relative strengths and weaknesses.

4. **Based on the literary and empirical findings, create a balanced approach for successfully transforming 2D polygons into 3D objects.**

The final objective of this research is to combine the theoretically and empirically obtained information into a balanced and properly substantiated approach for the creation of tGAP structured objects from 2D polygons.

1.5. Scope definition

Before answers to the research questions can be obtained, it must be established what exactly is to be investigated, and, perhaps more importantly, what is not. Therefore, this section will outline concretely which aspects that might be connected to this topic will be addressed and which will not. In doing so, it will be made clear where this research falls in reference to associated researches and where additional research might be necessary. Ultimately, the scope definition will clearly bound the research and its implications so that these cannot be taken out of context. This restriction is necessitated most primarily by the limited amount of time available for this research.

Table 1 – Scope Definition

Scope definition		
	In scope	**Out of scope**
1	Detecting and resolving narrow conflicts	Detecting and resolving distance and proximity conflicts
2	Using buffer-based, skeleton-based and Delaunay triangulation-based approaches to detect and resolve narrow conflicts	Using other methods to detect and resolve narrow conflicts
3	Resolving narrow conflicts at a polygonal level	Resolving narrow conflicts at a map level
4	Creating a 3D polyhedral representation of testing polygons with scale as the third dimension	Creating a tGAP data structure based on multiple polygons.
5	Determining the strengths and weaknesses of the different approaches with regard to the detection and resolution of narrow conflicts within the context of vario-scale maps	Evaluating any technical obstacles regarding the actual implementation of the different approaches for the creation of vario-scale map

1) As Chapter 2 will elucidate, narrow conflicts fall under the broader umbrella of geometric conflicts. These conflicts are all relevant to vario-scale maps. However, this research will be restricted in its focus to narrow conflicts. This decision is made for the sake of time management and in order to ensure that the issue can be explored in sufficient depth. On top of that, this type of conflict is particularly relevant because it is an issue that is inherent given the fundamental principles of vario-scale maps. This issue relates to the fact that smaller and narrower features become more or less relevant for display as scale is decreased or increased. Furthermore, other types of conflicts are generally concerned with distances between different features. As will be discussed shortly, this research is limited to a feature level of analysis.

2) A second important demarcation of scope relates to the choice of three types of approaches that play a central role in this research. These approaches are not picked arbitrarily. Instead, as will be discussed in Chapter 2, they are among the most prominent approaches within relevant fields like computational geometry, feature generalisation, conflict detection and conflicts resolution. Therefore, despite the fact that other methods might be effective for detecting and resolving narrow conflicts, the scope of this research is limited to buffer-, skeleton- and Delaunay triangulation-based approaches.

3) Perhaps the most critical issue concerning the scope definition has to do with the distinction between a polygonal level of analysis and a map wide perspective. Although this research is limited to a polygonal level of analysis, the extension of this perspective to a map wide one is likely to entail a range of consequences that are not considered in this research.

4) As a consequence of this decision, the objective of this research is to determine the efficacy of the three approaches when it comes to the creation of a 3D tGAP adequate object. Therefore, the creation of a tGAP dataset, which would contain multiple polyhedral features instead of merely one, is chosen to be out of scope.

5) Lastly, despite the fact that this research aims to obtain information that can be used for the eventual creation of properly functioning vario-scale maps, any technical concerns that relate to this issue will not be subject to consideration. Although creating a tGAP structured object is vastly different from using such features to create an actual vario-scale map, the challenges associated with that process fall outside of the scope of this research. The reason for this, once again, stems from a lack of time.

1.6. Relevance

The concept of vario-scale maps is a rather new one and current attempts at its implementation are far from perfect. Therefore, a lot of research needs to be undertaken with regard to its optimalisation and regarding the realisation of its theoretical principles. Within such research, the requirements for the creation of a vario-scale data structure, the application of different methods and their results are directly relevant. But why would the concept of vario-scale structures itself be relevant in the first place?

The potential positive consequences of the development of vario-scale structures are twofold. Firstly, the way in which data is stored in a vario-scale structure allows for the elimination of redundant information, which is always present in traditional multilayer maps. This is the case because every object is stored only once. Given the rapid increases in the amounts of data that are currently being collected, the creation of a more efficient data storage system would be a very welcome development.

Secondly, the concept of the tGAP data structure could bring forth innovative ways of extracting information from a dataset. The ability to view and work with a dataset at every scale level allows users to very precisely obtain information that might otherwise be 'hidden' in between layers. Not only can data be accessed at every scale imaginable but one might also be able to create a map with different scale levels across the map plane. For example, a slice could be made through the data structure so that areas closer to the observer are displayed at a larger scale then areas that are more distant (Harrie, Sarjakoski & Lehto, 2002). This exemplifies how vario-scale data structures could enable new ways of extracting information compared to traditional multilayer maps.

Given all this, a question that still remains is why narrow conflicts in particular are so crucially important to the successful actualisation of vario-scale concepts The reason for this lies in the process of aggregation and generalisation, which is fundamental and ever ongoing in the creation of proper vario-scale data structures. These space-scale cubes are built from the bottom up. Therefore, the creation of the complete data structure, containing the map features across all scale levels at once, requires continual aggregation, generalisation, removal, shrinking and enlargement of these features. Given the previously mentioned relation between a feature's size (or width) and its relevance, narrow (parts of) features are the focus of such operations. In other words, what narrow conflicts are and how they are handled are the two primary aspects of creating an actual vario-scale structured representation.

2. Theoretical framework

Building on the aforementioned problem definition and the research objectives that accompany it, a decision has to be made as to how these ought to be approached. In order to concretise this by means of a comprehensive methodological framework, the main concepts that are central to the research problem must first be addressed in the context of the relevant literature. This framing will yield clear and bounded definitions of important concepts as well as properly substantiated theoretical assumptions. On top of that, a theoretical exploration of associated researches is likely to bear information regarding the justification and application of different approaches for handling narrow conflicts in the context of vario-scale maps. For the sake of overview, this chapter will be structured like a funnel (Figure 3). This implies that the relevant concepts and associated findings will be discussed by proceeding from a rather broad and general perspective towards an increasingly narrow and specific focus.

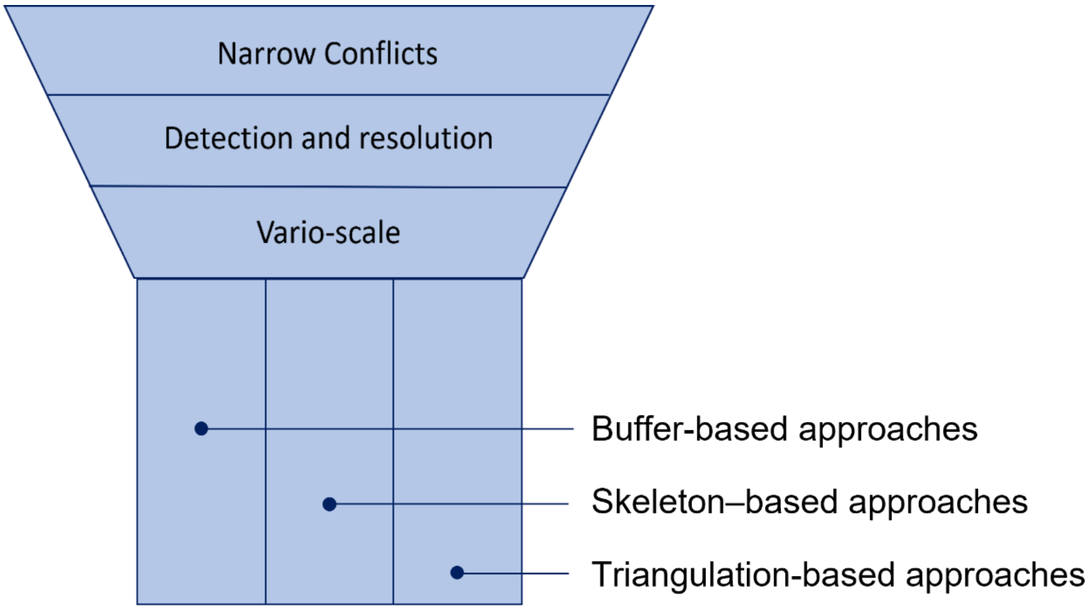


Figure 3 - Overview of the structure of the theoretical framework (source: Own work)

Speaking in terms of the concepts that are central to this research, this results in the following structure. Firstly, the concept of narrow conflicts will be framed in relation to other types of geometric conflicts that have to do with the representation of features across different scales. This section, therefore, aims to establish some fundamental terminology and clearly bound what is meant by narrow conflicts. Secondly, the concept of handling narrow conflicts, as mentioned in the research question, will be broken down into detection and resolution. This dichotomy is necessitated by the fact that these two issues are fundamentally different from one another and, thus, might require different approaches (Bader & Weibel, 1997). Thirdly, the concept of vario-scale maps will be framed from a theoretical perspective. This section will outline what is meant by the term vario-scale and associated terms such as tGAP structure and SSC (space-scale cube). Furthermore, this section aims to establish what consequences and additional requirements the focus on vario-scale maps might have on the issues of detecting and resolving narrow conflicts. Subsequently, the three approaches that are central to this research will be explored in relation to the concepts that preceded. Practically speaking, this means that their theoretical relevance in relation to the detection and resolution of narrow conflicts within a vario-scale environment will be elaborated. Lastly, this chapter will end with a brief but comprehensive summary of the major findings in the preceding sections of the theoretical framework.

2.1. Narrow Conflicts

Building on the introduction, in which the problem regarding narrow (parts of) polygons was elucidated, it is firstly important to clarify why they are problematic in the first place. For example, one could wonder why narrow sections might be less relevant on a map than larger sections. The literature surrounding narrow parts discusses a number of different reasons why these might be undesirably given a maps purpose. For example, from a rather general perspective on map principles, the presence of small areas can make the map look messy and chaotic. This could take away from the maps clarity and make it harder to interpret. Furthermore, excessive amounts of detail greatly increase the amount of data that is to be visualised, which is detrimental from a technical perspective (Galanda, 2003). On top of that, narrow parts are especially problematic for vario-scale maps due to the way in which features with larger area sizes "eat up" smaller features. Generally speaking, this is a good thing, since it is desirable to preserve larger features and have smaller features disappear. However, for long narrow sections with a large area size, such as rivers, having the river shape added to the areas of the surrounding features might be more appropriate. Not doing so could result in the whole map being dominated by rivers after a number of aggregation operations.

For the sake of clarity it is necessary to firmly establish what is being referred to when speaking of narrow polygons or narrow sections. In the literature around polygon generalisation, narrow parts are most commonly referred to as geometric conflicts. These occur when a certain threshold width between vertices that make up polygons, is not met (i.e. vertices are too close together). These vertices might be part of the same polygon or of different ones. Within the context of individual polygons, three kinds of conflicts can be differentiated, namely, size conflicts, distance conflicts and narrow conflicts (Gao et al., 2012)..

"Size conflict refers to polygons that are too small (smaller than a certain size threshold) to be displayed at the target scale. Both distance conflicts and narrow conflicts are caused by the distance between vertices of polygon boundaries being smaller than a certain threshold. Distance conflicts concern the distance between consecutive vertices, whereas narrow conflicts concern the distance between non-consecutive vertices." – Gao et al., 2012

When addressing the width of spaces between different polygons, locations that are too narrow are called proximity conflicts (Bader & Weibel, 1997). The terminological framework resulting from the conflicts mentioned by Bader and Weibel (1997), Gao et al. (2012) and Peter (2001), is presented in Table 2. Through consideration of the different types of conflicts, it becomes evident that narrow parts of polygons, as they have been discussed thus far, fall within the category of “narrow conflicts”. This is the case because this research (1) is focused on conflicts within the same polygon and (2) on the distance between non-consecutive vertices.

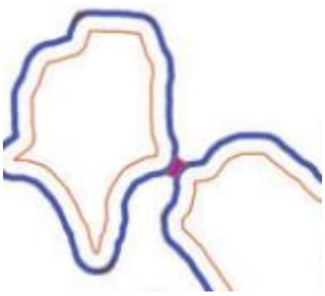
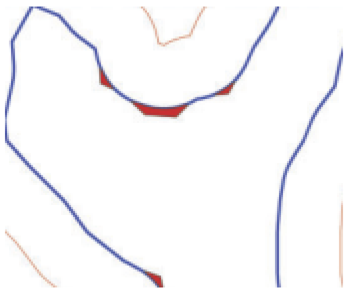
Table 2 – Overview of geometric conflict terminology (source: own work)

Geometric conflict	
Conflicts within individual polygons:	Conflicts between different polygons
Size conflict	**Proximity conflict**
Distance conflict	
Narrow conflict	

Even though this terminology yields some sense of demarcation from similar geometric conflicts, narrow conflicts themselves still occur in a number of ways.

Peter (2001) breaks down narrow conflicts even further by distinguishing narrow-corridor conflicts and narrow-jut conflicts. The former category refers to narrow sections that connect wider sections of the polygon in question, while the latter describes narrow sections that are only attached to a wider section at one end. These can be thought of as alike a peninsula sticking out from the larger landmass. The reason for this distinction is the fact that the removal of a narrow-corridor conflict would split the initial polygon into two or more parts while the removal of a narrow-jut conflict would not cause a split.

Table 3 – Subdivision of narrow conflicts (source: Gao et al.; own work)

Narrow conflict		
Narrow-corridor conflict	**Narrow-jut conflict**	**Jut-like bend**
Narrow conflict section which connects two or more core areas of a given polygon. Its removal would, therefore, split the initial polygon into multiple parts.	Narrow conflict that only attaches to a core area on one side. This means that it does not function as a bridge between multiple core areas and, thus, its removal does not cause a split in the initial polygon	Small sections at the outer edges of a polygon which might be too narrow depending on the method of conflict detection.
 *(Source: Gao et al., 2012)*	 *(Source: Gao et al., 2012)*	 *(Source: Gao et al., 2012)*

2.2. Detection and Resolution

What has been made clear is that the concept of handling narrow parts can be divided into two aspects. The first one is about the detection of narrow conflicts, while the second one relates to how these can be resolved. Resolving geometric conflicts implies that, given a polygon or map with one or more geometric conflicts, a version without geometric conflicts is derived. Although there is a relation between how conflicts are detected and how they are resolved, for the most part these are distinct processes, that, thus, may require different approaches (Bader and Weibel, 1997). Therefore, they will be discussed in separate sub-sections.

2.2.1. Conflict detection

As the term suggests, conflict detection is concerned with identifying which polygonal areas are too small or narrow to be displayed on the map at a given scale. Generally speaking, a certain threshold value is determined, which represents the minimum width that an area should have. All locations where this criterium is not met are appointed as conflict areas. Measuring the width of polygonal areas is not as straightforward as one might think. For example, where does one start measuring from and to where, exactly? Furthermore, manually reviewing potential conflict locations might be a viable solution for one or a few features, but as the number of features grows it becomes increasingly more laborious to pinpoint all geometric conflicts. As a consequence, a number of different methods have been developed that are aimed at effectively and accurately locating geometric conflicts at a map wide level.

Without prematurely addressing conflict detection methods from a vario-scale perspective, it is firstly relevant to discuss how certain methods might be used for the detection of narrow conflicts in the context of discrete scale levels. A quick overview of literature shows that there are many different approaches that have been proven capable of detecting conflicts in an automated manner. Among them are the rolling ball principle, the shrinking ball algorithm, the Voronoi cell approach, buffer-based approaches and skeletal-based approaches. Despite the substantially different ways in which these are implemented, all such methods can effectively detect narrow conflicts. However, among them, some methods feature more prominently and more frequently in the relevant literature. The buffer-, skeleton- and triangulation-based approaches specifically emerge in a number of different scientific publications (Altundag & Stoter, 2011; Bader & Weibel, 1997; Gao et al., 2012; Haunert & Sester, 2008, Peter, 2001). Due to the fact that they have repeatedly been implemented successfully and are therefore firmly established in the scientific literature, these categories of approaches will be explored in more detail.

Buffer-based operations can be applied in different ways to detect conflicts. One of the more straightforward implementations of buffer analysis in this regard is discussed by Peter (2001), who uses an inward facing buffer. The distance of this buffer is half the determined minimum width value. Peter bases this value on visual separability of objects in a map. The reason for using half the minimum width value is that the buffer is applied on all sides of the polygon. This means that if two sides of the buffer intersect, a narrow conflict is detected. If the polygon is separated into more than one area after the buffering operation, there is at least one narrow- corridor conflict in the original polygon. In fact, the number of narrow-corridor conflicts in the original polygon is *n*-1, where *n* is the number of polygons after the buffering operation. A benefit of this method is that it is really straightforward and does not necessitate any complex intermediate calculations in order to be able to measure width. A drawback, however, is the fact that it does not detect all kinds of narrow conflicts. Gao et al. (2012) distinguish narrow-corridor conflicts and narrow-jut conflicts (red dotted circles in Figure 5). A narrow-corridor conflict refers to a narrow part that bridges two larger parts of a shape. A narrow-jut conflict refers to a narrow part that extends out from a larger part, like a peninsula. This inward-buffering approach put forward by Peter (2001) is very efficient in detecting narrow-corridor conflicts, but very poor when it comes to narrow-jut conflicts. This is the case because these juts do not leave behind a core area after the buffering operation, which means that they are not recognised as being a conflict area.

Figure 4 – Internal Buffer approach (source: Gao et al., 2012)

To resolve this issue with the internal buffer approach by Peter (2001), Gao et al. (2012) developed an extension of this method, namely the inward-outward-buffering approach. As the name suggests, this method creates a buffer in two directions, both with a width of half the length of the narrow conflict threshold value. First the inward buffer is created, which detects any narrow-corridor conflicts. Using this inwardly directed buffer as input (not the original polygon), an outwardly directed buffer is created. By comparing the difference between this second buffer and the original polygon, narrow-jut conflicts can be detected. However, as a result of nature of round buffers, some narrow-jut conflicts are assigned to so called jut-like bends. These are locations where the original polygon is slightly outside of the outward facing buffer but not as much as to be labelled a jut. Consequentially, this method produces three types of narrow conflicts: corridor conflicts, jut conflicts and jut-like bends. The specification of the latter category can be useful in view of any subsequent generalisation.

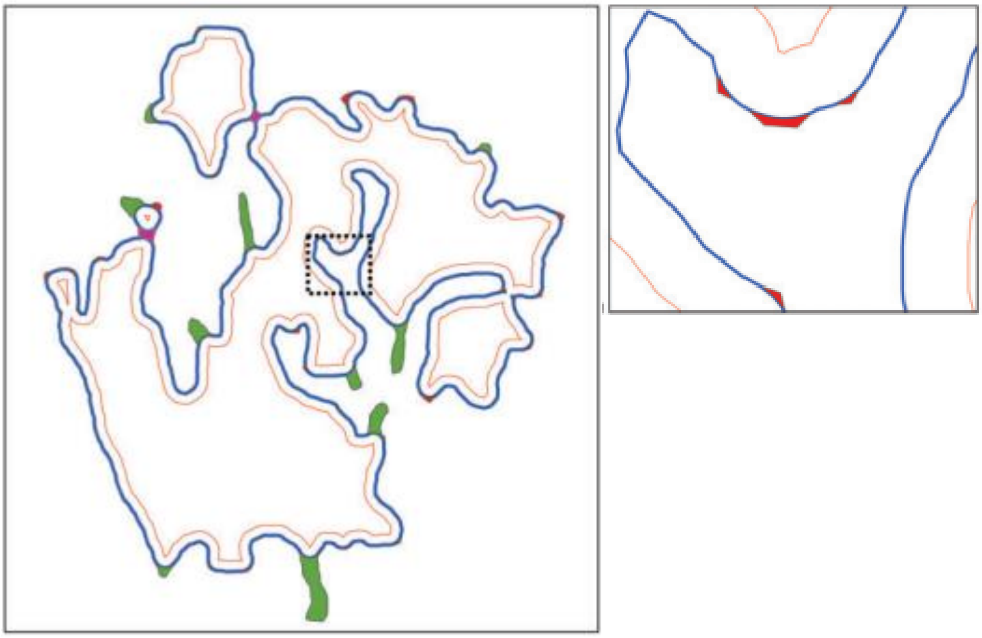


Figure 5 – Inward-Outward-buffering and three types of narrow conflicts (source: Gao et al., 2012)

Gao et al. also mention another relevant beneficial opportunity that comes with the use of this method.

"A notable point is that the outcome of the approach is highly determined by the buffer width which is further related to the minimum distance threshold. The larger the buffer width, the more conflicts will be detected including the number and area of the conflict regions. The value of buffer width should be decided according to the map scale of the target map. It also can be variable according to the importance of different categories expressed with polygons. For polygons belonging to the important categories, the buffer width can be set as a small value in order to keep more details on the target map."- Gao et al., 2012

A method that is closely related to the buffer-based approaches put forward by Peter (2001) and Gao et al. (2012) is the rolling ball approach developed by Bader and Weibel (1997). In this method, an imaginary ball with a certain diameter is rolled along the inside of a polygonal boundary. In doing so, an offset is created that has a width equal to the diameter of the ball. This offset is, for all intents and purposes, equivalent to the inward buffer mentioned by Peter (2001). In case no conflict areas exist withing the polygon, the offset line created by the ball never intersects itself. Were this to happen, however, a conflict area is present within the polygon. Where in the polygon this area is located can be determined by analysing the direction of vertices along the inside of the offset generated by the ball. This is the case because an intersection of its offset curve incurs a flipping of the direction of the vertices. A visual support for this phenomenon can be found in Figure 6.

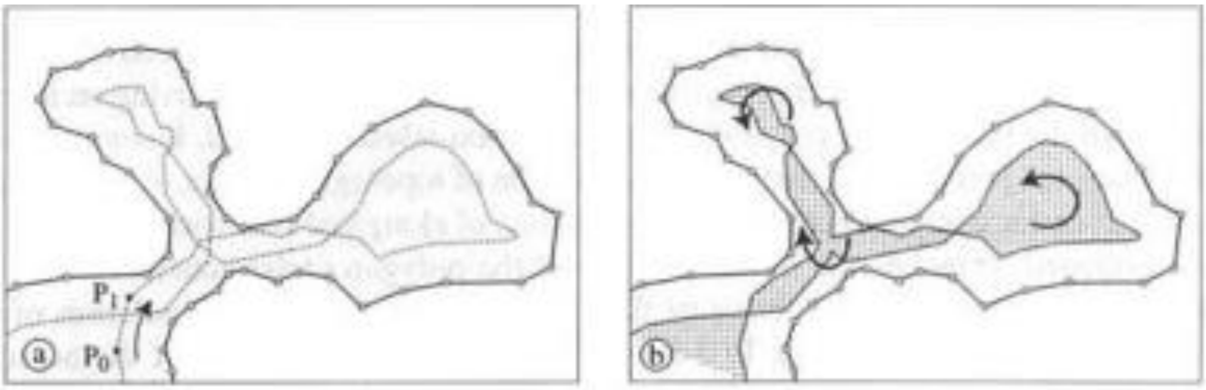


Figure 6 – Rolling Ball approach (source: Bader & Weibel, 1997)

In short, the following can be concluded regarding the usability of buffer-based approaches for the detection of narrow conflicts. Buffer-based approaches have been shown in different ways and by different authors to be capable of effectively detecting narrow conflicts in an automatic manner. Bader and Weibel (1997) even conclude, in their research, that "*…rolling ball buffers more neatly partition the polygons into regions of conflicts and 'safe areas'. This indicates that rolling ball buffers are superior for conflict detection…*".

Next to that, reflecting back on the categorisation of narrow conflicts in section 2.1, it can be concluded from this sub-section that inward facing buffers, such as those in Peter (2001) and Bader and Weibel (1997), detect narrow-corridor conflicts while the addition of an outward facing buffer in Gao et al. (2012) results in the detection of both narrow-corridor conflicts and narrow-jut conflicts. The latter category of conflicts is crucial to this research since narrow juts are directly relevant in the context of narrow parts. For that reason, the inward-outward-buffering approach promises to be most suitable for a comprehensive approach to conflict detection and, thus, will feature in the empirical part of this research.

Skeletal-based approaches have also been used repeatedly for the detection of narrow conflicts. In order to understand how it is that polygonal skeletons can be used for such a purpose, it is crucial to establish what skeletons are, how they are created and what information they contain in relation to the shape that they are derived from. In short, a skeleton can be described as the centre line of a shape. They can be calculated in a number of ways, but in general the skeleton is what remains of a polygon when all its vertices are moved inward until they collide in the middle. The connection of these collision points by means of a straight line results in the so called straight-skeleton, one of the more prominent skeletal variants. This skeletal variant is characterized by sharp corners. Closely related to this geometric entity is the medial axis (Blum, 1967), which similarly represents the centre line of a feature but in a more curved and smoothened manner. The medial axis of an object is the set of all points having more than one closest point on the object's boundary. This set of points is then connected by drawing a curved line segment through it. The visual difference between these two variants, which are the most frequently featured variants in the literature on computational geometry, can be seen in Figure 7.

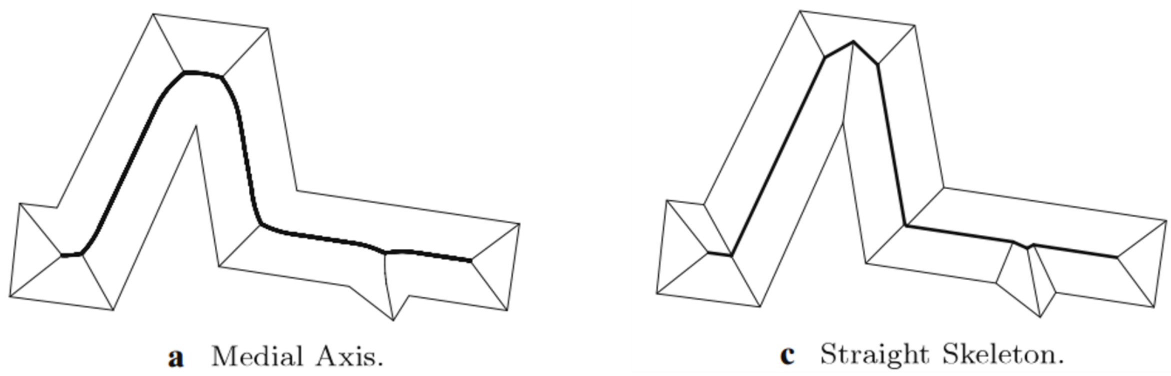


Figure 7 – Two skeletal variants visualised (source: Haunert & Sester, 2008)

Another aspect of skeletal variants and how they are applied in computational geometry relates to the so called pruning of the branches on the skeleton. As can be seen in Figure 7, skeletons do not always consist of merely one main branch, but can also be made up of side branches. Whereas the main branches represent the more general frame of the polygon, the smaller side branches represent the finer outer details of the polygons shape. The inclusion of these smaller branches might be either desired or undesired depending on the specific use purpose of the skeleton (Montero & Lang, 2012).

Zooming in on the issue of detection of narrow conflicts, the usefulness of skeletal variants has been well established in literature (Aigner, Aurenhammer & Jüttler, 2015; McAllister, 1999; McAllister & Snoeyink, 2000). As McAllister (1999) mentions, by adding a skeletal variant to a narrow polygon, the width of that polygon at any location can be calculated. This is done by measuring the distance from a point on the skeleton to the closest point on the polygon boundary. This distance can be compared against a minimum width value in order to distinguish precisely which areas should be appointed the label of narrow conflict and which should not. Altundag and Stoter (2011) discuss how they implement this technique in order to detect roads that are too narrow automatically based on a minimum width value. In short, they measure the distance from every vertex of the skeleton to every vertex of the polygon boundary. They compare these distances to a minimum width value of 2 meters. If this minimum distance is violated at any place in the road polygon, the polygon is deleted. This approach might work reasonably well for roads that have more or less the same width throughout the entire polygon. However, projecting this method onto the issue of narrow conflicts immediately becomes problematic, as a polygon might be very large in one location but narrow in another. Haunert and Sester (2008) make up for this limitation in their application of a skeleton (straight skeleton) for the detection of narrow conflict. Although their implementation is mainly the same as the one in Altundag and Stoter (2011), they do not remove the entire polygon when it contains any measurement that violates the minimum width criteria. Rather, they collapse only the parts where this happens into a line feature, which is the local skeleton. This splitting of polygons into wide and narrow parts fits well with the concept of detecting narrow parts. In fact, narrow parts are always part of a polygon that has wider areas (otherwise they would be deemed size conflicts (Gao et al., 2012)).

Haunert and Sester (2008) demonstrate that the polygon can simply be divided into conflict and non-conflict at the location where the minimum distance is violated. However, this still leaves the fact that distances are only calculated in between vertices. This means that width is not assessed everywhere and that the number of vertices in both the skeleton and the polygon outline directly influence not just the number of measurements but also the distances that are calculated. These two issues take away from the accuracy and integrity of the results that are obtained from the use of skeletal variants for the detection of narrow conflicts.

Another issue entailed by the use of skeletal variant for conflict detection relates to the previously discussed topic of pruning branches. This is the case because the more side branches are maintained, the closer the skeleton approximates the polygon boundary. Practically speaking, this means that there is a direct relation between the degree intricacy of the skeleton and the number of conflicts that will be detected. As a consequence, when using a skeleton-based approach for detecting narrow conflicts, there is not just the minimum width value parameter but also the skeleton's intricacy parameter. Moreover, the density of the vertices that make up the polygon boundary also determine the number vertices that make up its skeleton. Without making the definitive case that such additional parameters rule out the efficacy of skeleton-based approaches, it can be stated that these parameters ought to be taken into account when using this type of approach for the detection of narrow conflicts.

Another aspect to take into account would be the selection of a specific skeletal variant: Straight skeleton or medial axis. Both methods appear repeatedly throughout the literature on conflict detection and computational geometry in general. Therefore, finding out what circumstances or criteria render one variant favourable over the other would be relevant in view of any empirical analyses. Naturally, the use of either one of these variants depends on the intended application (Tănase & Veltkamp, 2004). However, within the literature on narrow (parts of) polygons, there are not a large number of sources discussing the differences on conflict detection. For example, Altundag and Stoter mention using both skeletal variants and treat them as equivalent. Haunert and Sester (2008), on the other hand, use the straight skeleton to assess local polygon width but this decision is based on subsequent resolution of the conflicts. They do state some possible downfalls of both variants which are the following. Firstly, the straight skeleton has the potential danger of containing sharp reflex angles that can approximate 360 degrees. This means that the local skeleton will run more closely to the polygon boundary then would be the case for the medial axis of the same polygon. Secondly, the medial axis, on the other hand, is less suitable for generating long continuous straight lines, which might be desirable in the context of long narrow polygons such as rivers or roads. However, it could be more suitable for shorter and rounder polygons. Although Haunert and Sester do not relate these observations back to their consequences on conflict detection, they might be useful for explaining differences in the results of both methods during the empirical part of this research.

Delaunay triangulation-based approaches also appear throughout the literature on conflict detection. Before diving into how Delaunay triangulation-based approaches can be implemented to detect narrow conflicts, a brief elucidation of what they are, is warranted. As the name suggests, Delaunay triangulation is a geometric operation that can be used to connect any set of points by means of a network that is made up of triangularly connected line segments. Due to the way these triangles are constructed, the have certain properties, the most relevant one being that the Delaunay triangulation maximizes the minimum angle. This means that the triangles will not be made unnecessarily narrow.

Although the central idea behind this phenomenon is rather straightforward, a swift review of literature reveals that the concept of Delaunay triangulation can be applied in a number of different ways: classical (Ruas, 1995), constrained (Jones et al, 1995) and conforming Delaunay triangulation (Bader & Weibel, 1997; Bern & Epstein, 1992).

In the classical implementation of the Delaunay triangulation, vertices are connected in such a way that there are no other vertices inside the circumcircle of each triangle. On top of that, the network must be constructed in such a way that the differences between the three angles that make up a triangle are minimised. In other words, the minimum angle inside the triangle is maximised. This way, unnecessarily narrow triangles are avoided. Lastly, no line segments within the network are allowed to intersect.

Whereas the classical approach is solely concerned with the connection of points, the constrained version of the Delaunay triangulation also uses line segments as input. These line segments bound the triangulation by functioning as constraining edges that cannot be crossed by the triangle edges. In the case of an individual polygon, this means that the polygon boundary cannot be intersected by any triangle edges. A visual representation of this can be seen in the Figure 8 (centre), where the vertical line segment separating the square is the constraining edge.

Something to consider when implementing a constrained Delaunay triangulation is the fact that, due to the introduction of constraining edges, the Delaunay criterion of maximised minimal angles may no longer hold true. Visually this means that “skinny” triangles might be created. The conforming Delaunay triangulation mitigates this problem by allowing new vertices to be inserted along the constraining edges, which subdivides them into multiple smaller edges. This way, the Delaunay criterion can be restored (Rognant et al., 1999).

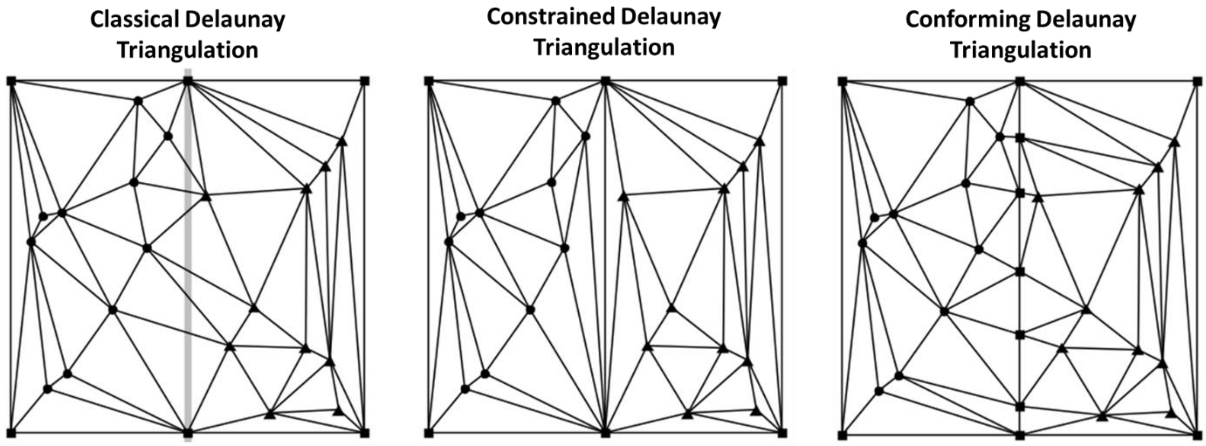


Figure 8 - Three types of Delaunay triangulation (source: Lucas, n.d.)

Delaunay triangulations are used all throughout the field of computational geometry but also features often in primary works related to the detection of narrow conflicts. Bader and Weibel (1997), in an often cited publication, use the Delaunay triangulation for this purpose in the following manner.

"Narrow sections are detected by an analysis of the width of triangles. Conflict regions are then formed by sets of (entire) triangles which have been identified as too narrow."

Although this approach is rather simple and quick to perform, it is characterised by problems similar to those regarding skeleton-based approaches. Namely, the fact that the conflict regions that are detected are demarcated by the edges of the triangles. This means that triangles in their entirety are detected as being either a narrow conflict or not. Thus, the way in which triangles are constructed directly influences the way in which conflicts are appointed. Consequentially, this approach yields a more coarse separation of conflicts areas and safe areas than was demonstrated by the buffer approaches by Gao et al. (2012) and Peter (2001). In fact, when it comes to the comparison of triangulation and buffer approaches, Bader and Weibel state that: "*...rolling ball buffers more neatly partition the polygons into regions of conflicts and 'safe areas'. This indicates that rolling ball buffers are superior for conflict detection…"*.

In closing off this subsection devoted to the detection of conflicts, the following findings have been established. All three methods have a rather straightforward way of approaching conflict detection. The skeleton-based approaches are somewhat similar to the Delaunay triangulation-based approach in the sense that both measure polygon width at discrete locations across the entire shape. Based on the distances measured at these discrete locations, conflicts areas are separated from safe areas. Furthermore, the results of both methods are directly influenced by the number, density and positioning of the vertices that make up the polygon in question. To be more specific, the higher the number of vertices, the more locations of measurement are included in the analyses. Naturally, this leads to a more detailed demarcation of narrow conflicts. The buffer-based approaches are inherently different as they rely on inward and outward facing offsets from the polygon boundary. They operate by finding the core area first and using that to detect which locations are not part of the core area, and, thus, narrow conflicts. Finally, through reviewing different publications discussing different approaches, it has become evident that buffer-based approaches create the neatest separation of conflict and non-conflict areas.

2.2.2. Conflict resolution

Once all conflicts have been established, the attention can be shifted towards resolving them. This process consists of different operations that have to do with the editing or manipulation of line segments or entire polygons. In one way, the outcomes of different resolving operations are the same, namely a feature or map without any conflicts. However, the number of ways in which this can be achieved are virtually endless. Moreover, the mere elimination of conflicts does not necessarily infer successful generalisation. In fact, polygons or parts of them may have been altered in ways that entail problems other than geometric conflicts (Peter, 2001). In fact, one could make the case that perfect polygon generalisation does not exist, merely because any simplification or alteration creates a deviation from the original polygon. This means that it provides untrue information regarding the feature it represents. Questions that an observation like this is likely to induce, are: what kinds of resolving operations can be used and what are the consequences with regard to the results they produce. Therefore, this section will touch on the more prominent ways of resolving conflicts and the results they entail. What is important to mention at this point, is that these resolving operations are not the same as the approaches that are central to this research. More precisely, the approaches are not resolving operations; they are general mechanisms that may or may not be suitable for carrying out these resolving operations. Consequentially, this subsection, alike the previous subsection on conflict detection, will be broken down into the three central approaches. Within each part, it will be elaborated to what extent the respective approach can be used to fulfil certain resolving operations.

Throughout the literature on conflict resolution, many different resolving operations are used in complement to one another. Badel and Weibel (1997) mention four resolving operations which are: elimination of small polygons, enlargement of narrow polygons, aggregation of polygons, displacement of polygons. Although all of these can be used to resolve narrow conflicts, the decision regarding which one to use is dependent on different aspects. These include the shape and size of the feature in question, the shapes and sizes of surrounding features, the topological relationship between a feature and its neighbours etcetera. This complexity demonstrates a crucial difference in perspective that is highly relevant to this research. Namely, whether one is attempting to resolve conflicts at a feature level or at a map level (which includes multiple neighbouring features). This issue is central to conflict resolution because of the fact that resolving conflicts in one feature is likely to produce new conflicts in surrounding features. Although resolving conflicts on a map wide level is outside the scope of this research, it is one of the most fundamental and challenging issues within the field of conflict resolution and vario-scale maps. Therefore, it is crucial to realise that resolving operations at a feature level is only the tip of the iceberg. Although being highly relevant, it is merely a part of a larger picture that has to be sketched out in order to successfully realise the concept of vario-scale maps. In short, this sub-section will discuss different resolving operations at a feature level of analysis. The consequences of these operations for hypothetical neighbours, although relevant from a practical perspective, will not be addressed.

Buffer-based approaches can be used effectively for the detection of narrow conflicts. When it comes to resolving them, contrarily, literature provides virtually no substantiation for the assumption that buffers can be of much use. Although Bader and Weibel (1997) use buffers for the aggregation of multiple small sections into one larger area, this resolving operation exceeds the scope of this research which limited to resolving conflicts at a polygonal level.

Skeletal-based approaches rely on centre line representations of polygonal features. The fact that they are line-like indicates that they are useful for particular resolving operations. To be more precise, different publications use skeletal variants for operations like the collapsing and splitting of features (Bundy, Jones & Furse, 1995). Haunert and Sester (2004) for instance, use the straight skeleton to partially collapse narrow features into a line. The fact that they implement partial collapse reflects that a distinction is made between narrow parts and wider parts of a polygon. Moreover, the goal of their analysis is to illustrate how the straight skeleton can be used to collapse narrow parts while simultaneously preserving core areas by representing them as polygons (Figure 9).

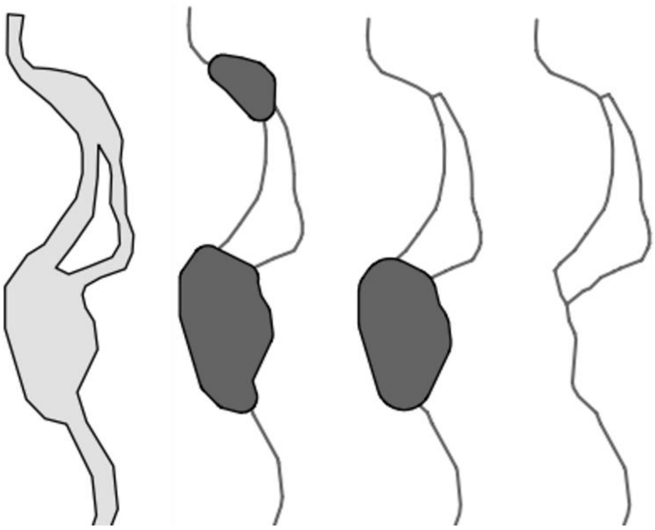


Figure 9 - Partial collapsing based on different width thresholds (source: Haunert & Sester, 2004)

This relates directly to the way in which narrow conflicts are framed in this research, namely as narrow sections that might be part of larger features. On top of that, given the fact that skeletons represent a feature as a line, they are particularly well suited to be used for narrow polygons (which are also line-like). These two characteristics point to efficacy of skeletons for the resolution of narrow conflicts.

Nevertheless, one might still wonder whether the use of different types of skeletons influences the results of any resolving operations. Comparing the most prominent ones, the straight skeleton and the medial axis, it appears that the straight skeleton is used more frequently in relevant literature (Haunert and Sester, 2004, 2008; Lewandowicz & Flisek, 2020). However, when it comes to using skeletons for the (partial) collapsing of features, a review of literature shows that different skeletal variants are used (Bader & Weibel, 1997; Johnston, Scott & Gibb, 1999; Lewandowicz & Flisek, 2020). However, most of these publications do not compare different skeletal variants and thorough justification for the use of a particular one is mostly absent. Haunert and Sester (2008), however, do evaluate different skeletal variants in light of collapsing and merging operations. Paradoxically, they deem the flexibility portrayed by triangulation-based skeletons as a disadvantage. They reason that because every application of a such a skeleton is different, the concept of a triangulation-based skeleton is ill defined and hard to reproduce consistently. On top of that they demonstrate how this skeletal variant is highly sensitive to small indentations or juts in the polygon boundary. Therefore, they state that in case of working with very detailed polygons (which have a lot of complex edges instead of straight line segments) the use of a straight skeleton should take preference.

The application of the medial axis in relation to the resolution of narrow conflicts is also mentioned in multiple different publications (Christensen, 2000; Fischer et al., 2005; Haunert and Sester, 2008; Szombara, 2013). Fisher et al. use the medial axis to create road centre lines for the generalisation of road networks. A noteworthy finding from their application is that the boundary of the input polygon needs to be comprised of a sufficient density of vertices in order to get a representative skeleton. They use vertex densification to ensure that this criteria is met. Haunert and Sester (2008) make another relevant remark by stating that the medial axis might be more suitable for less long but narrow features, while the straight skeleton is a better fit for long, line-like features such as roads and rivers.

Based on these findings, the following has been established. Skeletal variants are commonly used for the (partial) collapsing or narrow features, which makes them substantially relevant for resolving narrow conflicts at a feature level. Regarding different variants, literature indicates that different variants might be used depending on the feature characteristics such as length, width and vertex density.

Delaunay triangulation-based approaches are used by Van der Poorten & Jones (2002) to generalise polygonal outlines. They apply this approach in the following manner. By triangulating the vertices that make up polygons, the can distinguish three types of triangles (Figure 10).

1) Base triangles (surrounded by other triangles on all three sides)
2) Branch triangles (touches other triangles on two sides, one side is part of the polygon boundary)
3) Leaf triangles (only one side touches another triangle, the other sides are part of the polygon boundary)

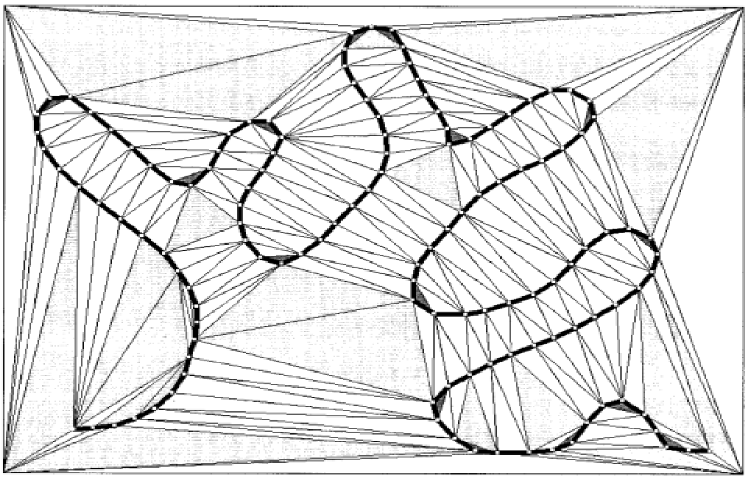


Figure 10 - Three types of triangles (source: Van der Poorten & Jones, 2002)

Van der Poorten and Jones use this framework to generalise polygons as follows. The area size of each branch (all branch and leaf triangles that stem from the same base) is calculated. This way, the smallest branches can be determined. By successively pruning away the smallest branch at its base, the shape is gradually generalised until only its main branch remains. Although this approach bears some relation to the issue of resolving narrow parts, it is not concerned with the distance between non-consecutive vertices. On the contrary, it focusses on the removal of sections based on area size rather than section width. Furthermore, even if the pruning of branches was done based on branch width, it would simply mean removing the narrow conflicts from the polygon This does not provide any further resolving abilities than just removing all detected conflicts. Therefore, this approach does not address narrow conflicts as they are framed in this research.

2.3. Vario-scale

The concept of vario-scale maps is a rather new concept, and, therefore, its full-fletched implementation might be considered a work in progress. However, given the fact that the foundational principles on which the concept relies are clearly established, any attempt at the implementation of methods or operations to a vario-scale map is subject to a number of inherent requirements. In order to get a grasp of what it is that such approaches ought to be capable of, the principles behind any vario-scale map will now be elucidated.

The creation of vario-scale maps relies on an underlying vario-scale data structure, which is generally referred to as a topological Generalized Area Partition structure (tGAP structure) (Haunert, Dilo & Van Oosterom, 2009; Suba, Meijers & Van Oosterom, 2014; Van Oosterom et al., 2014). This structure stores a 2D representation of space but adds scale as the 3rd dimension (rather than a z-value which is generally considered the 3rd dimension). This creates a space-scale cube (SSC) in which 3D objects feature as polyhedral representations of 2D polygons. This means that such a polyhedron contains the geometry of a feature at all scale levels at once. In the SSC, there are no gaps and no overlaps at any scale level, meaning that there is never no feature or multiple features at a given location at a given scale. At a map wide level, the SSC is always divided into an ever changing combination of polygonal covering. This means that, where some polygons shrink in area size as a consequence of a decrease in scale, others largen.

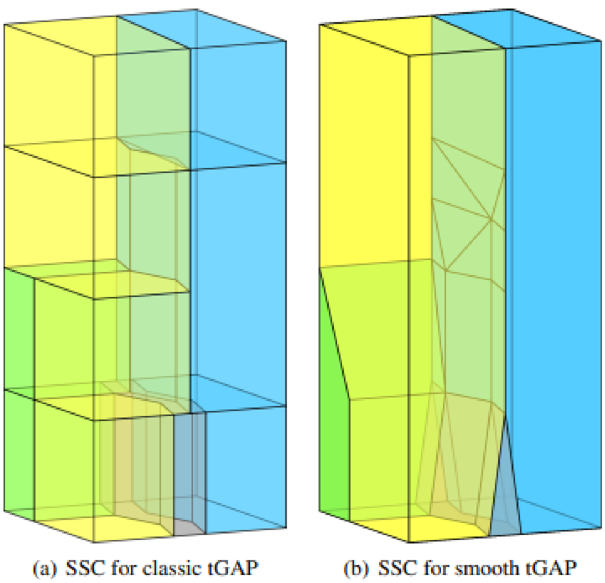


Figure 11 - Classical and smooth tGAP structures (source: Suba, Meijers, & Van Oosterom, 2014)

In order to display this tGAP structure at a particular scale, a horizontal slice of the SSC can be made. In classical digital maps, there is a fixed and limited number of slices. In vario-scale maps, however, the number of slices that can be made is infinite, as scale is a continuous phenomenon. Consequentially, zooming operations are characterised by less abrupt changes and a more gradual zooming process. Van Oosterom and Meijers (2011) even take this concept of smooth zooming one step further by developing the concept of a “smooth tGAP” structure. In this structure, sections that are added to surrounding features during aggregation are removed in a gradual manner. This entails that features, or parts of them, can never disappear into another feature abruptly. Instead they continuously shrink into a surrounding feature until they are fully removed. The result is a data structure without any horizontal faces and that does not produce any abrupt changes when zooming in or out. On top of that, the diagonal faces that separate the features ought to be free from any holes and intersections as this would result in problematic slices. This distinction between tGAP and smooth tGAP is crucial as it directly affects the ways in which features and aggregation operations are handled during the creation of the respective data structures.

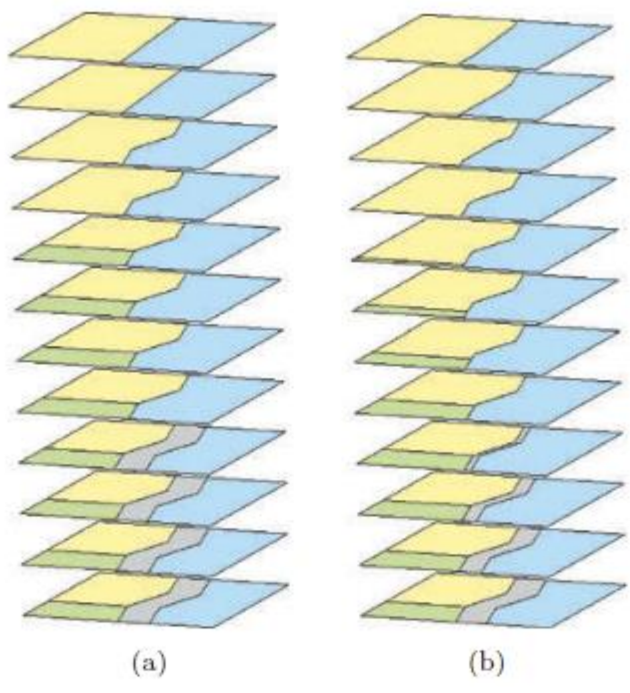


Figure 12 - Slices made from a classical tGAP SSC and a smooth tGAP SSC (source: Oosterom and Meijers, 2014)

2.3.1. Consequences for conflict detection

Now that the fundamental principle behind vario-scale maps and a tGAP data structure have been established, questions arise as to how this can be realised. In the ideal world, you would have the representation of a feature vary continuously in relation to the scale level. This would mean that, given any scale level, a corresponding representation of a feature without geometric conflicts is calculated. The realisation of this idea, however, is far from simple. Firstly because the creation of a satisfactory method is extraordinarily complex. On top of that, the computational demands it would entail would be tremendous. Therefore, a logical approach to creating a continuously changing representation of a feature is to calculate a set of representations at specific scale and to connect these into a 3D structure. This would mean that conflicts would have to be detected for every "step" in this model. In other words, conflicts are detected based on a given scale and a target scale. This process , then, is repeated up until the maximum scale level is reached. This reduces the issue of continuous conflict detection to multiple rounds of conflict detection. From a practical perspective, this entails that the methods for detection, as they have been addressed thus far, are still as viable as they were before the introduction of vario-scale requirements. However, given the nature of the smooth tGAP data structure, the resolution of conflicts in between these steps still has to be realised in a gradual manner.

2.3.2. Consequences for conflict resolution

In an attempt to frame the complex problem that results from the desire to resolve conflicts within the context of a vario-scale map, it is crucial to first break down the problem into its various challenges and decisions. The first distinction that has to be made is whether they have to be resolved at a polygonal level or on a map level, meaning the resolution has to take into account features that surround the feature containing the conflict(s) (Bader & Weibel, 1997; Peter, 2001). The latter option, naturally, takes preference over the former, since it takes into account issues that might occur in neighbouring features. Realising this when going from a specific scale level to a target scale level seems to be a somewhat straightforward operation. However, when projecting such a desire onto a vario-scale case (tGAP structured object) it becomes terribly more complex. This is the case because a tGAP structured map, ironically, does not allow for gaps in the maps coverage. Namely, every location must be covered by exactly one feature at a specific scale level (Van Oosterom & Meijers, 2011). This entails that, when increasing or decreasing the scale, the shrinking of one feature necessitates the growing of its neighbour, that has to occupy the empty space. Although this is a fundamental issue within the context of vario-scale maps, resolving conflicts at a map level is outside the scope of this research. Therefore, the consequences of resolving operations on surrounding features will be disregarded, despite their importance.

However, even at a polygonal level, the resolution of conflicts is not as straightforward as one might be tempted to believe. The introduction of the desire to incorporate any approach to conflict resolution in a vario-scale context, means that the conflicts cannot simply be resolved in an abrupt manner. Rather, the feature with the conflict(s) has to be gradually transformed into a derivative without any conflicts. How this might be achieved and what the consequences of different approaches might be is not easily determined. A logical way out of this problem could be to simply remove the conflicts from the initial polygon and to subsequently connect the two shapes in a way that creates a tGAP structured object. Although this might seem like an easy fix for the problem of gradual conflict resolution, the best way to transform one complex shape into another is by no means self-evident (Goldstein & Gotsman, 1995; Gotsman & Surazhsky, 2001; Guibas & Hershberger, 1994).

Moreover, simply removing narrow conflicts, though be it in a gradual manner, is likely to result in problems at a map wide level. Namely, the act of removing a conflict in one feature means that this space has to be occupied by a neighbouring feature. This is not necessarily an optimal solution as it could alter the map in ways that are problematic with regard to the information that the features represent.

Therefore, looking ahead to the creation of a conflict-free tGAP data structure from a 2D polygon, the following requirements must be taken into account. Firstly, the sudden resolution of conflicts, be it through removal, collapsing or any other resolving operation, would produce abrupt changes in the features geometry. In order to address this, the approaches should demonstrate a way of gradually implementing these operation, or at least smoothing out the changes in geometry. For example, if a narrow conflict should be collapsed, it should gradually shrink into a line segment. On top of that, features in a tGAP data structure can come in any shape or form, which means that the approaches should be able to resolve conflicts without creating problems in other parts of the polygon. Lastly, due to the fact that no holes or intersections can be present in a tGAP data structure, the approaches should create 3D polyhedral representations of 2D polygonal features that are free of either of those. Practically this would mean that the 3D object has no holes or intersections at any scale (z-axis) as this would lead to a hole or intersection in the slice at the respective scale level.

2.4. Buffer-based approach

Previous parts of the theoretical framework have established that buffer-based approaches are predominantly suited for the detection of narrow conflicts. On top of that, section 2.3 outlined that the process of conflict detection essentially does not change as a consequence of the introduction of vario-scale requirements, since it is broken down in consecutive steps. Because of this, what has been stated regarding buffer-based approaches in section 2.2.1 is still relevant. The only important additional requirement, thus, is that the approach should be capable of repeatedly detecting conflicts across a range of scales. This should result in the production of consecutive representations of an input feature as scale is increased. Looking back on the inward-outward-buffering approach by Gao et al. (2012), it becomes clear that by increasing the minimum width value simultaneously with increasing the scale, different representations of the input feature can be created.

2.5. Skeleton-based approach

Given the fact that conflict detection and conflict resolution, will be framed as a step by step process rather than being continuous, what has been established regarding conflict detection is still relevant after the introduction of a vario-scale context. Namely, consecutively detecting conflicts across a range of different scales is essentially equivalent to detecting conflicts at a given scale multiple times in a row. On top of that, it has been established that buffer-based approaches are better suited for the detection of conflicts than skeleton-based approaches. However, when it comes to resolving conflicts, the addition of requirements imposed by the nature of a tGAP data structure does bear consequences. To be precise, conflicts have to be resolved in a way that does not produce any abrupt changes in geometry. Therefore, this section will focus on assessing to what extent a skeleton-based approach can be used to do so.

Because the concepts of vario-scale maps and tGAP data structures are still in their infancy, the primary academic works are from the hand of a rather small number of authors. Therefore, the different approaches that are central to this research have not been used extensively within that specific context. Nevertheless, the geometric phenomena on which the approaches are based have been researched elaborately in other contexts within the field of computational geometry. For example, skeletal variants have been used repeatedly for the generation of roofs from polygonal footprints Held & Palfrader, 2017; Sugihara, 2019). To be more specific, given the way in which it is created, the straight skeleton generated from a building footprint perfectly represents the central ridge of the roof of that building (Held & Palfrader, 2017).

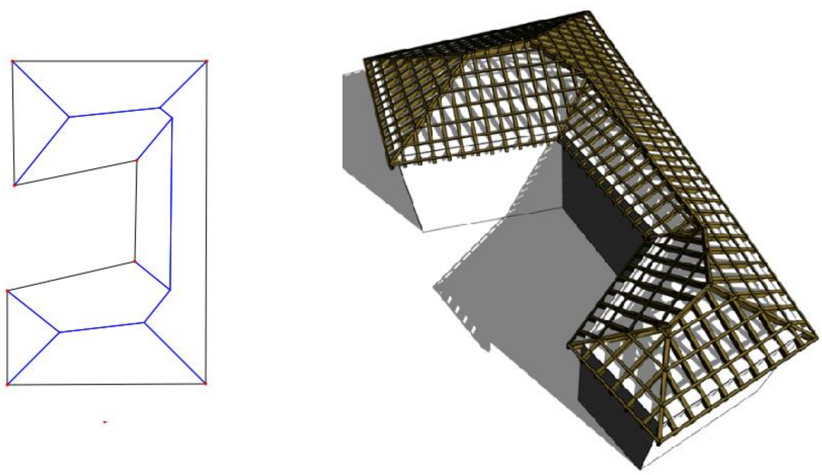


Figure 13 - Straight skeleton for roof generation (source: Geometry Factory, 2010)

This quality of the straight skeleton can be projected onto the issue of conflict resolution for a tGAP structure, in which the goal is also to create a representational 3D object from a 2D input polygon. In this model, the skeleton acts as a line of disappearance. This can be related back to the use of skeletal variants for the collapsing of narrow polygons (Haunert & Sester, 2008). By combining these operations, skeletons could be used to create a way of gradually collapsing polygons. Furthermore, Kuipers et al. (2020) demonstrate that the object that results from connecting an input shape to its skeleton can subsequently be sliced horizontally in order to obtain intermediate 2D representations of this shape. This process is visualised in Figure 14 and relates directly to the way 2D slices are made from a tGAP data structure in order to display the map at a given scale.

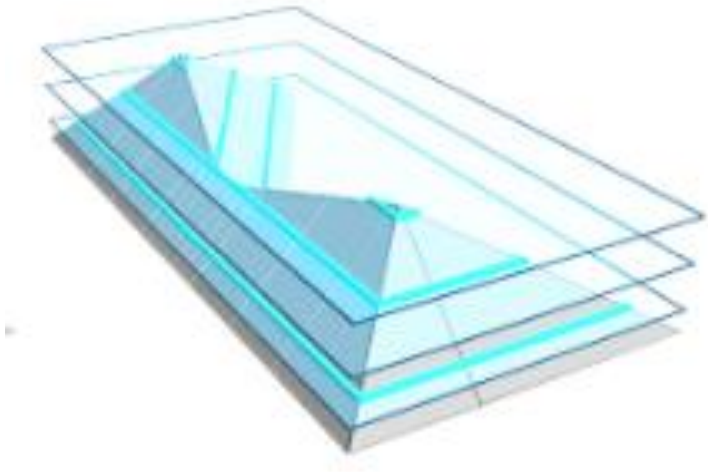


Figure 14 - Slicing of a 3D object created using the skeleton of a 2D shape (source: Kuipers et al., 2020)

However, despite the relevancy of these approaches to the issue of gradually resolving conflicts, a major challenge remains. To be specific, it is not self-evident how the input shape (or footprints in these example) are to be connected to their derived skeletons. For example, this process might be somewhat straightforward for buildings but is likely to become tremendously more challenging when the input shape is far more complex. The issue of gradually transforming one polygonal shape into another is referred to as polygon morphing. Within the field of polygon morphing, there are, broadly speaking, two main ways of approaching the problem of connecting shapes. The first one focusses on connecting the vertices that make up the first polygon to vertices in the second polygon, creating a smooth transition. The second approach relies on a line-based approach, which treats the outline of a polygon as a line rather than a set of vertices. This way, more instinctive generalization can be applied. However, the problem with this approach is that it is aimed at producing a given number of intermediate representations that gradually change from the original polygon into the target polygon. Given the context of this research, however, the desire is to obtain a 3D object from which an infinite number of intermediate representations can be derived. Therefore, the vertex-based approach fits more neatly with the goal of this research. Nevertheless, a vertex-based approach still entails a large number of possible connections between the vertices of the bottom and top feature. However, it turns out that the Delaunay triangulation is one of these ways.

2.6. Delaunay triangulation-based approach

As a consequence of the decision to break conflict detection down into a step by step process, the use of triangulation-based approaches in this regard does not change. Therefore, the issue that remains is to determine how Delaunay triangulation can be used for the gradual resolution of narrow conflicts. This means discussing how different representations of a given shape can be connected in a way that is true to the smooth tGAP principles. As it turns out, the relevant literature features a number of publications that discuss how Delaunay triangulation-based approaches can address this issue.

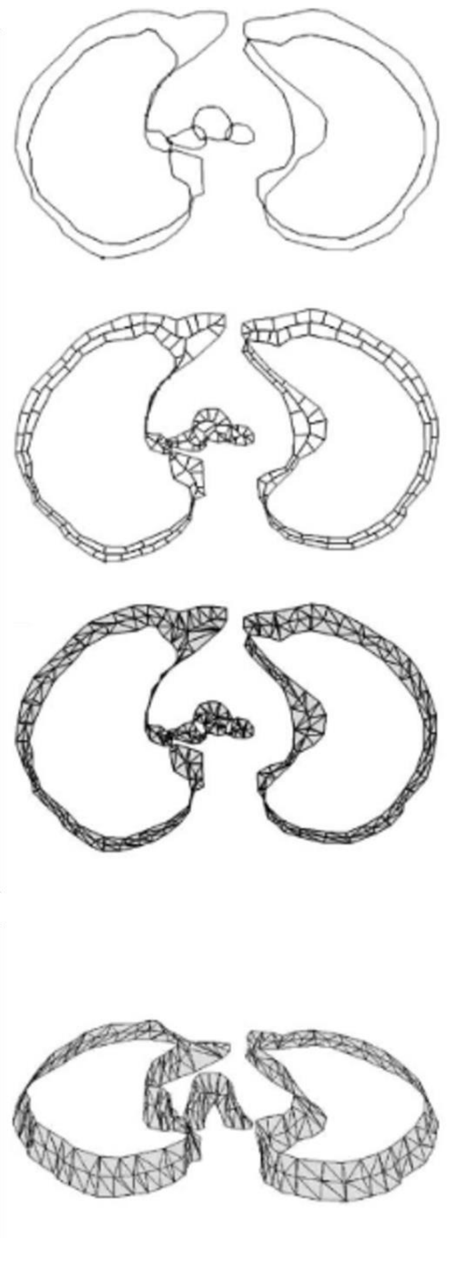


Figure 15 - Connecting two slices through triangulation (source: Barequet et al., 2004)

One way is related to the phenomenon of Triangulated Irregular Networks (TINs). This term refers to a triangulation application that is used to connect points (often located at different elevations) in a network of triangles. This operation creates a surface from a set of points, which can be used to perform different analyses then could be done with the initial set of points (Ali & Mehrabian, 2009). The qualities of a TIN are such that they are relevant to the issue of conflict resolution as it is framed in this research. To be more specific, the creation of a TIN from a set of points results in a surface that connects vertices at different elevations. This surface has a complete cover of the area in question (meaning no holes) and is free of any intersecting line segments. These characteristics match precisely with the requirements of a tGAP data structure.

However, connecting vertices is not necessarily the same thing as connecting layers, even though these layers consist of vertices. Barequet et al. (2004) discuss an implementation of a Delaunay triangulation for the connection of multiple horizontal slices (CT scans). By connecting these layers, they recreate the 3D object from which the slices were obtained. Their approach is as follows. First, two slices are overlayed (on the same elevation). From this overlay, it can be determined for every area of the overlay whether it belongs to the first slice, second slice, both or neither. Then, the locations that are only covered by one of the slices are selected. Subsequently, the boundaries of these areas are connected by means of triangulation. Lastly, the triangulation is lifted upwards to represent the slices at their original elevations. By applying this process to multiple layers, they can be turned into a 3D object.

Although this approach is highly relevant to conflict resolution, the slices in Barequet et al. (2004) all have geometries that do not diverge a substantial amount in 2D terms. The layers that result from different rounds of conflict detection, on the other hand, can vary tremendously in terms of relative shape. Moreover, it is not in any way self-evident that the Delaunay triangulation can be used to connect such deviating layers.

A possible solution to this issue might be to 'guide' the triangulation in some way. This idea is described by Barequet and Sharir (1994), Barequet et al. (2004) and Boissonnat and Geiger (1993), who all use a skeleton in order to make the connection between different layers happen more smoothly. In both of these articles, the skeleton functions as a bridge or series of points for the triangulation to grapple onto. The meaningful combination of these two phenomena speaks to the complementariness of the skeleton- and triangulation-based approaches. In fact, looking back on previous findings, this relation becomes even stronger. Namely, it was ascertained that implementing skeletons for the (partial) collapsing of features cannot be done smoothly without some way of connecting the input shape to its local skeleton. The efficacy of the Delaunay triangulation when it comes to connecting different layers, might be able to fulfil this demand.

Another noteworthy finding that can be made from the three papers on connecting CT scans relates to the type of Delaunay triangulation that was applied. Although none of these papers mention this explicitly, it can be inferred from their studies that all of them applied either a constrained or conforming Delaunay triangulation. The reason being that all of the triangle edges begin or end at a vertex on the outline of either layer and no edges cross them. On top of that, in concave edges, no triangulation outside of the polygon appears, which would be the case with a classical Delaunay triangulation.

2.7. Theoretical framework summarised

Given the length of this chapter and the multiplicity of concepts that are central to it, a concise summary is warranted for the sake of overview. Therefore, this section will feature such a summary by going over the main findings once more.

First of all, the terminology around narrow conflicts, in relation to other types of geometric conflicts was elucidated. In short, the term narrow conflict refers to parts of polygons that are too narrow relative to a given minimum width value. These are problematic even in classical fixed scale maps and occur continuously in vario-scale maps. Therefore, continuous detection and resolution of these conflicts is desired. However, this desire cannot be fulfilled due to the fact that the scale of a map has to be fixed in order to measure distances. As a consequence, both the processes of detection and resolution have to be broken up into consecutive steps. The results of these steps then have to be connected in order to create an object that can be sliced at any scale. Furthermore, the nature of a smooth tGAP data structure demands that such objects are free of any horizontal faces, intersections or gaps in outer faces of the object (Van Oosterom & Meijers, 2014). These requirements, according to the relevant literature, can be addressed by the central approaches in the following ways.

The buffer-based approaches are said to be best suited for the detection of conflicts. To be more precise, the relevant literature suggests that the inward-outward-buffering approach put forth by Gao et al. (2012) is most complete when it comes to detecting all three types of narrow conflicts. By changing the minimum width criteria, conflicts can be detected at different scales and in a consecutive manner. These resulting shapes then have to be connected to one another in a gradual manner in order to resolve the conflicts. The skeleton-based approaches can address this issue by providing a centre line to which the input shapes can connect (Held & Palfrader, 2017; Sugihara, 2019). This creates a gradual collapsing of the narrow conflicts to its skeleton. The Delaunay triangulation-based approach addresses the issue of conflict resolution as follows. By connecting the vertices of the top and bottom features in a network of triangles, a surface is created that gradually connects all vertices (Barequet and Sharir (1994), Barequet et al. (2004) and Boissonnat). This can be used to connect the input shape to its skeleton, but also to connect the bottom layer to the top layer if the top layer is not a skeleton but a core area.

Overall, the different approaches display characteristics that are relevant to conflict detection and resolution in the context of vario-scale maps. On top of that, they have strengths and weaknesses but also seem to possess complementary qualities.

3. Methodology

Since the theoretical underpinnings of the concepts that are central to this paper have been established, the attention can be shifted towards the empirical part of this research. This part is concerned with what is to be done (and how) in order to answer the questions that initiated this research. Therefore, this chapter will outline the operationalisation of the central concepts all the way down to the concrete steps that will be taken with regard to empirical analyses. This broad-to-specific nature is reflected in the structure of this chapter, which is comprised of the following sections. Firstly, the general methodological approach and the analyses that are part of this will be discussed. This section will also include a discussion of the data and software that were used for across the different analyses. Subsequently, the concrete operationalisation of this approach will detail how the analysis will be carried out and how the concepts themselves will be made measurable. This last section will feature a step-by-step description of how the analyses and calculations were carried out.

3.1. General methods

This section will outline what methods will be used in order to obtain answers to the central research question. This will be done by discussing the overall approaches, why these are justifiable and how they serve to address the central issue. As the introduction and the conceptual model that featured in it made clear, the general principle behind this research' methodology is to perform a comparative study on three approaches. This research started off with an exploration of literature, which is to be followed by the application of these approaches to different test cases. Because of the fact that multiple approaches will be analysed on different aspects, it is especially important to maintain a clear overview of what will be done. Therefore, a summary of which approaches will be tested on which aspects related to narrow parts can be found in Table 5.

Table 5 – Overview of the analyses that will be performed

	Conflict detection	**Conflict resolution**
Buffer-based approach	Yes	No
Skeleton-based approach	Yes	Yes
Delaunay triangulation-based approach	Yes	Yes
Combinational approach	Yes	Yes

Three important remarks have to be made with regard to Table 6. Firstly, one might wonder why the buffer-based approach will not be tested on its usability for conflict resolution. The reason for this stems from the fact that there is no literary substantiation for the assumption that a buffer-based approach can even be used to resolve conflicts.

Secondly, it might be questioned why the skeleton- and triangulation-based approaches will be tested on conflicts resolution. Namely, buffer based approaches were said to be more accurate and effective in partitioning polygonal shapes into core areas and narrow conflicts. The reason for this is the fact that both methods have been used repeatedly to detect narrow conflicts in an effective manner. Given this abundance of usage, it is warranted to assess the efficacy of both approaches with regard to the aspect of conflicts detection.

Thirdly, Table 6 also contains a combinational approach. The reason for the addition of this approach is twofold. Firstly, the theoretical framework has pointed out that different methods have different strengths and weaknesses when it comes to different aspects of handling narrow parts. Therefore, a combinational approach might be able to harness these strength, which could make up for the weaknesses. On top of this, to answer the central question of this research thoroughly, it is warranted to fully explore the efficacy of the different methods. Carrying out the analyses in this way will allow for meaningful conclusions to be made regarding all approaches, both in isolation and in combination.

3.1.1. Data

The availability and quality of data are two of the more prominent factors influencing the results of most GIS-related researches. However, this research does not focus on either a specific location or a certain period of time. Therefore, the quest for usable data is certainly not a problematic one. On top of that, finding a dataset of a particular quality does not promise to be a challenge either, as the only requirement for this research is that it contains polygon features. Something that is important, however, is the shape of the testing polygons that will be extracted out of the dataset. The reason is that the approaches that will be subject to experimentation might perform satisfactorily when applied to simple shapes but less so for shapes that are more complex. The words simple and complex, in this case, refer to the presence of multiple narrow parts and awkward shapes in the polygon's geometry. To exemplify this, one might think of a square piece of farm land as a simple shape and a river, with lots of little meandering arms extending from it, as a more complex shape. The selection of a sufficiently complex polygon is essential for thoroughly testing the functioning of the methods in question. Moreover, one might argue that using one testing polygon is problematic, merely because every shape is different. Consequentially, they could pose different challenges when it comes to the detection and resolution of narrow conflicts. In order to avert this problem, the methods will be applied to three different polygons. Figures showing the shapes of these polygons can be found in appendix 8.1. They have been selected to contain different phenomena related to narrow parts. These are size conflicts (where the entire polygon is too small to be displayed), narrow-corridor and narrow-jut conflicts but also polygons with large core areas and narrow conflicts. This diversity in the testing shapes is necessary in order to be able to draw meaningful conclusions, as features in vario-scale maps can also come in all shapes and sizes imaginable.

3.1.2. Software

The selection of the appropriate software packages is also of consequence to the way in which the research is carried out and therefore to the results. On top of that, it is necessary to describe the software that was used for the sake of replicability. To be brief, all analyses and experiments were done using either ArcGIS Pro or in Python. ArcGIS Pro was used primarily to visualise analyses and their results while most of the actual implementation of the different approaches was done in different python scripts. A detailed step by step description of these scripts is provided in section 3.3 and the full scripts can be found in the appendices. Apart from common Python libraries such as NumPy and Pandas, the more specific libraries that were used were the following.

Table 6 – Used Python libraries

Library	Used for
Matplotlib.tri	Classical Delaunay triangulation
Tri	Constrained Delaunay triangulation
Trimesh	Mesh creation and slicing cross sections
Trimesh	Calculating medial axis points
Shapely	For storing point, line and polygon features

3.2. Operationalisation

The previous section stated the array of analyses that will be performed in the practical part of this research. What remains to be done, however, is outline how these analyses will be carried out and how the approaches will be operationalised. For example, a number of different triangulation-based approaches have featured in the theoretical framework. Which of these will be implemented? A general overview of how the different approaches will be filled in can be found in Table 7. These decisions as to which specific approaches are to be implemented are based on the findings obtained throughout the theoretical framework.

Table 7 – Concretised overview of the analyses that will be performed

	Conflict detection	**Conflict resolution**
Buffer-based approach	Inward-Outward-buffering approach	Not applicable
Skeleton-based approach	Medial axis	Medial axis
Delaunay triangulation-based approach	Classical Delaunay triangulation	Constrained Delaunay triangulation
Combinational approach	Inward-Outward-buffering approach	Medial axis & Constrained Delaunay

Something that is worth mentioning with regard to Table 7 is the fact that the medial axis chosen over the straight skeleton. This is based on the discussed literature, which pointed out that the medial axis is likely to fit better with rounder shapes, while the straight skeleton is more suited for more sharply angled shapes. The buffer-based approach that is used, naturally, produces such round shapes.

Although the methodological structure of this research has now been elucidated, what has yet to be provided is a concrete description of exactly how the approaches mentioned above are to be implemented. Therefore, the remainder of this section will feature a step by step description of this process.

3.2.1. Buffer-based approach

The buffer-based approach was only used to detect narrow conflicts, as the review of literature provided no reason to assume that it can be used to resolve narrow conflicts. To be precise, the type of buffer approach that will be used is the inward-outward-buffering approach proposed by Gao et al. (2012). This method was implemented to three testing polygons in order to locate narrow conflicts. This was done in ArcGIS Pro since this software allows for quick and easy creation and manipulation of buffers.

The first step in this process was creating an inward facing buffer from the input polygons with a size that is half the minimum width value. This value was chosen arbitrarily for every testing polygon as the goal of this analysis is not to find actual narrow conflicts given a certain scale. Rather the aim is to determine what the quality of the inward-outward-buffering approach is in relation to conflict detection. A properly substantiated value should be used in case this method was to be used in the actual detection of conflicts in a real map.

The second step was creating an outward facing buffer from the shape that resulted from the inward facing buffer. This buffer had the same size as the inward facing buffer did. By subtracting the resulting shape from the initial testing polygon, the narrow conflict areas remain. These resulting shape represents the core area of the input polygon. By applying the inward-outward-buffering approach multiple times with increasing buffer sizes, consecutive versions of the input shape can be created.

3.2.2. Skeleton-based approach

Conflict detection

The Medial Axis was implemented to detect narrow conflicts using a python script, which can be found in Appendix 2. First, the testing shapes were imported, after which the coordinates of their vertices were extracted from the shapefiles. Using these vertices the medial axis was calculated with the medial axis function from the trimesh module. Based on the coordinates of the vertices of both the input shape and the vertices of the medial axis, distances between them could be calculated using a simple Euclidean distance calculation. These distances could then be assessed against a user defined minimum width value. At locations where this minimum width value was not met, the local vertex was selected in order to determine where the narrow conflicts were located.

Conflict resolution

The medial axis was also analysed on its usability regarding the resolution of narrow conflicts. As was the case with the detection analysis, the resolution analysis was once again done using a python script. This code can be found in Appendix 3.

Also in this code, the shapes were imported and the medial axis was generated using the trimesh module. Given the fact that the goal of this analysis was to assess to what extent skeletons can be used to resolve conflicts in a vario-scale context, the input shapes had to be turned into 3D polyhedral objects in which the narrow conflicts were gradually eliminated. In this 3D object, the third dimension is scale. This means that the at $z = 0$ should represent the initial testing polygons and at $z = max$ should be its conflict-free representation. The exact value of $z(max)$ is not of relevance to this analysis as this value would depend on the scale ranges used in the actual tGAP data structure. For the sake of this analysis it suffices to use an arbitrary value for $z(max)$ as this value does not influence the usability of this particular approach.

In order to create the desired 3D object, the vertices of the testing polygons were given a z-coordinate of 0 while the skeleton was given a z-coordinate of 10. The second step was to connect these “layers” in a gradual manner without creating holes or intersections. This was done using a Delaunay triangulation. From this triangulation, the faces could be extracted and combined into a mesh using the mesh function from the trimesh module. This mesh represents the 3D object containing all representations of the testing polygon from its initial geometry all the way to its conflict-free geometry.

3.2.3. Delaunay triangulation-based approach

Conflict detection

The analysis of the efficacy of a Delaunay triangulation-based approach for detecting narrow conflicts was performed in ArcGIS Pro. The first step was to extract the vertices from the testing polygons. These vertices were then used as input for the Delaunay triangulation. The lengths of the edges of the triangles that were created through this operation could then be measured and compared against a minimum width value. Edges that violated this criteria were detected as conflict edges. By selecting the areas between conflict edges, narrow conflict areas could be detected. The Delaunay triangulation that was used was a constrained one, in order to avoid line segments crossing the polygon outline.

Conflict resolution

Regarding conflict resolution, the Delaunay triangulation was implemented in the following manner. In ArcGIS Pro, two representations of the testing shapes were places on top of one another like they would be in a SSC. Then, the vertices of both layers were extracted from the polygons. These points were used as input for the Delaunay triangulation. The edges from the triangulation were then also displayed in 3D so that they attached to the polygons at their respective z-levels. The Delaunay triangulation that was applied was a constrained one, in order to avoid line segments crossing the outlines of either one of the polygons.

3.2.4. Combinational approach

In this final approach, the three approaches that have been discussed so far were combined in order to determine to what extent they can be effective when it comes to narrow conflicts when implemented complementarily.

Conflict detection

For the detection of narrow conflicts, the inward-outward-buffering approach was found to be very effective. Therefore, this approach will be used to create multiple conflict-free representations of the testing shapes across a range of scale levels. This was done in the manner that was described in subsection 3.3.1. After the different iterations of the testing shapes were created, they were imported into a python script for conflict resolution.

Conflict resolution

The first step in this script consisted of turning the shapefiles into shapely Polygons. From these Polygons, the vertices were extracted and stored in two lists, one for the bottom shape and one for the top shape. Then, the medial axis was calculated for the bottom shape, which can be a MultiPolygon consisting of multiple separate Polygons. Only the medial axis points that were outside of the top shape were selected. This was done in order to ensure that the medial axis points were only used as points of disappearance when there was no top shape present at a particular location. This process resulted in a third list, namely, a list containing the selected skeleton points. Afterwards, the option was included to create centroids for small islands that had to be completely removed by the resolving operation. The reason for the addition of this option is that, for small islands (small number of vertices), it might make more sense to have them disappear into a central point, rather than into a line segment.

These three lists, or optionally four, were all given a z-value, in order to create the eventual 3D object. The bottom vertices got a z-value of 0 and the other points got an arbitrary z-value of 10. These lists were combined into one list, which was then used as input for the creation of the 3D triangulation. This triangulation resulted in a list with triangles, based on vertex indices. By matching these triangles with their original coordinates, they could be turned into a mesh that represents the 3D object. The resulting object was sliced at different z-values between 0 and 10 in order to determine how the shape progresses across a range of scale levels.

4. Results

The previous chapter detailed which analyses have been carried out and in what order. In that same order, this chapter will provide an elaboration of the findings that have resulted from these analyses. This will be done in concert with visual images that were obtained through visualising the data that the analyses produced. The relevant findings will be related back to the findings that resulted from the theoretical discussion in order to frame to what extent these are in correspondence. In doing so, this chapter builds toward the conclusion of this research.

4.1. Buffer-based approach

In order to assess the efficacy of buffer-based approaches conflict detection, the inward-outward-buffering approach (Gao et al., 2012) was performed on the three testing shapes. Figures 16, 17 and 18 contain the results of this analysis which show what areas were designated as narrow conflicts (red). What can be observed is that all three types of narrow conflicts are present in this shape. To be more specific, the buffering approach produced mainly narrow-jut conflicts, a few narrow-corridor conflicts, and a couple of jut-like bends that are the result of sharp corners in the input shapes. Looking at Figure 15 specifically, it can be observed how three narrow-corridor conflicts were detected. This finding, however, is in conflict with a statement from Gao et al. (2012) who stated that the number of detected corridor-conflicts will be equal to the number of polygons after the inward-buffer minus the number of polygons after the outward buffer. Applying this principle to the results in Figure 15, there should be zero narrow-corridor conflicts, as the number of polygons is three after both the first and the second buffer (3 - 3 = 0). The number of actual narrow-conflicts in this case, however, is four. This indicates that the statement by Gao et al. is merely based on a coincidental result from their testing polygon and not a definitive rule of thumb. Nevertheless, their approach was effective at detecting all three types of narrow conflicts, which is of primary importance.

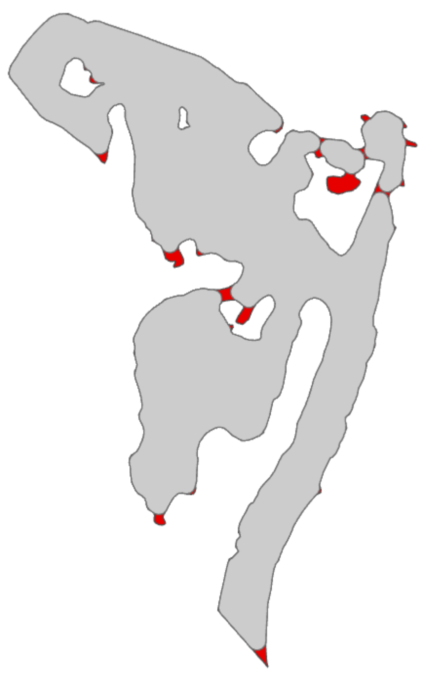


Figure 16 - Conflicts detected in shape 1

Figure 17 - Conflicts detected in shape 2

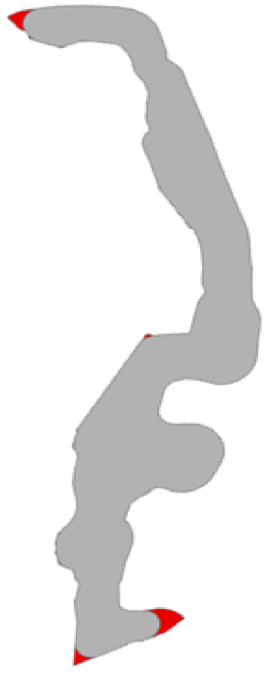


Figure 18 - Conflicts detected in shape 3

The inward-outward-buffering approach can be applied once, in order to detect conflicts at a given scale. However, in the context of a vario-scale map, the detection of narrow conflicts is an ongoing process. This process, as has been established, has to be broken down into consecutive steps, since truly continuous detection cannot be achieved. Therefore, the inward-outward-buffering approach should be capable of repeatedly detecting narrow conflicts. Figures 19 and 20 contain the results of consecutive rounds of detection. The layers in these figures represent different versions of the input polygons across a scale range. The layers are obtained by gradually increasing the minimum width values. By displaying the layers at different levels of elevation, it becomes clear how the layers can be used as points of reference for the creation of a 3D object. These shapes could be turned into solid objects by extruding each layer upwards until it meets the layer above. This would create an object that meets the requirements of a classical tGAP data structure. To generate a smooth tGAP adequate object, however, the layers would have to be connected in a gradual manner.

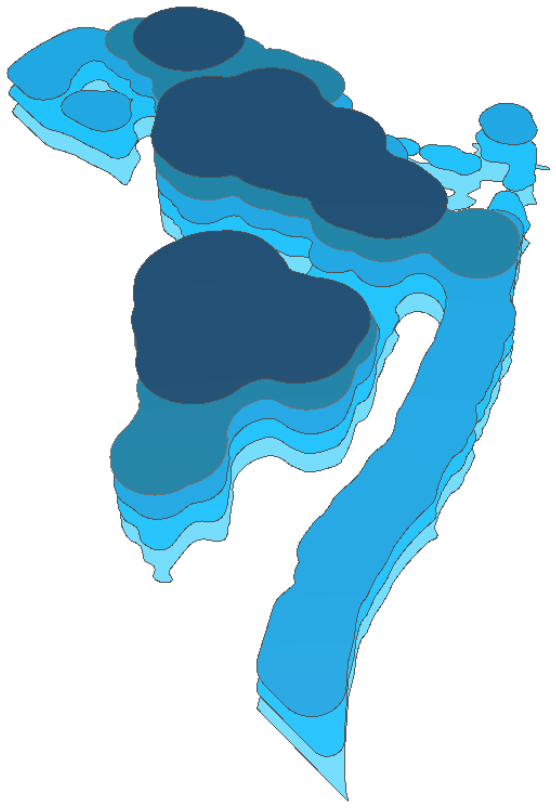


Figure 19 - Multiple representations of shape 1 in 3D

Figure 20 - Multiple representations of shape 2 in 3D

The consecutive layers illustrate how the initial polygons progress towards polygons that represent core areas based on an increasing minimum width criteria. Although this is what the buffer approach was selected for, it can be debated to what extent the upper layers satisfactorily represent the input features at a larger scale. For instance, shape 1 has lost more than half its area size over the course of 5 rounds of conflict detection. Shape 2 on the other hand has maintained almost all of its area size, due to the fact that it contained less narrow sections. This difference illustrates the one-sidedness that results from purely focussing on removing narrow conflicts. In fact, it might make more sense to enlarge some narrow sections rather than remove them. This underlines the complexity of the issue of narrow conflicts as well as the tremendous variety in possible resolving operations.

4.2. Skeleton-based approach

Conflict detection

Next to the buffer-based approach, a skeleton-based approach was also applied to the issue of detecting narrow conflicts. A visualisation of the results that this analysis garnered can be found in Figure 21. In this image, the red dots represent conflict vertices, the green dots represent the safe vertices and the black dots are the points that make up the medial axis. The first observation that can be made from these results is that the vertices that make up sections that visually appear narrow are predominantly conflict vertices. Next to that, when comparing the locations of the narrow conflicts to the ones that were detected by the buffer-based approach (Figure 22), it becomes clear that approximately the same locations have been detected as narrow conflicts. These results indicate that the skeleton-based approach produces a reliable distinction between narrow conflicts and core areas. However, when it comes to the way in which these areas are specifically separated, the results of the skeleton-based approach are significantly less neat. To be more precise, in some sections that are made up of mostly conflict vertices, there are also a small number of safe vertices scattered throughout. This makes it difficult to derive concrete conflict areas from the conflict vertices. This is problematic due to the fact that a precise and accurate distinction between conflict and non-conflict is required in order to determine the core area.

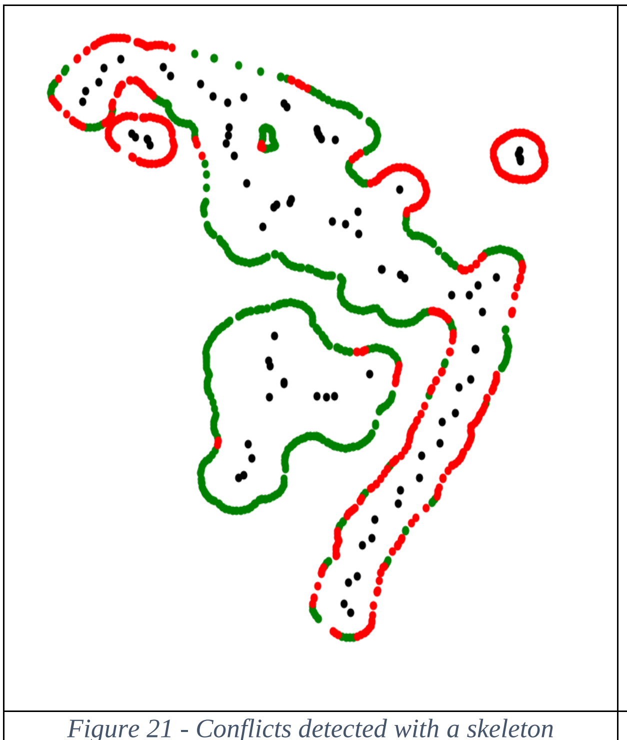


Figure 21 - Conflicts detected with a skeleton

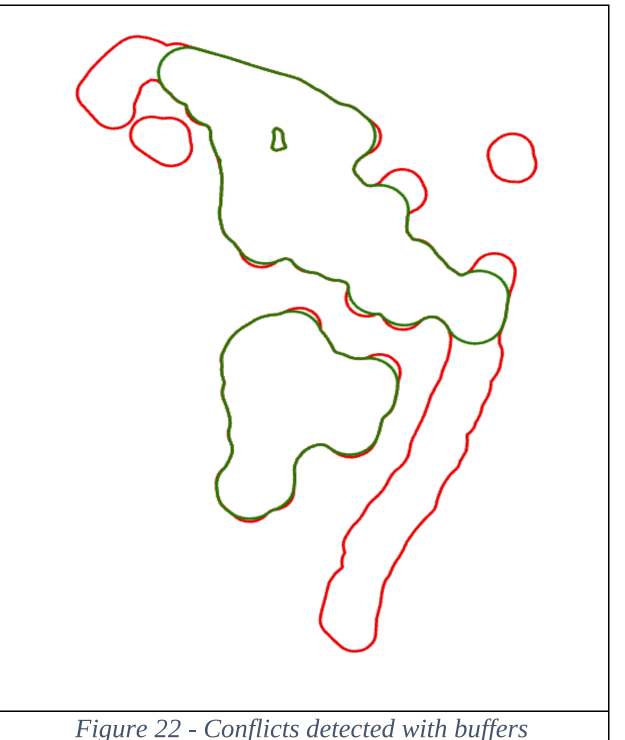


Figure 22 - Conflicts detected with buffers

Conflict resolution

A skeleton-based approach was also used for the resolution of the detected conflicts. Namely, it was used to collapse the polygons to a line. Figures 23, 24 and 25 show the 3D objects that were created by connecting the input shapes to their medial axis points by means of triangulation. There are a few noteworthy observations that can be made from these objects. Firstly, due to the fact that the medial axis is not represented as a line, but rather as a set of points, the polygons are not fully collapsed into a line. Instead, some core areas (plateaus at z(max)) are created at locations where internal triangulation occurs between points of the medial axis. However, if one were to use a line representation of the medial axis or a straight skeleton, the polygons would fully collapse into line features. This distinction provides some additional parameterisation, which might be useful in light of the creation of a tGAP data structure. Nevertheless, the creation of these core areas through internal triangulation of the medial axis points is rather arbitrary and hard to control. Therefore, a more deliberate way of preserving core areas is called for.

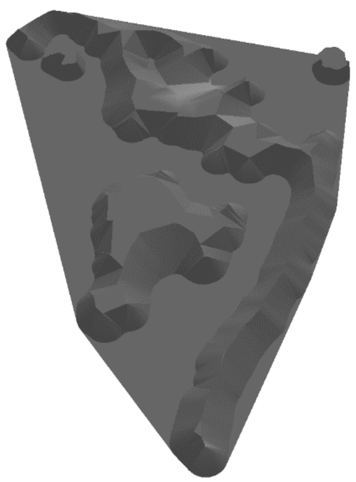


Figure 23 - 3D object created from shape 1

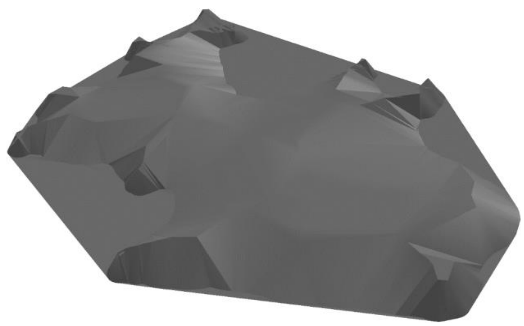


Figure 24 - 3D object created from shape 2

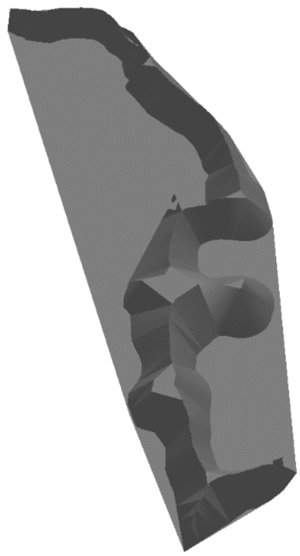


Figure 25 - 3D object created from shape 3

Another observation that can be made from these objects is that, as a consequence of the triangulation, there are no gaps in the objects. In other words, if they were to be sliced horizontally, fully closed polygons would be created. Figures 26, 27 and 28 display five of combined slices that are obtained through slicing the objects at a range of z-levels. These combined cross-sections illustrate how the polygons gradually transform from their initial shape towards a feature that is either comprised only of line segments or of a combination of line segments and core areas.

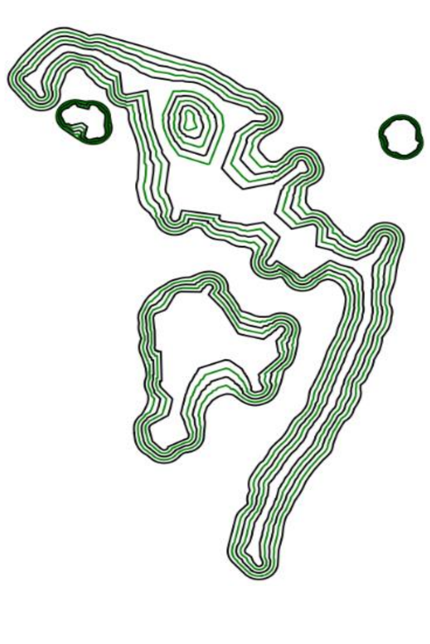
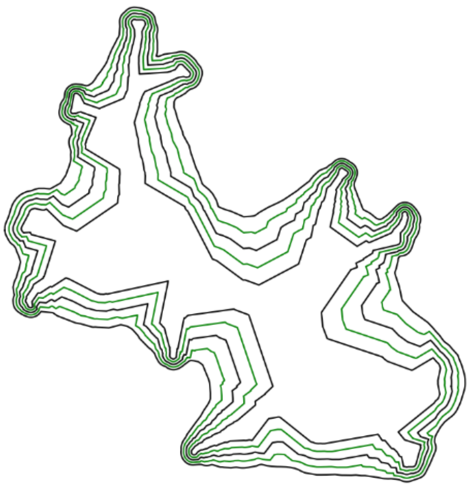
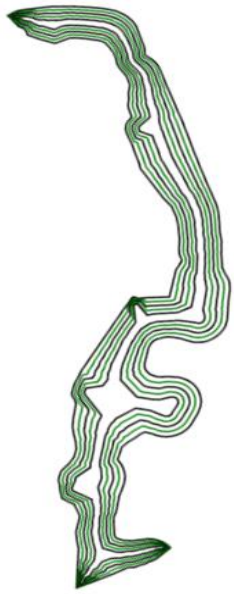
		

Figure 26 - Five consecutive slices from the 3D object of shape 1	*Figure 27 - Five consecutive slices from the 3D object of shape 2*	*Figure 28 - Five consecutive slices from the 3D object of shape 3*

Overall, the results of using the medial axis for the resolution of conflicts indicate that skeleton-based approaches are effective for the (partial) collapsing of polygons. They can be used to create 3D objects that gradually transform and that can be sliced horizontally to produce 2D polygonal representations without any gaps in their outline. However, in order to connect the input polygon to its skeleton in a gradual manner, Delaunay triangulation is required. This means that skeletal variants on their own can only be used for abrupt collapsing. However, in concert with a Delaunay triangulation, they can be implemented effectively for the creation of smooth tGAP structured objects.

However, it is important to nuance that collapsing entire polygons would assume that the entire polygon is too narrow and, thus, not a narrow conflict but a size conflict. Therefore, a skeleton-based approach, even though combined with a Delaunay triangulation, has not yet proven capable of properly resolving narrow conflicts. Moreover the medial axis created some core areas. However, these areas were the result of coincidental internal triangulation between its points and not the result of intentionally designated core areas. Therefore, this approach needs to be expanded with an approach that allows for such intentional preservation.

Another problem with the results of the skeleton-based approach is produced as a consequence of the presence of an island in the core area of shape 1. Namely, this inner ring gradually moves inwards to the skeleton, which is what is to be expected but not necessarily what is desired. Therefore, the approach should not only be extended with a way of preserving core areas but also with a way of constraining the inward movement of island.

4.3. Delaunay triangulation-based approach

Conflict detection

Lastly, the Delaunay triangulation was also used for the detection of narrow conflicts. The results can be seen in Figure 28, which depicts the conflict in red. The first relevant finding in these results is the fact that this approach, alike the buffer-based approach, detected all three types of narrow conflicts. In fact, it even detected virtually the exact same locations as the buffer-based approach did, which speaks to the stability of both methods. Furthermore, in contrast to the skeleton-based approach, which did not result in a clear demarcation of conflicts areas, the triangulation-based approach did allow for the easy creation of conflict areas. An important deviation from the buffer-based approach, however, is the fact that the conflict areas are demarcated by straight line segments (Figures 29 and 30).

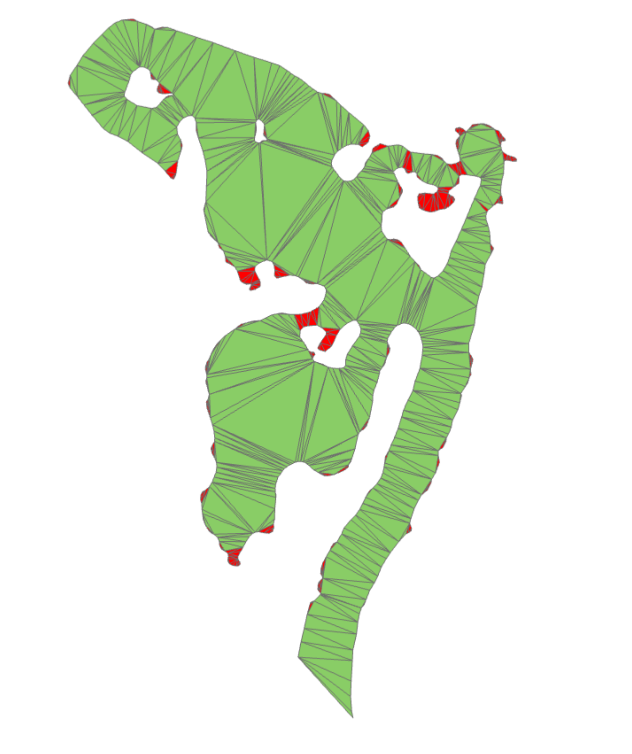


Figure 29 - Conflicts detected with triangulation

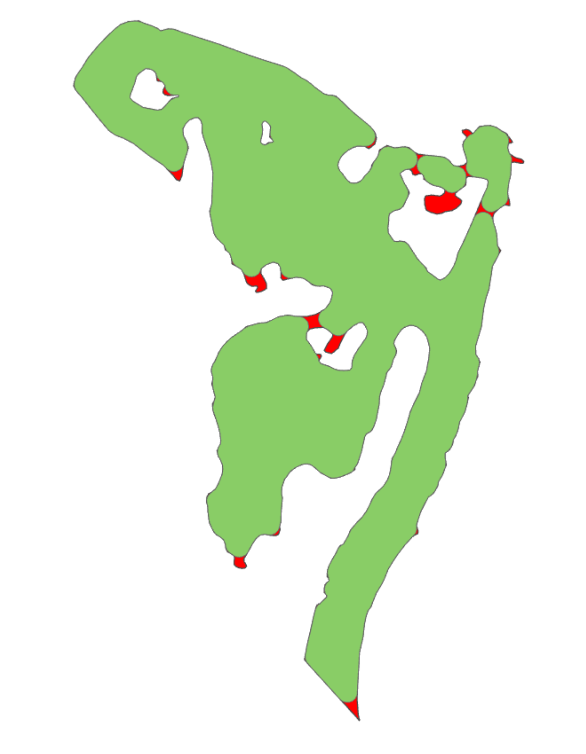


Figure 30 – Conflicts detected with buffers

One consequence is that the safe areas, or core areas, are not as curved as they were after applying the inward-outward-buffering approach. A second consequence is that the resulting core areas are comprised of a smaller number of vertices. These two consequences entail the following implications.

Firstly, the fact that the buffer approach produced rounded output shapes means that a larger amount of detail from the input feature is preserved. The sharp corners that are characteristic of the triangulation approach, on the other hand, indicate that the input shape is generalised more aggressively. However, this difference does not necessarily speak to a difference in quality between both approaches. In fact, in some cases it might be preferable to maintain less detail while other situations might require a larger amount of detail to be preserved. For example, sharply angled objects like buildings could fit better with a triangulation-based approach while more complex shapes might be more properly addressed by a buffer-based approach.

The difference in preservation of detail can also be expressed from a more quantitative perspective. Namely, the features that are produced by the triangulation-based approach are made up of a smaller number of vertices than the ones produced by the buffer approach. Apart from visual differences, this is also of consequence to the amount of computational power required to work with them or display them. Naturally, a larger number of vertices means more information and more information means more processing.

Therefore, the following can be stated regarding the efficacy and usability of the different approaches. The decision as to which approach should be given preference to is subject to considerations of the following aspects: the nature of the input features (sharply angled or round), the desired amount of detail that is to be preserved and any computational limitations.

Conflict resolution

The Delaunay triangulation was not only implemented to detect conflicts, it was also used to resolve them. This was done by connecting different layers in order to create a tGAP structured object. Figure 31 shows how two representations of an input shape can be connected by means of triangulating their combined sets of vertices. The following observations can be made from this image. Firstly, using the Delaunay triangulation, the two layers can be connected to create a 3D object. However, in some locations, the triangulation occurs between vertices of the bottom layer (red ovals) instead of vertices of different layers. This is problematic for the following reason. The fact that internal triangulation occurs means that the bottom layer at this location is not connected to the upper layer. If one were to slice this 3D object at a given z-level above z(min), this location would abruptly disappear. The reason for this internal triangulation is that the vertices of the bottom layer, at these locations, have no vertices in the upper layer to which they can be connected through the Delaunay triangulation. In order to get them to connect to the upper layer, additional points at z(max) have to be inserted into the triangulation in such a way that the bottom vertices will connect to them. This observation leads to the following finding. The more closely that the geometry of the upper shape matches the geometry of the bottom shape, the better the connection between the layers through the Delaunay triangulation. This means that fewer problematic areas are created. Logically, this entails that the Delaunay triangulation becomes less effective for connecting layers when the bottom and upper shapes differ more severely in terms of their geometries. This finding is in correspondence with the publications on the connection of slices using the Delaunay triangulation. In these papers the connection occurred accurately due to the fact that the geometries of the slices did not diverge strongly.

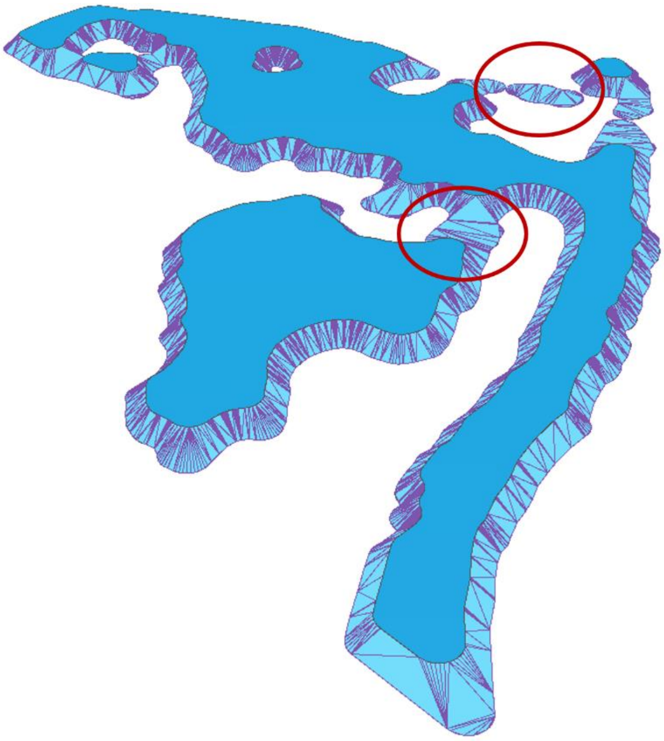


Figure 31 - Connection of two representations through triangulation

Nevertheless, the use of the Delaunay triangulation does create a 3D object without any holes or intersections in the diagonal faces. This means that any slices that are made from it will produce fully closed polygonal shapes. This characteristic does make Delaunay triangulation-based approaches useful for the resolution of conflicts, even though additional operations have to be performed for a truly satisfactory result.

4.4. Combinational approach

As the previous sections of this chapter have demonstrated, the different approaches can be implemented in order to address narrow conflicts in different ways. For example, skeletons have proven to be useful for the collapsing of narrow polygons while triangulation can be used to connect vertices at different elevations. However, it has also been established that all approaches are subject to inherent limitations. Therefore, this final approach aims to combine the strengths of the different approaches with the goal of making up for their limitations.

The results of this combinational approach can be seen in Figures 32 through 37. These images display the layers that are to be turned into 3D objects as well as five horizontal slices that were made from them. The outer slices represent the feature at a lower scale level while the inner ones represent the feature at a higher scale level. What can be observed is the following. At the locations that were appointed narrow conflicts, the slices gradually progress towards the local skeleton. The skeleton functions as a line consisting of all the points of disappearance. At the core polygonal areas on the other hand, the slices do not progress inwards. This is what is desired as the boundaries of the two iterations of the polygon have virtually the same geometry. Furthermore, at places where narrow parts and core areas meet, no gaps or overlaps are present. This means that the transition from skeleton to core area, as grappling points for the triangulation, happens smoothly. Lastly, the two island polygons, which can be categorised as size conflicts, gradually disappear into the centroids that have been created for them. The combination of these three observations indicates that this combinational approach is able to overcome all the issues that were found to be characteristic of the separate approaches. To be more specific, the three approaches work together in the following way.

1) The buffer approach is used to effectively detect narrow conflicts and to create multiple consecutive representations of a shape that are free of conflicts.
2) These layers are then transformed into a 3D object by using a Delaunay triangulation-based approach:
    - At the conflicts areas, the vertices of the bottom layer are connected to the skeleton in order to avoid internal triangulation between the vertices of the bottom layer
    - At the core areas, the vertices of the bottom layer are connected to the vertices of the top layer in order to preserve the core areas

Figure 32 - Two iterations of testing shape 1

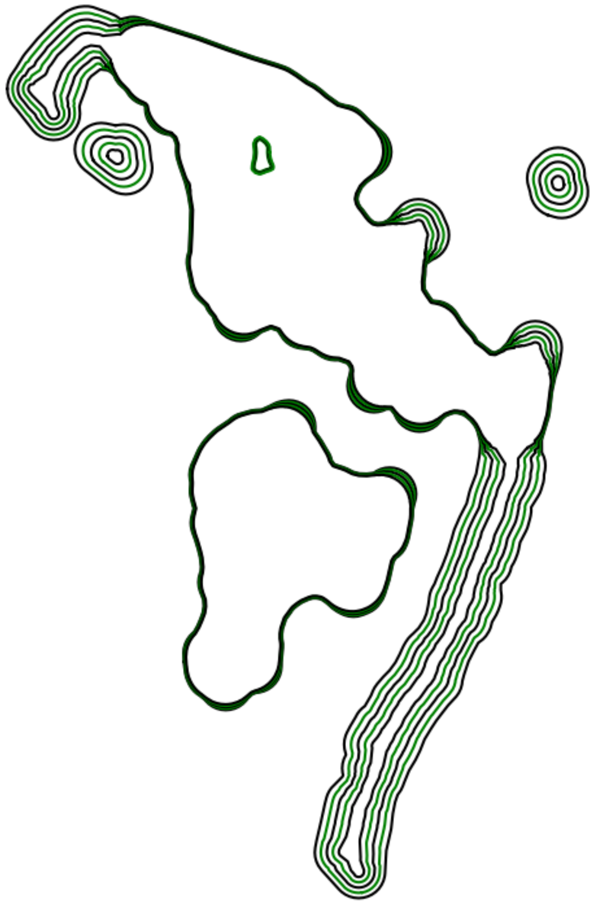


Figure 33 - Five consecutive slices of shape 1

Figure 34 - Two iterations of testing shape 2

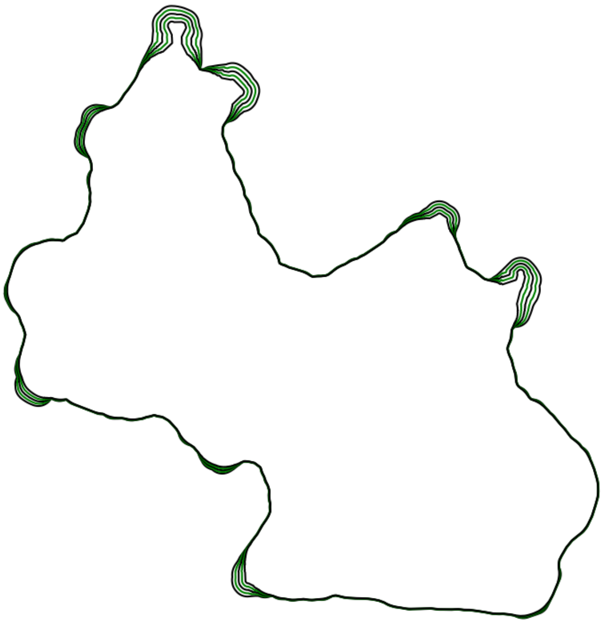


Figure 35 - Five consecutive slices of shape 2

Figure 36 - Two iterations of testing shape 3

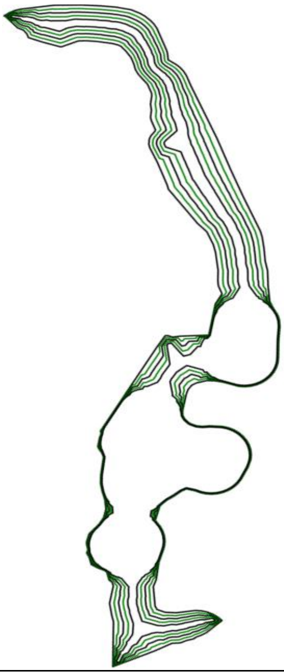


Figure 37 - Five consecutive slices of shape 3

To further illustrate how this combinational approach allows for the intentional preservation of core areas, when compared to the skeleton-based approach, Figures 38 to 43 can be consulted. As can be seen in the images on the right, the 3D object only progresses inward in places that were previously assigned the label of narrow conflicts. This way, all the core areas are maintained without losing any area size.

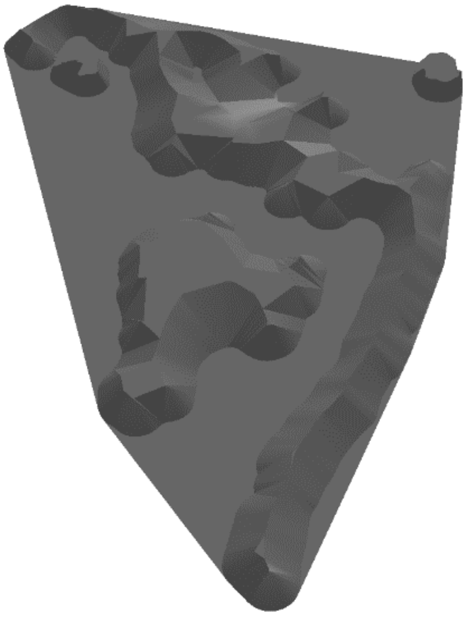


Figure 38 - 3D object of shape 1 created by the skeleton-based approach

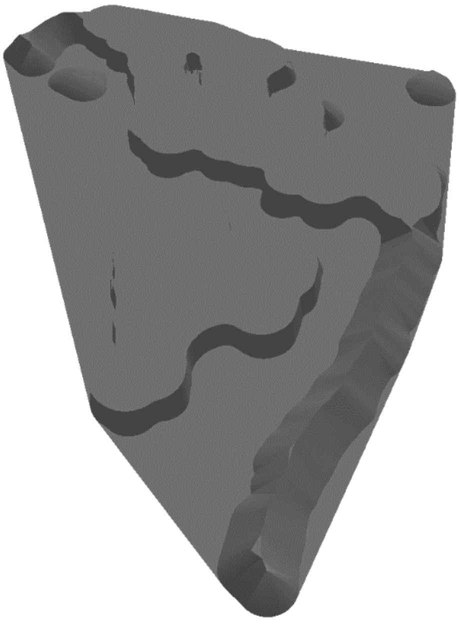


Figure 39 - 3D object of shape 1 created by the combinational approach

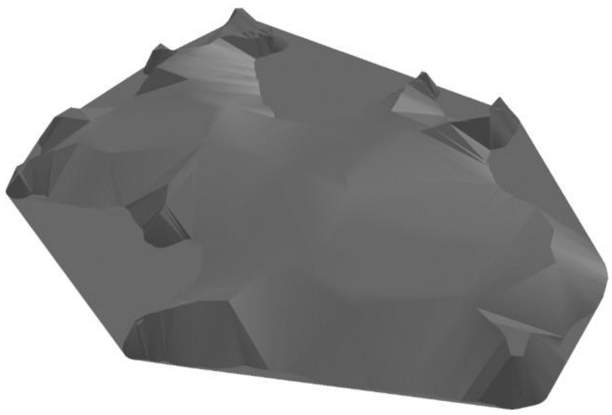
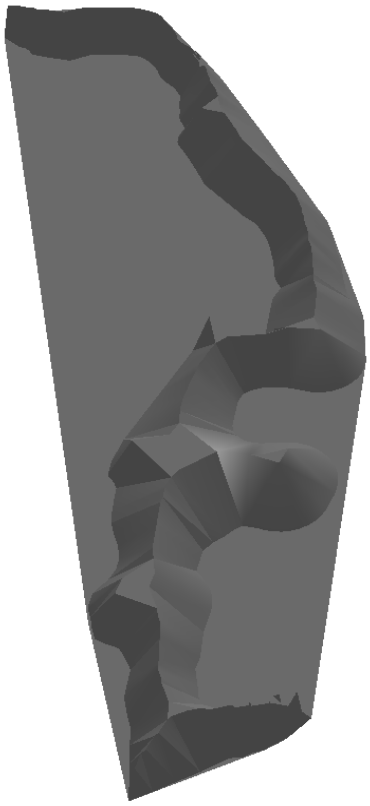


Figure 40 - 3D object of shape 2 created by the skeleton-based approach	Figure 41 - 3D object of shape 2 created by the combinational approach



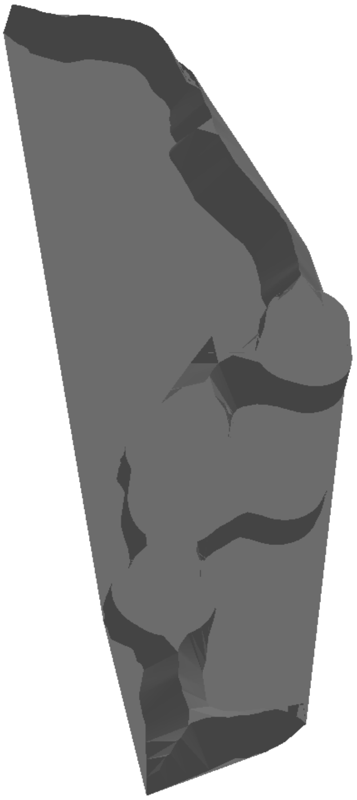


Figure 42 - 3D object of shape 3 created by the skeleton-basedl approach	Figure 43 - 3D object of shape 3 created by the combinational approach

Up to this point, it has been left ambiguously which type of Delaunay triangulation was used to connect the different layers. This is the case not only for the combinational approach but also for the connection of a feature to its skeleton. This is a highly relevant issue, however. The reason for this ambiguity is the following. The implementation of a constrained Delaunay triangulation is significantly more complex than a classical one. The reason is that more information has to be passed into the triangulation algorithm. To be specific, for a classical triangulation, all that is required are the points that are to be taken into account. For a constrained triangulation, especially for more complex shapes, information such as inner rings (holes in the shape), additional constraining edges and even the order of the vertices are required. This complexity is amplified because of the nature of this combinational approach, which also demands skeleton points or edges to be inserted into the triangulation. Despite this complexity, a constrained triangulation would produce results that are preferable to those of a classical triangulation. To be precise, it would mitigate the following issues that are associated with the use of a classical triangulation.

The first risk relates to concave edges in the shape. If these are present, there will be triangles created in these indentations. These triangles could form between vertices of the bottom and top layer (Figure 44). If this happens, it will result in holes in the 3D object and therefore the slices (Figure 46). This means that the 3D object no longer meets the requirements imposed by the nature of the tGAP data structure. Using a constrained triangulation would circumvent this problem because it prevents triangulation in concave edges by bounding the triangulation.

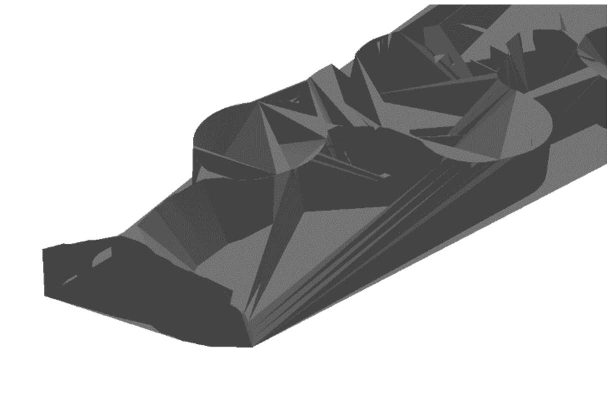


Figure 44 - Unwanted triangulation in concave edges

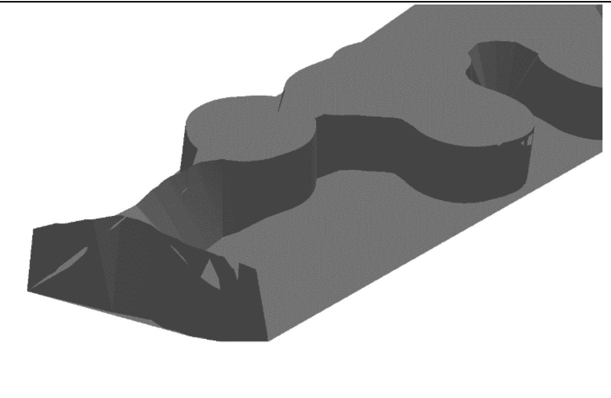


Figure 45 - No unwanted triangulation in concave edges

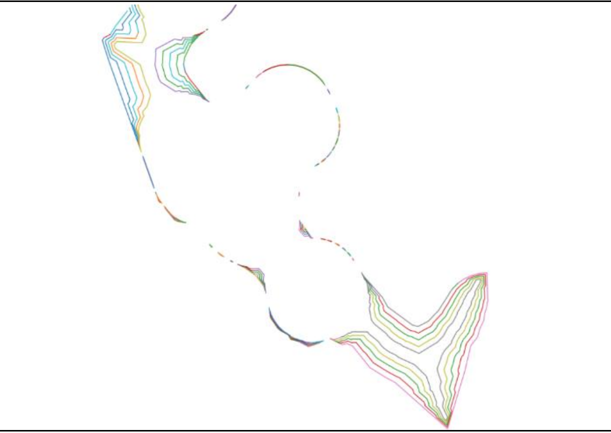


Figure 46 - Holes in the slices, produced by unwanted triangulation

Figure 47 - No holes in the slices by slightly shrinking the upper layer inward

In order to avoid this issue while still using a classical triangulation, the following modification could be made. By shrinking the upper layer inward, the gap between the bottom and the top layer is increased. Given the fact that the vertices of the bottom layer are now more outward than those of the upper layer, the triangulation that occurs in concave edges happens only between vertices of the bottom layer. This does not result in any holes in the outline of the 3D object and therefore the slices. The price that one pays for this fix is that a small part of the core area is lost, as a consequence of slightly shrinking the upper layer. Although this loss might not be much, successfully integrating a constrained triangulation into this combinational approach would prevent it altogether. Given the time span available for this research, such successful implementation has not been fully realised. However, the theoretical and empirical findings that this research has obtained do indicate that applying a constrained Delaunay triangulation in this combinational approach is possible and would produce the most optimal results.

A last noteworthy remark with regard to the results of the combinational approach is the following. This approach can not only be used to resolve narrow conflicts by a way of gradually removing them, it can also be applied in a reverse manner. Looking back on Figures 32 to 37, this means that the top layer becomes the bottom layer and vice versa. The resulting 3D objects and slices are the same as they have been addressed thus far. However, they progress outwards instead of inwards as their z-value is increased. This resolving operation, namely enlargement, is outside of the scope of this research. However, it does speak to the fact that the approaches, as they have been framed in this research, might have a useful role in addressing narrow conflicts by means of other resolving operations.

5. Conclusion

Now that the concepts central to this research have been explored from both a theoretical perspective and an empirical perspective, the attention can be refocused on the central question, which is: *To what extent can buffer- and skeleton- and Delaunay triangulation-based approaches be effective for handling narrow conficts in the context of vario-scale maps?* In order to answer this question in a comprehensive manner, the efficacy of the three central approaches will be discussed seperately after which their complementariness will be addressed.

Buffer-based approaches are particularly effective when it comes to the detection of narrow conflicts. This has been ascertained from both theoretical and empirical exploration. The approach was succesfully used to detect narrow conflicts in multiple consecutive rounds of detection. This resulted in the creation of multiple representations of the testing shapes across a range of scales. Buffer-based approaches, however, have shown no way of being able to resolve conflicts from a vario-scale perspective.

Skeleton-based approaches have proven to be capable of addressing the detection as well as the resolution of narrow conflicts. Nevertheless, literature made it evident that their efficacy when it comes to conflict detection is subordinate to that of the other approaches. This finding was substantiated by the emperical findings that the implementation of this approach yielded. Namely, skeletons can be used to detect narrow conflicts but the partitioning of these regions is not nearly as precise and effective as is the case with other approaches. Regarding conflict resolution, it can be concluded that skeletons can be used effectively for (partially) collapsing narrow conflicts into lines. However, in order to create a gradual connection between the input shape and the skeleton, Delaunay triangulation is required. Therefore, skeleton-based approaches play a crucial role in the resolution of narrow conflicts but cannot be used in isolation, given the requirements of tGAP data structures.

Delaunay triangulation-based approaches, have been found to be capable of both detecting narrow conflicts and resolving them. Regarding conflict detection, they differ from the buffer-based approach in the following ways. The demarcation of conflicts is more rigid than is the case with buffer approaches. This means that the resulting core areas are characterised by straight line segments and sharp angles rather than detailed and rounded edges. This can be a downside in view of preservation of detail but also an upside. Namely, the more radical generalisation of the input shapes results in less geometry (less vertices).

This can be beneficial from a technical perspective, as less information means that less computation power is required in order to work with the features.

When it comes to the resolution of conflicts, the use of triangulation-based approaches stems primarily from their ability to connect vertices at different elevations. This connection occurs in a way that is in correspondence with the requirements necessitated by the nature of a smooth tGAP data structure. In fact, triangulation has been succesfully used for connecting the input shape to its local skeleton and for connecting the core areas of the bottom and top layer. This can be concluded from both the theoretical framework and the emperical application of this type of approach. This also relates to an important conclusion that can be drawn from the results of this research. Namely, the fact that the skeleton should be used to guide the triangulation in places where the bottom and top shape differ substantially in terms of geometry, while direct triangulation between bottom and top shape should be used if these differ to a limited extent. This is the case because only using the skeleton would collapse core areas and only using triangulation would result in a problematic connection between the bottom and top layer. When it comes to the combinational approach that was implemented, the results indicate that this approach can be used effectively for the creation of the most optimal 3D objects from their 2D input shapes. Most optimal meaning that the objects are characterised by the following attributes. Firstly, the objects gradually change in terms of geometry as scale in decreased or increased. Secondly, the core areas that are obtained through the buffer-based approach are fully preserved. This means that this approach allows for the intentional preservation of these core areas. Thirdly, the use of skeletons ensures that narrow parts can be resolved no matter how complex their shape and no matter how far they extend out from the core areas. Lastly, the transition from skeleton to core area, as grappling points for the triangulation, occurs flawlessly. This means that there are no holes or intersecting lines in the diagonal faces of the 3D object, which ensures fully closed polygons when the object is sliced.

However, an important annotation to be made in this regard is the fact that using a classical triangulation requires the upper layer to be shrunk inwards to a minimal degree. This issue can be avoided by successfully applying a constrained triangulation, which has not been fully realised in this research.

All in all, returning to the central question of this research, the following can be concluded. All three types of approaches have their strenghts and weaknesses, which have been mentioned above. Therefore, they can all be used effectively to handle narrow conflicts though in different ways. Regarding their complementariness, is has become evident that combining the strenghts of the approaches allows for the mitigation of their weaknesses. On top of that, it can be concluded that the approaches cannot be used in isolation, since differences in shape characteristics require different approaches to be implemented.

6. Discussion

This research has demonstrated implementations of the three approaches in a fairly successful manner. Naturally however, there are also limitations associated with the methods and results in this research. This chapter will elaborate on these primary limitations and, subsequently, make recommendations regarding ways in which future research might address them.

One of the main limitations of this research is the fact that it specifically addresses both conflicts detection and resolution at a polygonal level. The time frame that was available for this research did not allow for the investigation of these aspects at a map wide level. However, this extension of scope would entail additional consequences regarding the efficacy of the methods as well as practical challenges that were not encountered as a consequence of this restriction of scope. To be more precise, addressing narrow conflicts at a map level would mean that the detection or resolution of a narrow conflict in one polygon could result in the creation of new conflicts in surrounding polygons. For example, where the area size of one polygon shrinks, the area sizes of its neighbours necessarily increase. Therefore, additional research is required in order to determine to what extent the efficacy of the three approaches holds up when applied to a multi-feature environment.

Related to this limitation is one that has to do with the different resolving operations that can be used on narrow conflicts. Namely, this research focussed on gradually removing the narrow conflicts from the polygonal area. However, other resolving operations, such as enlargement or displacement of areas, have been left outside of consideration. These alternative actions might be preferable depending on specific factors. Therefore, the three approaches that were central to this research may or may not be usefully implementable for such other operations, this is an issue that will have to be investigated in future research.

Another issue that is worth mentioning relates to the decision to break continuous conflict detection and resolution down into step by step processes. This decision has been substantiated from a theoretical perspective and also makes sense from a practical standpoint. However, it does entail a discussion on how to structure scale in a vario-scale environment. Moreover, this research has addressed scale as a range going from small to large and vice versa. However exact scale ratios have not been discussed, primarily due to the fact that the approaches were assessed regardless of exact scales. From the perspective of practical implementation, however, it is crucial to determine how scale can be represented in a vario-scale environment. This is not so much a limitation of this research as it is an injunction for future research to investigate.

Another limitation that this research has faced relates to the theoretical framework specifically, which is heavily dominated by a limited number of authors. Although this is primarily due to the fact that the concept of vario-scale maps is still in its infancy, this one-sidedness does produce a limited variation of different views and perspectives.

An additional limitation stems from the fact that only three testing shapes were used for the assessment of the approaches. The number of testing cases had to be bounded and these shapes were selected to contain a whole range of possible obstacles. Nevertheless, other shapes could pose problems that were not encountered in these tests.

Last but certainly not least, this research was limited by the amount of time that was available. This significantly limited the range of relevant concepts that could be addressed, which meant that this research has only been able to address a small part of the discussion on vario-scale maps and narrow conflicts. More time would have also allowed for a deeper exploration of the topics in this research, such as the integration of a constrained triangulation in the combinational approach.

All in all, by answering some questions related to narrow conflicts and vario-scale maps, this research has also definitely opened the door to more questions. In hopes of answering these questions, more research in this direction is required.

7. References

Ai, T., & Guo, R. (2000). A constrained Delaunay partitioning of areal objects to support map generalization. *Journal of Wuhan University: Information Science Edition*, *25*(1), 35-41.

Aigner, W., Aurenhammer, F., & Jüttler, B. (2015). On triangulation axes of polygons. *Information Processing Letters*, *115*(1), 45-51.

Ali, T., & Mehrabian, A. (2009). A novel computational paradigm for creating a Triangular Irregular Network (TIN) from LiDAR data. *Nonlinear Analysis: Theory, Methods & Applications*, *71*(12), 624-629.

Altundag, D., & Stoter, J. (2011). Automated generalisation of the 1: 10k topographic map from municipal data. *Proceedings of the 14th Workshop of the ICA Commission on Generalisation and Multiple Representation*.

Bader, M., & Weibel, R. (1997). Detecting and resolving size and proximity conflicts in the generalization of polygonal maps. *Proceedings 18th International Cartographic Conference*, *23*, 27-38.

Barequet, G., Goodrich, M. T., Levi-Steiner, A., & Steiner, D. (2004). Contour interpolation by straight skeletons. *Graphical Models*, *66*(4), 245-260

Barequet, G., & Sharir, M. (1994). Piecewise-linear interpolation between polygonal slices. *Proceedings of the tenth annual symposium on Computational geometry*, 93-102.

Bern, M., & Eppstein, D. (1995). Mesh generation and optimal triangulation. *Computing in Euclidean geometry*, 47-123.

Blum, H. (1967). *Models for the Perception of Speech and Visual Form* (W. Wathen-Dunn, ed.). Cambridge, Mass.: M.I.T. Press. A transformation for extracting new descriptors of shape.

Boissonnat, J. D., & Geiger, B. (1993). Three-dimensional reconstruction of complex shapes based on the Delaunay triangulation. *Biomedical image processing and biomedical visualization*, *1905*, 964-975.

Bundy, G. L., Jones, C. B., & Furse, E. (1995). A topological structure for the holistic generalization of large-scale cartographic data. *Innovations in GIS*, *2*, 19-31.

Christensen, A. H. (2000). Line generalization by waterlining and medial-axis transformation. Successes and issues in an implementation of Perkal's proposal. *The Cartographic Journal*, *37*(1), 19-28.

Du, J., Wu, F., Xing, R., Li, J., & Gong, X. (2021). An Automated Approach to Coastline Simplification for Maritime Structures with Collapse Operation. *Marine Geodesy*, *44*(3), 157-195.

Fisher, P. F., Roberts, S. A., Hall, G. B., & Boots, B. (2005). Street centreline generation with an approximated area Voronoi diagram. In *Developments in Spatial Data Handling: 11 th International Symposium on Spatial Data Handling* (pp. 435-446). Springer Berlin Heidelberg.

Galanda, M. (2003). Modelling constraints for polygon generalization. In *Proceedings 5th ICA Workshop on Progress in Automated Map Generalization* (pp. 28-30).

Gao, W., Gong, J., Yang, L., Jiang, X., & Wu, X. (2012). Detecting geometric conflicts for generalisation of polygonal maps. *The Cartographic Journal*, *49*(1), 21-29.

Geometry Factory (2010). Straight skeleton for modelling roofs. Retrieved from https://geometryfactory.com/portfolio/roofs-and-the-straight-skeleton/.

Goldstein, E., & Gotsman, C. (1995). Polygon morphing using a multiresolution representation. In *Graphics Interface* (pp. 247-247). Canadian Information Processing Society.

Gotsman, C., & Surazhsky, V. (2001). Guaranteed intersection-free polygon morphing. *Computers & Graphics*, *25*(1), 67-75.

Guibas, L., & Hershberger, J. (1994). Morphing simple polygons. In *Proceedings of the tenth annual Symposium on Computational Geometry* (pp. 267-276).

Harrie, L., Sarjakoski, L.T., and Lehto, L. (2002) A variable-scale map for small-display cartography. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, *34*, 237–242.

Haunert, J. H. (2008). *Aggregation in map generalization by combinatorial optimization*. Fachrichtung Geodäsie und Geoinformatik der Leibniz-Universiteit.

Haunert, J. H., Dilo, A., & Van Oosterom, P. (2009). Constrained set-up of the tGAP structure for progressive vector data transfer. *Computers & Geosciences*, *35*(11), 2191-2203.

Haunert, J. H. & Sester, M. (2008). Area collapse and road centerlines based on straight skeletons. *GeoInformatica*, *12*(2), 169-191.

Haunert, J. H. & Sester, M. (2004). Using the straight skeleton for generalisation in a multiple representation environment. *ICA Workshop on Generalisation and Multiple Representation.*

Held, M., & Palfrader, P. (2017). Straight skeletons with additive and multiplicative weights and their application to the algorithmic generation of roofs and terrains. *Computer-Aided Design*, *92*, 33-41.

Hofman, A. M. (2008). Developing a vario-scale IMGeo using the constrained tGAP structure.

Højholt, P. (2000). Solving space conflicts in map generalization: Using a finite element method. *Cartography and Geographic Information Science*, *27*(1), 65-74.

Johnston, M. R., Scott, C. D., & Gibb, R. (1999). Problems arising from a simple GIS generalisation algorithm. *Proceedings of the Eleventh Annual Colloquium of the Spatial Information Research Centre*, 191-200.

Jones, C. B., Bundy, G. L. & Ware, M. J. (1995). Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, *22*(4), 317-331.

Kuipers, T., Doubrovski, E. L., Wu, J., & Wang, C. C. (2020). A framework for adaptive width control of dense contour-parallel toolpaths in fused deposition modeling. *Computer-Aided Design*, *128*.

Lewandowicz, E., & Flisek, P. (2020). Base point split algorithm for generating polygon skeleton lines on the example of lakes. *ISPRS international journal of geo-information*, *9*(11), 680.

Lucas, G.W. (n.d.). What is the Constrained Delaunay Triangulation and why would you care? Retrieved from https://gwlucastrig.github.io/TinfourDocs/DelaunayIntroCDT/index.html.

McAllister, M. (1999). The computational geometry of hydrology data in geographic information systems (Doctoral dissertation, University of British Columbia).

McAllister, M., & Snoeyink, J. (2000). Medial axis generalization of river networks. *Cartography and Geographic Information Science*, *27*(2), 129-138.

Montero, A. S., & Lang, J. (2012). Skeleton pruning by contour approximation and the integer medial axis transform. *Computers & Graphics*, *36*(5), 477-487.

Palfrader, P. (2014). A simple algorithm for computing positively weighted straight skeletons of monotone polygons. Retrieved from. https://www.researchgate.net/publication/268753016_A_simple_algorithm_for_computing_positively_weighted_straight_skeletons_of_monotone_polygons/figures?lo=1..

Peter, B. (2001). Measures for the generalization of polygonal maps with categorical data. *Fourth ICA Workshop on Progress in Automated Map Generalization, Beijing*, 2-4.

Rognant, L., Chassery, J. M., Goze, S., & Planes, J. G. (1999). The Delaunay constrained triangulation: the Delaunay stable algorithms. *1999 IEEE International Conference on Information Visualization*, 147-152.

Ruas, A. (1995). Multiple paradigms for automating map generalization: Geometry, topology, hierarchical partitioning and local triangulation. *ACSM/ASPRS Annucal Convention and Exposition*.

Suba, R., Meijers, M., & Van Oosterom, P. (2013). 2D vario-scale representations based on real 3D structure. In *16th ICA Workshop on Generalisation and Multiple Representation.*.

Sugihara, K. (2019). Straight skeleton Computation Optimized for Roof Model Generation. *WSCG, 27*,101-109.

Tănase, M., & Veltkamp, R. C. (2004). A straight skeleton approximating the medial axis. *Algorithms–ESA 2004: 12th Annual European Symposium*, 809-821.

Tran, Q. A., & Fan, H. (2013). Automatic generalization of map polygon resident in digital environment. *MIPPR 2013: Remote Sensing Image Processing, Geographic Information Systems, and Other Applications*, *8921*, 36-42.

Van der Poorten, P. M. & Jones, C. B. (2002). Characterisation and generalisation of cartographic lines using Delaunay triangulation. *International Journal of Geographical Information Science, 16*(8), 773-794.

Van Kreveld, M. (2001). Smooth generalization for continuous zooming. In *Proc. 20th Intl. Geographic Conference*, 2180-2185.

Van Oosterom, P. & Meijers, M. (2011). Towards a true vario-scale structure supporting smooth-zoom. *Proceedings of the 14th Workshop of the ICA Commission on Generalisation and Multiple Representation.*

Van Oosterom, P. & Meijers, M. (2014). Vario-scale data structures supporting smooth zoom and progressive transfer of 2D and 3D data. *International Journal of Geographical Information Science*, *28*(3), 455-478.

Van Oosterom, P., Meijers, M., Stoter, J. & Šuba, R. (2014). Data structures for continuous generalisation: tGAP and SSC. *Abstracting Geographic Information in a Data Rich World: Methodologies and Applications of Map Generalisation*, 83-117.

8. Appendices

Appendix 1: Three testing polygons

Shape 1:

Shape 2:

Shape 3:

Appendix 2: Skeleton-based conflict detection code

```
import math
import fiona
import trimesh
import shapely
import numpy as np
import matplotlib.pyplot as plt
from shapely.geometry import shape


#import shape and extract the vertices

c = fiona.open(r"C:\GIMA\Thesis\Nieuwe_Exports\Bottom_Shape.shp")
pol = c.next()
full_shape = shape(pol['geometry'])
polygons = list(full_shape.geoms)

vertices = [shapely.geometry.mapping(polygons[i]) for i in
range(len(polygons))]
vertices_coords = [vertices[i] for i in range(len(vertices))]
vertices_extract = [vertices_coords[i]['coordinates'] for i in
range(len(vertices_coords))]
vertices_final_coords = []

for i in range(len(vertices_extract)):
    vertices_seperated_polygons = vertices_extract[i]
    for p in range(len(vertices_seperated_polygons)):
        vertices_seperated_coords = vertices_seperated_polygons[p]
        for o in range(len(vertices_seperated_coords)):
            vertices_final_coords.append(vertices_seperated_coords[o])

vertices_list = [list(vertices_final_coords[i]) for i in
range(len(vertices_final_coords))]
vertices_array = np.array(vertices_list)

#create a skeleton for this shape

medial_axis= [trimesh.path.polygons.medial_axis(polygons[i], resolution =
20) for i in range(len(polygons))]

medial_axis_coords = [medial_axis[i][1] for i in range(len(medial_axis))]
medial_axis_sep_lists = [medial_axis_coords[i].tolist() for i in
range(len(medial_axis_coords))]
medial_axis_list = []

for i in range(len(medial_axis_sep_lists)):
    for j in range(len(medial_axis_sep_lists[i])):
        medial_axis_list.append(medial_axis_sep_lists[i][j])

#create a way of calculating distance polygon vertices to the skeleton
(points)

distances = []

for i in range(len(vertices_list)):
```

```
    dis = math.inf
    distances.append([i])
    for j in range(len(medial_axis_list)):
        dis_iter = math.sqrt((medial_axis_list[j][0]-
vertices_list[i][0])**2 + (medial_axis_list[j][1] -
vertices_list[i][1])**2)
        if dis_iter < dis:
            dis = dis_iter
            distances[i].append(dis)

distances_output = []

for i in range(len(distances)):
    distances_output.append([i])
    distances_output[i].append(distances[i][-1])

#determine a minimum width value that makes sense and check where in the
polygon this value is not met

minimum_width = 14

conflict_vertices = []
safe_vertices = []

for i in range(len(distances_output)):
    if distances_output[i][1] < minimum_width:
        conflict_vertices.append(distances_output[i][0])
    else:
        safe_vertices.append(distances_output[i][0])

#select conflict vertices

conflict_coords = [vertices_list[conflict_vertices[i]] for i in
range(len(conflict_vertices))]
safe_coords = [vertices_list[safe_vertices[i]] for i in
range(len(safe_vertices))]
conflict_x = [conflict_coords[i][0] for i in range(len(conflict_coords))]
conflict_y = [conflict_coords[i][1] for i in range(len(conflict_coords))]
safe_x = [safe_coords[i][0] for i in range(len(safe_coords))]
safe_y = [safe_coords[i][1] for i in range(len(safe_coords))]
medial_axis_x = [medial_axis_list[i][0] for i in
range(len(medial_axis_list))]
medial_axis_y = [medial_axis_list[i][1] for i in
range(len(medial_axis_list))]

#visualise the conflict areas and safe areas

plt.scatter(safe_x, safe_y, color = 'green')
plt.scatter(conflict_x, conflict_y, color = 'red')
plt.scatter(medial_axis_x, medial_axis_y, color = 'black')
plt.show()
```

Appendix 3: Skeleton-based conflict resolution code

```
import fiona
import trimesh
import shapely
import numpy as np
import geopandas as gpd
import matplotlib.tri as mtri
import matplotlib.pyplot as plt
from shapely.geometry import shape


c = fiona.open(r"C:\GIMA\Thesis\Nieuwe_Exports\Bottom_Shape.shp")
pol = c.next()
bottom_shape = shape(pol['geometry'])
if bottom_shape.geom_type == 'MultiPolygon':
    bottom_polygons = list(bottom_shape.geoms)
elif bottom_shape.geom_type == 'Polygon':
    bottom_polygons = [bottom_shape]
else:
    raise IOError('Shape is not a polygon.')

medial_axis = []
for i in range(len(bottom_polygons)):
    medial_axis_iter =
trimesh.path.polygons.medial_axis(bottom_polygons[i])
    medial_axis.append(medial_axis_iter)

medial_axis_coords = [medial_axis[i][1] for i in range(len(medial_axis))]
medial_axis_sep_list = [medial_axis_coords[i].tolist() for i in
range(len(medial_axis_coords))]

medial_axis_list = []

for i in range(len(medial_axis_sep_list)):
        for p in range(len(medial_axis_sep_list[i])):
            medial_axis_sep_list[i][p].append(10)
            medial_axis_list.append(medial_axis_sep_list[i][p])

medial_axis_array = np.array(medial_axis_list)

bottom_vertices = [shapely.geometry.mapping(bottom_polygons[i]) for i in
range(len(bottom_polygons))]
bottom_vertices_coords = [bottom_vertices[i] for i in
range(len(bottom_vertices))]
bottom_vertices_extract = [bottom_vertices_coords[i]['coordinates'] for i
in range(len(bottom_vertices_coords))]
bottom_vertices_final_coords = []

for i in range(len(bottom_vertices_extract)):
    for p in range(len(bottom_vertices_extract[i])):
        for o in range(len(bottom_vertices_extract[i][p])):

bottom_vertices_final_coords.append(bottom_vertices_extract[i][p][o])

bottom_vertices_list = [list(bottom_vertices_final_coords[i]) for i in
range(len(bottom_vertices_final_coords))]
```

```
bottom_vertices_with_z = [bottom_vertices_list[i].append(0) for i in
range(len(bottom_vertices_list))]
bottom_vertices_array = np.array(bottom_vertices_list)

merged_points = np.concatenate((bottom_vertices_array, medial_axis_array),
axis = 0)

triang = mtri.Triangulation(merged_points[:,0], merged_points[:,1])
faces = triang.get_masked_triangles()
mesh = trimesh.Trimesh(vertices = merged_points, faces = faces)
edges = triang.edges
triangles = triang.triangles
triangles_list = [list(triangles[i]) for i in range(len(triangles))]

z_extents = mesh.bounds[:,2]
z_levels  = np.arange(*z_extents, step = 2)
sections = mesh.section_multiplane(plane_origin = mesh.bounds[0],
                                   plane_normal = [0,0,1],
                                   heights = z_levels)
combined_sections = np.sum(sections)
combined_sections.show()

mesh.show(viewer = 'gl')
```

Appendix 4: Combinational approach full code

```
import fiona
import trimesh
import shapely
import numpy as np
import matplotlib.tri as mtri
from shapely.geometry import shape

# loading bottom and top shape and turning them into list of POLYGONs

with fiona.open(r"C:\GIMA\Thesis\Nieuwe_Exports\Bottom_Shape.shp") as c:
    for pol in c:
        bottom_shape = shape(pol['geometry'])
        if bottom_shape.geom_type == 'MultiPolygon':
            bottom_polygons = list(bottom_shape.geoms)
        elif bottom_shape.geom_type == 'Polygon':
            bottom_polygons = [bottom_shape]
        else:
            raise IOError('Shape is not a polygon.')

with fiona.open(r"C:\GIMA\Thesis\Nieuwe_Exports\Internal_Shape_X.shp") as
c:
    for pol in c:
        top_shape = shape(pol['geometry'])
        if top_shape.geom_type == 'MultiPolygon':
            top_polygons = list(top_shape.geoms)
        elif top_shape.geom_type == 'Polygon':
            top_polygons = [top_shape]
        else:
            raise IOError('Shape is not a polygon.')

# creating medial axis points for all indices of bottom_polygons

for i in range(len(bottom_polygons)):
    medial_axis = trimesh.path.polygons.medial_axis(bottom_polygons[i])

# storing their coordinates as a list of lists and adding a z-value (10)

medial_axis_coords = medial_axis[1]
medial_axis_list = medial_axis_coords.tolist()
medial_axis_with_z = [medial_axis_list[i].append(10) for i in
range(len(medial_axis_list))]

# selecting the medial axis points that are located outside any of the top
shapes

selected_ma_coords = []

for i in range(len(medial_axis_list)):
    if top_shape.covers(shapely.geometry.Point(medial_axis_list[i][0],
medial_axis_list[i][1])) == False:
        selected_ma_coords.append(medial_axis_list[i])

medial_axis_array = np.array(selected_ma_coords)

# Extracting vertices from bottom polygon and adding z-value (0)
```

```
bottom_vertices = [shapely.geometry.mapping(bottom_polygons[i]) for i in
range(len(bottom_polygons))]
bottom_vertices_coords = [bottom_vertices[i] for i in
range(len(bottom_vertices))]
bottom_vertices_extract = [bottom_vertices_coords[i]['coordinates'] for i
in range(len(bottom_vertices_coords))]
bottom_vertices_final_coords = []

for i in range(len(bottom_vertices_extract)):
    for p in range(len(bottom_vertices_extract[i])):
        for o in range(len(bottom_vertices_extract[i][p])):

bottom_vertices_final_coords.append(bottom_vertices_extract[i][p][o])

bottom_vertices_list = [list(bottom_vertices_final_coords[i]) for i in
range(len(bottom_vertices_final_coords))]
bottom_vertices_with_z = [bottom_vertices_list[i].append(0) for i in
range(len(bottom_vertices_list))]
bottom_vertices_array = np.array(bottom_vertices_list)

# Extracting vertices from top polygon and adding z-value (10)

top_vertices = [shapely.geometry.mapping(top_polygons[i]) for i in
range(len(top_polygons))]
top_vertices_coords = [top_vertices[i] for i in range(len(top_vertices))]
top_vertices_extract = [top_vertices_coords[i]['coordinates'] for i in
range(len(top_vertices_coords))]
top_vertices_final_coords = []

for i in range(len(top_vertices_extract)):
    for p in range(len(top_vertices_extract[i])):
        for o in range(len(top_vertices_extract[i][p])):
            top_vertices_final_coords.append(top_vertices_extract[i][p][o])

top_vertices_list = [list(top_vertices_final_coords[i]) for i in
range(len(top_vertices_final_coords))]
top_vertices_with_z = [top_vertices_list[i].append(10) for i in
range(len(top_vertices_list))]
top_vertices_array = np.array(top_vertices_list)

# finding sub-polygons of bottom_polygons for which a centroid has to be
created

island_polygons = []

# check for each sub-polygon whether it contains a Medial Axis point
for i in range(len(bottom_polygons)):
    island = True
    for p in range(len(selected_ma_coords)):
        if
bottom_polygons[i].covers(shapely.geometry.Point(selected_ma_coords[p][0],
selected_ma_coords[p][1])):
            island = False
    if island == True:
        island_polygons.append(bottom_polygons[i])
```

```
island_polygons_2 = []

# check for each sub-polygon whether it contains a top vertex
for i in range(len(island_polygons)):
    island = True
    for p in range(len(top_vertices_list)):
        if
island_polygons[i].covers(shapely.geometry.Point(top_vertices_list[p][0],
top_vertices_list[p][1])):
            island = False
    if island == True:
        island_polygons_2.append(island_polygons[i])

centroids = [island_polygons_2[i].centroid for i in
range(len(island_polygons_2))]
centroids_coords = [shapely.geometry.mapping(centroids[i]) for i in
range(len(centroids))]
centroids_extract = [centroids_coords[i]['coordinates'] for i in
range(len(centroids_coords))]
centroids_list = [list(centroids_extract[i]) for i in
range(len(centroids_extract))]
centroids_with_z = [centroids_list[i].append(10) for i in
range(len(centroids_list))]
centroids_array = np.array(centroids_list)

# combining the bottom and top vertices with the medial axis points and
centroids (if those were created)

if len(centroids_array) == 0 and len(medial_axis_array) == 0:
    merged_points = np.concatenate((bottom_vertices_array,
top_vertices_array), axis=0)

elif len(centroids_array) == 0 and len(medial_axis_array) != 0:
    merged_points = np.concatenate((bottom_vertices_array,
top_vertices_array, medial_axis_array), axis=0)

elif len(medial_axis_array) == 0 and len(centroids_array) != 0:
    merged_points = np.concatenate((bottom_vertices_array,
top_vertices_array, centroids_array), axis=0)

else:
    merged_points = np.concatenate((bottom_vertices_array,
top_vertices_array, medial_axis_array, centroids_array), axis=0)

# triangulating the combined set of points and extracting the faces for the
mesh

triang = mtri.Triangulation(merged_points[:, 0], merged_points[:, 1])
faces = triang.get_masked_triangles()
edges = triang.edges
triangles = triang.triangles
triangles_list = [list(triangles[i]) for i in range(len(triangles))]

mesh = trimesh.Trimesh(vertices = merged_points,
                       faces = faces)
mesh.show(viewer='gl')
```

```
cross_section = mesh.section(plane_origin = (mesh.centroid),
                             plane_normal = [0, 0, 1])

slice_2D, to_3D = cross_section.to_planar()
z_extents = mesh.bounds[:, 2]
z_levels = np.arange(*z_extents, step = 2)
sections_1 = mesh.section_multiplane(plane_origin = mesh.bounds[0],
                                     plane_normal = [0, 0, 1],
                                     heights = z_levels)

combined_sections_1 = np.sum(sections_1)
combined_sections_1.show()
```