J Geogr Syst (2009) 11:273–289
DOI 10.1007/s10109-009-0078-8

ORIGINAL ARTICLE

Simultaneous curve simplification

Christopher Dyken · Morten Dæhlen ·
Thomas Sevaldrud

Received: 21 August 2007 / Accepted: 20 January 2009 / Published online: 11 February 2009

Abstract In this paper we present a method for simultaneous simplification of a collection of piecewise linear curves in the plane. The method is based on triangulations, and the main purpose is to remove line segments from the piecewise linear curves without changing the topological relations between the curves. The method can also be used to construct a multi-level representation of a collection of piecewise linear curves. We illustrate the method by simplifying cartographic contours and a set of piecewise linear curves representing a road network.

Keywords Simplification · Digital line graph · Multi-resolution · Triangulation

JEL C63

1 Introduction

A simplification of a piecewise linear curve is a piecewise linear approximation using fewer segments than the original, where the distance between the curve and its approximation is within a prescribed tolerance. In addition, when simplifying sets of

C. Dyken (✉)
SINTEF ICT Applied Mathematics, P.O. Box 124, Blindern, 0314 Oslo, Norway
e-mail: erikd@ifi.uio.no

M. Dæhlen
Department of Informatics, University of Oslo, P.O. Box 1080, Blindern, 0316 Oslo, Norway
e-mail: mortend@ifi.uio.no

T. Sevaldrud
World Beside AS, Gaustadalléen 25, 0373 Oslo, Norway
e-mail: thomas@worldbeside.com

curves, we want to maintain the relationships between the curves, thus keeping the topology of the curve set constant. In this article we will describe a method for performing a simultaneous simplification of a curve set, that is, all the curves are simplified in parallel while enforcing the given topology of the curve set. A piecewise linear curve is defined by a sequence of points where the convex hull pairs of two consecutive points define the line segments. We create the approximation by finding a subsequence of the original sequence of points by discarding points one-by-one until no more points can be removed without violating the tolerance.

An important issue is the strategy for finding which point to remove at each step of the process. Usually, such processes are guided by evaluating the error induced by removing each candidate point, and choosing the point inducing the smallest error. This is a greedy optimization approach, which not necessarily finds a global optimum, but gives good results in practice. In most cases the traditional Euclidean norm or variations over the Hausdorff metric is used to measure distance between curves, see Arge and Dæhlen (1997). In this paper we will use a variation over these measures. The distance measure is particularly important when performing simultaneous curve simplification since we must measure the distance to neighbouring curves. Our solution for handling neighbouring relations, both topological and geometrical, is based on representing the curve set as a triangulation, i.e. subdividing the plane into triangles using the points and segments of the curve set.

Simplification of piecewise linear curve sets, and in particular curve sets representing level curves or cartographic contours, was one of the early challenges of computer-based cartography. Other typical examples of curve sets which appear in cartography and geographical information systems are river networks, various types of boundary curves describing property borders, forest stands, etc. Contour data and boundary curves are also important within other applications areas, e.g. when modeling geological structures.

The most commonly used method for curve simplification is the well known Douglas–Peucker algorithm (Douglas and Peucker 1973; Ramer 1972), which was developed in the early seventies. Other methods have also been developed as well, see Arge and Dæhlen (1997) and references therein. Our method for simultaneous simplification of a set of curves is based on measuring the distance between points and curves using variations over the Euclidean norm, in addition to relationships between points and curves, which are efficiently found by using triangulations. We also illustrate how this method can be used to construct a nested multi-level representation of curve networks.

The outline of this paper is as follows: in the next section we give some background information on refinement and decimation of curves and triangulations, followed by a section defining the problem statement and further comments the motivation behind this work. In Sect. 4 we describe the triangulation into which the curve set is embedded. Then, in Sect. 5 we describe the simultaneous curve simplification algorithm. Finally, we conclude the paper with a few examples in Sect. 6 and some remarks on future work in Sect. 7.

2 Refinement and decimation

Various algorithms have been constructed for simplification of piecewise linear curves. The most well-known is the Douglas–Peucker algorithm (Douglas and Peucker 1973; Ramer 1972), of which we will give an overview.

We begin by defining a piecewise linear planar curve. Given a sequence of points in the plane

$$\mathbf{p}_1, \ldots, \mathbf{p}_n, \quad \mathbf{p}_i \in \mathbb{R}^2,$$

the curve is defined by the line segments

$$[\mathbf{p}_i, \mathbf{p}_{i+1}], \quad i = 1, 2, \ldots, n-1,$$

where $[\cdot]$ is the convex hull.

Then, given a tolerance we proceed as follows. The initial approximation is the single segment $[\mathbf{p}_1, \mathbf{p}_n]$. We then pick the point $\mathbf{p}_k$ in $\mathbf{p}_2, \ldots, \mathbf{p}_{n-1}$ that is farthest away from $[\mathbf{p}_1, \mathbf{p}_n]$, using e.g. the Euclidean distance measure.

If the distance between $\mathbf{p}_k$ and $[\mathbf{p}_1, \mathbf{p}_n]$ is greater than the tolerance, we insert $\mathbf{p}_k$ into the approximation, and get two new line segments $[\mathbf{p}_1, \mathbf{p}_k]$ and $[\mathbf{p}_k, \mathbf{p}_n]$. We continue this process recursively on every new line segment until the tolerance is met for every line segment.

The two first steps of the process is shown in Fig. 1 where first $\mathbf{p}_k$ is inserted into $[\mathbf{p}_1, \mathbf{p}_n]$ yielding the two line segments $[\mathbf{p}_1, \mathbf{p}_k]$ and $[\mathbf{p}_k, \mathbf{p}_n]$. Then, $\mathbf{p}_l$ is inserted into $[\mathbf{p}_k, \mathbf{p}_n]$, and we end up with an approximation defined by the three line segments $[\mathbf{p}_1, \mathbf{p}_k]$, $[\mathbf{p}_k, \mathbf{p}_l]$, and $[\mathbf{p}_l, \mathbf{p}_n]$. Variations over the Douglas-Peucker algorithm and other methods can be found in Arge and Dæhlen (1997) , and references therein.

The Douglas–Peucker method is a *refinement* procedure, where we start with a coarse approximation (one single line segment in our example) and insert the most significant points, getting increasingly finer approximations, until the prescribed tolerance is met. Another approach is the converse process, called a *decimation* procedure. We start with the original curve as the initial approximation and remove the least significant points progressively, giving increasingly coarser approximations

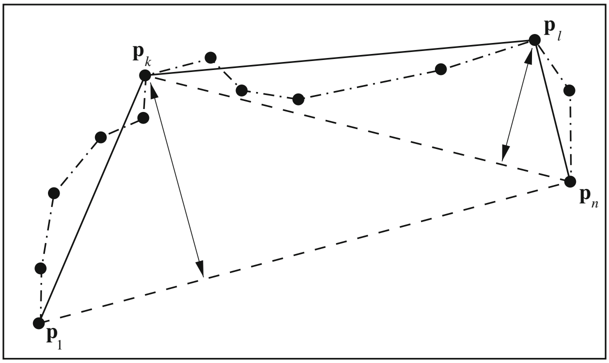

Fig. 1 Two steps of the Douglas–Peucker simplification algorithm

until no more points can be removed without violating the prescribed tolerance. The simultaneous simplification algorithm presented in this paper is based on decimation of triangle meshes. Similarly to curve simplification, we can start with two triangles covering the domain as our initial approximation and insert the most significant points followed by a re-triangulation one by one until the tolerance is met. Or we can start with the full triangle mesh and remove points and re-triangulate while the tolerance is not violated. A variety of methods has been developed for this purpose, and details on these algorithms can be found in Heckbert and Garland (1997) and references therein.

3 Problem statement

We are given a set of piecewise linear curves in the plane, and we want to simplify these curves without modifying the topological relationship between the curves. Such a curve set is the lake in the top row of Fig. 2 where each curve defines a contour of a complex polygon. When simplifying it is important that islands in the lake remain inside the lake, and furthermore that the lake polygon remain a valid polygon. Another example is the road network in the bottom row of Fig. 2. The road network is a curve set where each curve describes the path between two junctions. Thus, we are given a set Q of M piecewise linear curves,

$$Q = \{P^j, \quad j = 1, \ldots, M\}, \tag{1}$$

where each curve P^j in Q is a sequence of N^j points,

$$P^j = \{\mathbf{p}_1^j, \ldots \mathbf{p}_{N^j}^j\}, \quad \mathbf{p}_i^j \in D, \quad D \subset \mathbb{R}^2 \tag{2}$$

contained in some suitable domain D in the plane.

We want to create a simplified approximation of Q, which we denote $\hat{Q}$,

$$\hat{Q} = \{\hat{P}^j, \quad j = 1, \ldots, M\},$$

where $\hat{P}^j$ approximates P^j using a subset of the points of P^j.

Given a suitable distance measure $|\cdot|$ and a tolerance ϵ, we create $\hat{Q}$ from Q by removing points from the curves one-by-one while $|\hat{Q} - Q| < \epsilon$. When removing points one-by-one, we inherently create a sequence of increasingly coarser approximations, and thus, we can easily obtain a multiple-level-of-detail representation of Q by instead of using a single fixed tolerance ϵ, we use a sequence of monotonically increasing tolerances $0 < \epsilon_1 < \epsilon_2 < \cdots < \epsilon_L$ where $|\hat{Q}_i - Q| < \epsilon_i$. Initially, we let $\hat{Q}_0 = Q$, and using ϵ_1 we create $\hat{Q}_1$ from $\hat{Q}_0$. Further, we create $\hat{Q}_2$ from $\hat{Q}_1$ using ϵ_2, and so on. Since we only remove points and never move a point, the points of $\hat{Q}_i$ is a subset of the points of $\hat{Q}_{i-1}$, and thus, we have a *nested* multiple-level-of-detail representation.

If this is done carefully we can introduce the curve sets into a triangle based multi-level terrain model. Alternatively, a sequence of approximations can also be obtained by removing a fixed number of points successively, and hence avoid using an error tolerance at all. In Fig. 3, a fixed number of points were removed, while in Figs. 4, 5, and 6, a sequence of tolerances were used.

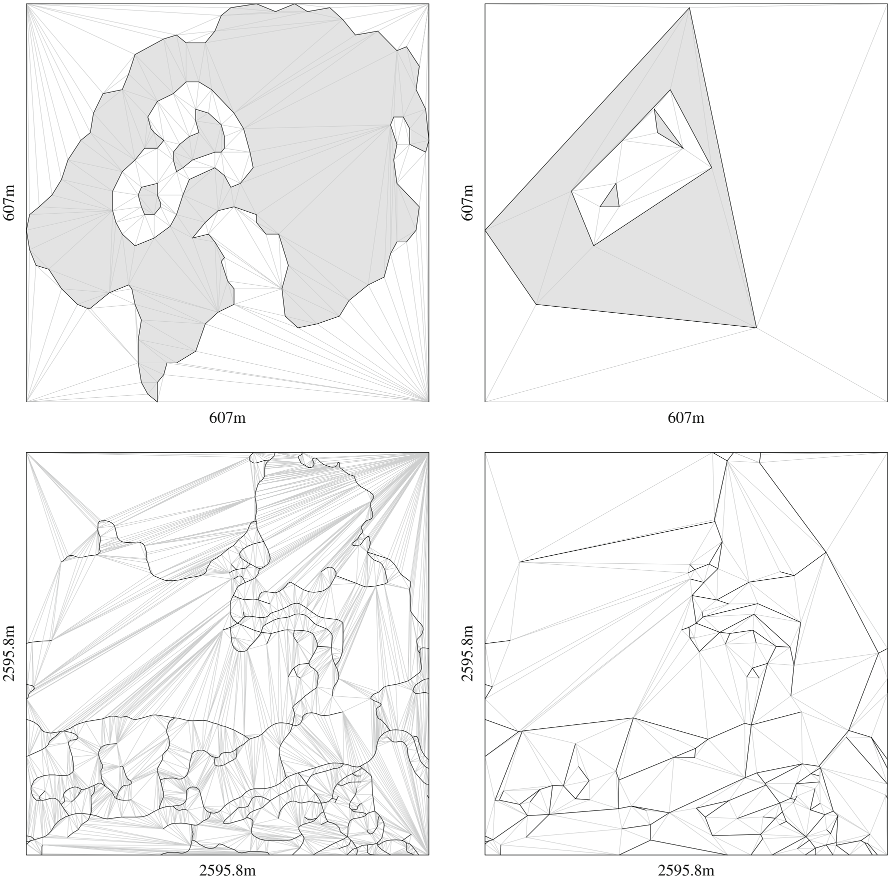

Fig. 2 Polygonal networks with constrained Delaunay triangulations. The *top row* depicts a polygonal lake, and the *bottom row* depicts a small part of a road network. The *left column*show the original datasets, while the *right column* show their maximally simplified counterparts

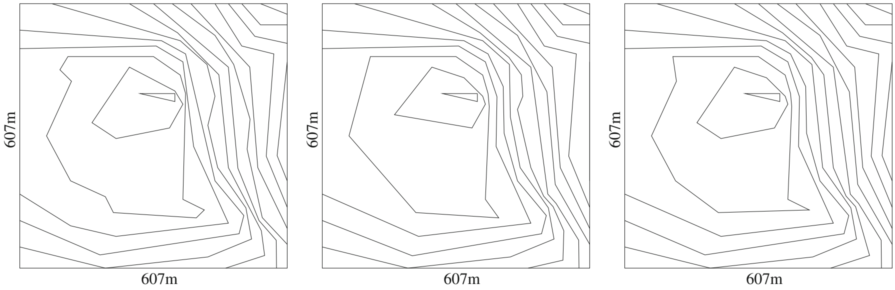

Fig. 3 Comparison of the three distance measures. The curve set given in Fig. 7 left was simplified to 90 points using the three different norms. *Left* Maximum distance measure, ϵ_M = 16.7 m. *Center* Scaled maximum distance measure, ϵ_s = 44.5 m. *Right* Hybrid distance measure (R = 20 m), ϵ_H= 22.3 m

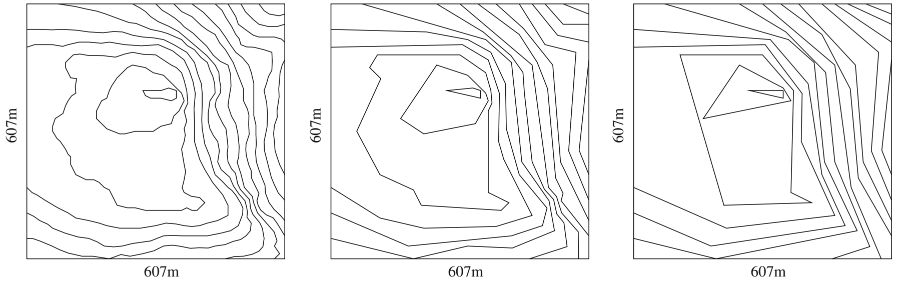

Fig. 4 Multi-resolution simplification of cartographic contours, created using the hybrid norm ($R = 20$ m). *Left* Original set of cartographic contours, 470 points. *Center* Simplification $\epsilon_1 = 15.4$ m, 108 points. *Right* Further simplification $\epsilon_2 = 100$ m, 66 points. Here, no more points can be removed without violating the topological constraints

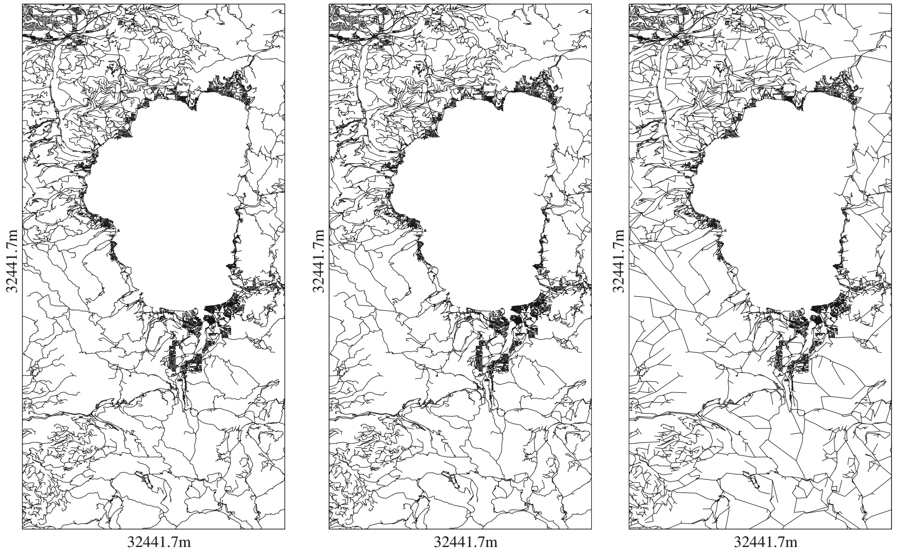

Fig. 5 Multi-resolution simplification of the road network of the Lake Tahoe dataset US Geological Survey (1997), created using the hybrid distance measure ($R = 500$ m). *Left* Original curve set, 123,038 points. *Middle* Simplification $\epsilon_1 = 100$ m, 33,014 points. *Right* Further simplification $\epsilon_2 = 1{,}000$ m, 12,839 points

However, care must be taken when we remove points. Figure 7 shows the result of an unfortunate point removal. By approximating the two consecutive line segments [**p**, **q**] and [**q**, **r**] with the single line segment [**p**, **r**], we introduce two new intersections, and thus, change the topological relationship between the curves. From this we see that in order to simplify one curve segment in a network, we must consider all curves simultaneously.

To satisfy that the topological relationship between curves remain constant when we remove a point, we never remove any point whose removal would violate any of the following four requirements:

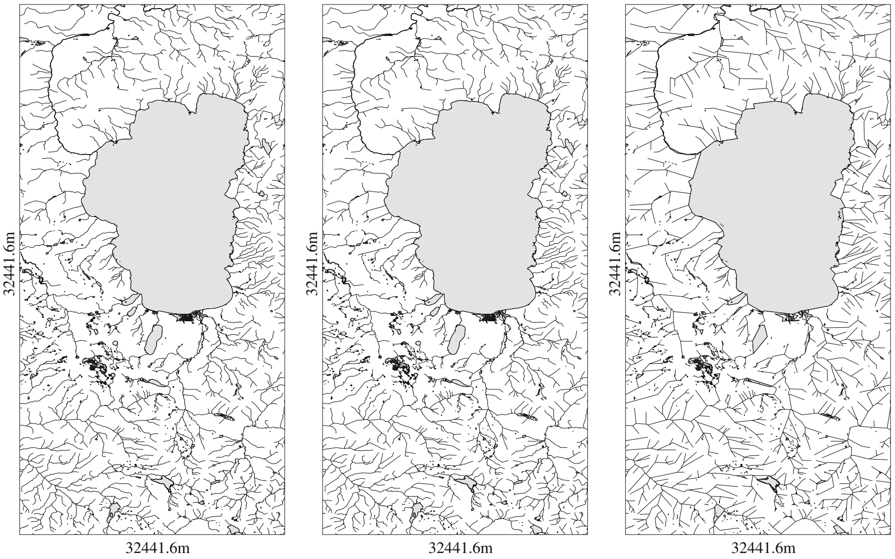

Fig. 6 Multi-resolution simplification of the lake and river network combined of the Lake Tahoe dataset US Geological Survey (1997), created using the hybrid distance measure (R = 500 m). *Left* Original network, 144,293 points. *Middle* Simplification $\epsilon_1 = 100$ m, 31,794 points. *Right* Further simplification $\epsilon_2 = 1{,}000$ m, 8,022 points

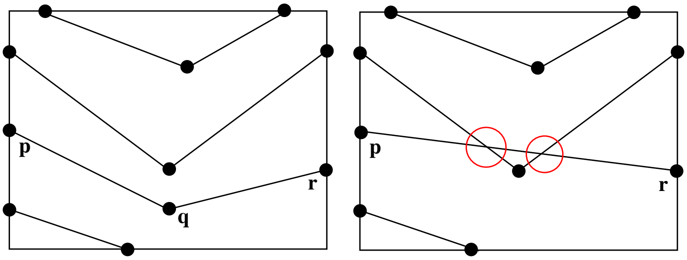

Fig. 7 Approximating the two consecutive line segments. [**p**, **q**] and [**q**, **r**] with [**p**, **r**] introduces two new intersections, and thus is an *illegal simplification* which changes the topology of the curve set

1. All intersections between curves shall be kept fixed.
2. No new intersections between curves shall be generated.
3. No curves shall degenerate into one single point.
4. No closed curves shall degenerate into a single line segment.

This asserts that the topology of the curve set as a whole does not change.

For example, in the case of the road network, Requirements 1 through 3 ensures that all existing junctions are maintained, no new junctions are introduced, or that two junctions do not collapse into one single junction. Further, in the case of areal objects like a polygonal lake, Requirement 4 ensures that they remain areal objects.

In order to continuously enforce the four requirements while we remove points, we need a suitable method of encoding the geometric relationship between the curves. This encoding can be handled by using the concept of triangulations from graph theory, on which we will elaborate in the next section.

4 Triangulation

To encode the geometrical relationship between the curves of Q, we represent Q as a triangulation (Hjelle and Dæhlen 2006) and use triangle decimation to remove points and line segments. Thus, the initial triangulation must contain all points of Q and every line segment of every curve P^j of Q must be an edge in the triangulation. The edges not corresponding to line segments encodes the spatial relationship between the curves. Figure 2 shows examples of such triangulations. Notice that a curve can have itself as a neighbour.

We assume that all curve intersections are also points on the curves. If this is not the case, the intersection points must be found and inserted into their respective curves. With this assumption, it is always possible to construct such a triangulation of Q. This follows from the fact that the collection of curves can be extended to a set of closed non-overlapping polygons by including the boundary of the convex hull of the curve set. Moreover, any closed polygon in the plane can be partitioned into a set of triangles (Nielson 1997), and hence it follows that we can construct a triangulation where every line segment of the curve set is an edge in the triangulation.

For practical purposes we embed the entire curve set in a rectangular region by introducing the boundary of this rectangle as a curve in the the curve set (see Fig. 2). In most examples this boundary curve will be a rectangle represented as a polygon containing the four corner points of the rectangle plus points where the curve network intersects the boundary, e.g. where a road or a river leaves the rectangular region.

Several methods exists for creating a triangulation with prescribed edges. A well-known approach is to create a Delaunay triangulation followed by a successive phase of edge swapping that includes the prescribed edges into the triangulation (Nielson 1997). Any non-boundary edge is always a diagonal of a quadrilateral composed of two adjacent triangles. If this quadrilateral is convex we can swap the edge to form two new triangles. Alternatively, we can directly construct a constrained Delaunay triangulation, using all the line segments of the curves as initial edges and fill inn edges between the curves (Lee and Lin 1986). Figure 2 shows the original and decimated versions of two curve sets.

5 Curve set decimation

We have a curve set Q of piecewise linear curves embedded in a triangulation T such that each line segment in Q corresponds to an edge in T. The triangulation T encodes the geometrical relationship between the curves of Q. In this section we describe how we use triangle decimation to simplify the curves while enforcing the topology of the curve set.

We begin by giving a general description of the decimation algorithm, and continue with discussing the various components of the algorithm in detail. The structure of the decimation algorithm is as follows:

1. Determine the set of points that are candidates for removal, and assign to each candidate a weight based on a suitable error measure.
2. While the list of candidates for removal is non-empty and the smallest weight have an error less than the prescribed tolerance, do:
   (a) Remove the candidate with the smallest weight from its curve and the triangulation.
   (b) Update the list of candidates and calculate the new weights for points in the neighbourhood of the removed point.

Notice that there are two error measures at play. One error measure is used to determine the sequence in which points are removed, and another error measure is used to describe the global error of the approximation, and thus is used to terminate the decimation process. Alternatively, we can use a prescribed number of points to be removed as termination criteria.

5.1 Nodes and vertices

We assume that the initial curve set Q satisfy the four requirements given in Sect. 3. The triangulation T is a planar graph consisting of points and edges, and Q is a subgraph of T.

We classify the points in T into two sets, nodes and vertices. The set of vertices is the points that are not nodes. A node is a point in T that is also an endpoint of a curve or a junction in Q. The four corner points of the rectangular domain are also included in the set of nodes.

Further, some points are classified as nodes to handle closed loops. We must make sure that every closed loop consists of at least three nodes that are not co-linear. This requirement asserts that Requirement 4 will not be violated in the decimation process.

If a single curve itself forms a closed loop (and the two endpoints of the curve are the same point), we tag two of the interior points of the curve as nodes such that the three nodes of the curve are not co-linear and form a triangle. Further, if two curves together form a closed curve, e.g. when two curves describes the perimeter of a lake, one interior point of one curve that is not co-linear with the endpoints must be classified as a node. For three or more curves forming a loop, the problem does not exists.

5.2 Removable vertices

Nodes are never considered for removal. Further, at each step in the decimation process, only some of the vertices can be removed without violating some of the requirements.

Let **q** be a vertex. The *platelet* of **q** is the union of triangles of T that has **q** as a corner, see Fig. 8. The geometry of the platelet defines whether or not a vertex can be removed. Vertices are always in the interior of curves, and thus, they belong to one single curve and always have two immediate neighbouring points on the same curve.

Let **q** be the vertex on the curve P^j considered for removal and **p** and **r** be the two immediate neighbours on P^j. The points **p** and **r** is situated on the border of the platelet of **q** (refer to Fig. 8). Removing **q** implies that we simplify the two consecutive line segments [**p**, **q**] and [**q**, **r**] of P^j with the single line segment [**p**, **r**]. The line segments of neighbouring curves are located on the boundary of the platelet of **q**. Thus, if [**p**, **r**] is *strictly inside* the platelet of **q**, the line segment [**p**, **r**] never touches any other curve, and therefore can be safely removed without violating any of the four criteria. Contrary, if [**p**, **r**] intersects the boundary, the approximation intersects one or more of the neighbouring curves, and the removal of **q** would change the topology of the curve set.

We check if the line segment [**p**, **r**] is strictly inside the platelet of **q** in the following way. Let $\mathbf{s}_1,\ldots,\mathbf{s}_L$ be the points on the boundary of the platelet of **q** from **p** to **r**, counter-clockwise organized, and similarly, let $\mathbf{t}_1,\ldots,\mathbf{t}_R$ be the points on the boundary of the platelet of **q** from **r** to **p**, also counter-clockwise organized. That is, the sequence

$$\mathbf{p}, \mathbf{s}_1, \ldots, \mathbf{s}_L, \mathbf{r}, \mathbf{t}_1, \ldots, \mathbf{t}_R,$$

describes the complete border of the platelet of **q**. Then, the line segment [**p**, **r**] is strictly inside the platelet of **q** if either

$$\det(\mathbf{r}-\mathbf{p}, \mathbf{s}_i-\mathbf{p}) < 0, \quad i = 1, \ldots, L \quad \text{and}$$
$$\det(\mathbf{r}-\mathbf{p}, \mathbf{t}_j-\mathbf{p}) > 0, \quad j = 1, \ldots, R,$$

or

$$\det(\mathbf{r}-\mathbf{p}, \mathbf{s}_i-\mathbf{p}) > 0,\ i = 1, \ldots, L \text{ and}$$
$$\det(\mathbf{r}-\mathbf{p}, \mathbf{t}_i-\mathbf{p}) < 0,\ j = 1, , R,$$

where $\det(\cdot,\cdot)$ is the determinant of a 2 × 2 matrix.

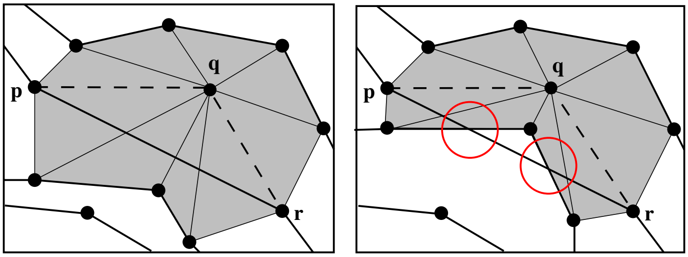

Fig. 8 The geometry of the platelet of **q** allows or prohibits removal of **q**. In the *left figure*, [**p**,**r**] is strictly inside the platelet of **q**, and thus, **q** can safely be removed. However, in the *right figure*, [**p**,**r**] intersects the boundary of the platelet of **q** twice, and thus removal of **q** would introduce two new intersections, and therefore **q** cannot be removed without changing the topology of the curve set

5.3 Weight calculation

Some points are more significant that others, and we handle this by associating a weight to each vertex. The decimation process removes the vertex with the smallest weight and updates the weight of the remaining vertices.

It is natural that weights are based on some appropriate error measure. If approximation error is the single interesting property, using the approximation error induced by the removal of a vertex as the weight of that vertex is appropriate. However, in some applications, other properties like separation of adjacent curves are important. In this case more elaborate weight expressions are appropriate.

In the following we present three different error measures based upon Euclidean distances in the plane.

Maximum distance This distance measure is the maximum distance between a curve and the approximation of the curve we get if the vertex is removed. Let **q** be a vertex on the curve P^j. The points defining P^j are either remaining or have been removed in preceding decimation steps.

Let **p** be the last remaining point on P^j before **q**, and let **r** be the first remaining point after **q**. Moreover, let $\mathbf{s}_1,\ldots,\mathbf{s}_n$ be the sequence of removed points of P^j from **p** to **q**.

The removal of **q** provides that the two line segments [**p**, **q**] and **q**, **r**] is replaced by the single line segment [**p**, **r**]. The maximum distance between the original curve P^j and the approximation induced by the removal of **q** is

$$v_1 = \max\{\text{dist}\{l, s_i\}, \quad i = 1, \ldots, n\}, \tag{3}$$

where l is the line segment [**p**, **r**]. Figure 9 illustrates the calculation of the maximum distance.

Scaled maximum distance The maximum distance measure only describes local approximation error. However, in some cases the separation of curves, that is, the distance between adjacent curves, is important, for example when approximating cartographic contours. To accommodate this, we introduce an additional error measure, a scaled version of the maximum distance measure.

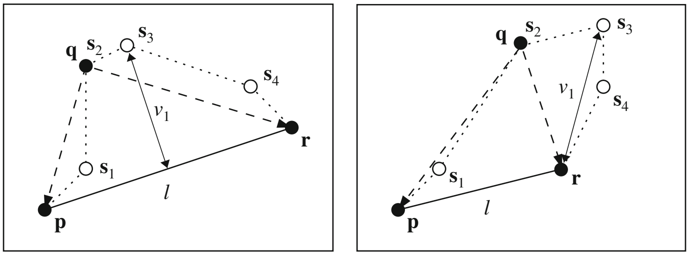

Fig. 9 The maximum distance v_1 is the maximum distance between the approximating line segment l = [**p**, **r**] and the vertices $\mathbf{s}_1,\ldots,\mathbf{s}_n$ between **p** and **r** that is removed from the curve segment. In the *left figure*, the maximum distance is perpendicular to l, while in the *right figure* it is not

Similarly to the maximum distance calculation, let **q** be the point on P^j we are calculating the weight for, and let **p** and **q** be the remaining points before and after **q** on P^j. The points **p** and **q** and segments of curves close to P^j lie on the border of the platelet of **q**.

Let $\mathbf{t}_1,\ldots,\mathbf{t}_m$ be the vertices on the border of the platelet of **q**, excluding the two points **p** and **r**. Then, assuming that $l = [\mathbf{p}, \mathbf{r}]$,

$$v_2 = \min\{\text{dist}\{l, \mathbf{t}_j\}, \quad j = 1, \ldots, m\}, \tag{4}$$

is the minimum distance between the approximating line segment [**p**, **r**] and the points on the border of the platelet of **q**, as illustrated in the left of Fig. 10.

The scaled maximum distance is then defined as

$$w_s = \frac{v_1}{v_2},$$

where v_1 is the maximum distance (3).

The scaled maximum distance measure scales the maximum distance measure proportionally to the minimum distance to neighbouring curves, and thus makes it less likely that **q** on P^j is removed if other curves are situated close to P^j. In this way, the scaled maximum distance measure tends to preserve separation of adjacent curves.

Hybrid distance measure The scaled measure tries to maintain the spatial separation between neighbouring curves. A drawback is that this scaling also has influence on the separation between curves in areas where curves are relatively far apart, in areas that the maximum distance measure would be more appropriate. For example, when simplifying cartographic contours, the scaled maximum distance measure is appropriate in areas dense with curves in order to maintain a good separation of adjacent curves, while in areas sparse with curves, the maximum distance measure is appropriate. The hybrid distance measure solves this.

The hybrid distance measure introduces a parameter R, which can be regarded as a radius of separation. If the minimum distance v_2 is smaller than R, we use the scaled maximum distance measure, otherwise, if v_2 is larger than R, we use the maximum distance measure (see Fig. 10, right).

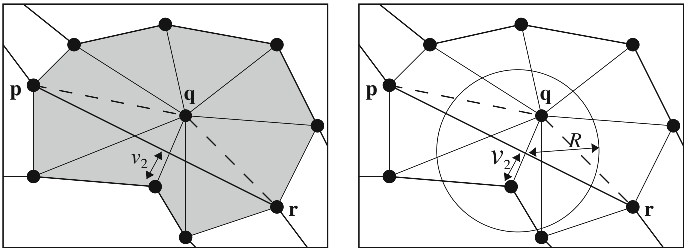

Fig. 10 The scaled maximum distance measure is the maximum distance scaled by v_2, which is the minimum distance between the nodes on the border of the platelet of **q** and the approximating line segment [**p**, **q**]

First, we let v_3 be v_2 clamped to the interval [0, R],

$$v_3 = \begin{cases} v_2 & \text{if } v_2 < R \\ R & \text{otherwise.} \end{cases}$$

Then, hybrid distance measure w_h is defined as

$$w_h = \frac{v_1}{v_3}.$$

In effect, R chooses whether the maximum distance measure or the scaled distance measure is used.

The choice of R is highly dependent on context. For plotting purposes, R should be dependent on pen size, or for visualization on-screen, R should be dependent on pixel size. If the resulting curve set should be represented on a grid, the grid size should dictate the size of R. At other times, it is appropriate that R should be proportional to the error threshold ϵ. However, this can be a bit dicy when doing multi-resolution simplifications, since R can be too small at the finer resolutions resulting in bad spatial separation at coarser resolutions.

Figure 3 shows the dataset of Fig. 4a where 90 points were removed using the three different distance measures. We see that the approximation from the maximum distance measure has the best overall shape, but the contours along the gorge in the right part of the contours end up almost on top of each other. The approximation of the scaled maximum distance measure has distinctly better separation between the contours, however, at the cost of an overall coarser shape. The result of the hybrid measure maintains good separation between adjacent contours while also preserving the overall shape of contours considerably better than the scaled maximum norm.

5.4 Implementational details

We will here give a short description of the actual implementation used to produce the results in Sect. 6.

From a given set of piecewise linear curves we establish a network of curve segments. Points are classified as nodes or vertices, and a curve segment (also referred to as a chain) is a sequence of vertices forming a piecewise linear curve with one node at each end. Thus, the problem is reduced to simplify the interior of chains without violating any of the requirements introduced in Sect. 3.

We start by building a constrained Delaunay triangulation where the chains are the constraints. The triangulation is represented using an half-edge data structure, see Hjelle and Dæhlen (2006). This data structure allows efficient navigation through the triangulation. In theory, the removal of a vertex is bounded by $\mathcal{O}(n \log n)$, see (Chew 1987). However, a vertex removal strategy with a direct re-triangulating locally around the removed vertex is generally very fast. In practice, just a few edge-swaps are required.

The simplification process is built around maintaining a heap, where both removal of the minimum element and the update of elements are of $\mathcal{O}(\log n)$ complexity. The heap contains all removable vertices organized by the error induced by the removal of that particular vertex. While the vertex with the minimum induced error can be removed without violating the prescribed error threshold, the vertex is removed and the induced error of the neighboring vertices are updated.

Since a vertex always belongs to a single chain, the induced maximum distance can be found by finding the maximum distance between the original chain and the simplified chain with that vertex removed. Also, the minimum distance to neighboring chains is found by traversing the platelet of that vertex. Further, when a vertex is removed, only the induced error of the remaining vertices in a small neighborhood are affected.

Using the half-edge structure, both traversing platelets and finding these neighborhoods are straightforward. Thus, encoding the curve set in a triangulation does not only enforce a consistent topology, it also serves as a tool for neighborhood queries, allowing efficient evaluation of the error measures.

6 Numerical results

In this section we present three examples: a set of cartographic contours, a road network, and a lake and river network. The two first examples consist of polygonal curves, while the last example is a mix of polygonal curves as well as complex polygons with holes.

Cartographic contours The cartographic contour curve set is a curve set of 15 cartographic contours, where the contours are a mix of open and closed piecewise linear curves. The curve set is originally defined by 470 points (Fig. 4, left) over an area of 607 × 607 m.

First, we simplified the original curve set three times, removing 380 points (and thus leaving 90 points to define the curve set) using the three different distance measures to guide the simplification. The results are shown in Fig. 3. We used the maximum distance measure to define the approximation error. It is no surprise that the simplification guided by the maximum distance measure got the smallest approximation error ($\epsilon_m = 16.7$ m). However, as mentioned in Sect. 3, the maximum distance measure has problems preserving separation of adjacent curves, which is clearly visible in Fig. 3, left. In Fig. 3, center, the result of using the scaled maximum distance measure as a guide is shown. The separation of adjacent curves is maintained, but the approximation error is considerably worse ($\epsilon_s = 44.5$ m), which is particularly visible on the gentle side of the mountain top where the contours are unnecessarily coarse (the approximation error on this part is scaled such that it becomes very small since the distance between adjacent contours is large). However, simplification guided by the hybrid distance measure, which is depicted in the right of Fig. 3, maintains a good balance between approximation error ($\epsilon_H = 22.3$ m) and separation of adjacent contours. The shapes are almost as good as the result of simplification guided by the maximum distance measure, while also the contours along the gorge are well separated.

Note in particular that even though adjacent contours may end up close to each other, which is the case in particular when the maximum distance measure is used to guide the simplification, any contour *never* end up on top of another contour, which would cause new intersections to form. This is because the four requirements of Sect. 3 is honored, and points that whose removal would result in a contour set of a different topology is never removed, and thus, the simplification maintains a constant topology of the contour set.

Further, Fig. 4 shows a multi-resolution simplification of the set of cartographic contours, where the hybrid distance measure ($R = 20$ m) was used to guide the simplification. The sequence of simplifications is nested in that sense that the points of a coarse simplification is a subset of the points of any of the finer simplifications. In addition, each line segment of a coarse simplification has a one-to-one correspondence to a sequence of one or more line segments in a finer simplification. To the left, the original contour set is shown. In the center, the first simplification with tolerance $\epsilon_1 = 15.4$ m, which gives a contour set of 108 points (23%), and to the right, a further simplification with tolerance $\epsilon_2 = 100$ m, using 66 points (14%). This simplification cannot be further simplified since removal of any of the remaining points would change the topological type.

Road network The road network of Fig. 5 is the road network of the Lake Tahoe dataset (US Geological Survey 1997), covering a domain of approximately 32×32 km^2. Note that the network is plotted using the geographic coordinate system and thus appears to be rectangular even though the domain is square. The original network, which is depicted in Fig. 5, is defined by 123,038 points.

The road network was consecutively simplified two times using the hybrid distance measure ($R = 500$ m). The first simplification ran with a tolerance $\epsilon_1 = 100$ m, which produced an approximation of 33,014 points (27%), depicted in the center of Fig. 5. The original road network and the first approximation are visually indistinguishable at this scale, even though the approximation only uses one fourth of the geometry. The second simplification is a further simplification of the first simplification, using a tolerance $\epsilon_2 = 1{,}000$ m, and results in a road network of 12,839 points (10%), about one-tenth of the geometry of the original network, and with the same topology as the original road network.

Lake and river network The final example is the lake and river networks of the Lake Tahoe dataset (US Geological Survey 1997) combined into one network, a network consisting of both piecewise linear curves and complex polygons with holes.

A clean polygon is a set of closed non-intersecting curves, called rings or contours, consistently oriented such that the inside of the polygon always is on a consistent side when one “walks” along the curves. Simplification of polygons using our technique is simple. We let each contour be a closed curve in the contour set. Since the topology of the curve set is constant during the simplification, the polygon defined by the simplified curves is always a clean consistent polygon.

The original network is shown in the left of Fig. 6, and the two consecutive simplifications are shown in the middle and right of the same figure. The simplifications were guided by the hybrid distance measure ($R = 500$ m), and the tolerances was the same as for the road network. The original network consists of 144,293 points, the first approximation with a tolerance of $\epsilon_1 = 100$ m consists of

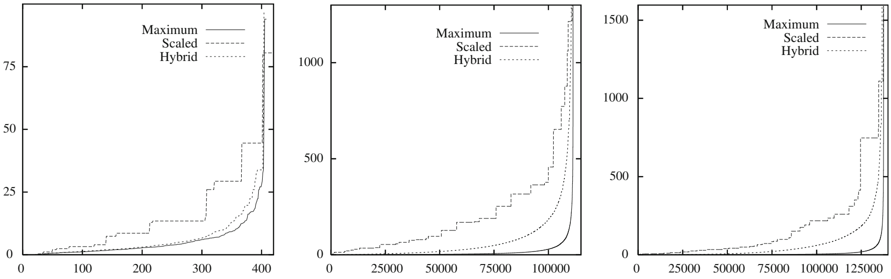

Fig. 11 Approximation errors resulting from using the different distance measures to guide the simplification process. From *left* to *right*, the cartographic contours, the road network, and the lake and river network is shown respectively

31,794 points (22%) and the second approximation with a tolerance of ϵ_1 = 1,000 m consists of 8,022 points (6%).

Similarly to the road network, the original lake and river network is visually indistinguishable at this scale from the first approximation, which contains significantly less geometry.

Approximation error In Fig. 11, the approximation error (the maximum distance measure) is plotted as a function of the number of the points removed, for each of the three examples. From left to right, the plots for the cartographic contours, the road network, and the lake and river networks are shown respectively. Each plot shows the approximation error of the three different norms as the simplification progresses, and thus gives an indication of the performance of the three distance measure.

As expected, simplifications guided directly by the maximum distance give the best performance in terms of approximation error, but do not preserve curve separation. The scaled maximum distance measure preserves curves separation better, but has a significantly worse performance when comparing approximation error, which is quite evident in all three plots. However, the hybrid distance measure embraces the best from both worlds, both preserving good separation between curves, and gives a reasonable performance in terms of approximation error.

7 Concluding remarks

In this paper we presented a method for simultaneous simplification of curve sets in the plane, that is, simplification of curves while maintaining the geometric relationship between curves.

In addition, we introduced three different distance measures that can be used to guide the simplification process, choosing which point to remove at each step. The distance measures have different characteristics. The maximum distance measure minimizes approximation error, while the scaled maximum distance measure preserves separation of adjacent curves. These two measures were successfully combined in the hybrid distance measure, which both preserves separation of adjacent curves while giving a reasonable performance in regard to approximation error. However, the hybrid distance measure requires that a radius of separation be specified, which is highly dependent on context.

In addition to creating single simplifications with a given tolerance or a given number of points, the method can be used to create multi-level representations of collections of curves.

References

Arge E, Dæhlen M (1997) Data reduction of piecewise linear curves. In: Tveito A, Dæhlen M (eds) Numerical methods and software tools in industrial mathematics, Birkhauser, Boston, pp 347–364

Chew L (1987) Constrained delaunay triangulations. In: SCG’87: Proceedings of the third annual symposium on Computational geometry, ACM, New York, pp 215–222

Douglas D, Peucker T (1973) Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Can Cartogr 10(2):112–122

Heckbert P, Garland M (1997) Survey on polygonal surface simplification algorithms. Technical report, Carnegie Mellon University, Multiresolution Surface Modeling Course, SIGGRAPH’97

Hjelle Ø, Dæhlen M (2006) Triangulations and applications. Springer, Berlin

Lee D, Lin A (1986) Generalized delaunay triangulation for planar graphs. Discret Comput Geom 1:201–217

Nielson G (1997) Tools for triangulations and tetrahedrizations and constructing functions defined over them. CS Press, Washington, pp 429–525

Ramer U (1972) An iterative procedure for the polygonal approximation of plane curves. Comput Graphics Image Process 1:244–256

US Geological Survey (1997) Lake Tahoe data clearinghouse website. http://tahoe.usgs.gov