Progressive Simplification of Polygonal Curves

Kevin Buchin[1], Maximilian Konzack[2], and Wim Reddingius[3]

1 **TU Eindhoven, The Netherlands**
`k.a.buchin@tue.nl`
2 **TU Eindhoven, The Netherlands**
`m.p.konzack@tue.nl`
3 **TU Eindhoven, The Netherlands**
`wimreddingius@gmail.com`

Abstract

Simplifying polygonal curves at different levels of detail is an important problem with many applications. Existing geometric optimization algorithms are only capable of minimizing the complexity of a simplified curve for a single level of detail. We present an $O(n^3m)$-time algorithm that takes a polygonal curve of n vertices and produces a set of consistent simplifications for m scales while minimizing the cumulative simplification complexity. This algorithm is compatible with distance measures such as Hausdorff, Fréchet and area-based distances, and enables simplification for continuous scaling in $O(n^5)$ time.

1 Introduction

Given a polygonal curve as input, the curve simplification problem asks for a polygonal curve that approximates the input well using as few vertices as possible. Because of the importance of data reduction, curve simplification has a wide range of applications. One such application is cartography, where the visual representation of line features like rivers, roads, and region boundaries needs to be reduced. Most maps nowadays are interactive and incorporate zooming, which requires curve simplification that facilitates different levels of detail. A naive approach would be to simplify for each zoom level independently. This however has the drawback that the resulting simplifications are not consistent between different scales. Therefore, we require *progressive simplification*, that is, a series of simplifications for which the level of detail is progressively increased for higher zoom-levels. This is shown in Figure 1a.

Progressive simplifications are used in cartography [7]. Existing algorithms for progressive simplification (e.g. Cao et al. [2]) work by simplifying the input curve, then simplifying this simplification, and so on. Cao et al. [2] referred to progressive curve simplification as "aging". More concretely, a common approach is to iteratively discard vertices, such that we always discard the vertex whose removal introduces the smallest error (according to some criterion). For example, the algorithm by Visvalingam and Whyatt [9] always removes the vertex which together with its neighboring vertices forms a triangle with the smallest area.

Such approaches stand in stark contrast to (non-progressive) curve simplification algorithms that aim to minimize the complexity of the simplification while guaranteeing a (global) bound on the error introduced by the simplification. The most prominent algorithm with a preset error bound was proposed by Douglas and Peucker [5]. However, while heuristically aiming at a simplification with few vertices, this algorithm does not actually minimize the number of vertices. A general algorithm for the problem of minimizing the number of vertices was introduced by Imai and Iri [6]. Their approach uses *shortcut graphs*, which we describe in more detail below. An efficient algorithm to compute shortcut graphs for the Hausdorff distance was presented by Chan and Chin [3]. Inspired by the work of Visvalingam and Whyatt, Daneshpajouh et al. [4] defined an error measure for non-progressive simplification by

34th European Workshop on Computational Geometry, Berlin, Germany, March 21–23, 2018.

measuring the sum or the difference in area between a simplification and the input curve. In the line of these algorithms, the goal of our work is to develop algorithms that solve progressive simplification as an optimization problem.

A (vertex-restricted) *simplification* $\mathcal{S}$ of a polygonal curve $\mathcal{C}$ is an ordered subsequence of $\mathcal{C}$ (denoted by $\mathcal{S} \sqsubseteq \mathcal{C}$) that includes the first and the last point of $\mathcal{C}$. An *ε-simplification* $\mathcal{S}$ is a simplification that ensures that each edge of $\mathcal{S}$ has a distance of at most ε to its corresponding subcurve, where the distance measure can for instance be the Hausdorff or the Fréchet distance [1]. For an ordered pair of vertices (p_i, p_j) of $\mathcal{C}$ we denote the distance between the segment (p_i, p_j) and the corresponding subchain by $\varepsilon(p_i, p_j)$. We denote by $(p_i, p_j) \in \mathcal{S}$ that (p_i, p_j) is an edge of $\mathcal{S}$.

We next define the *progressive simplification problem* in the plane. Given a polygonal curve $\mathcal{C} = \langle p_1, \dots, p_n \rangle$ in $\mathbb{R}^2$ and a sequence $\mathcal{E} = \langle \varepsilon_1, \dots, \varepsilon_m \rangle$ with $\varepsilon_i \in \mathbb{R}_{>0}$ where $0 < \varepsilon_1 < \dots < \varepsilon_m$, we want to compute a sequence of (vertex-restricted) simplifications $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_m$ of $\mathcal{C}$ such that

1. $\mathcal{S}_m \sqsubseteq \mathcal{S}_{m-1} \sqsubseteq \dots \sqsubseteq \mathcal{S}_1 \sqsubseteq \mathcal{C}$ (*monotonicity*),
2. $\mathcal{S}_k$ is an ε_k-simplification of $\mathcal{C}$,
3. $\sum_{k=1}^{m} |\mathcal{S}_k|$ is minimal.

We refer to a sequence of simplifications fulfilling the first two conditions as *progressive simplification*. A sequence fulfilling all three conditions is called a *minimal progressive simplification*, and the problem of computing such a sequence is called the *progressive simplification problem*. We present an $O(n^3 m)$-time algorithm for the progressive simplification problem in the plane.

The cornerstone of progressive simplification is that we require monotonicity. This guarantees that, when "zooming out", vertices are only removed and cannot (re)appear. As error measure, we will mostly use the Hausdorff distance. This is not essential to the core algorithm, and we will discuss how to use the Fréchet distance [1] or area-based measures [4] without affecting the worst-case running time. Furthermore, our algorithm generalizes to the *weighted* version of the problem in which $\sum_{i=1}^{m} w_i |\mathcal{S}_i|$ with positive weights w_i is minimized, and to the *continuous* version, where $\mathcal{S}_\varepsilon$ needs to be computed for all $0 \leq \varepsilon < \varepsilon_M$. As in the discrete setting, we require $\mathcal{S}_{\varepsilon'} \sqsubseteq \mathcal{S}_\varepsilon$ for $\varepsilon' > \varepsilon$; the resulting algorithm minimizes $\int_0^{\varepsilon_M} |\mathcal{S}_\varepsilon| \, d\varepsilon$ in $O(n^5)$ time. Note that ε_M is the error at which we can simplify the curve by the single line segment (p_1, p_n); thus, we have $\varepsilon_M = \varepsilon(p_1, p_n)$.

In our algorithms we will make use of the *shortcut graph* as introduced by Imai and Iri [6]. For a given curve $\mathcal{C}$, a *shortcut* (p_i, p_j) is an ordered pair $(i < j)$ of vertices. Given an error $\varepsilon > 0$, a shortcut (p_i, p_j) is *valid* if $\varepsilon(p_i, p_j) \leq \varepsilon$. The *shortcut graph* $G(\mathcal{C}, \varepsilon)$ [6] as shown in Figure 1b represents all valid shortcuts (p_i, p_j) with $1 \leq i < j \leq n$. A bottleneck in computing (progressive) simplifications is the construction and space usage of these graphs.

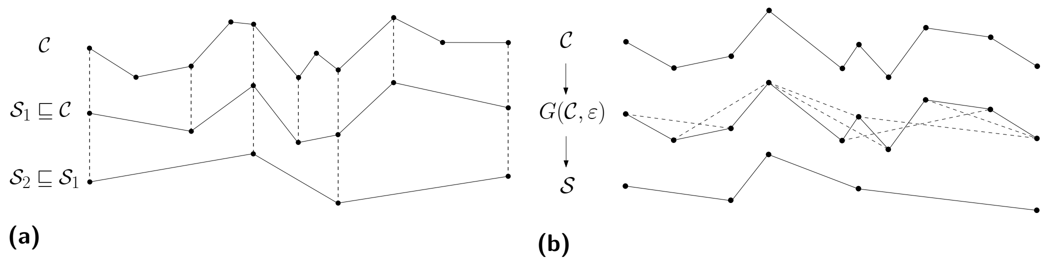
$\mathcal{C}$
$\mathcal{S}_1 \sqsubseteq \mathcal{C}$
$\mathcal{S}_2 \sqsubseteq \mathcal{S}_1$
(a)
$\mathcal{C}$
$G(\mathcal{C}, \varepsilon)$
$\mathcal{S}$
(b)

Figure 1 (a) a progressive simplification and (b) curve simplification using the shortcut graph.

2 Optimal Progressive Simplification

We show how to solve the progressive simplification problem in $O(n^3m)$ time in this section. The same running time holds for the weighted version, and based on this we show that the continuous progressive simplification problem can be solved in $O(n^5)$ time, see Section 3.

By the monotonicity property of the progressive simplification problem (see condition 1 in the definition in Section 1), we require that all vertices within a simplification $\mathcal{S}_k$ of the sequence must also occur within all subsequent simplifications $\mathcal{S}_l$ with $k < l$. Adding shortcuts to a specific simplification thus influences the structure of the other simplifications. We therefore associate a cost value $c^k_{i,j} \in \mathbb{N}$ for each shortcut (p_i, p_j) in the shortcut graph $G(\mathcal{C}, \varepsilon_k)$ that relates to the cost of including (p_i, p_j) in $\mathcal{S}_k$. We use the Hausdorff distance as an error measure to determine whether a shortcut is valid, but since the shortcut graph is flexible to use any error measures, we can employ any other distance measure for our algorithms. In particular for the Fréchet distance [1] and area-based distances [4], we can use brute-force to compute whether a shortcut is valid in $O(n)$ time, and therefore use these measures without changing the worst-case running time. We obtain a cost value $c^k_{i,j}$ for a shortcut $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_k)$ by minimizing the costs of all possible shortcuts in $\langle p_i, \dots, p_j\rangle$ at lower scales recursively. The dynamic program is defined as follows:

$$c^k_{i,j} = \begin{cases} 1 & \text{if } k = 1 \\ 1 + \min\limits_{\pi \in \prod^{k-1}_{i,j}} \sum\limits_{(p_x,p_y)\in\pi} c^{k-1}_{x,y} & \text{if } 1 < k \leq m \end{cases}$$

We use $\prod^k_{i,j}$ to denote the set of all paths in $G(\mathcal{C}, \varepsilon_k)$ from p_i to p_j.

The algorithm starts with constructing the shortcut graphs $G(\mathcal{C}, \varepsilon_1), \dots, G(\mathcal{C}, \varepsilon_m)$. For most distance measures, the distance of shortcut (p_i, p_j) to the subcurve $\langle p_i, \dots, p_j\rangle$ can be determined in $O(j - i)$ time. For such measures, constructing these graphs naively takes $O(n^3m)$ time. By employing the algorithm by Chan and Chin [3] we can compute it in $O(n^2m)$ time for the Hausdorff distance.

We compute all cost values from scale $k = 1$ up to m by assigning a weight $c^k_{i,j}$ to each shortcut $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_k)$. For each shortcut $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_k)$, we compute $c^k_{i,j}$ by finding a shortest path π in $G(\mathcal{C}, \varepsilon_{k-1})$ from p_i to p_j, minimizing $\sum_{(p_x,p_y)\in\pi} c^{k-1}_{x,y}$ thereby.

We can use any shortest path algorithm, such as Dijkstra's algorithm. On each scale k, we need to run Dijkstra's algorithm on $O(n)$ source nodes of $G(\mathcal{C}, \varepsilon_k)$. This yields a worst case running time of $O(n^3m)$, because Dijkstra's algorithm runs in $O(n^2)$ time on weighted shortcut graphs with integer weights.

We increment $c^k_{i,j} = c^{k-1}_{i,j} + 1$ for any shortcut $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_{k-1})$. By doing so, we avoid recomputations of shortest paths and reuse cost values whenever necessary.

We construct the sequence of simplifications from $\mathcal{S}_m$ down to $\mathcal{S}_1$. First, we compute $\mathcal{S}_m$ by returning the shortest path from p_1 to p_n in $G(\mathcal{C}, \varepsilon_m)$ using the computed cost values at scale m. Next, we compute a shortest path P from p_i to p_j in $G(\mathcal{C}, \varepsilon_{m-1})$ for all shortcuts $(p_i, p_j) \in \mathcal{S}_m$. Simplification $\mathcal{S}_{m-1}$ is then constructed by linking these paths P with each other. We build all other simplifications in this manner until $\mathcal{S}_1$ is constructed.

If (p_i, p_j) is a valid shortcut in $G(\mathcal{C}, \varepsilon_{k-1})$ for any $1 < k \leq m$, then it follows that $c^k_{i,j} = c^{k-1}_{i,j} + 1$. We prove this in [8].

Correctness

We prove that our simplification algorithm returns a valid and minimal solution for the progressive simplification problem. Let $\langle \mathcal{S}_1, \dots, \mathcal{S}_m\rangle$ be a sequence of simplifications computed by our algorithm. By constructing the simplifications from scale m down to 1, it follows that for any shortcut $(p_i, p_j) \in \mathcal{S}_k$ with $1 < k \leq m$, there exists a subsequence $\langle p_i, \ldots, p_j \rangle \sqsubseteq \mathcal{S}_{k-1}$, and thus $\mathcal{S}_k \sqsubseteq \mathcal{S}_{k-1}$. Furthermore, each simplification $\mathcal{S}_k$ has a maximum Hausdorff distance ε_k to $\mathcal{C}$ since it contains only edges from $G(\mathcal{C}, \varepsilon_k)$.

It remains to show that we minimize $\sum_{i=1}^{m} |\mathcal{S}_i|$. We therefore define a set of shortcuts $\mathcal{S}_k^{i,j}$ for any $1 \leq i < j \leq n$ and $1 \leq k \leq m$ as $\mathcal{S}_k^{i,j} = \{\ (p_x, p_y) \in \mathcal{S}_k \mid x \leq i < j \leq y\ \}$.

Thus, $\mathcal{S}_k^{i,j}$ includes all line segments of $\mathcal{S}_k$ that span the subcurve $\langle p_i, \ldots, p_j \rangle$ with an error of at most ε_k to $\mathcal{C}$. $|\mathcal{S}_k^{i,j}|$ then is the number of shortcuts in simplification $\mathcal{S}_k$ covering (p_i, p_j).

▶ **Lemma 2.1.** *If the line segment (p_i, p_j) is part of simplification $\mathcal{S}_k$, then the associated cost value $c_{i,j}^k = \sum_{\ell=1}^{k} |\mathcal{S}_\ell^{i,j}|$ for any $1 \leq k \leq m$ and $1 \leq i < j \leq n$.*

Proof. We show $c_{i,j}^k = \sum_{\ell=1}^{k} |\mathcal{S}_\ell^{i,j}|$ by induction on k using the following induction hypothesis: For any $(p_x, p_y) \in \mathcal{S}_k$, it holds that $c_{x,y}^k = \sum_{\ell=1}^{k} |\mathcal{S}_\ell^{x,y}|$ (IH).
Base $k = 1$: Take any shortcut $(p_i, p_j) \in \mathcal{S}_1$. It follows that $\mathcal{S}_1^{i,j} = \{(p_i, p_j)\}$, and therefore $|\mathcal{S}_1^{i,j}| = 1$. We deduce that $c_{i,j}^1 = 1 = \sum_{\ell=1}^{k} 1 = \sum_{\ell=1}^{k} |\mathcal{S}_1^{i,j}|$.
Step $k > 1$: Take any line segment $(p_i, p_j) \in \mathcal{S}_{k+1}$. Thus, we observe $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_{k+1})$, $\mathcal{S}_{k+1}^{i,j} = \{(p_i, p_j)\}$, and $|\mathcal{S}_{k+1}^{i,j}| = 1$.

Consider any $1 \leq \ell \leq k$ and a path $\pi \in \prod^k(p_i, p_j)$ such that $\sum_{(p_x,p_y)\in\pi} |\mathcal{S}_\ell^{x,y}|$ is minimal. We now derive that $\pi = \mathcal{S}_\ell^{i,j}$ such that $\mathcal{S}_\ell^{x,y}$ is minimal for all $(p_x, p_y) \in \pi$. Note that $\pi = \mathcal{S}_\ell^{i,j} \subseteq G(\mathcal{C}, \varepsilon_\ell) \subseteq G(\mathcal{C}, \varepsilon_k)$ since $\varepsilon_k \geq \varepsilon_\ell$. We observe that π is both in $\prod^\ell(p_i, p_j)$ and $\prod^k(p_i, p_j)$. It thus follows that:

$$\min_{\pi\in\prod_{i,j}^{k}} \sum_{(p_x,p_y)\in\pi} |\mathcal{S}_\ell^{x,y}| = \min_{\pi\in\prod_{i,j}^{\ell}} \sum_{(p_x,p_y)\in\pi} |\mathcal{S}_\ell^{x,y}| \tag{1}$$

From $\pi = \mathcal{S}_\ell^{i,j}$ it follows that $\mathcal{S}_\ell^{x,y} \cap \mathcal{S}_\ell^{y,z} = \emptyset$ for any (p_x, p_y) and (p_y, p_z) in π. Combining $\mathcal{S}_\ell^{x,y}$ for all $(p_x, p_y) \in \pi$ yields a non-overlapping sequence of shortcuts from p_i to p_j. This gives us:

$$|\mathcal{S}_\ell^{i,j}| = \min_{\pi\in\prod_{i,j}^{\ell}} \sum_{(p_x,p_y)\in\pi} |\mathcal{S}_\ell^{x,y}| \tag{2}$$

We now derive the following:

$$c_{i,j}^{k+1} \overset{\text{(IH)}}{=} 1 + \min_{\pi\in\prod_{i,j}^{k}} \sum_{(p_x,p_y)\in\pi} \sum_{\ell=1}^{k} |\mathcal{S}_\ell^{x,y}| \overset{(1)}{=} 1 + \sum_{\ell=1}^{k} \min_{\pi\in\prod_{i,j}^{\ell}} \sum_{(p_x,p_y)\in\pi} |\mathcal{S}_\ell^{x,y}| \overset{(2)}{=} 1 + \sum_{\ell=1}^{k} |\mathcal{S}_\ell^{i,j}|$$
$$\overset{|\mathcal{S}_{k+1}^{i,j}| = \{(p_i,p_j)\}}{=} \sum_{\ell=1}^{k+1} |\mathcal{S}_\ell^{i,j}|$$

◀

▶ **Theorem 2.2.** *Given a polygonal curve with n points in the plane, and $0 \leq \varepsilon_1 < \ldots < \varepsilon_m$, a minimal progressive simplification can be computed in $O(n^3 m)$ time under distance measures for which the validity of a shortcut can be computed in $O(n)$ time. This includes the Fréchet, Hausdorff and area-based measures.*

Proof. It remains to be proven that the combined size of the simplifications computed by our algorithm is minimal. Let $\langle \mathcal{S}'_1, \ldots, \mathcal{S}'_m \rangle$ be a sequence of simplifications of a minimal progressive simplification, and let $\langle \mathcal{S}_1, \ldots, \mathcal{S}_m \rangle$ be the sequence computed by our algorithm.

Let us derive the following:

$$\min_{\pi \in \prod_{1,n}^{m}} \sum_{(p_x,p_y)\in\pi} c_{x,y}^{m} \overset{(2.1)}{=} \min_{\pi \in \prod_{1,n}^{m}} \sum_{(p_x,p_y)\in\pi} \sum_{k=1}^{m} |\mathcal{S}_k^{x,y}| \overset{(1)}{=} \sum_{k=1}^{m} \min_{\pi \in \prod_{1,n}^{\ell}} \sum_{(p_x,p_y)\in\pi} |\mathcal{S}_k^{x,y}| \overset{(2)}{=} \sum_{k=1}^{m} |\mathcal{S}_k|$$

Hence, the algorithm produces a simplification that minimizes the cumulative cost of shortcuts in $\mathcal{S}_m$. Because $\mathcal{S}_{i+1} \sqsubseteq \mathcal{S}_i$; the algorithm produces a set of simplifications in which each simplification consists of edges from the corresponding shortcut graph such that the cumulative number of vertices is minimized.

We further know that any minimal simplification $\mathcal{S}'_k$ is a path in $G(\mathcal{C}, \varepsilon_k)$ since it strictly connects shortcuts with an error of at most ε_k.

We conclude that $\sum_{k=1}^{m} |\mathcal{S}_k| \leq \sum_{k=1}^{m} |\mathcal{S}'_k|$ holds. ◀

3 Continuous and Weighted Progressive Simplification

We now consider two versions of the progressive simplification problem: the *weighted* progressive simplification, where the objective is to minimize $\sum_{k=1}^{m} w_k|\mathcal{S}_k|$ (with $w_k \geq 0$), thus the weighted cumulative size of the simplifications; and the *continuous* progressive simplification, which is an instance of the weighted progressive simplification where $\int_0^m |\mathcal{S}_i|\, d\varepsilon$ is minimal. For both problems, we can employ our preceding algorithm to compute simplifications progressively. We first show how to adapt our algorithm for the weighted progressive simplification problem; then we prove how to solve the continuous simplification problem.

For the weighted progressive simplification, we use the following cost function for each shortcut $(p_i, p_j) \in G(\mathcal{C}, \varepsilon_k)$: if $k = 1$, $c_{i,j}^{k} = w_1$ else $c_{i,j}^{k} = w_k + \min_{\pi \in \prod_{i,j}^{k-1}} \sum_{(p_x,p_y)\in\pi} c_{x,y}^{k-1}$. Note that the proofs above are trivially extended to apply to this updated cost function. The main reason to consider the weighted case is that it helps us solving the continuous progressive simplification problem.

▶ **Theorem 3.1.** *Given a polygonal curve with n points in the plane, a minimal continuous progressive simplification can be computed in $O(n^5)$ time under distance measures for which the validity of a shortcut can be computed in $O(n)$ time. This includes the Fréchet, Hausdorff and area-based measures.*

Proof. Consider the maximal errors $\varepsilon(p_i, p_j)$ of all possible line segments (p_i, p_j) with $i < j$ with respect to the Hausdorff distance (or another distance measure). Then let $\mathcal{E} := \langle \varepsilon_1, \ldots, \varepsilon_{\binom{n}{2}} \rangle$ be the sorted sequence of these errors based on their value. Let M be the index of the corresponding ε_M in this sorted sequence $\mathcal{E}$ for the line segment (p_1, p_n); thus $\varepsilon_M = \varepsilon(p_1, p_n)$. Note that it is possible that $M < \binom{n}{2}$, but there is no reason to use any $\varepsilon > \varepsilon_M$, since at this point we already have simplified the curve to a single line segment, (p_1, p_n).

In a minimal-size progressive simplification it holds that $\mathcal{S}_\varepsilon = \mathcal{S}_{\varepsilon_i}$ for all $\varepsilon \in [\varepsilon_i, \varepsilon_{i+1})$. This can be shown by contradiction: if $\mathcal{S}_\varepsilon$ would be smaller, we could decrease the overall size by setting all $\mathcal{S}_{\varepsilon'}$ with $\varepsilon' \in [\varepsilon_i, \varepsilon]$ to $\mathcal{S}_\varepsilon$. Therefore, in a minimal continuous progressive simplification we have $\int_0^{\varepsilon_M} |\mathcal{S}_\varepsilon|\, d\varepsilon = \sum_{k=1}^{M-1} (\varepsilon_{k+1} - \varepsilon_k)|\mathcal{S}_{\varepsilon_k}|$. Thus, we can solve the continuous progressive simplification problem by reducing it to the weighted progressive simplification problem with $O(n^2)$ values ε_k. ◀

4 Discussion

We present the first algorithm to compute minimum-complexity progressive simplifications given a polygonal curve with n points in the plane. Our algorithm runs in $O(n^3m)$ time for m discrete scales and $O(n^5)$ time for continuous scaling.

In the following, we survey further results from [8]. To facilitate progressive simplifications on many scales, in [8] we present a technique for computing all $\varepsilon(p_i, p_j)$ efficiently in $O(n^2 \log n)$ time instead of $O(n^3)$ time [3]. This is in particular useful for continuous progressive simplification, where we would otherwise need to compute a quadratic number of shortcut graphs, thus spending $O(n^4)$ time on computing shortcut graphs.

Furthermore, we developed a storage-efficient representation of the shortcut graph that is capable of finding shortest paths in $O(n \log n)$ time, which is also applicable to any simplification algorithm that uses a shortcut graph.

The experimental evaluation on a trajectory of a migrating griffon vulture shows that our progressive algorithm is effective, yet too slow for larger trajectory data, and provides similar cumulative simplification sizes as an optimal non-progressive simplification algorithm. We discuss all experiments, algorithms, and results in [8].

As future work, it would be of interest to improve the running time of the minimal progressive simplification algorithm to facilitate real-world application.

Acknowledgments. We thank Michael Horton for our discussions on this topic. Kevin Buchin and Maximilian Konzack are supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 612.001.207.

References

1 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *IJCGA*, 5(1–2):78–99, 1995.

2 Hu Cao, Ouri Wolfson, and Goce Trajcevski. Spatio-temporal data reduction with deterministic error bounds. *VLDB J*, 15(3):211–228, 2006. doi:10.1007/s00778-005-0163-7.

3 Wing Shiu Chan and F Chin. Approximation of polygonal curves with minimum number of line segments or minimum error. *IJCGA*, 6(01):59–77, 1996.

4 Shervin Daneshpajouh, Mohammad Ghodsi, and Alireza Zarei. Computing polygonal path simplification under area measures. *Graphical Models*, 74(5):283–289, 2012.

5 David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973.

6 Hiroshi Imai and Masao Iri. Polygonal approximations of a curve – formulations and algorithms. In G. T. Toussaint, editor, *Computational Morphology*, pages 71–86. Elsevier, 1988.

7 Guo Qingsheng, Christoph Brandenberger, and Lorenz Hurni. A progressive line simplification algorithm. *Geo-spatial Information Science*, 5(3):41–45, 2002.

8 Wim Reddingius. Progressive minimum-link simplification of polygonal curves. Master's thesis, Eindhoven University of Technology, 2017. URL: https://tue.on.worldcat.org/oclc/1016161122.

9 Maheswari Visvalingam and James D Whyatt. Line generalisation by repeated elimination of points. *Cartogr J*, 30(1):46–51, 1993.