EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science
Algorithms

# The effect of area preservation on polygon simplification

*Master's thesis*

Damian M. Buzink

15-08-2023

*Supervision:*
Wouter Meulemans
Willem M. Sonke

*Assessment committee:*
Wouter Meulemans
Willem M. Sonke
Huub van de Wetering

*Credits:* 30

This is a public Master's thesis.

This Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct.

# Abstract

Simplification algorithms have many use cases such as in trajectory analysis and simulations. Some of these algorithms preserve area. We present an overview of related work as well as investigate the effect of area preservation on the quality of simple polygon simplifications. To investigate the effect, we implemented the simplification algorithms proposed by by Imai and Iri (1988), Kronenfeld et al. (2020) and Buchin et al. (2016) and create an additional variant for each algorithm such that we have two versions: one with area preservation and one without. We apply these algorithms on two different data sets: one consisting of building outlines and the other consisting of lakes. The polygons from these data sets we simplify to 25% and 75% of the original number of vertices. We observe that our area preserving variant of Imai and Iri produces simplifications of low quality compared to any of the other algorithms. The area preserving variant of Kronenfeld et al. outperforms its non-area preserving counterpart, although it was the opposite for the Buchin et al. algorithm, if only slightly. The algorithm by Buchin et al. does modify twice the area, thus it was unexpected that it was as close as it is, particularly in regards to the symmetric difference. Due to this, we suspect that area preservation does help, but possibly not enough to counteract the cost of the additional modification to preserve area in this algorithm. We also observe that the area preserving variants do seem to lessen compounding errors in the iterative algorithms of Kronenfeld et al. and Buchin et al. To further investigate this hypothesis, we created a simple iterative version of Imai and Iri and here we see that the area preserving variant is significantly closer to the non-area preserving variant. The area preserving variant for the iterative version also produced a better simplification more often than in the normal version of Imai and Iri. These results suggest that area preservation limits compounding errors in iterative simplification algorithms when applied on simple polygons.

# Acknowledgments

For his overall guidance and specifically invaluable feedback regarding my thesis, I would like to extend my sincere thanks to my graduation supervisor Wouter Meulemans. I would also like to express my deepest gratitude to my other graduation supervisor Willem M. Sonke for his help during the execution of the research and the many meetings we held, in which he helped guide me to a good quality thesis.

# Contents

# Chapter 1

# Introduction

When we have polylines, polygons or subdivisions there are situations where the need arises to reduce the number of vertices. For example during movement analysis we might want to apply an algorithm on a polyline. However, the polyline may have so many vertices that the algorithms require a significant amount of time to compute a result [8]. A similar situation might occur in simulations with polygons and subdivisions, where the time to compute the simulation might be high, due to the many vertices or edges. The reduction of the number of vertices in a polyline, polygon or subdivision is what we call simplification. To automate the process, simplification algorithms have been created.

Recently there have been a number of simplification algorithms proposed that preserve the area during the simplification process [3, 4, 13, 15, 16]. It is however not clear if area preservation improves the quality of simplifications, although Kronenfeld et al. do conjecture as such [13]. The assumption is supported by the expectation that high quality simplifications have a similar amount of area, if not the same. In this study, we investigate this conjecture through experimentation to determine what the effect is of area preservation in simplification algorithms on simple polygons.

**Contributions** To investigate the effect of area preservation in simplification algorithms on simple polygons, we implement multiple simplification algorithms. We either find or newly propose a counterpart such that, if the algorithm preserves area, the counterpart does not. If the algorithm does not preserves area, we find or create an area preserving counterpart. This allows us to compare the simplifications and analyze their differences. Based on this, we formulated the following research questions to investigate the effect of area preservation on polygon simplification:

- What is the difference between the simplifications of the algorithm by Imai and Iri [12] with Hausdorff as the error measure and with area preservation as proposed by Daneshpajouh et al. [4]?
- What is the difference between the simplifications of the algorithm by Kronenfeld et al. [13] with area preservation and without area preservation?

- What is the difference between the simplifications of the algorithm by Buchin et al. [3] with area preservation and without area preservation?

We discuss in Chapter 3 why we selected the two data sets: one consisting of outlines of landmarks of the District of Columbia and the other of lakes from within the United States of America. Likewise, we discuss why we chose to simplify the polygons to 25% and 75% of the original number of vertices. Lastly, we discuss the metrics we use to analyze the simplifications, which are: Hausdorff distance (see Definition 1), Modified Hausdorff distance (see Definition 2), Discrete Fréchet distance (see Definition 3) and symmetric difference (see Definition 4).

In Chapter 4 we discuss the algorithms we implemented in detail: Imai and Iri [12], Kronenfeld et al. [13] and Buchin et al. [3]. For the area preserving variant of Imai and Iri, we use the difference in area between the original polygon and the polygon with the simplification as the error threshold for a shortcut. In the non-area preserving variant of the Kronenfeld et al. algorithm we place the Steiner point in a three edge segment, not such that the area is preserved, as the area preserving variant does, but to exclusively preserve the shape of the segment. In the Buchin et al. algorithm we do not do both edge moves at each iteration, such that area is preserved, but instead only the edge move required to reduce the number of vertices by one. Lastly, we discuss how some of our implementations of the algorithms and the corresponding variants differ slightly from the proposed algorithms to simplify implementation. The algorithms do compute the exact simplifications as the proposed algorithms.

We show the results of the metric computations based on the simplifications in Chapter 5. From the results we observe that Imai and Iri's area preserving counterpart produces simplifications of significantly inferior quality compared to all other algorithms and counterparts. Kronenfeld et al.'s algorithm does do slightly better than the non-area preserving counterpart consistently. The variants of the Buchin et al. algorithm stay close together with the non-area preserving variant showing slightly better results.

Based on the fact that the area preserving variant of Kronenfeld et al. performs better and that Buchin et al. modifies twice as much area per iteration and yet still is only slightly behind its non-area preserving counterpart, we formulated the following hypothesis in Chapter 6: we hypothesize that area preservation helps in iterative simplification algorithms to limit compounding errors. This hypothesis was further investigated by implementing a simple iterative version of Imai and Iri and comparing the area preserving and non-area preserving variants for that version. Here, we see that the quality of the area preserving simplifications are much closer to the non-area preserving variant than they are for the normal version of Imai and Iri.

**Related work** A well known polyline simplification algorithm is one proposed by Imai and Iri [12]. There are two variants: min-vertices and min-$\varepsilon$. For the min-

vertices variant we compute for each combination of vertices, with some specified measure, the error incurred by the application of the shortcut. If that error is below the $\varepsilon$ value given as a parameter to the algorithm, we include the shortcut into a shortcut graph. To create the simplification, we find the shortest path through the shortcut graph. We create the simplification by creating a polygon consisting of the vertices of the shortest path. The min-$\varepsilon$ applies a binary search on the min-vertices variant to find the simplification with the lowest error, which has the number of vertices as given as a parameter.

Daneshpajouh et al. [4] propose two area measures intended to be used with the aforementioned Imai and Iri algorithm. The first area measure is *sum area*, which we will refer to as *areal displacement*. Areal displacement is the total area that is changed. The other area measure they describe is *diff-area*: the difference in area. Note that if and only if *diff-area* is zero, area has been preserved exactly. They also present an efficient method for computing these error measures for polylines.

Bose et al. [2] propose a polyline algorithm to simplify $x$-monotone paths which can preserve area within a specified limit. A polygon is $x$-monotone if any line drawn perpendicular to the x-axis intersects the polygon at most twice. They were also able to show that for polylines, determining if it can be simplified while preserving area, without using Steiner points, is NP-hard.

Buchin et al. [3] propose two algorithms: one to simplify subdivisions and another to schematize subdivisions. Schematization is about modifying a polyline, polygon or subdivision, such that all edges are at one of the given orientations. For example, an axis-aligned polygon would have all its edges be horizontal or vertical (thus parallel to one of the axes). A subdivision is a planar straight-line embedding of a graph. This means that we can consider a simple polygon to be a basic version of a subdivision. They propose an iterative simplification algorithm. In this algorithm, they make use of a concept called an *edge move*. An *edge move* is where you move, in a three edge segment, the inner edge along the lines of the first and last edge. The inner edge is not allowed to move past the first or last vertex and stays parallel to its original position. If we move the inner edge to either the first or last vertex, we have an edge of length zero and two vertices at same location. This is what we call a full contraction. In this case we can remove the vertex of the inner edge, which is in the same position as either the first of last vertex of the segment. We can also move the inner edge only partially towards the first or last vertex, such that none of the edges have a length of zero. Such a move we call a partial contraction. Both the full and partial contraction either adds area to the polygon or removes it. If we then apply both a full and partial contraction, such that the partial contraction compensates the change in area of the full contraction, we simplify the polygon while preserving area. Note that since an edge move changes the location of the vertices of the inner edge, we may have points at positions none were previously. Such points are called Steiner points. We choose the full and partial contractions based on which have the lowest areal displacement associated with their contractions. The algorithm

halts when the desired number of vertices is reached.

Tutić and Lapaine [17] propose an area preserving polyline simplification algorithm that sequentially collapses zig-zag sections with the help of Steiner points. An iterative algorithm proposed by Kronenfeld et al. [13] shares some similarities with Tutić and Lapaine's algorithm, but it aims to minimize areal displacement while preserving area. The algorithm categorizes three edge segments into S-shapes and C-shapes, then computes a location for the Steiner point that preserves area and minimizes areal displacement, if the Steiner point where to replace the vertices of the inner edge. The segment with the lowest areal displacement when simplified is selected iteratively until the desired number of vertices is reached.

Funke et al. [9] present a subdivision simplification algorithm with topology constraints. They do this with the help of a constrained Delaunay triangulation they continuously maintain. Each iteration they remove a vertex from a degree-2-chain, where the resulting shortcut is valid and some specified conditions hold for the vertex. Valid means that there are no self-intersections introduced and the topology constraint is not invalidated. One of the authors of the aforementioned paper, Mendel [15], proposes a modification to this algorithm to support approximate area preservation. It focuses on adding a constraint that requires the original area to be $\delta$ close to the new area (i.e. $\frac{1}{1+\delta} \cdot \text{originalSize} \leq \text{newSize} \leq (1+\delta) \cdot \text{originalSize}$). Mendel also created an Integer Linear Program to compare his simplifications with.

Haunert and Wolff [10] present both an Integer Linear Program and a heuristic algorithm for simplifying building ground plans. They also prove that the problem of minimizing the number of edges according to a user-defined error tolerance is NP-hard when the output must consist of non-intersecting simple polygons. The heuristic algorithm computes a shortcut graph, which might result in buildings intersecting. If there are any intersections, it removes one of the two arcs involved in each intersection from the shortcut graph and computes a new solution. It will repeat this until a solution is found without intersecting polygons.

Tong et al. [16] propose a radically different solution to the problem. They use *structured total least squares adjustment with constraints* (STLSC) to simplify a polygon while preserving the area. Their framework for their STLSC method consists of a linear fitting model, combined with a constraint to ensure that the new polygon is close to the original polygon and a constraint for area preservation, and the optimization model of STLSC. With this framework they are able to simplify a polygon and also preserve area.

Ma et al. [14] compare four simplification algorithms with three measures. The measures are based on the Douglas-Peucker simplification algorithm [6]. Based on this, they define three measures. The first is the $x$ measure which is the maximum distance of a point on a segment to the line formed by the line through the first and last vertex of the segment. The length of the segment from first vertex of the aforementioned segment to the last vertex of the segment is the measure $d$. Finally the area formed by the $x$ and $d$ is the measure they call *area*.

The simplification algorithms they use are Douglas–Peucker [5], Visvalingam and Whyatt [18], Wang and Müller [20] and Ai et al. [1]. The simplifications created with the aforementioned *area* measure are the lowest in quality.

# Chapter 2

# Preliminaries

Before proceeding further, we need to define area preservation. We define area preservation as ensuring that the area of the simplification is equal to the area of the original polygon. It is not always possible to preserve area exactly. To ensure exact area preservation, we need to use *Steiner points*. *Steiner points* are vertices that did not exist in the original polygon. Certain algorithms that we will discuss later use Steiner points.

An area-based measure, that is used by some simplification algorithms we will discuss later, is *areal displacement* [13, 3]: *areal displacement* is the total area that is changed, also known as the symmetric difference. If we use the simplification in Figure 2.1 as an example, where we assume the interior of the polygon to be to the left, we have two regions: region $A$ (adding area to the polygon) and region $R$ (removing area from the polygon). The area preservation in the example means that $|A| - |R| = 0$, where $|X|$ denotes the area of a region $X$. Areal displacement can be computed as $|A| + |R|$. Note that the style used in the figure will be used throughout the thesis. Black dots are unchanged points, grey dots and line segments are points and edges that are being removed during the simplification process and blue vertices and edges are those that are added as part of the simplification. In the figures we will assume the interior of the polygon to be to the left unless stated otherwise.

The notation that we use to refer to the edge between vertices $a$ and $b$ is $ab$. To refer to a line segment between points $a$ and $b$ we use $\overline{ab}$. When we want to refer to a line that intersects points $a$ and $b$ we use $\overleftrightarrow{ab}$ and to refer to a new line we use $\overleftrightarrow{x}$, where $x$ is the variable name given to that line.

We will focus on the algorithms by Imai and Iri, Kronfeld et al. and Buchin et al. and for conciseness we will refer to them as: *II* for Imai and Iri, *KSBB* for Kronenfeld et al. and *BMRS* for Buchin et al.

We will use metrics as part of our experiment to measure the quality of simplifications. In Chapter 3 we will discuss why we selected these metrics, but we will provide definitions here. The selected metrics are as follows:

- Hausdorff distance as defined in Definition 1 (from Huttenlocher et al.[11]).

- Modified Hausdorff distance as defined in Definition 2 (from Dubuisson and Jain [7].
- Discrete Fréchet distance as defined in Definition 3 (from Vodolazskiy [19]).
- Symmetric difference as defined in Definition 4.

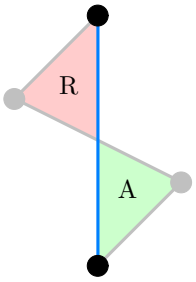



Figure 2.1: An example of a simplification, where we assume the left side to be the interior of the polygon.

**Definition 1** *Given two polygons A and B, the Hausdorff distance is* $H(A,B) = \max(h(A,B), h(B,A))$*, where h is the directed Hausdorff distance. The directed Hausdorff distance is defined as* $h(A,B) = \max_{a \in A}(d(a,B))$*, where* $d(a,B)$ *is the shortest distance from point a to polygon B.*

**Definition 2** *Given two polygons A and B, the Modified Hausdorff distance is* $MH(A,B) = \max(mh(A,B), mh(B,A))$*, where mh is the directed Modified Hausdorff distance. The directed Modified Hausdorff distance is defined as* $mh(A,B) = \frac{\sum_{a \in A} d(a,B)}{|A|}$*, where* $d(a,B)$ *is the shortest distance from point a to polygon B.*

**Definition 3** *Given a sequence A consisting of points* $(a_1, a_2, ..., a_n)$ *and another sequence B likewise consisting of points* $(b_1, b_2, ..., b_m)$*, we define a coupling of these two sequences. Specifically a coupling* $L = ((a_{x_1}, b_{y_1}), (a_{x_2}, b_{y_2}), ..., (a_{x_l}, b_{y_l}))$ *of distinct pairs, such that* $x_1 = 1$*,* $y_1 = 1$*,* $x_l = n$ *and* $y_l = m$ *and* $x_{t+1} = x_t$ *or* $x_{t+1} = x_t + 1$ *and either* $y_{t+1} = y_t$ *or* $y_{t+1} = y_t + 1$ *for all* $1 \leq t \leq l-1$*, where l and t are integers. We define the length* $||L||$ *of a coupling L as the largest distance between pairs of points in the sequence:* $||L|| = max_{t \in \{1,...,l\}}(d(a_{x_t}, b_{y_t}))$*. The set of all couplings for sequences A and B we denote as* $C(A,B)$*. With these definitions we can define the Discrete Fréchet distance for two sequences of points as:* $\delta_{dF}(A,B) = min_{L \in C(A,B)}||L||$*. To extend this definition to closed sequences we need to define a cyclic shift: a cyclic shift* $S(A,s)$ *of a sequence A, as defined earlier, by an integer number* $s \in \{0, 1, ..., n\}$ *is a sequence* $A' = S(A,s) = (a'_1, a'_2, ..., a'_n)$ *such that* $u'_i = u_{i+s}$ *for* $i + s \leq n$ *and* $u'_i = u_{i+s-n}$ *for* $i + s > n$*. Given closed sequences A and B consisting of points* $(a_1, a_2, ..., a_n)$ *and* $(b_1, b_2, ..., b_m)$ *respectively, we define the discrete Fréchet distance for closed sequences as:* $\delta_{dcF}(A,B) = min_{s_A \in \{0,...,n\}}(min_{s_B \in \{0,...,m\}}(\delta_{dF}(S(A,s_A), S(B,s_B))))$*.*

**Definition 4** *Given two polygons A and B, we define the symmetric difference as the sum of area of all regions that are in the interior of polygon A or B, but not both.*

# Chapter 3

# Methodology

We will do experiments with the algorithms, which necessitates three things besides the algorithms themselves. Firstly, we need data sets to run the algorithm on. Secondly, we need to give the algorithms the number of vertices the algorithms should simplify to, which allows us to ensure that the simplifications have a similar number of vertices. As a result, we must determine a number or multiple numbers, with which we will run the algorithms. Lastly, we need to be able to analyze the simplifications objectively, thus requiring us to choose metrics. We defined the metrics earlier on in Chapter 2, but we will end this section with the reasoning behind our rationale for selecting the chosen metrics. We implement the metrics and the algorithms in the C++ library called CartoCrow, which can be found on Github at `https://github.com/tue-alga/cartocrow`. The implementation of the simplification algorithms and metrics can be found at: `https://github.com/dmbuzink/cartocrow`.

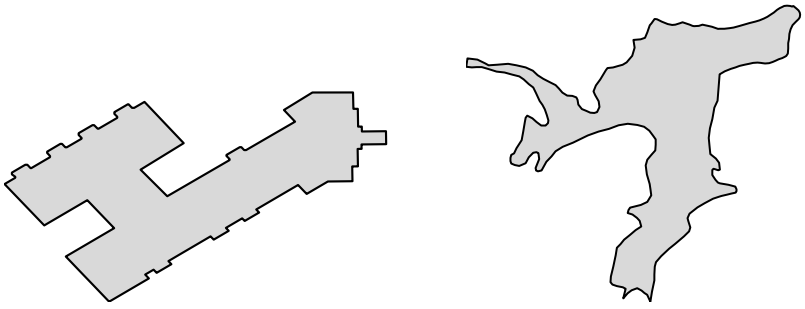
**Data sets** We want to apply the algorithms on more than one data set. Specifically at least two different types of data sets. By using different data sets, we prevent making conclusions that are applicable only to one type of input. However, we will limit the data to simple polygons as there is an increased complexity when dealing with complex polygons. Let us discuss the data sets we use. The first data set is of the District of Columbia and contains information about historic landmarks. We will use the outlines of these landmarks as our first data set. More information on this data set can be viewed in the description below. Our second data set consists of lakes from the National Hydrography Dataset. This data set contains water-related geographic data from the United States of America. More information about this data set can be seen below. A polygon from each of the respective data sets is shown in Figure 3.1 for illustrative purposes. To allow for proper comparisons, we will scale all inputs to an area of 10,000. We do not simplify the inputs if the input has less than 100 vertices and if the file size is larger than 20 KB due to high run times on the II algorithm.

**Name** District of Columbia Landmarks (DCL)

**Description** This data set contains information, including building outlines, of historic landmarks as officially designated by the District of Columbia. More information about the data set can be found at `https://data.world/codefordc/historic-landmarks-polygons`.

**Name** National Hydrography Dataset (NHD)

**Description** This data set contains water-related geographic data in the US, enabling us to test simplifications on naturally occurring geographic shapes. More information about the NHD can be found at `https://www.usgs.gov/national-hydrography/national-hydrography-dataset`.



(a) (b)

Figure 3.1: Example of (a) a building from the historic landmarks data set of the District of Columbia (with id 130581) and (b) a lake from the National Hydrography Dataset (with id 9279).

**Size of simplifications** We require simplifications with similar numbers of vertices, so we can properly analyze them. The number of vertices we will compute the inputs to is 25% and 75% of the original number of vertices. This allows us to both analyze simplifications where many vertices need to be simplified away and where few vertices need to be simplified away.

**Metrics** For objective analysis of the simplifications, appropriate metrics are necessary, which we defined earlier in Chapter 2. The first metric we will use is the Hausdorff distance. This tells us about how far apart the polygons are at the maximum. To ascertain the average distance between the polygons, we use the Modified Hausdorff distance. According to Dubuisson and Jain [7], this metric is effective for measuring the similarity between two polygons. To get information about the distance between curves we use the Discrete Fréchet distance. We use the Discrete Fréchet distance and not the continuous Fréchet distance as it is significantly easier to implement. Lastly to know how much area is not shared between the polygons we can use the symmetric difference.

# Chapter 4

# Algorithms

In this chapter, we discuss the three implemented algorithms: Imai and Iri [12], Kronenfeld et al. [13], and Buchin et al. [3]. This section provides a detailed description of the algorithms, including the area preserving and non-area preserving variants. We will likewise discuss some implementation details where our implementation differs from the proposed algorithm. Despite any variants in implementation, all algorithms yield correct simplifications.

## 4.1 Imai and Iri

The min-vertices variant minimizes the number of vertices for a given error threshold. This error threshold we refer to as $\varepsilon$. We then, for each pair of distinct vertices, compute the error that a shortcut between these two vertices would incur. It is a shortcut as this means we will skip all vertices between the aforementioned vertices. If the error is below $\varepsilon$ we add it to a graph. We will do this for each pair of distinct vertices, such that we have a graph with the original edges and all shortcuts with an error below $\varepsilon$. An example of such a shortcut graph is shown in Figure 4.1a. To create a simplification with the error of each shortcut below $\varepsilon$, we find the shortest path through the shortcut graph from the start point to the end point and let the vertices that comprise this path, be the vertices that comprise our simplified polygon. The simplification resulting from the shortcut graph of Figure 4.1a is shown in Figure 4.1b. This is appropriate for polylines, which the algorithm was originally intended for. However, when applied to polygons, we do not have a start or end point. Therefore, we need to find the shortest cycle through the shortcut graph instead.

The min-$\varepsilon$ variant minimizes the error for a simplification with a given number of vertices. We can minimize the error by applying binary search on the error in the min-vertices variant. In our experimentation we will use the min-$\varepsilon$ variant as this allows us to properly compare the simplifications of the algorithms as the simplifications will have the same (or at least a similar) number of vertices.

The error in the min-$\varepsilon$ variant is computed using the Hausdorff distance as a measure (see Definition 1).

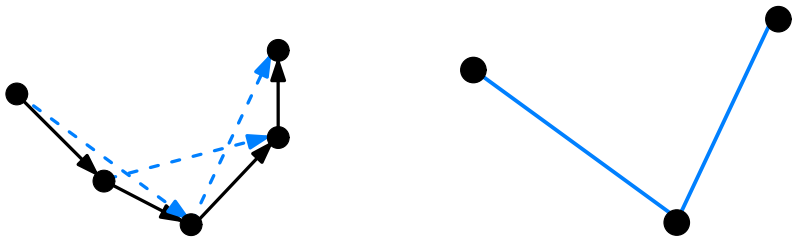



Figure 4.1: Example of: (a) a polyline with shortcuts that are not edges as the dashed blue lines and (b) the simplified polyline as a result of the Imai and Iri algorithm.

**Area preserving variant** The area preserving variant of the algorithm by Imai and Iri uses a measure to compute the difference in area between the original polygon and the polygon if we were to use the shortcut in question. Our implementation of the area preserving variant follows the area difference algorithm proposed by Daneshpajouh et al. [4]. This means that when we apply the algorithm, a shortcut is only added to the shortcut graph, if and only if the difference in area is below $\varepsilon$. For the example in Figure 4.2, we would compute the error as: $||A| - |R||$, where $|X|$ is the area of region $X$. The rest of the algorithm works exactly the same and we only use a different measure for the error computation during the creation of the shortcut graph.

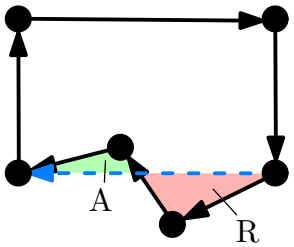



Figure 4.2: An example of an area preserving shortcut, with the error $||A| - |R||$, where $|X|$ is the area of region $X$.

**Implementation** Daneshpajouh et al. also propose how to compute the area difference [4]. Since run time is not a crucial factor for our research, we compute the area difference by straightforwardly computing the area of the polygon with the shortcut applied and taking the absolute difference with the area of the original polygon.

## 4.2 Kronenfeld et al.

Kronenfeld et al. [13] propose an iterative simplification algorithm that aims to minimize areal displacement while preserving area. The algorithm distinguishes three edge segments into S-shaped curves and C-shaped curves, which are illustrated in Figures 4.3a and 4.3b. We want to collapse the segment in such a way, that we place the Steiner point where area is preserved. The vertices of the segment we will call: $a$, $b$, $c$ and $d$. To preserve area, we compute a line parallel to $\overleftrightarrow{ad}$ such that, if we place the Steiner point somewhere on that line, we will preserve area. This line we will call $\overleftrightarrow{e}$. We place the Steiner point such that areal displacement is minimized. Kronenfeld et al. show that the areal displacement is minimized for the point of intersection of $\overleftrightarrow{e}$ with either $\overleftrightarrow{ab}$ or $\overleftrightarrow{cd}$. Which of the two depends on some conditions described in further detail in their paper. Note that the only case where this intersection point does not exist, is when two edges are collinear. If we have two edges that are collinear, we can simply remove the middle vertex and we preserve area with no areal displacement. An example for both an S-shaped segment and a C-shaped segment is shown in Figure 4.3. We see in the example how the additional region $A$ is equal in size to the removed region $R$ in both cases, thus preserving area. We repeat collapsing segments with the lowest associated areal displacement until the desired number of vertices is reached.

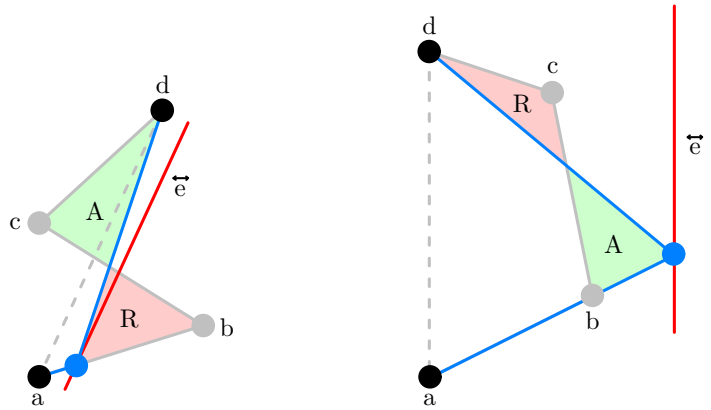



Figure 4.3: Examples of: (a) S-shaped segment Kronenfeld et al. simplification and (b) C-shaped segment Kronenfeld et al. simplification.

There is one case where it is unclear where exactly the Steiner point should be placed. In this case we have a C-shaped segment, where the distance from $b$ to $\overleftrightarrow{ad}$ is equal to the distance from $c$ to $\overleftrightarrow{ad}$. This is equivalent to $\overleftrightarrow{ad}$ being parallel to $\overleftrightarrow{bc}$. In this case we can minimize areal displacement by placing the Steiner point anywhere on $\overleftrightarrow{e}$ between the points of intersection of $\overleftrightarrow{e}$ with $\overleftrightarrow{ab}$ and $\overleftrightarrow{cd}$. This line

segment we will refer to as $\bar{e}$. This case is shown in Figure 4.4a, with $\bar{e}$ shown as the full green line segment. The paper does not specify where the Steiner point should be placed on $\bar{e}$, therefore we have to decide where we will place it. We could place it, for example in the middle as shown in Figure 4.4b, which means we get three regions that are modified. To stay consistent with the rest of the algorithm and simplify our implementation, we choose to place the Steiner point on either the intersection $\overleftrightarrow{e}$ with $\overleftrightarrow{ab}$ or $\overleftrightarrow{cd}$. This simplifies implementation as this means that we will have two regions in all cases. Since we want consistent results we have to pick one of the two and we arbitrarily choose $\overleftrightarrow{ab}$ as we see no reason to chose one above the other.

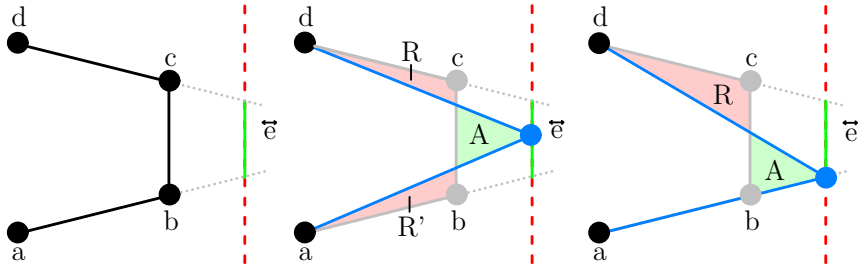



Figure 4.4: Examples of: (a) a C-shaped segment where the distance from $b$ to $\overline{ad}$ is the same as from $c$ to $\overline{ad}$, (b) the aforementioned segment simplified by placing the Steiner point in the middle and (c) the aforementioned segment simplified by our chosen method.

**Non-area preserving variant** The non-area preserving variant works similarly to the area preserving variant. It differs only in where it places the Steiner point. We still use areal displacement as the error to minimize per iteration to limit the differences between the algorithms.

Where we place the Steiner point for S-shaped and C-shaped segments is different. Let us first look at the S-shaped segments. Here we place the Steiner point in the middle of edge $bc$. This results in a simplification as shown in Figure 4.5. We place it here as it equally considers the locations of points $b$ and $c$. For the placement of the C-shaped segments we experimented with different methods. We tried to find a placement such that the Hausdorff distance is minimized, but we were not successful. As part of our experimentation to find such a placement, we created many other methods to determine the location for the Steiner point. Some simplification examples for these methods are shown in Figure 4.7. We use following method as it seems to us to preserve the shape of the C-shaped segment the best. Firstly, we obtain lines $\overleftrightarrow{ac}$ and $\overleftrightarrow{bd}$ and move them such that they pass through vertices $b$ and $c$ respectively. This results in an intersection on the "outside" side of the C-shape, which is a suitable simplification based on the

compromise between simply removing vertex $b$ and simply removing vertex $c$. In Figure 4.6a the line $\overleftrightarrow{ac}$ is shown as the orange dashed line and $\overleftrightarrow{bd}$ as the purple dotted line. The lines going through $b$ and $c$ always result in an intersection on the "outside" of the C-shape. This follows from the fact that since we have a C-shape, there must be an intersection caused by $\overleftrightarrow{ac}$ and $\overleftrightarrow{bd}$, where the point of intersection is in the polygon of vertices $a$, $b$, $c$ and $d$. Since we move the lines such that they go through $b$ and $c$, the intersection point must also be on the other side of the $\overleftrightarrow{bc}$. We give some examples in Figure 4.6.

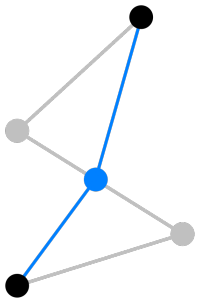


Figure 4.5: Simplification of a S-shaped segment by our non-area preserving variant of the Kronenfeld et al. algorithm.

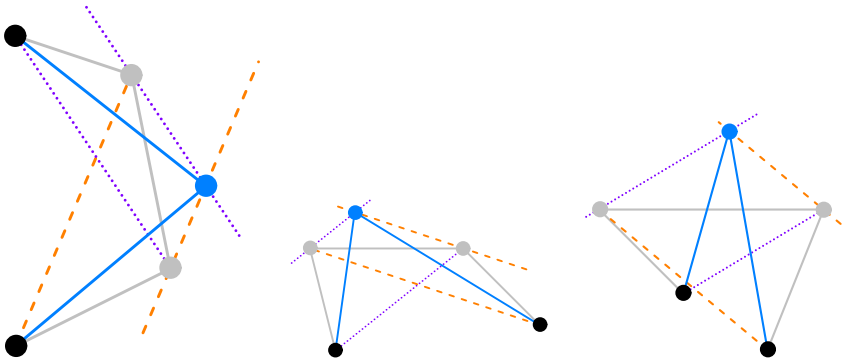
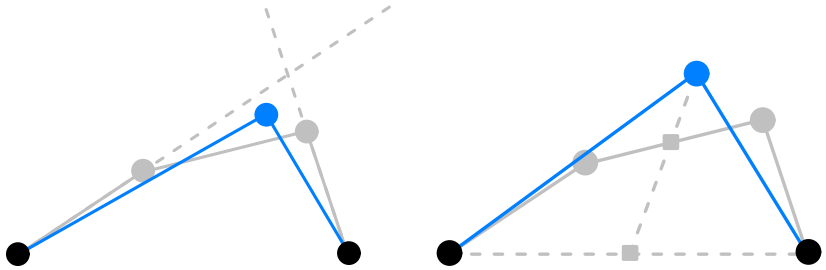



Figure 4.6: Different types of C-shaped segment and how we simplify them in our non-area preserving variant of the Kronenfeld et al. algorithm.

## 4.3 Buchin et al.

Buchin et al. [3] propose an iterative area preserving simplification algorithm. We start by going through all three edge segments. For each segment we consider moving the inner edge along the lines of the first and last edge, while not changing the orientation of the inner edge. We limit the movement of the inner



(a) (b)

Figure 4.7: Some alternative methods of simplifying C-shaped segments. In (a) we form a polygon on the outside of $bc$ and place the Steiner point at the centroid. For the method of (b) we draw a line through the middle of $\overline{ad}$ and $\overline{bc}$, with a length between the Steiner point and the middle of $\overline{bc}$ that is half of the distance between the middle of $\overline{ad}$ and $\overline{bc}$.

edge to the first and last vertices. This is what we call an *edge move*. Due to this edge move we either add or remove area. In Figure 4.8a we have an example of a three edge segment. If we move the inner edge fully towards the right most vertex, we add the area of region $A$. Instead, if we move it fully towards the left most vertex, we remove the area of region $R$. If we were to move the edge fully to the right, we get the segment in Figure 4.8b. When we fully move the inner edge to one of the outer vertices of the segment, we have two vertices at the same location, thus allowing us to remove one of them. We call such an edge move a *full contraction*. By applying such a full contraction, we simplify the polygon by a single vertex. At each iteration, we apply the full contraction with the lowest areal displacement. We can also move an edge only partially towards the first or last vertex. This we refer to as a *partial contraction*. To ensure area preservation, we apply a partial contraction, where we move the inner edge such that the area added or removed by the edge move, compensates the change in area of the full contraction. We apply the partial contraction with the lowest areal displacement, which also does not share an edge with the three edge segment of the full contraction. See Figure 4.8c for an example of a partial contraction. These examples are S-shaped segments, but we can also apply edge moves on C-shaped segments. This is shown in Figure 4.9. As shown in Figure 4.9c, depending on the orientation and positions of the edges, there could be no intersection at the outward side of the inner edge. In that case there will only be a positive or negative contraction region. After having done both the full and partial contraction, we have simplified the polygon by a single vertex while preserving the area. We then repeat applying these steps until we have reached the desired amount of vertices.

**Non-area preserving variant** In the non-area preserving variant, we apply only the full contraction at each iteration, simplifying the polygon by removing one vertex, without considering area preservation. This allows us to explore

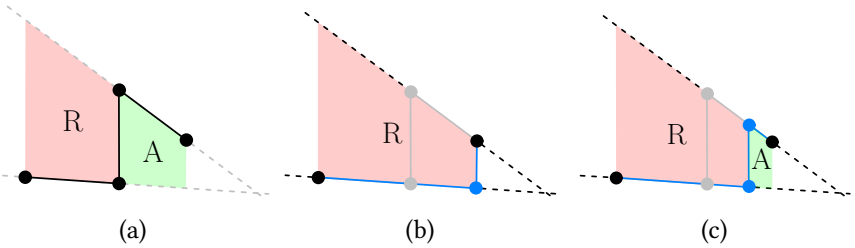



Figure 4.8: Examples of: (a) a S-shaped three edge segment with both contraction regions, (b) full contraction of the segment from a and (c) a partial contraction of the segment from a.

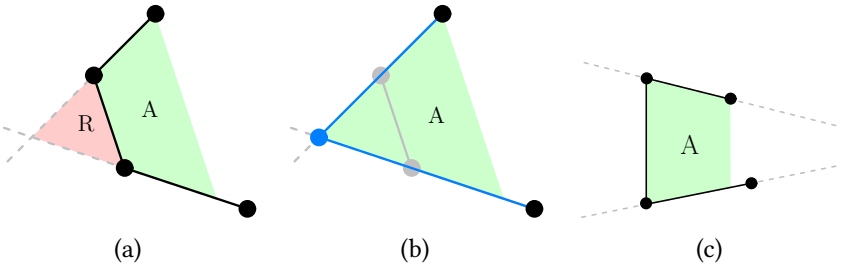



Figure 4.9: Examples of: (a) a C-shaped segment with both contraction regions, (b) a full contraction of the segment from a (c) a partial contraction of the segment from a.

whether it is advantageous to modify the polygon more than necessary at each iteration to preserve area, as is done in the area preserving variant.

**Implementation** There is one way our implementation differs from the algorithm described by Buchin et al. It is important to state explicitly that the simplifications that are computed are exactly the same. To ensure that we do not have self intersections, we need to be careful of which three edge segments we contract. The algorithm keeps track of the number of edges that a contraction region intersects with and calls this the *blocking number*. With this we can compute the blocking numbers once for each three edge segment, but only update what might have changed per iteration, improving the run time. However, to simplify implementation we currently recompute all blocking numbers at each iteration. This implementation is fast enough for the data sets we used and the resulting simplifications are exactly the same.

# Chapter 5

# Results

In this chapter, we present the results of our experiments. We ran for our experiments both area preserving and non-area preserving variants of II, KSBB, BMRS on 42 polygons from the DCL data set and 46 polygons from the NHD data set. To refer to area preservation we will use the abbreviation *AP* and for non-area preservation we will use *NAP*. We simplified each input polygon for every combination of algorithm and variant to 25% and 75% of the original number of vertices. For each simplification we compute the Hausdorff distance (HD), Modified Hausdorff distance (MHD), Discrete Fréchet distance (DFD) and symmetric difference (SD).

We focus primarily on the differences between the average and maximum values of the 25% similarity metrics of the AP and NAP variants. The detailed averages, maxima and direct comparisons, including the 75% simplification results, can be found in Appendix A. In Figure 5.1 we show the average and maximum values for the algorithms for both data sets. In Figures 5.2 and 5.3 we show the percentages of simplifications produced by the area preserving variant having a lower score, for the corresponding metric, than the non-area preserving variant for the DCL and NHD data sets respectively. We disregard results where the values were equal. To also get a visual understanding of how the simplifications differ we show for input 131054 from the DCL data set all 25% simplifications in Figure 5.4. This helps with understanding where weaknesses may lie for certain algorithms and specific variants.


Figure 5.1: Averages and maxima for the DCL and NHD data sets for the metrics Hausdorff distance (a) and (b), Modified Hausdorff distance (c) and (d), and symmetric difference (e) and (f).

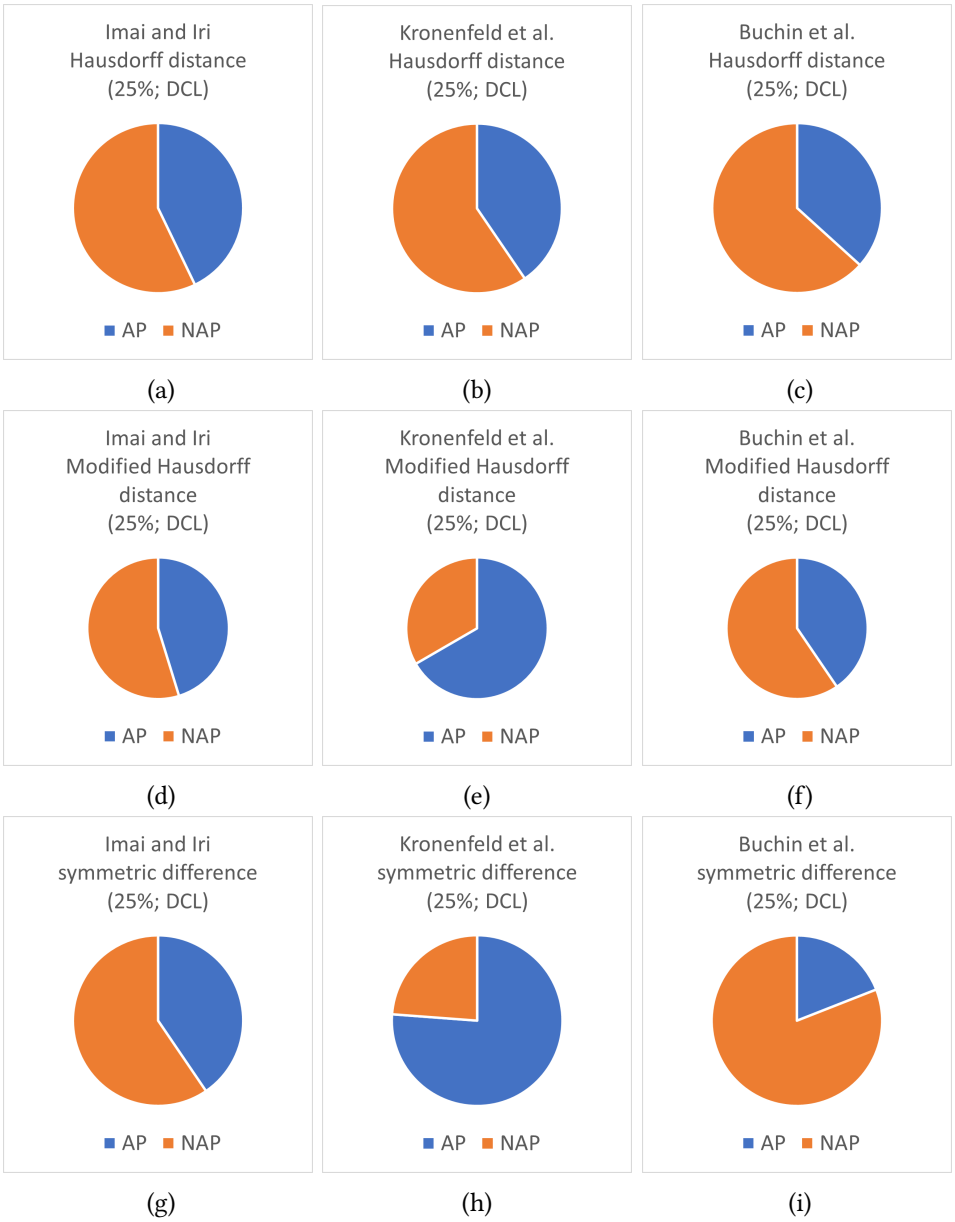



Figure 5.2: Direct comparisons between AP and NAP variants for the metrics: Hausdorff distance, Modified Hausdorff distance and symmetric difference for the DCL data set. With Imai and Iri for Hausdorff distance (a), with Kronenfeld et al. (b), with Buchin et al. (c), with Imai and Iri for Modified Hausdorff (d), with Kronenfeld et al. (e), with Buchin et al. (f), with Imai and Iri for symmetric difference (g), with Kronenfeld et al. (h) and with Buchin et al. (i).

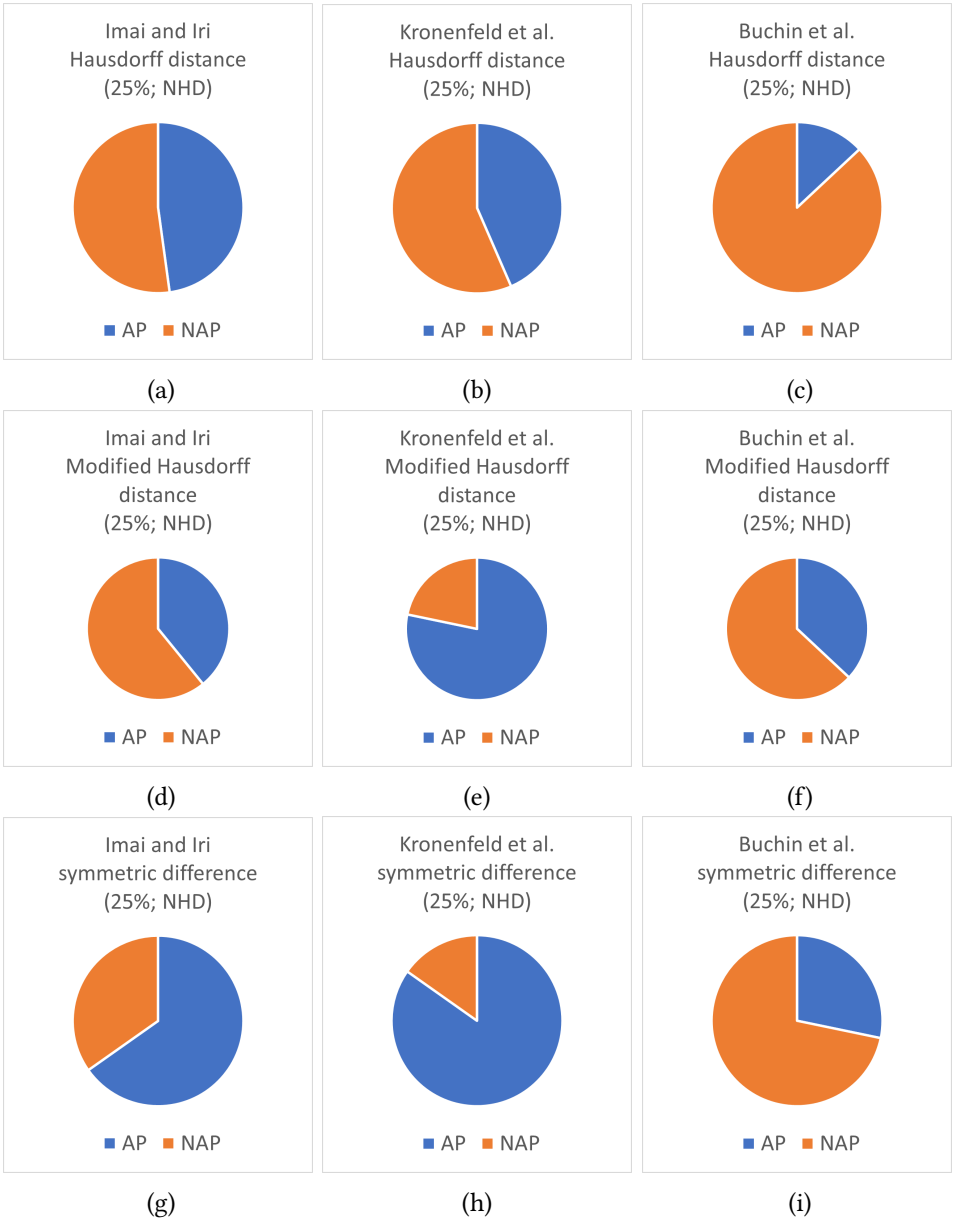



Figure 5.3: Direct comparisons between AP and NAP variants for the metrics: Hausdorff distance, Modified Hausdorff distance and symmetric difference for the NHD data set. With Imai and Iri for Hausdorff distance (a), with Kronenfeld et al. (b), with Buchin et al. (c), with Imai and Iri for Modified Hausdorff (d), with Kronenfeld et al. (e), with Buchin et al. (f), with Imai and Iri for symmetric difference (g), with Kronenfeld et al. (h) and with Buchin et al. (i).

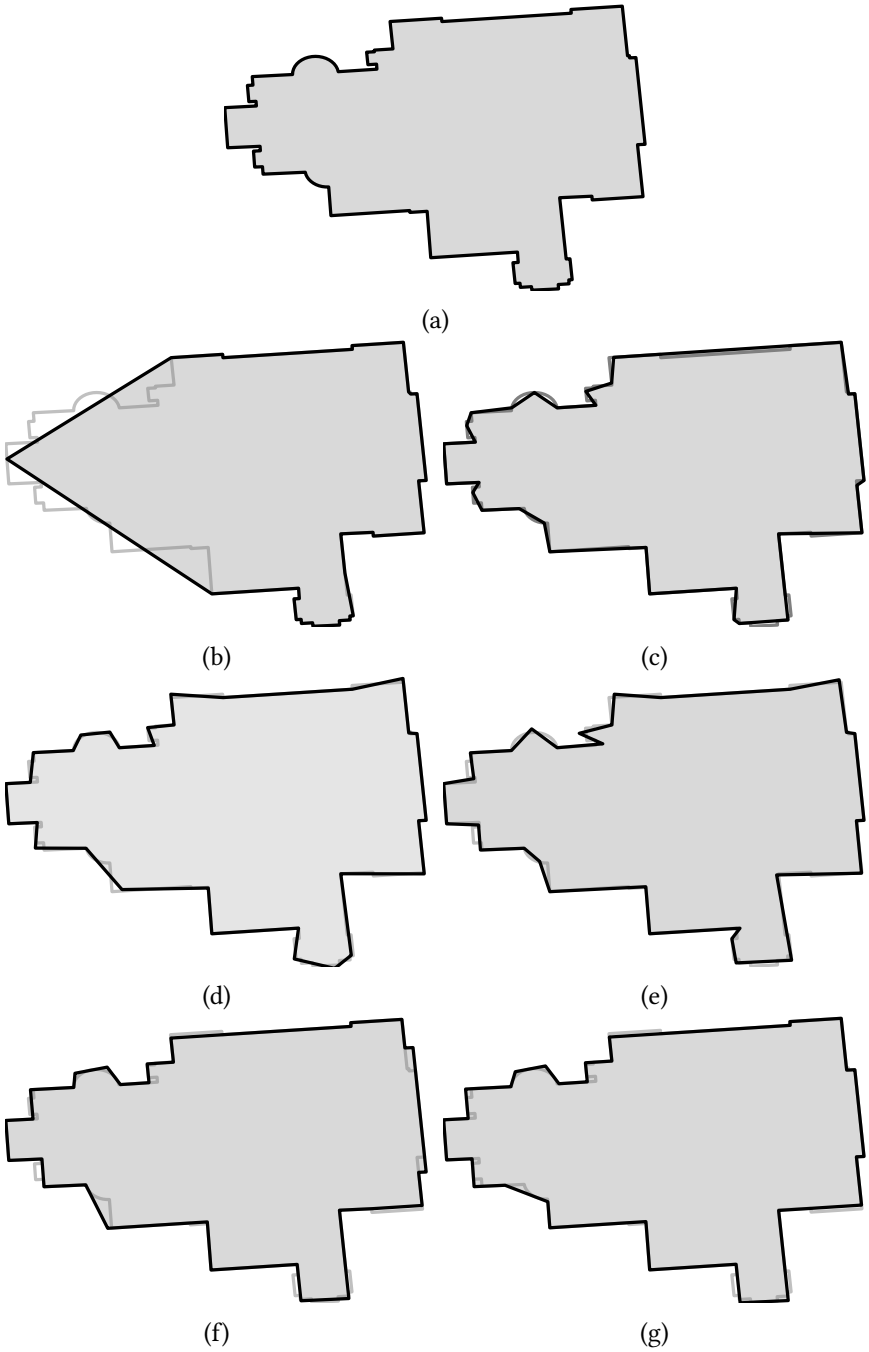



Figure 5.4: Original polygon 131054 from data set DCL (a) and simplifications produced by: AP variant of II (b), NAP variant of II (c), AP variant of KSBB (d), NAP variant of KSBB (e), AP variant of BMRS (f) and NAP variant of BMRS (g). The grey line in the simplifications is the original polygon.

# Chapter 6

# Discussion

We showed the results of the metrics of the simplifications created by the implemented algorithms in the previous chapter. In this chapter we discuss the results and the overall research, including its limitations.

**Discrete Fréchet distance** The Discrete Fréchet distance proved to be an unreliable metric for the quality of simplifications. The problem with this metric is its discrete nature. In Figure 6.1 we have an example of two polylines and the Discrete Fréchet distance as the dashed red line. These segments are similar and thus we would expect the Discrete Fréchet distance to be low. Yet it turns out to be high, because it compares the location of points. Due to this issue, the Discrete Fréchet distance is an unreliable metric for simplification quality and we will not use it to discuss the quality of the simplifications.

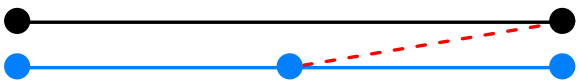


Figure 6.1: An example of the Discrete Fréchet distance, shown as the dashed red line, between two similar segments.

## 6.1 Research questions

We will address each research question and provide answers with the help of the data from the previous chapter. Firstly, we look at the differences between the non-area preserving II algorithm and the area preserving variant, as proposed by Daneshpajouh. After that we look at the differences between the area preserving KSBB algorithm and our non-area preserving counterpart. Lastly, we look at the area preserving BMRS algorithm and our non-area preserving counterpart.

**Imai and Iri** We start by looking at the averages. The average Hausdorff distance for the AP variant is very significantly bigger than for the NAP variant. This is also the case for the averages of the Modified Hausdorff distance in both

data sets and the symmetric difference in both data sets. The relative difference between the AP and NAP variants is smaller for the symmetric difference, compared to the Hausdorff and Modified Hausdorff distance, yet still large. Next we look at the maxima. The maximal Hausdorff distance is also very significantly greater for the AP variant in both data sets than the NAP variant. The same holds for the maximal Modified Hausdorff distance and symmetric difference in both data sets, where we again see a lesser relative difference for the symmetric difference than the Hausdorff or Modified Hausdorff distance. Finally we look at the direct comparisons. We see that the Hausdorff distance is more frequently lower for the NAP variant than the AP variant in the DCL data set, although not by that much. When we look at the comparison for the NHD data set, we see that the NAP variant is still more frequently the one with a lower Hausdorff distance, yet it is near 50%. The direct comparison for the Modified Hausdorff is equal between the data sets and we see that the NAP variant more frequently produces simplifications with a lower Modified Hausdorff distance, yet again not by that much. The direct comparison for the symmetric difference has a really interesting result. For the DCL data set we see the NAP variant outperform the AP variant on the symmetric difference about three out of five times. However, for the NHD data set we see a reversal, where about three out of five times the AP variant produces a simplification with a lower symmetric difference than the NAP variant.

Based on these observations, it seems like the AP variant of II is inconsistent in regards to the quality of the simplifications. For the direct comparison the AP variant does not do particularly bad, even outperforming the NAP variant for the symmetric difference for the NHD data set. However, we see big differences in the averages. Taking the maxima also into account, it seems like the AP variant can produce good simplifications, but also regularly produces extremely poor simplifications. Most likely this is due to the fact that the area difference is sometimes equal (or close to equal) to areal displacement which, based on the quality of the other algorithms, is a good measure. In the cases where many shortcuts are not, we get the low quality simplifications.

**Kronenfeld et al.** We, again, start by looking at the averages. The average Hausdorff distance is slightly lower for the NAP variant than the AP variant for both data sets. For the Modified Hausdorff distance we see the reverse as it is slightly higher for the NAP variant than the AP variant for both data sets. We also see that the AP variant has a lower average for the symmetric difference for both data sets than the NAP variant. Continuing on to the maximal values, we see that the maximum Hausdorff distance is less for the AP variant than for the NHD data set, but greater for the DCL data set. The AP variant has a slightly lower maximal Modified Hausdorff distance than the NAP variant in both data sets. The AP variant also has lower maximal symmetric difference scores in both data sets, but with bigger relative differences between the scores. Finally for the direct comparisons we see that for both data sets, the NAP has a lower Hausdorff

distance about three out of five times. For the Modified Hausdorff distance we see that three out of five times the AP variant has a lower score for the DCL data set and for the NHD data set the AP variant is better three out of four times. The symmetric difference is lower for the AP variant about three out of four times in the DCL data set and seven out of eight times in the NHD data set.

We see that the AP variant does better in both data sets than the NAP variant overall, with the AP variant performing especially well for the NHD data set. More specifically, although it more commonly has a slightly higher Hausdorff distance, the AP variant has frequently a slightly lower Modified Hausdorff distance and symmetric difference. Therefore it seems that the AP variant seems to sacrifice being close to the original polygon (and thus having a higher Hausdorff distance), for staying closer to the shape overall. It could also be the case that how the NAP variant collapses segments results in a lower Hausdorff distance and higher Modified Hausdorff distance and symmetric difference, irrelevant of the algorithm preserving area or not.

**Buchin et al.** For the Buchin et al. algorithm we see that the average Hausdorff distance for the AP variant is slightly higher for the DCL data set and slightly lower for the NHD data set. The average Modified Hausdorff distance is only by a tiny amount lower for the NAP variant in both data sets. For the symmetric difference we see that the NAP variant is slightly lower for both data sets. Continuing on to the maximal scores, we see that the maximal Hausdorff distance is significantly greater for the AP variant than the NAP variant. For the maximal Modified Hausdorff distance we have almost the same score for the DCL data set and the score of the NAP variant is lower by a small margin for the NHD data set. The maximal symmetric difference of the NAP variant has lower scores for both data sets. Lastly, we see that for the direct comparisons, we have frequently lower scores for the NAP variant for the Hausdorff distance measure, especially for the NHD data set. For the Modified Hausdorff distance the NAP variant outperforms the AP variant about five out of eight times for both data sets. The symmetric difference is very frequently lower for the NAP variant for the DCL data set and frequently for the NHD data set.

From this we can see that the AP variant does generally perform worse than the NAP variant. This is not entirely surprising, since the AP variant of BMRS, modifies twice as much area per iteration compared to the NAP variant. Therefore, we modify area that is not required to simplify. What is somewhat unexpected is that the symmetric difference scores are not that different between the variants. Since the AP variant modifies twice as much area, we would expect the symmetric difference to be about twice as big as the NAP variant, yet this is clearly not the case. We have two explanations for why this might be the case. Firstly, we know that the partial contraction is applied on a segment with one of the lowest areal displacements. This means that if we modify the contraction region for the segment, we lower the areal displacement of the segment for the next iteration

even further. Therefore. it is likely to become the next full contraction in one of the following iterations. Because of this, much of the added areal displacement might be compensated by a full contraction with a lower areal displacement in of the later iterations. It should be noted that this will not happen every time, as the next iteration might have another segment with a lower areal displacement due to the change of the polygon. Another reason for the close symmetric difference score could be that area preservation prevents the shape of the polygon straying too far from the polygon across iterations. Therefore, for example, preventing compounding errors from occurring and thus improving the simplification quality. If this is the case, it would seem that the cost of modifying the polygons additionally as is proposed by the authors, is higher than the benefit overall for simple polygons.

## 6.2 The difference between area preservation and non-area preservation across the algorithms

We will now look at the differences between the AP and NAP variants across algorithms. To understand what the effect is of area preservation, it is also important to consider *why* area preservation might help with polygon simplification. One issue we can theorize that area preservation can help with is in regards to compounding errors. When you have an iterative simplification algorithm, it will find the best next simplification for the *current* version of the polygon. Current as there might have been a number of simplifications that have been already applied on the polygon. Hence we might have compounding errors, where we get many simplifications in a specific region, such that each step has a low error or areal displacement. However, the polygon may have shifted away from the original polygon across iterations. An example of such an compounding error is shown in Figure 6.2 in the circle in blue. We theorize that it is here where area preservation makes a difference. This could also explain why the area preserving variant of KSBB outperforms its non-area preserving variant more frequently than the BMRS variant, as KSBB preserves area at the simplification, whereas BMRS does not.

To investigate this hypothesis we modify the Imai and Iri algorithm. We create an iterative version of the Imai and Iri algorithm, that at each iteration applies the shortcut with the lowest error. The error is still computed in the same manner as the normal Imai and Iri variants. In this iterative version we see that the AP variant does better in relation to the NAP variant (with the Hausdorff distance as the error measure) compared to the variants of the normal Imai and Iri algorithm. This is derived from the data in Figure 6.3. It even does better in the 25% case for data set DCL. This data suggests that area preservation does indeed help prevent compounding errors in iterative algorithms.

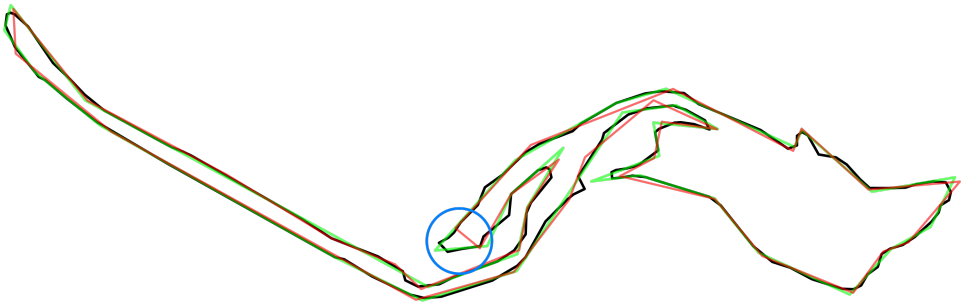


Figure 6.2: The simplification of input 27184 from the NHD data set. The black line is the original polygon, the green line is the simplification of the area preserving variant of the Imai and Iri algorithm and the red dashed line is the non-area preserving simplification. An example of a compounded error is found in the blue circle.

## 6.3 Limitations

The research has some limitations. We have only three algorithms for which we could compare the AP and NAP variants. In our experiment we have only one algorithm that was originally non-area preserving and changed to be area preserving, yet it is one of a significantly inferior quality compared to all other algorithms. This brings into question the value of those results. The NAP variant of II also does not preserve area exactly. It only attempts to preserve area as much as it is able to.

Another limitation is in regards to data sets. We have two data sets that are quite different, thus ensuring that the results are unlikely to be only relevant to these types of polygons. However, more data sets with different types of polygons ensure the generality of these results even further.

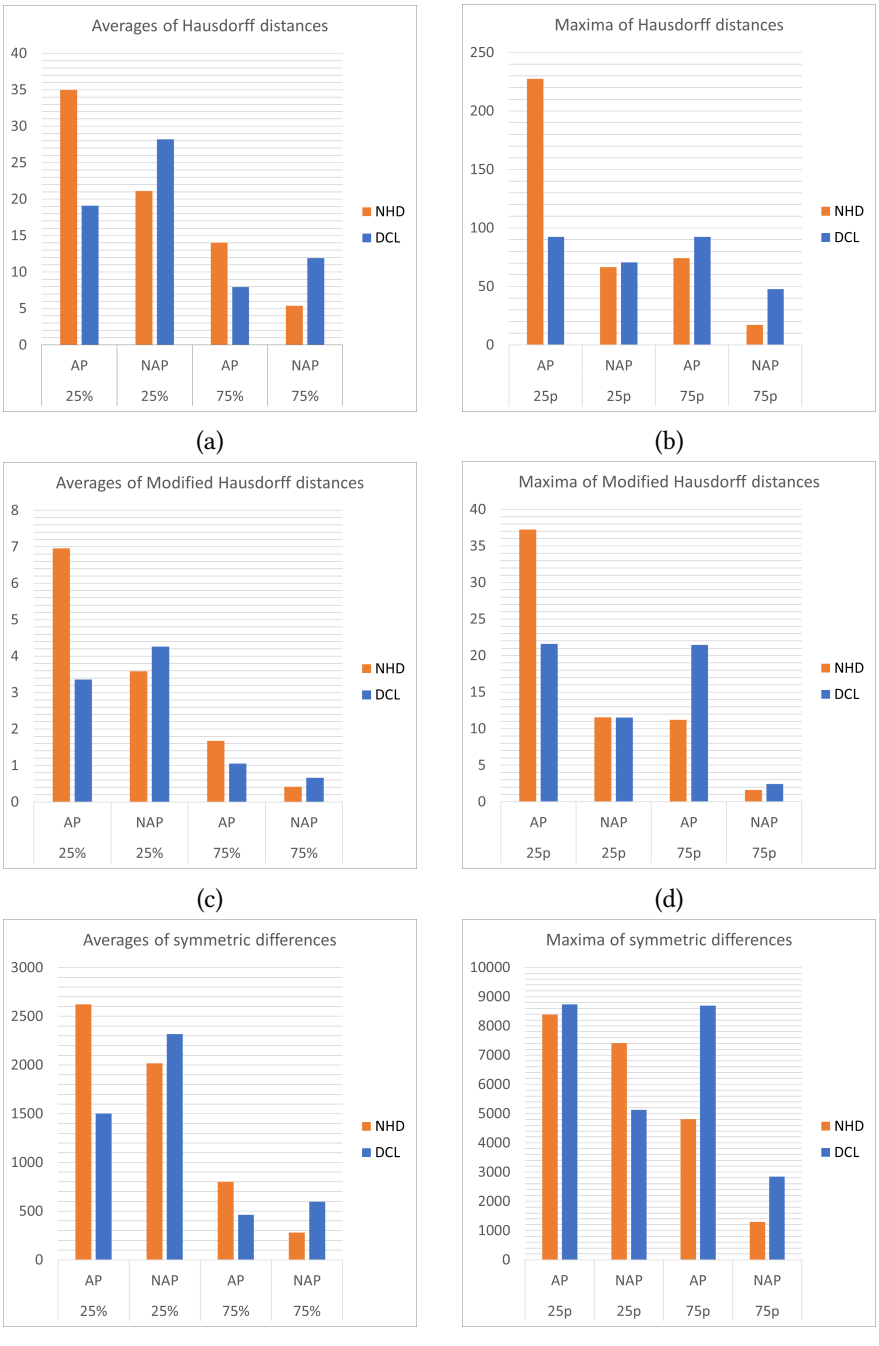



Figure 6.3: Averages and maxima for our iterative version of the Imai and Iri algorithm for the DCL and NHD data sets for the metrics Hausdorff distance (a) and (b), Modified Hausdorff distance (c) and (d), and symmetric difference (e) and (f).

# Chapter 7

# Conclusion

We have implemented three different simplification algorithms: Imai and Iri, Kronenfeld et al. and Buchin et al. For each of these we created an area preserving or non-area preserving counterpart (whichever is applicable) and compared the quality of the simplifications against each other. We obtained varying results. The area preserving counterpart of Imai and Iri produces low quality simplifications as the difference in area seems to be a bad error measure. We see Kronenfeld et al. outperform its non-area preserving counterpart. The non-area preserving counterpart to Buchin et al. has only slightly better results, even though the area preserving variant of Buchin et al. modifies twice as much area per iteration. All the results led us to the hypothesis that area preservation in iterative simplification algorithms might help with limiting compounding errors. We found an example of a simplification where a compounded error only occurred in the non-area preserving variant and not in the area preserving variant. To further investigate this hypothesis, we created an iterative version of the Imai and Iri algorithm to see if an iterative version would have the area preserving variant produce better quality simplification compared to the non-area preserving variant. The non-area preserving simplifications are for the most part still better, yet the differences are significantly smaller, lending credence to our hypothesis. Therefore we hypothesize that the effect area preservation has on iterative simplification algorithms on simple polygons is that it is able to lessen compounded errors from occurring. We do not have enough results to make definitive conclusions regarding non-iterative polygon simplification algorithms.

Subsequent research can delve into further verifying our hypothesis. Another topic of further research is if this conclusion also holds for complex polygons and subdivisions, where the results might be different due to relative size being more important. This research was also limited in regards to creating an area preserving variant to a non-area preserving simplification algorithm. Another point of interest is in regards to how we preserve area. A difference between the Kronenfeld et al. algorithm and the Buchin et al. algorithm is the former preserves area where the simplification occurs, whereas Buchin et al. does not. Research can be conducted to examine the effect of preserving area near the location of simplification and preserving it somewhere else.

# Bibliography


[1] T. Ai, Zhilin Li, and Y.L. Liu. *Progressive transmission of vector data based on changes accumulation model*, pages 85–96. Springer, 2005. International Symposium on Spatial Data Handling ; Conference date: 01-01-2005.

[2] P. Bose, S. Cabello, O. Cheong, J. Gudmundsson, M. van Kreveld, and B. Speckmann. Area-preserving approximations of polygonal paths. *Journal of Discrete Algorithms*, 4(4):554–566, 2006.

[3] K. Buchin, W. Meulemans, A. Van Renssen, and B. Speckmann. Area-preserving simplification and schematization of polygonal subdivisions. *ACM Transactions on Spatial Algorithms and Systems*, 2(1):1–36, 2016.

[4] S. Daneshpajouh, M. Ghodsi, and A. Zarei. Computing polygonal path simplification under area measures. *Graphical Models*, 74(5):283–289, 2012.

[5] D.H. Douglas and T.K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, 1973.

[6] D.H. Douglas and T.K. Peucker. *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature*, chapter 2, pages 15–28. John Wiley and Sons, Ltd, 2011.

[7] M.-P. Dubuisson and A.K. Jain. A modified hausdorff distance for object matching. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 566–568 vol.1, 1994.

[8] Y. Tao et al. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58(5):643–669, 2021.

[9] S. Funke, T. Mendel, A. Miller, S. Storandt, and M. Wiebe. Map simplification with topology constraints : Exactly and in practice. In S. Fekete and V. Ramachandran, editors, *19th Workshop on Algorithm Engineering and Experiments 2017 (ALENEX17) : Barcelona, Spain, 17-18 January 2017*, pages 185–196, Red Hook, NY, 2017. Curran Associates.

[10] J.-H. Haunert and A. Wolff. Optimal simplification of building ground plans. In *XXI ISPRS Congress, Commission II*, volume XXXVII-B2, pages 373–378, 2008.

[11] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[12] H. Imai and M. Iri. Polygonal approximations of a curve — formulations and algorithms. *Machine Intelligence and Pattern Recognition*, 6:71–86, 1988.

[13] B.J. Kronenfeld, L.V. Stanislawski, B.P. Buttenfield, and T. Brockmeyer. Simplification of polylines by segment collapse: minimizing areal displacement while preserving area. *International Journal of Cartography*, 6(1):22–46, 2020.

[14] D. Ma, Z. Zhao, Y. Zheng, R. Guo, and W. Zhu. Polysimp: A tool for polygon simplification based on the underlying scaling hierarchy. *ISPRS International Journal of Geo-Information*, 9(10), 2020.

[15] T. Mendel. Area-preserving subdivision simplification with topology constraints: Exactly and in practice. In Rasmus Pagh and Suresh Venkatasubramanian, editors, *ALENEX*, pages 117–128. SIAM, 2018.

[16] X. Tong, Y. Jin, L. Li, and T. Ai. Area-preservation simplification of polygonal boundaries by the use of the structured total least squares method with constraints. *Transactions in GIS*, 19(5):780–799, 2015.

[17] D. Tutic and M. Lapaine. Area preserving cartographic line generalizaton. *Kartografija i Geoinformacije*, 8:84, 2009.

[18] Maheswari Visvalingam and J. D. Whyatt. Line generalisation by repeated elimination of points. *The cartographic journal*, 30:46–51, 1993.

[19] Evgeniy Vodolazskiy. Discrete frechet distance for closed curves. *CoRR*, abs/2106.02871, 2021.

[20] Zeshen Wang and Jean-Claude Müller. Line generalization based on analysis of shape characteristics. *Cartography and Geographic Information Systems*, 25(1):3–15, 1998.

# Appendix A

# Detailed simplification results

We have the 25% and 75% simplification averages for the DCL data set in Tables A.1 and A.2 respectively. The same results for the NHD data set are in Tables A.3 and A.4. Besides the averages we also have the maxima in Tables A.5, A.6, A.7 and A.8. We also have direct comparisons in Table A.9 for the DCL data set and in Table A.10 for the NHD data set. The averages for our iterative version of Imai and Iri are shown in Tables A.11 and A.12 for the DCL and NHD data set respectively. The maxima are shown in Tables A.13 and A.14 for the DCL and NHD data set respectively.

Table A.1: Averages for 25% simplifications on the DCL data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 13,2190 | 0,030592 | 46,1543 | 944,3945 | 99,98% |
| Imai and Iri (NAP) | 1,6097 | 0,006443 | 32,7838 | 260,6031 | 99,78% |
| Kronenfeld et al. (AP) | 3,6640 | 0,005610 | 26,72858 | 175,1547 | 100,00% |
| Kronenfeld et al. (NAP) | 3,4226 | 0,006435 | 23,17820 | 232,3735 | 99,73% |
| Buchin et al. (AP) | 4,4876 | 0,006317 | 30,0425 | 207,7617 | 100,00% |
| Buchin et al. (NAP) | 3,3842 | 0,005942 | 29,3171 | 177,2310 | 99,77% |

Table A.2: Averages for 75% simplifications on the DCL data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 1,4183 | 0,001605 | 15,9867 | 36,8061 | 100,00% |
| Imai and Iri (NAP) | 0,1115 | 0,000149 | 16,4356 | 4,0773 | 99,99% |
| Kronenfeld et al. (AP) | 0,1577 | 0,000143 | 10,9680 | 2,2857 | 100,00% |
| Kronenfeld et al. (NAP) | 0,1691 | 0,000199 | 9,7246 | 3,4095 | 99,99% |
| Buchin et al. (AP) | 0,2546 | 0,000140 | 11,6686 | 2,5443 | 100,00% |
| Buchin et al. (NAP) | 0,2584 | 0,000127 | 10,9442 | 2,2645 | 100,01% |

Table A.3: Averages for 25% simplifications on the NHD data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 31,0242 | 0,001201 | 69,9415 | 2317,0680 | 99,99% |
| Imai and Iri (NAP) | 1,8071 | 0,010067 | 27,6580 | 522,4945 | 99,53% |
| Kronenfeld et al. (AP) | 3,7909 | 0,006392 | 22,6703 | 296,8254 | 100,00% |
| Kronenfeld et al. (NAP) | 3,5088 | 0,008026 | 20,3553 | 382,5889 | 99,60% |
| Buchin et al. (AP) | 5,9247 | 0,007933 | 26,3984 | 409,6645 | 100,00% |
| Buchin et al. (NAP) | 6,1342 | 0,007598 | 26,3296 | 395,9581 | 100,15% |

Table A.4: Averages for 75% simplifications on the NHD data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 9,4225 | 0,013836 | 30,1217 | 441,8437 | 100,00% |
| Imai and Iri (NAP) | 0,3376 | 0,000709 | 10,4902 | 33,3097 | 99,97% |
| Kronenfeld et al. (AP) | 0,4731 | 0,000558 | 6,9024 | 13,7739 | 100,00% |
| Kronenfeld et al. (NAP) | 0,4801 | 0,000648 | 5,9248 | 20,5347 | 100,00% |
| Buchin et al. (AP) | 0,7451 | 0,000811 | 7,2244 | 14,7509 | 100,00% |
| Buchin et al. (NAP) | 0,7464 | 0,000803 | 7,1742 | 14,5610 | 100,01% |

Table A.5: Maxima for 25% simplifications on the DCL data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 92,3947 | 0,4173 | 117,2900 | 8709,0100 | 99,80% |
| Imai and Iri (NAP) | 5,63807 | 0,0175 | 68,1075 | 1065,9700 | 103,13% |
| Kronenfeld et al. (AP) | 18,6015 | 0,0160 | 69,0938 | 749,5710 | 100,00% |
| Kronenfeld et al. (NAP) | 11,6717 | 0,0173 | 69,0926 | 1198,9900 | 98,09% |
| Buchin et al. (AP) | 22,0984 | 0,01842 | 69,0884 | 1117,6100 | 100,00% |
| Buchin et al. (NAP) | 11,7648 | 0,0188 | 69,0857 | 867,3200 | 96,13% |

Table A.6: Maxima for 75% simplifications on the DCL data set.

| **Algorithm** | **HD** | **MHD** | **DFD** | **SD** | **Area %** |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 21,8650 | 0,032908 | 40,0024 | 859,8470 | 100,02% |
| Imai and Iri (NAP) | 0,9460 | 0,001427 | 68,1075 | 56,7640 | 99,86% |
| Kronenfeld et al. (AP) | 1,7220 | 0,001253 | 37,9536 | 37,1867 | 100,00% |
| Kronenfeld et al. (NAP) | 1,2773 | 0,002091 | 37,8018 | 55,0202 | 99,90% |
| Buchin et al. (AP) | 2,2984 | 0,001151 | 37,9536 | 42,4570 | 100,00% |
| Buchin et al. (NAP) | 2,48618 | 0,001201 | 37,9536 | 38,3073 | 100,31% |

Table A.7: Maxima for 25% simplifications on the NHD data set.

| Algorithm | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 227,4800 | 0,276645 | 227,4800 | 7658,6700 | 99,67% |
| Imai and Iri (NAP) | 5,6580 | 0,019167 | 62,5304 | 2677,1000 | 95,79% |
| Kronenfeld et al. (AP) | 8,6601 | 0,012650 | 66,7696 | 1755,5400 | 100,00% |
| Kronenfeld et al. (NAP) | 13,5388 | 0,019426 | 55,2136 | 2208,0700 | 95,36% |
| Buchin et al. (AP) | 23,9359 | 0,019523 | 66,1131 | 2677,0000 | 100,00% |
| Buchin et al. (NAP) | 14,2526 | 0,015937 | 66,1131 | 2544,4600 | 104,51% |

Table A.8: Maxima for 75% simplifications on the NHD data set.

| Algorithm | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|
| Imai and Iri (AP) | 27,0930 | 0,059206 | 98,8378 | 4042,8900 | 100,02% |
| Imai and Iri (NAP) | 1,0092 | 0,001469 | 48,1397 | 150,2550 | 99,55% |
| Kronenfeld et al. (AP) | 1,2287 | 0,001082 | 16,4368 | 77,4419 | 100,00% |
| Kronenfeld et al. (NAP) | 1,4685 | 0,001048 | 15,6623 | 120,3530 | 100,13% |
| Buchin et al. (AP) | 2,4804 | 0,001767 | 18,7207 | 87,1802 | 100,00% |
| Buchin et al. (NAP) | 2,4804 | 0,001709 | 18,7207 | 91,6955 | 100,08% |

Table A.9: Area preserving vs non-area preserving simplifications in the DCL data set. Percentages means: $x$% of simplifications have a lower score than their non-area preserving counterpart in the metric.

| Algorithm | Size | HD | MHD | DFD | SD |
|---|---|---|---|---|---|
| Imai and Iri | 25% | 42,86% | 45,24% | 42,11% | 40,48% |
| Kronenfeld et al. | 25% | 40,48% | 66,66% | 25,00% | 76,19% |
| Buchin et al. | 25% | 50,00% | 47,83% | 19,04% | 19,05% |
| Imai and Iri | 75% | 48,72% | 30,95% | 50,00% | 64,29% |
| Kronenfeld et al. | 75% | 36,84% | 83,33% | 44,73% | 78,57% |
| Buchin et al. | 75% | 57,89% | 43,90% | 25,00% | 33,33% |

Table A.10: Area preserving vs non-area preserving simplifications in the NHD data set. Percentages means: $x$% of simplifications have a lower score than their non-area preserving counterpart in the metric.

| Algorithm | Size | HD | MHD | DFD | SD |
|---|---|---|---|---|---|
| Imai and Iri | 25% | 47,83% | 39,13% | 21,74% | 65,22% |
| Kronenfeld et al. | 25% | 43,48% | 78,26% | 32,61% | 84,78% |
| Buchin et al. | 25% | 31,58% | 36,96% | 63,16% | 28,26% |
| Imai and Iri | 75% | 54,35% | 60,87% | 53,33% | 47,83% |
| Kronenfeld et al. | 75% | 54,35% | 65,22% | 19,57% | 80,43% |
| Buchin et al. | 75% | 37,50% | 30,43% | 50,00% | 21,74% |

Table A.11: The averages of the simplifications of the Imai and Iri iterative variant for the DCL data set.

| Size | Variant | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|---|
| 25% | AP | 19,0935 | 3,359474 | 49,4481 | 1501,2060 | 100,02% |
| 25% | NAP | 28,2192 | 4,262100 | 43,4644 | 2316,8969 | 89,38% |
| 75% | AP | 7,9421 | 1,052984 | 29,0384 | 461,1046 | 100,00% |
| 75% | NAP | 11,9079 | 0,662990 | 20,9584 | 597,0470 | 97,71% |

Table A.12: The averages of the simplifications of the Imai and Iri iterative variant for the NHD data set.

| Size | Variant | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|---|
| 25% | AP | 34,9939 | 6,95526 | 70,9336 | 2623,6405 | 100,00% |
| 25% | NAP | 21,1074 | 3,581951 | 48,4469 | 2018,2961 | 96,25% |
| 75% | AP | 14,0486 | 1,676134 | 40,4051 | 801,9019 | 100,00% |
| 75% | NAP | 5,3697 | 0,415823 | 21,9814 | 280,2989 | 99,29% |

Table A.13: The maxima of the simplifications of the Imai and Iri iterative variant for the DCL data set.

| Size | Variant | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|---|
| 25% | AP | 92,3946 | 21,585100 | 136,1020 | 8733,6400 | 100,35% |
| 25% | NAP | 70,4771 | 11,511600 | 86,9147 | 5128,9900 | 59,07% |
| 75% | AP | 92,3946 | 21,455300 | 117,2900 | 8695,8800 | 99,98% |
| 75% | NAP | 47,6261 | 2,448220 | 58,8365 | 2841,6300 | 80,17% |

Table A.14: The maxima of the simplifications of the Imai and Iri iterative variant for the NHD data set.

| Size | Variant | HD | MHD | DFD | SD | Area % |
|---|---|---|---|---|---|---|
| 25% | AP | 227,4800 | 37,257500 | 227,4800 | 8388,8300 | 100,19% |
| 25% | NAP | 66,3969 | 11,564800 | 108,0880 | 7414,2200 | 71,54% |
| 75% | AP | 74,1055 | 11,224200 | 98,8378 | 4810,3800 | 100,02% |
| 75% | NAP | 17,0340 | 1,624360 | 55,0396 | 1294,2200 | 96,39% |