Mapping Polygons to the Grid with Small Hausdorff and Fréchet Distance*

Quirijn W. Bouts[1], Irina Kostitsyna[2], Marc van Kreveld[3], Wouter Meulemans[4], Willem Sonke[5], and Kevin Verbeek[6]

1 **Dept. of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands**
q.w.bouts@tue.nl
2 **Dept. of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands**
i.kostitsyna@tue.nl
3 **Dept. of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands**
m.j.vankreveld@uu.nl
4 **giCentre, City University London, London, United Kingdom**
wouter.meulemans@city.ac.uk
5 **Dept. of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands**
w.m.sonke@tue.nl
6 **Dept. of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands**
k.a.b.verbeek@tue.nl

Abstract

We show how to represent a simple polygon P by a (pixel-based) grid polygon Q that is simple and whose Hausdorff or Fréchet distance to P is small. For any simple polygon P, a grid polygon exists with constant Hausdorff distance between their boundaries and their interiors. Moreover, we show that with a realistic input assumption we can also realize constant Fréchet distance between the boundaries. We present algorithms accompanying these constructions, heuristics to improve their output while keeping the distance bounds, and experiments to assess the output.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases grid mapping, Hausdorff distance, Fréchet distance, digital geometry

Digital Object Identifier 10.4230/LIPIcs.ESA.2016.22

1 Introduction

Transforming the representation of objects from the real plane onto a grid has been studied for decades due to its applications in computer graphics, computer vision, and finite-precision computational geometry [14]. Two interpretations of the grid are possible: (i) the grid graph, consisting of vertices at all points with integer coordinates, and horizontal and vertical edges

* Research on the topic of this paper was initiated at the 1st Workshop on Applied Geometric Algorithms (AGA 2015) in Langbroek, The Netherlands, supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.023.208. NWO is supporting Q. W. Bouts, I. Kostitsyna, and W. Sonke under project no. 639.023.208, and K. Verbeek under project no. 639.021.541. W. Meulemans is supported by Marie Skłodowska-Curie Action MSCA-H2020-IF-2014 656741.

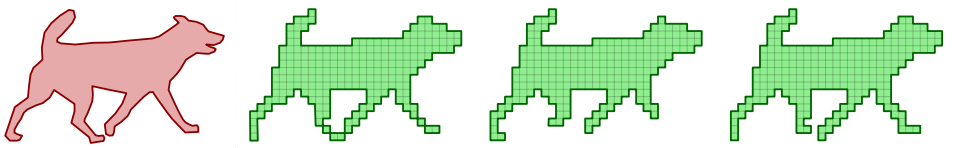


Figure 1 From left to right: input; symmetric-difference optimal result is not a grid polygon; grid polygon computed by our Fréchet algorithm; grid polygon computed by our Hausdorff algorithm.

between vertices at unit distance; (ii) the pixel grid, where the only elements are pixels (unit squares). In the latter, one can choose between 4-neighbor or 8-neighbor grid topology. In this paper we adopt the pixel grid view with 4-neighbor topology.

The issues involved when moving from the real plane to a grid begin with the definition of a line segment on a grid, known as a *digital straight segment* [18]. For example, it is already difficult to represent line segments such that the intersection between any pair is a connected set (or empty). In general, the challenge is to represent objects on a grid in such a way that certain properties of those objects in the real plane transfer to related properties on the grid; connectedness of the intersection of two line segments is an example of this.

While most of the research related to *digital geometry* has the graphics or vision perspective [17, 18], computational geometry has made a number of contributions as well. Besides finite-precision computational geometry [12, 14] these include snap rounding [11, 13, 16], the integer hull [4, 15], and consistent digital rays with small Hausdorff distance [10].

Mapping polygons. We consider the problem of representing a simple polygon P as a similar polygon in the grid (see Fig. 1). A *grid cycle* is a simple cycle of edges and vertices of the grid graph. A *grid polygon* is a set of pixels whose boundary is a grid cycle. This problem is motivated by schematization of country or building outlines and by nonograms.

The most well-known form of schematization in cartography is called a metro map, in which metro lines are shown in an abstract manner by polygonal lines whose edges typically have only four orientations. It is common to also depict region outlines with these orientations on such maps. It is possible to go one step further in schematization by using only integer coordinates for the vertices, which often aligns vertices vertically or horizontally, and leads to a more abstracted view. Certain types of cartograms like mosaic maps [8] are examples of maps following this visualization style. The version based on a square grid is often used to show electoral votes after elections. Another cartographic application of grid polygons lies in the schematization of building outlines [20].

Nonograms – also known as Japanese or picture logic puzzles – are popular in puzzle books, newspapers, and in digital form. The objective is to reconstruct a pixel drawing from a code that is associated with every row and column. The algorithmic problem of solving these puzzles is well-studied and known to be NP-complete [6]. To generate a nonogram from a vector drawing, a grid polygon needs to be made on a coarse grid. We are interested in the generation of grid polygons from shapes like animal outlines, which could be used to construct nonograms. To our knowledge, two papers address this problem. Ortíz-García et al. [22] study the problem of generating a nonogram from an image; both the black-and-white and color versions are studied. Their approach uses image processing techniques and heuristics. Batenburg et al. [5] also start with an image, but concentrate on generating nonograms from an image with varying difficulty levels, according to some definition of difficulty.

Considering the above, our work also relates to image downscaling (e.g. [19]), though this usually starts from a raster image instead of continuous geometric objects. Kopf et al. [19]

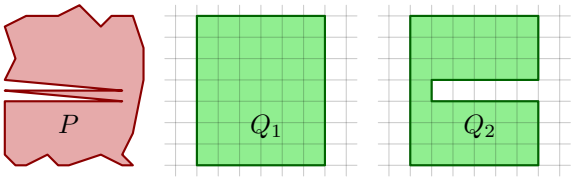


Figure 2 $d_H(P, Q_1)$ is small but $d_H(\partial P, \partial Q_1)$ is not. $d_H(P, Q_2)$ and $d_H(\partial P, \partial Q_2)$ are both small but the Fréchet distance $d_F(\partial P, \partial Q_2)$ is not.

apply their technique to vector images, stating that the outline remains connected where possible. In contrast to our work, the quality is not measured as the geometric similarity and the conditions necessary to guarantee a connected outline remain unexplored.

Similarity. There are at least three common ways of defining the similarity of two simple polygons: the symmetric difference[1], the Hausdorff distance [1], and the Fréchet distance [2]. The first does not consider similarity of the polygon boundaries, whereas the third usually applies to boundaries only. The Hausdorff distance between polygon interiors and between polygon boundaries both exist and are different measures; this distance can be directed or undirected. Let X and Y be two closed subsets of a metric space. The (directed) *Hausdorff distance* $d_H(X, Y)$ from X to Y is defined as the maximum distance from any point in X to its closest point in Y. The undirected version is the maximum of the two directed versions. To define the *Fréchet distance*, let X and Y be two curves in the plane. The Fréchet distance $d_F(X, Y)$ is the minimum leash length needed to let a man walk over X and a dog over Y, where neither may walk backwards (a formal definition can be found in [2]).

Contributions. In Section 2 we show that any simple polygon P admits a grid polygon Q with $d_H(P, Q) \leq \frac{1}{2}\sqrt{2}$ and $d_H(Q, P) \leq \frac{3}{2}\sqrt{2}$ on the unit grid. Furthermore, the constructed polygon satisfies the same bounds between the boundaries ∂P and ∂Q. This is not equivalent, since the point that realizes the maximum smallest distance to the other polygon may lie in the interior (Fig. 2). Our proof is constructive, but the construction often does not give intuitive results (Fig. 2, P and Q_2). Therefore, we extend our construction with heuristics that reduce the symmetric difference whilst keeping the Hausdorff distance within $\frac{3}{2}\sqrt{2}$. The Fréchet distance d_F [2] between two polygon boundaries is often considered to be a better measure for similarity. Unlike the Hausdorff distance, however, not every polygon boundary ∂P can be represented by a grid cycle with constant Fréchet distance. In Section 3 we present a condition on the input polygon boundary related to fatness (in fact, to κ-straightness [3]) and show that it allows a grid cycle representation with constant Fréchet distance. Finally, in Section 4 we evaluate how our algorithms perform on realistic input polygons.

2 Hausdorff distance

We consider the problem of constructing a grid polygon Q with small Hausdorff distance to P. Though minimizing the Hausdorff distance is NP-hard (Theorem 1, see [7]), we present an algorithm that achieves low, constant Hausdorff distance between both the boundaries and the interiors of the input polygon P and the resulting grid polygon Q. We first show

[1] The symmetric difference between two sets A and B is defined as the set $(A \setminus B) \cup (B \setminus A)$. When using symmetric difference as a quality measure, we actually mean the area of the symmetric difference.

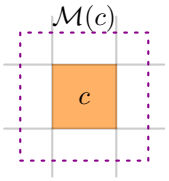


Figure 3 Module $\mathcal{M}(c)$ (dashed) of a cell c.

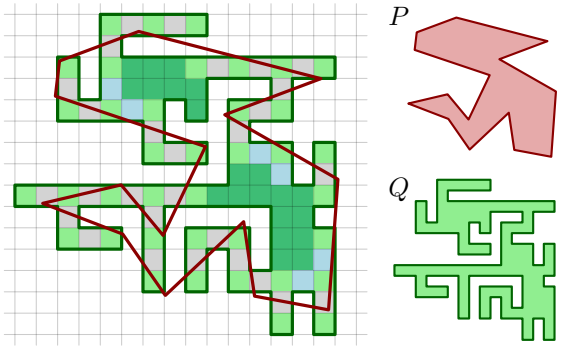


Figure 4 Example of the Hausdorff algorithm; the input and output are shown on the right. Colors: Q_1, Q_2, Q_3, Q_4.

how to construct such a grid polygon. Then, we provide an efficient algorithm to compute Q. Finally, we describe heuristics that can be used to improve the results in practice.

▶ **Theorem 1.** *Given a polygon P, it is NP-hard to decide whether there exists a grid polygon Q such that both $d_H(\partial P, \partial Q) \leq \frac{1}{2}$ and $d_H(\partial Q, \partial P) \leq \frac{1}{2}$.*

2.1 Construction

We represent the grid polygon Q as a set of cells (or pixels). We say that two cells are *adjacent* if they share a segment. If two cells share only a point, then they are *point-adjacent*. If two cells $c_1 \in Q$ and $c_2 \in Q$ are point-adjacent, and there is no cell $c \in Q$ that is adjacent to both c_1 and c_2, then c_1 and c_2 share a *point-contact*. We construct Q as the union of four sets Q_1, Q_2, Q_3, Q_4 (not necessarily disjoint). To define these sets, we define the *module* $\mathcal{M}(c)$ of a cell c as the 2×2-region centered at the center of c (see Fig. 3). Furthermore, we assume the rows and columns are numbered, so we can speak of even-even cells, odd-odd cells, odd-even cells, and even-odd cells. The four sets are defined as follows; see also Fig. 4.

Q_1: All cells c for which $\mathcal{M}(c) \subseteq P$.
Q_2: All even-even cells c for which $\mathcal{M}(c) \cap P \neq \emptyset$.
Q_3: For all cells $c_1, c_2 \in Q_1 \cup Q_2$ that share a point-contact, the two cells that are adjacent to both c_1 and c_2 are in Q_3.
Q_4: A minimal set of cells that makes Q connected, and where each cell $c \in Q_4$ is adjacent to two cells in Q_2 and $\mathcal{M}(c) \cap P \neq \emptyset$.

Set $Q_1 \cup Q_2$ is sufficient to achieve the desired Hausdorff distance. We add Q_3 to resolve point-contacts, and Q_4 to make the set Q simply connected (a polygon without holes). The lemmas below show that Q is indeed a grid polygon.

▶ **Lemma 2.** *The set $Q_1 \cup Q_2$ is hole-free, even when including point-adjacencies.*

Proof. For the sake of contradiction, let H be a maximal set of cells comprising a hole. Let set B contain all cells in $Q_1 \cup Q_2$ that surround H and are adjacent to a cell in H. Since Q_2 contains only even-even cells, every cell in $Q_2 \cap B$ is (point-)adjacent to two cells in $Q_1 \cap B$ (see Fig. 5). Hence, the outer boundary of the union of all modules of cells in $Q_1 \cap B$ is a single closed curve C. Since $C \subset P$ by the definition of Q_1, the interior of C must also be in P. Since all modules of cells in H lie completely inside C, they are also in P, so the cells in H must all be in Q_1. This contradicts that H is a hole. ◀

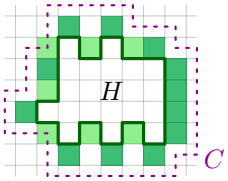


Figure 5 A hole in Q. Colors: $Q_1 \cap B$; $Q_2 \cap B$.

▶ **Lemma 3.** *The set Q is simply connected and does not contain point-contacts.*

Proof. Consider a point-contact between two cells $c_1, c_2 \in Q_1 \cup Q_2$ and a cell $c \notin Q_1 \cup Q_2$ that is adjacent to both c_1 and c_2 (so $c \in Q_3$). Since Q_2 contains only even-even cells, we may assume that $c_1 \in Q_1$. Recall that $\mathcal{M}(c_1) \subseteq P$ by definition. We may further assume that c_1 is an odd-odd cell, for otherwise a cell in Q_2 would eliminate the point-contact. Hence, all cells point-adjacent to c_1 are in $Q_1 \cup Q_2$, and thus c has three adjacent cells in $Q_1 \cup Q_2$. This implies that adding $c \in Q_3$ to $Q_1 \cup Q_2$ cannot introduce point-contacts or holes. Similarly, cells in Q_4 connect two oppositely adjacent cells in Q_2, and thus cannot introduce point-contacts (or holes, by definition). Combining this with Lemma 2 implies that Q is hole-free and does not contain point-contacts.

It remains to show that Q is connected, that is, the set Q_4 exists. Consider two cells $c_1, c_2 \in Q$. We show that c_1 and c_2 are connected in Q. We may further assume that $c_1, c_2 \in Q_2$, as cells in $Q_1 \cup Q_3 \cup Q_4$ must be adjacent or point-adjacent to a cell in Q_2. Let $p \in \mathcal{M}(c_1) \cap P$, $q \in \mathcal{M}(c_2) \cap P$ and consider a path π between p and q inside P. Every even-even cell c with $\mathcal{M}(c) \cap \pi \neq \emptyset$ must be in Q_2. Furthermore, the modules of even-even cells cover the plane. Every cell connecting a consecutive pair of even-even cells intersecting π satisfies the conditions of Q_4, and thus can be added to make c_1 and c_2 connected in Q. ◀

Upper bounds. To prove our bounds, note that $\mathcal{M}(c) \cap P \neq \emptyset$ for every cell $c \in Q$. This is explicit for cells in Q_1, Q_2, and Q_4. For cells in Q_3, note that these cells must be adjacent to a cell in Q_1, and thus contain a point in P.

▶ **Lemma 4.** $d_H(P, Q), d_H(\partial P, \partial Q) \leq \frac{1}{2}\sqrt{2}$.

Proof. Let $p \in P$ and consider the even-even cell c such that $p \in \mathcal{M}(c)$. Since $c \in Q_2$, the distance $d_H(p, Q) \leq d_H(p, c) \leq \frac{1}{2}\sqrt{2}$. Now consider a point $p \in \partial P$. There is a 2×2-set of cells whose modules contain p. This set contains an even-even cell $c \in Q$ and an odd-odd cell $c' \notin Q$. The latter is true, because odd-odd cells in Q must be in Q_1. Therefore, the point q shared by c and c' must be in ∂Q. Thus, $d_H(p, \partial Q) \leq d_H(p, q) \leq \frac{1}{2}\sqrt{2}$. ◀

▶ **Lemma 5.** $d_H(Q, P), d_H(\partial Q, \partial P) \leq \frac{3}{2}\sqrt{2}$.

Proof. Let q be a point in Q and let $c \in Q$ be the cell that contains q. Since $\mathcal{M}(c) \cap P \neq \emptyset$, we can choose a point $p \in \mathcal{M}(c) \cap P$. It directly follows that $d_H(q, P) \leq d_H(q, p) \leq \frac{3}{2}\sqrt{2}$. Now consider a point $q \in \partial Q$, and let $c \in Q$ and $c' \notin Q$ be two adjacent cells such that $q \in \partial c \cap \partial c'$. We claim that $(\mathcal{M}(c) \cup \mathcal{M}(c')) \cap \partial P \neq \emptyset$. If $c \notin Q_1$, then $\mathcal{M}(c) \nsubseteq P$. As furthermore $\mathcal{M}(c) \cap P \neq \emptyset$, we have that $\mathcal{M}(c) \cap \partial P \neq \emptyset$. On the other hand, if $c \in Q_1$, then $\mathcal{M}(c) \subseteq P$, so $\mathcal{M}(c') \cap P \neq \emptyset$. As furthermore $\mathcal{M}(c') \nsubseteq P$ (otherwise $c' \in Q_1$), we have that $\mathcal{M}(c') \cap \partial P \neq \emptyset$. Let $p \in (\mathcal{M}(c) \cup \mathcal{M}(c')) \cap \partial P$. Then $d_H(q, \partial P) \leq d_H(q, p) \leq \frac{3}{2}\sqrt{2}$. ◀

▶ **Theorem 6.** *For every simple polygon P a simply connected grid polygon Q without point-contacts exists such that $d_H(P, Q), d_H(\partial P, \partial Q) \leq \frac{1}{2}\sqrt{2}$ and $d_H(Q, P), d_H(\partial Q, \partial P) \leq \frac{3}{2}\sqrt{2}$.*

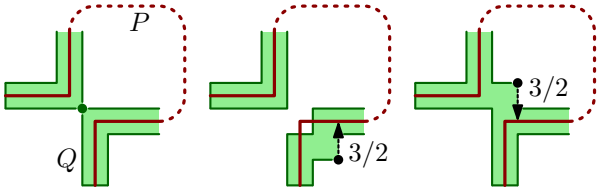


Figure 6 A polygon that does not admit a grid polygon with Hausdorff distance smaller than 3/2. The brown line signifies an infinitesimally thin polygon.

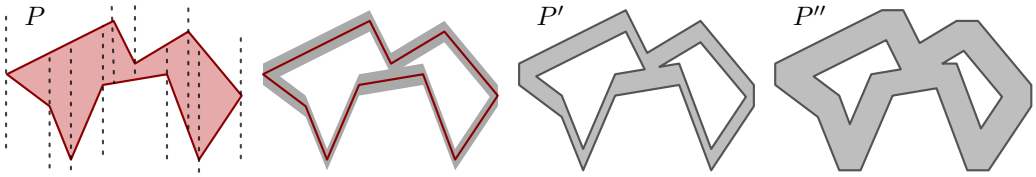


Figure 7 A simple polygon P with its vertical decomposition, and the construction of P' and P''.

Lower bound. Fig. 6 illustrates a polygon P for which no grid polygon Q exists with low $d(Q, P)$. A naive construction results in a nonsimple polygon (left). To make it simple, we can either remove a cell (center) or add a cell (right). Both methods result in $d_H(Q, P) \geq 3/2 - \epsilon$. Alternatively, we can fill the entire upper-right part of the grid polygon (not shown), resulting in a high $d_H(Q, P)$. This leads to the following theorem.

▶ **Theorem 7.** *For any $\epsilon > 0$, there exists a polygon P for which no grid polygon Q exists with $d(Q, P) < 3/2 - \epsilon$.*

In the L_∞ metric, the lower bound of $3/2 - \epsilon$ given in Fig. 6 also holds. A straightforward modification of the upper-bound proofs can be used to show that the Hausdorff distance is at most 3/2 in the L_∞ metric. In other words, our bounds are tight under the L_∞ metric.

2.2 Algorithm

To compute a grid polygon for a given polygon P with n edges, we need to determine the cells in the sets Q_1–Q_4. This is easy once we know which cells intersect ∂P. One way to do this is to trace the edges of P in the grid. The time this takes is proportional to the number of crossings between cells and ∂P. Let us denote the number of grid cells that intersect ∂P by b. Clearly, there are simple polygons with $\Theta(nb)$ polygon boundary-to-cell crossings. We show how to achieve a time bound of $O(n + B)$, where B is the number of cells in the output. The key idea is to first compute the Minkowski sum of ∂P with a square of side length 2 and use that to quickly find the cells intersecting ∂P.

To compute this Minkowski sum we first compute the vertical decomposition of ∂P, see Fig. 7. For every of the $O(n)$ quadrilaterals, determine the parts that are within vertical distance 1 from the bounding edges. The result P' is a simple polygon with holes with a total of $O(n)$ edges, and $\partial P \subset P'$. We compute the horizontal decomposition of every hole and the exterior of P' and determine all parts that are within horizontal distance 1 from the bounding edges. We add this to P', giving P''. These steps take $O(n)$ time if we use Chazelle's triangulation algorithm [9]. Essentially, the above steps constitute computing the Minkowski sum of ∂P with a square of side length 2, centered at the origin and axis-parallel.

▶ **Lemma 8.** *For any cell c, at most four edges of P'' intersect its boundary twice.*

Proof. For any edge of P'', by construction, the whole part vertically above or below it over distance at least 2 is inside P'', and the same is true for left or right. For any edge e that intersects the boundary of c twice, one side of that edge is fully in the interior of P'', and hence, cannot contain other edges of P''. Hence, e can be charged uniquely to a corner of c. ◀

▶ **Corollary 9.** *The number of polygon boundary-to-cell crossings of P'' is $O(n+b)$, where b is the number of grid cells intersecting ∂P.*

By tracing the boundary of P'', we can identify all cells that intersect it. Then we can determine all cells that intersect the boundary of P, because these are the cells that lie fully inside P''. The modifications needed to find all cells whose module lies inside P are straightforward. In particular, we can find all cells whose module lies inside P, but have a neighbor for which this is not the case in $O(n+b)$ time. This allows us to find the $O(B)$ cells selected in step Q_1 in $O(n+B)$ time. Steps Q_2 and Q_3 are now straightforward as well.

We now have a number of connected components of chosen grid cells. No component has holes, and if there are k components, we can connect them into one with only $k-1$ extra grid cells. We walk around the perimeter of some component and mark all non-chosen cells adjacent to it. If a cell is marked twice, it is immediately removed from consideration. Cells that are marked once but are adjacent to two chosen cells will merge two different components. We choose one of them, then walk around the perimeter of the new part and mark the adjacent cells. Again, cells that are marked twice (possibly, both times from the new part, or once from the old and once from the new part) are removed from consideration. Continuing this process unites all components without creating holes.

▶ **Theorem 10.** *For any simple polygon P with n edges, we can determine a set of B cells that together form a grid polygon Q in $O(n+B)$ time, such that $d_H(P,Q)$, $d_H(\partial P, \partial Q) \leq \frac{1}{2}\sqrt{2}$ and $d_H(Q,P)$, $d_H(\partial Q, \partial P) \leq \frac{3}{2}\sqrt{2}$.*

2.3 Heuristic improvements

The grid polygon Q constructed in Section 2.1 does not follow the shape of P closely (see Fig. 4). Although the boundary of Q remains close to the boundary of P, it tends to zigzag around it due to the way it is constructed. As a result, the symmetric difference between P and Q is relatively high. We consider two modifications of our algorithm to reduce the symmetric difference between P and Q while maintaining a small Hausdorff distance:

1. We construct Q_4 with symmetric difference in mind.
2. We post-process the resulting polygon Q by adding, removing, or shifting cells.

Construction of Q_4. Instead of picking cells arbitrarily when constructing Q_4 we improve the construction with two goals in mind: (1) to directly reduce the symmetric difference between P and Q, and (2) to enable the post-processing to be more effective. To that end, we construct Q_4 by repeatedly adding the cell c (not introducing holes) that has the largest overlap with P. These cells are the ones that reduce the symmetric difference between P and Q the most.

Post-processing. After computing the grid polygon Q, we allow three operations to reduce the symmetric difference: (1) adding a cell, (2) removing a cell, and (3) shifting a cell to a neighboring position. These operations are applied iteratively until there is no operation that can reduce the symmetric difference. Every operation must maintain the following conditions:

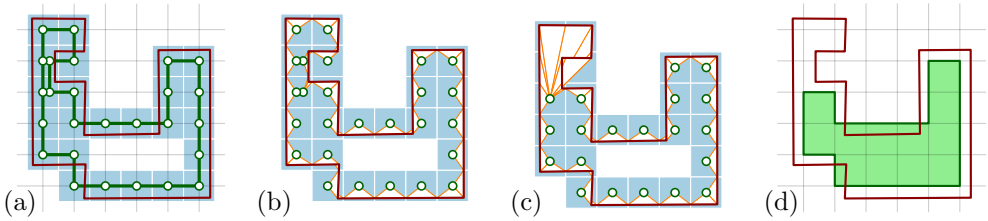


Figure 8 Constructing Q for the upper bound on the Fréchet distance. (a) Input polygon on the grid and the squares it visits (shaded); initial state of C with revisited vertices slightly offset for legibility. (b) Initial mapping μ (white triangles) between the vertices of C and ∂P. (c) Removal of duplicate vertices in C, and its effect on μ. (d) Resulting cycle represents a grid polygon.

(1) Q is simply connected, and (2) the Hausdorff distance between P (∂P) and Q (∂Q) is small. For the second condition we allow a slight relaxation with regard to the bounds of Lemma 4: $d_H(P, Q)$ and $d_H(\partial P, \partial Q)$ can be at most $\frac{3}{2}\sqrt{2}$ (like $d_H(Q, P)$ and $d_H(\partial Q, \partial P)$). This relaxation gives the post-processing more room to reduce the symmetric difference.

3 Fréchet distance

The Fréchet distance d_F between two curves is generally considered a better measure for similarity than the Hausdorff distance. For an input polygon P, we consider computing a grid polygon Q such that $d_F(\partial P, \partial Q)$ is bounded by a small constant. We study under what conditions on ∂P this is possible and prove an upper and lower bound. However, if ∂P zigzags back and forth within a single row of grid cells, any grid polygon must have a large Fréchet distance: the grid is too coarse to follow ∂P closely. To account for this in our analysis, we introduce a realistic input model, as explained below.

Narrow polygons. For $a, b \in \partial P$, we use $|ab|_{\partial P}$ to denote the perimeter distance, i.e., the shortest distance from a to b along ∂P. We define *narrowness* as follows.

▶ **Definition 11.** A polygon P is (α, β)-*narrow*, if for any two points $a, b \in \partial P$ with $|ab| \leq \alpha$, $|ab|_{\partial P} \leq \beta$.

Given a value for α, we refer to the minimal β as the α-narrowness of a polygon. We assume $\alpha < \beta$, to avoid degenerately small polygons. We note that narrowness is a more forgiving model than straightness [3]. A polygon P is κ-*straight* if for any two points $a, b \in \partial P$, $|ab|_{\partial P} \leq \kappa \cdot \|a - b\|$. A κ-straight polygon is $(\alpha, \kappa\alpha)$-narrow for any α, but not the other way around. In particular, a finite polygon that intersects itself (or comes infinitesimally close to doing so) has a bounded narrowness, whereas its straightness becomes unbounded.

Upper bound. With our realistic input model in place, we can bound the Fréchet distance needed for a grid polygon from above. In particular, we prove the following theorem.

▶ **Theorem 12.** *Given a $(\sqrt{2}, \beta)$-narrow polygon P with $\beta \geq \sqrt{2}$, there exists a grid polygon Q such that $d_F(\partial P, \partial Q) \leq (\beta + \sqrt{2})/2$.*

Proof. To prove the claimed upper bound, we construct Q via a grid cycle C that defines ∂Q. The construction is illustrated in Fig. 8. We define the *square* of a grid-graph vertex v to be the 1×1-square centered on v. Let C be the cyclic chain of vertices whose square is intersected by ∂P, in the order in which ∂P visits them. We define a mapping μ between the

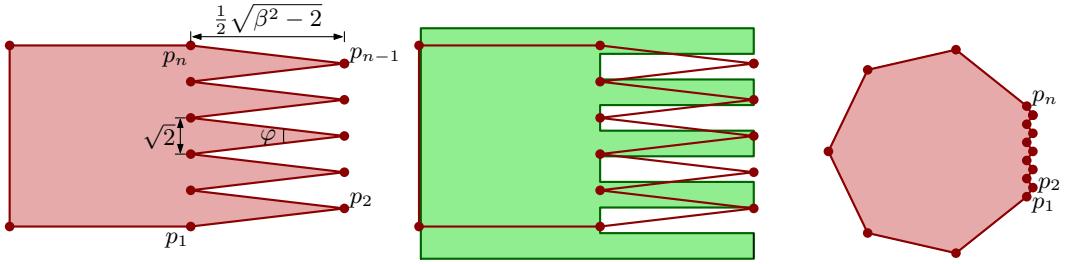


Figure 9 Polygon P (left) for which any grid polygon will have high Fréchet distance (center); polygon P for $\beta < 2$ (right).

vertices of C and ∂P. In particular, for each $c \in C$, let $\mu(c)$ be the "visit" of ∂P that led to c's existence in C, that is, the part of ∂P within the square of c. By construction, we have that $\|c - p_c\| \leq \sqrt{2}/2$ for all $c \in C$ and $p_c \in \mu(c)$. The visits $\mu(c)$ and $\mu(c')$ for two consecutive vertices, c and c', in C intersect in a point (or, in degenerate cases, in a line segment) that lies on the common boundary of the squares of c and c'; let p denote such a point. For any point σ on the line segment between c and c', we have that $\|\sigma - p\| \leq \max\{\|c-p\|, \|c'-p\|\} \leq \sqrt{2}/2$, as the Euclidean distance is convex (i.e., its unit disk is a convex set). Hence, μ describes a continuous mapping on ∂P and acts as a witness for $d_F(\partial P, C) \leq \sqrt{2}/2$.

However, C may contain duplicates and thus not describe a grid polygon Q. We argue here that we can remove the duplicates and maintain μ in such a way that it remains a witness to prove that $d_F(\partial P, C) \leq (\beta + \sqrt{2})/2$. Let c and c' be two occurrences in C of the same vertex v. Let $p \in \mu(c)$ and $p' \in \mu(c')$, both in the square of v. As they lie within the same square, $\|p - p'\| \leq \sqrt{2}$ and hence we know that $|pp'|_{\partial P} \leq \beta$. Hence, at least one of the two subsequences of C strictly in between c and c' maps via μ to a part of ∂P that has length at most β. We pick one such subsequence and remove it as well as c' from C. We concatenate to $\mu(c)$ the mapped parts of ∂P from the removed vertices. As the length of the mapped parts is bounded by β, the maximal distance between any point on these mapped parts is $\beta/2 + \sqrt{2}/2$. Hence, after removing all duplicates, we are left with a cycle C, with μ as a witness to testify that $d_F(\partial P, C) \leq (\beta + \sqrt{2})/2$.

If C contains at least three vertices, it describes a grid polygon and we are done. However, if C consists of at most two vertices, then it does not describe a grid polygon. We can extend C easily into a 4-cycle for which the bound still holds (see [7] for details). ◀

The proof of the theorem readily leads to a straightforward algorithm to compute such a grid polygon. The construction poses no restrictions on the order in which to remove duplicates and the decisions are based solely on the lengths of $\mu(v)$. Hence, the algorithm runs in linear time by walking over P to find C and handling duplicates as they arise.

Lower bound. To show a lower bound, we construct a $(\sqrt{2}, \beta)$-narrow polygon P for which there is no grid polygon with Fréchet distance smaller than $\frac{1}{4}\sqrt{\beta^2 - 2}$ to P, for any $\beta > \sqrt{2}$. First, construct a polygonal line $L = (p_1, \ldots, p_n)$, where $n = 2\left\lceil \frac{1}{4}\sqrt{\beta^2 - 2} \right\rceil + 1$. Vertex p_i is $(0, i/2)$ if i is odd and $(\frac{1}{2}\sqrt{\beta^2 - 2},\ i/\sqrt{2})$ otherwise. Now, consider a regular k-gon with side length $(n-1)/\sqrt{2}$ and $k \geq 4$ such that its interior angles are at least $\varphi = \arccos(1 - 4/\beta^2)$. Assume the k-gon has a vertical edge on the right-hand side. We replace this edge by L to construct our polygon P. Fig. 9 shows a polygon for $k = 4$ ($\beta \geq 2$) and for $k = 7$ ($\beta < 2$).

The two lemmas below readily imply our lower bound on the Fréchet distance. We omit proof of the first, but details can be found in [7].

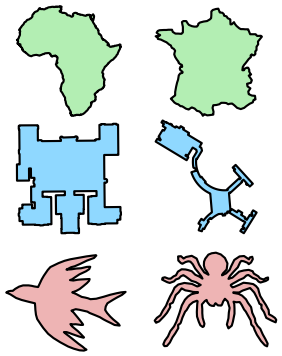


Figure 10 The input categories.

▶ **Lemma 13.** *The constructed polygon P described above is $(\sqrt{2}, \beta)$-narrow.*

▶ **Lemma 14.** *For constructed polygon P and any grid polygon Q, $d_F(\partial P, \partial Q) \geq \frac{1}{4}\sqrt{\beta^2 - 2}$.*

Proof. We show this by contradiction: assume that a grid polygon Q exists with $d_F(\partial P, \partial Q) = \varepsilon < \frac{1}{4}\sqrt{\beta^2 - 2}$. For any vertex p_i of P, there must be a point $q_i \in \partial Q$ (not necessarily a vertex) such that $\|p_i - q_i\| < \varepsilon$. Moreover, these points $q_1, \ldots, q_n$ need to appear on ∂Q in order. Equivalently, if we draw disks with radius ε centered at $p_1, \ldots, p_n$, curve ∂Q needs to visit these disks in order.

The disks centered at $p_1, p_3, \ldots, p_n$ never intersect the disks centered at $p_2, p_4, \ldots, p_{n-1}$. In particular, the disks centered at $p_1, p_3, \ldots, p_n$ are all to the left of the vertical line $v\colon x = \frac{1}{4}\sqrt{\beta^2 - 2}$, and all disks centered at $p_2, p_4, \ldots, p_{n-1}$ are all to the right of this line. Hence, between q_1 and q_2, ∂Q must contain at least one horizontal line segment crossing line v to the right, and between q_2 and q_3 there must be at least one horizontal segment crossing v to the left, and so on until we reach q_n. Since Q is simple, this requires that the difference between the maximum and the minimum y-coordinate of the these horizontal segments on ∂Q is at least $n - 1$. The y-difference between p_1 and p_n is only $(n-1)/\sqrt{2}$. This implies $d_F(\partial P, \partial Q) \geq n - 1 - (n-1)/\sqrt{2} > \frac{1}{4}\sqrt{\beta^2 - 2}$ and thus contradicts our assumption. ◀

▶ **Theorem 15.** *For any $\beta > \sqrt{2}$, there exists a $(\sqrt{2}, \beta)$-narrow polygon P such that $d_F(\partial P, \partial Q) \geq \frac{1}{4}\sqrt{\beta^2 - 2}$ holds for any grid polygon Q.*

4 Experiments

Here, we apply our algorithms to a set of polygons that can be encountered in practice. We investigate the performance of the Hausdorff algorithm and its heuristics as well as the Fréchet algorithm. Moreover, we consider the effects of grid resolution and the placement of the input. Full details on the experiments can be found in [7].

Data set. We use a set of 34 polygons: 14 territorial outlines (countries, provinces, islands), 11 building footprints and 9 animal silhouettes (see Fig. 10 for six examples). We scale all input polygons such that their bounding box has area r; we call r the resolution. Unless stated otherwise, we use $r = 100$. This scaling is used to eliminate any bias introduced from comparing different resolutions.

Table 1 Normalized symmetric difference, as an increase percentage w.r.t. optimal, of the algorithms. Note that "optimal" here means optimal for the symmetric difference when not insisting on a connected set of cells. For the Hausdorff algorithm, results for the various heuristic improvements are shown. In the second row, *None* means that no postprocessing heuristic was used; A, R and S mean additions, removals and shifts, respectively. In the third row, ✓ and ✗ indicate whether Q_4 was chosen arbitrarily (✗) or using the symmetric difference heuristic (✓).

	Optimal	**Hausdorff**						**Fréchet**
postproc.		*None*		*A / R*		*A / R / S*		
Q_4 *heur.*		✗	✓	✗	✓	✗	✓	
Maps	0.223	+ 316 %	+ 238 %	+ 39 %	+ 3 %	+ 11 %	+ 3 %	+ 23 %
Buildings	0.257	+ 270 %	+ 197 %	+ 47 %	+ 9 %	+ 21 %	+ 8 %	+ 17 %
Animals	0.333	+ 246 %	+ 188 %	+ 60 %	+ 12 %	+ 29 %	+ 11 %	+ 8 %

4.1 Symmetric difference

We start our investigation by measuring the symmetric difference between the input and output polygon. If the symmetric difference is small, this indicates that the output is similar to the input. We normalize the symmetric difference by dividing it by the area of the input polygon. The results of our algorithms depend on the position of the input polygon relative to the grid. Hence, for every input polygon we computed the average normalized symmetric difference over 20 random placements.

Computing a (simply connected) grid polygon that minimizes symmetric difference is NP-hard [21]. Hence, as a baseline for our comparison, we compute the set of cells with the best possible symmetric difference by simply taking all cells that are covered by the input polygon for at least 50 %. This set of cells is optimal with respect to symmetric difference but may not be simply connected. It can hence be thought of as a lower bound.

Overview. In Table 1, we compare the Fréchet algorithm and the various instantiations of the Hausdorff algorithm in terms of the (normalized) symmetric difference. The second column lists the average symmetric difference of the symmetric-difference optimal solution, calculated as described above. The other columns are hence given as a percentage representing the increase with respect to the optimal value. We aggregated the results per input type.

The table tells us that, with the use of heuristics, the Hausdorff algorithm gets quite close to the optimal symmetric difference, while still bounding the Hausdorff distance and guaranteeing a grid polygon. The Fréchet algorithm is performing more poorly in comparison, though interestingly performs *better* on the animal contours.

Fig. 11 shows three solutions for one of the input polygons: symmetric-difference optimal, Fréchet algorithm and Hausdorff algorithm with heuristics. The symmetric-difference optimal solution looks like the input, but consists of multiple disconnected polygons. The result of the Fréchet algorithm is a single grid polygon, but the algorithm cuts off narrow parts. The result of the Hausdorff algorithm is also a single grid polygon, but does not have to cut off parts when input is narrow.

Below, we examine the effect of the different heuristics for the Hausdorff algorithm to explain their success. Moreover, we show that the performance of the Fréchet algorithm is highly dependent on the grid resolution.

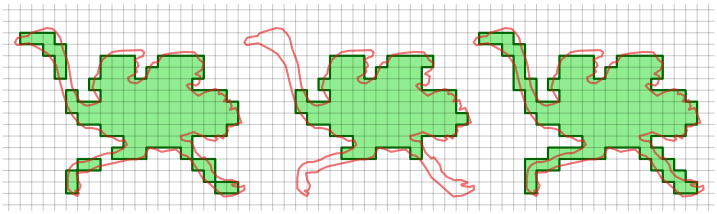


Figure 11 Example outputs for the symmetric-difference optimal algorithm (left), the Fréchet algorithm (center) and the Hausdorff algorithm (right). Note that the first does not yield a grid polygon.

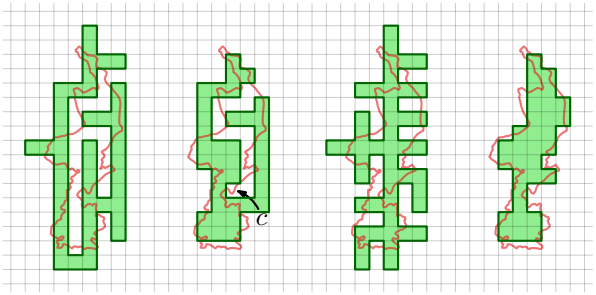


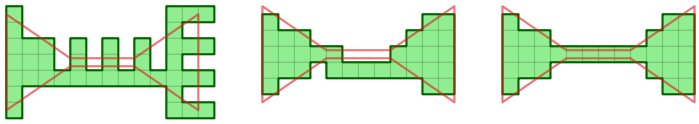
Figure 12 Without the heuristic for the Q_4 construction (a), the algorithm gets stuck in the post processing phase (b). The smart Q_4 construction gives a better starting point (c) resulting in the desired shape (d).

Hausdorff heuristics. Table 1 shows that using the heuristic for Q_4 makes a tremendous difference, especially if a postprocessing heuristic is used as well. Fig. 12 illustrates this finding with four results on the same input. In (a–b) Q_4 is chosen arbitrarily and the resulting shape does not look like the input – even after postprocessing. In particular, the postprocessing heuristic cannot progress further: the cell marked c cannot be added to Q since that would increase the symmetric difference. In (c–d) Q_4 is chosen using the heuristic; it provides a better initial solution which allows the postprocessing to create a nice result.

In the postprocessing heuristic, allowing or disallowing shifts can influence the result. See for example Fig. 13. Without shifts, the heuristic cannot move the connection between the two ends of the input polygon to the correct location as it would first need to increase the symmetric difference. With a series of diagonal shifts this can be achieved. Our experiments show that in practice allowing shifts indeed decreases the symmetric difference. However, the effect is only marginal if we use the heuristic for the Q_4 construction. Hence, we conclude that shifts only significantly improve the result if Q_4 is chosen badly.

Resolution and placement. While developing our algorithm we noticed that not just the grid resolution but also the placement of the input polygon effected the symmetric difference. Hence we set up experiments to investigate these factors. First we tested how much the resolution influences the symmetric differences. In Table 2, the results are shown, averaged over all 34 inputs. As expected, for all algorithms, the normalized symmetric difference decreases when the resolution increases.

To investigate how much the results of our algorithms depend on the input placement, we compared the minimal, maximal and average symmetric difference over 20 runs of the algorithms. The polygons were placed randomly for each run, but per polygon the same 20 positions were used for all three algorithms. We found that the difference between the



no post-processing additions / removals shifts

Figure 13 Without allowing shifts, the post-processing phase cannot move the cells in the middle to coincide with the input polygon. With shifts, this is possible.

Table 2 Normalized symmetric difference for the various algorithms on five resolutions.

	$r = 100$	$r = 225$	$r = 400$	$r = 625$	$r = 900$
Optimal	0.263	0.188	0.147	0.119	0.101
Hausdorff	0.282	0.201	0.155	0.123	0.103
Fréchet	0.306	0.227	0.184	0.148	0.122

minimum and the maximum symmetric difference for each algorithm / polygon combination is rather large. We hence concluded that placement can have a significant effect on the achieved symmetric difference. Hence, if the application permits us to choose the placement, it is advisable to do so to obtain the best possible result. This leads to an interesting open question of whether we can algorithmically optimize the placement, to avoid the need to find a good placement with trial and error. In the upcoming analysis, we also consider the effect of resolution and placement, with respect to the Fréchet distance.

4.2 Fréchet analysis

Theorem 12 predicts an upper bound on the Fréchet distance based on $\sqrt{2}$-narrowness. However, if the points defining the narrowness lie within different squares of grid vertices, this bound may be naive. Moreover, it assumes a worst-case detour, going away in a thin triangle to maximize the distance between the detour and a doubly-visited cell. Hence, the algorithm has the potential to perform better, depending on the actual geometry and its placement with respect to the grid. Here, we discuss our investigation of these effects.

Procedure. We use all 34 polygons for our experiments. As we may expect the grid resolution to significantly affect results, we used 20 different resolutions. In particular, we use resolutions varying from 10 000 to 25, using $(100/s)^2$ with scale $s \in \{1, \ldots, 20\}$.

For each resolution-polygon combination (case), we measure its $\sqrt{2}$-narrowness (see [7] for details on how to compute narrowness) and derive the predicted upper bound. Then, we run the Fréchet algorithm, using the 25 possible offsets in $\{0, 0.2, 0.4, 0.6, 0.8\}^2$, and measure the precise Fréchet distance between input and output. We keep track of three summary statistics for each case: the minimum (best), average ("expected") and maximum (worst) measured Fréchet distance.

Effect of placement. We consider placement with respect to the grid (offset) to have a significant effect on the result computed for a polygon, if the difference between the maximal and minimal Fréchet distance over the 25 offsets is at least 2. Almost 30 % of cases exhibit such a significant effect, with the animal contours being particularly affected (35 % significant). Again, this raises the question of whether we can algorithmically determine a good placement.

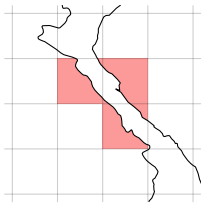


Figure 14 Red cells cause a cut-off and have high symmetric difference.

Upper bound quality. We define the performance as the measured Fréchet distance as a percentage of the upper bound. We consider a performance of 40 % significantly better than the upper bound. Using the best placement, over 95 % of cases perform significantly better. Averaging performance over placement, we still find such a majority (over 81 %). Interestingly, this drop is mostly due to the animal contours, of which only 63 % now perform significantly better. Thus, although we have a provable upper bound, we may typically expect our simple algorithm to perform significantly better than the upper bound. This holds even without any postprocessing to further optimize the result and when taking a random offset.

Effect of resolution. The influence of the resolution on the above results does not seem to exhibit a clear pattern. Nonetheless, resolution likely plays an important role in these results, but not as straightforward as either low or high resolution being more problematic. Instead, it is likely the most problematic resolutions are those at which the $\sqrt{2}$-narrowness of the polygon jumps as a new pair of edges comes within distance $\sqrt{2}$ of each other. However, an in-depth investigation of this is beyond the scope of this paper.

Heuristic improvement. In contrast to the Hausdorff algorithm, the Fréchet algorithm needs no heuristic improvement on inputs that are not too narrow. However, badly placed narrow polygons can be problematic: large parts of the polygon may be cut, greatly diminishing similarity. A solution may be to select an appropriate resolution (if our application permits us to). In our experiments the algorithm tends to perform well at resolutions where the symmetric-difference optimal solution is a single grid polygon. The advantage of our Fréchet algorithm is that it guarantees a grid polygon on all outputs and bounds the Fréchet distance.

Nonetheless, we may want to consider heuristic postprocessing to obtain a locally-optimal result. If we want to do this in terms of the symmetric difference, we may use similar techniques as for the Hausdorff algorithm. However, this does not perform well: the narrow strip that causes the Fréchet algorithm to perform badly tends to effect a high symmetric difference for the nearby grid cells (Fig. 14). As such, the result is already (close to) a local optimum in terms of the symmetric difference.

5 Conclusion

We presented two algorithms to map simple polygons to grid polygons that capture the shape of the polygon well. For measuring the distance between the input and the output, we considered the Hausdorff and the Fréchet distance. We achieved a constant bound on the Hausdorff distance; for the Fréchet distance we require a realistic input assumption to achieve a constant bound. We also evaluated our algorithms in practice. Although the Hausdorff algorithm does not produce great results directly, the algorithm achieves good results when combined with heuristic improvements. The Fréchet algorithm, on the other

hand, struggles with narrow polygons, and it is not clear how to improve the results using heuristics. Designing an algorithm for the Fréchet distance that also works well in practice remains an interesting open problem. Another interesting open problem is to algorithmically optimize the placement of the input polygon, for the best results of both the Hausdorff and the Fréchet algorithm.

References

1. Helmut Alt, Bernd Behrends, and Johannes Blömer. Approximate matching of polygonal shapes. *Annals of Mathematics & Artificial Intelligence*, 13(3-4):251–265, 1995.
2. Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5:75–91, 1995.
3. Helmut Alt, Christian Knauer, and Carola Wenk. Comparison of distance measures for planar curves. *Algorithmica*, 38(1):45–58, 2003.
4. Ernst Althaus, Friedrich Eisenbrand, Stefan Funke, and Kurt Mehlhorn. Point containment in the integer hull of a polyhedron. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 929–933, 2004.
5. K. Joost Batenburg, Sjoerd Henstra, Walter A. Kosters, and Willem Jan Palenstijn. Constructing simple nonograms of varying difficulty. *Pure Mathematics and Applications (Pu. MA)*, 20:1–15, 2009.
6. Daniel Berend, Dolev Pomeranz, Ronen Rabani, and Ben Raziel. Nonograms: Combinatorial questions and algorithms. *Discrete Applied Mathematics*, 169:30–42, 2014.
7. Quirijn W. Bouts, Irina Kostitsyna, Marc van Kreveld, Wouter Meulemans, Willem Sonke, and Kevin Verbeek. Mapping polygons to the grid with small hausdorff and fréchet distance. *Computing Research Repository (arXiv)*, abs/1606.06660, 2016.
8. Rafael G. Cano, Kevin Buchin, Thom Castermans, Astrid Pieterse, Willem Sonke, and Bettina Speckmann. Mosaic drawings and cartograms. *Computer Graphics Forum*, 34(3):361–370, 2015.
9. Bernard Chazelle. Triangulating a simple polygon in linear time. *Discrete & Computational Geometry*, 6:485–524, 1991.
10. Jinhee Chun, Matias Korman, Martin Nöllenburg, and Takeshi Tokuyama. Consistent digital rays. *Discrete & Computational Geometry*, 42(3):359–378, 2009.
11. Mark de Berg, Dan Halperin, and Mark H. Overmars. An intersection-sensitive algorithm for snap rounding. *Computational Geometry*, 36(3):159–165, 2007.
12. Olivier Devillers and Philippe Guigue. Inner and outer rounding of boolean operations on lattice polygonal regions. *Computational Geometry*, 33(1-2):3–17, 2006.
13. Michael T. Goodrich, Leonidas J. Guibas, John Hershberger, and Paul J. Tanenbaum. Snap rounding line segments efficiently in two and three dimensions. In *Proceedings of the 13th Annual Symposium on Computational Geometry (SoCG)*, pages 284–293, 1997.
14. Daniel H. Greene and F. Frances Yao. Finite-resolution computational geometry. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 143–152, 1986.
15. Warwick Harvey. Computing two-dimensional integer hulls. *SIAM Journal on Computing*, 28(6):2285–2299, 1999.
16. John Hershberger. Stable snap rounding. *Computational Geometry*, 46(4):403–416, 2013.
17. Reinhard Klette and Azriel Rosenfeld. *Digital Geometry – geometric methods for digital picture analysis*. Morgan Kaufmann, 2004.
18. Reinhard Klette and Azriel Rosenfeld. Digital straightness – a review. *Discrete Applied Mathematics*, 139(1-3):197–230, 2004.

19 Johannes Kopf, Ariel Shamir, and Pieter Peers. Content-adaptive image downscaling. *ACM Transactions on Graphics*, 32(6):Article No. 173, 2013.

20 Wouter Meulemans. *Similarity Measures and Algorithms for Cartographic Schematization*. PhD thesis, Technische Universiteit Eindhoven, 2014.

21 Wouter Meulemans. Discretized approaches to schematization. *Computing Research Repository (arXiv)*, abs/1606.06488, 2016.

22 Emilio G. Ortíz-García, Sancho Salcedo-Sanz, José M. Leiva-Murillo, Ángel M. Pérez-Bellido, and José Antonio Portilla-Figueras. Automated generation and visualization of picture-logic puzzles. *Computers & Graphics*, 31(5):750–760, 2007.