

Shortcut Hulls: Vertex-restricted Outer Simplifications of Polygons

Annika Bonerath*

Jan-Henrik Haunert†

Joseph S. B. Mitchell‡

Benjamin Niedermann§

Abstract

Let P be a crossing-free polygon and \mathcal{C} a set of shortcuts, where each shortcut is a directed straight-line segment connecting two vertices of P . A shortcut hull of P is another crossing-free polygon that encloses P and whose oriented boundary is composed of elements from \mathcal{C} . Shortcut hulls find their application in geo-related problems such as the simplification of contour lines. We aim at a shortcut hull that linearly balances the enclosed area and perimeter. If no holes in the shortcut hull are allowed, the problem admits a straight-forward solution via shortest paths. For the more challenging case that the shortcut hull may contain holes, we present a polynomial-time algorithm that is based on computing a constrained, weighted triangulation of the input polygon's exterior. We use this problem as a starting point for investigating further variants, e.g., restricting the number of edges or bends. We demonstrate that shortcut hulls can be used for drawing the rough extent of point sets as well as for the schematization of polygons.

1 Introduction

The simplification of polygons finds a great number of applications in geo-related problems. For example in map generalization it is used to obtain abstract representations of area features such as lakes, buildings, or contour lines. A common technique, which originally stems from polyline simplification, is to restrict the resulting polygon Q of a polygon P to the vertices of P , which is also called a *vertex-restricted simplification* [21, 25, 39]. In that case Q consists of straight edges¹ that are *shortcuts* between vertices of P . In the classic problem definition of line and area simplification the result Q may cross edges of P .

In this paper, we consider the vertex-restricted crossing-free simplification of a polygon P considering only shortcuts that lie in the exterior of P or are part of

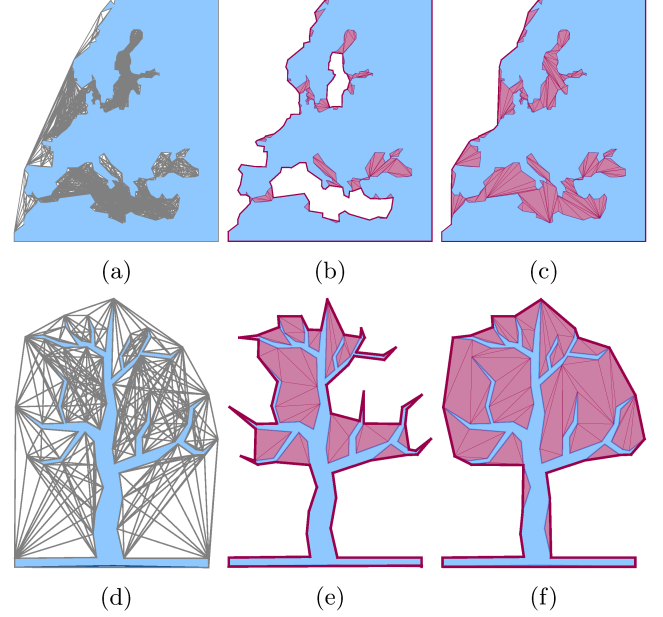


Figure 1: 1st column: Input polygon (blue) with a set \mathcal{C} of all possible shortcuts (gray). 2nd–3rd columns: Optimal \mathcal{C} -hulls (blue and red area) for different λ .

the boundary of P . In contrast to other work, we consider the shortcuts as input for our problem and do not require special properties, e.g., that they are crossing-free, or that they comprise all possible shortcuts. The result of the simplification is a *shortcut hull* Q of P possibly having holes. We emphasize that the edges of a shortcut hull do not cross each other. Figure 1 shows polygons (blue area) with all possible shortcuts and different choices of shortcut hulls (blue and red area). Such hulls find their application when it is important that the simplification contains the polygon. Figure 2 shows the simplification of a network of lakes. We emphasize that the lakes are connected to the exterior of the green polygon at the bottom side. In that use case, it can be desirable that the water area is only decreased to sustain the area of the land occupied by important map features. The degree of the simplification of Q can be measured by its perimeter and enclosed area. While a small perimeter indicates a strong simplification of P , a small area gives evidence that Q adheres to P . In the extreme case Q is either the convex hull of P minimizing the possible perimeter, or Q coincides with P minimizing the enclosed area. We present algorithms that

*Institute of Geodesy and Geoinformation, University of Bonn, bonerath@igg.uni-bonn.de

†Institute of Geodesy and Geoinformation, University of Bonn, haunert@igg.uni-bonn.de

‡Stony Brook University joseph.mitchell@stonybrook.edu

§Institute of Geodesy and Geoinformation, University of Bonn, niedermann@igg.uni-bonn.de

¹Throughout this paper, we use the term *edge* instead of *straight-line segment*.

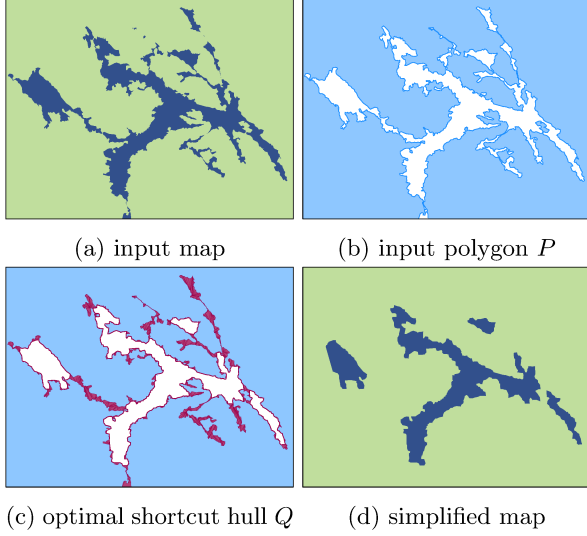


Figure 2: Simplification of a network of lakes in Sweden.

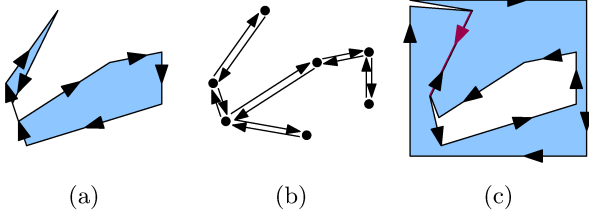


Figure 3: Weakly-simple polygons. (a)–(b) Valid input polygon as the exterior is a connected region. (c) Invalid input polygon as the exterior consists of two regions.

construct shortcut hulls of P that linearly balance these two contrary criteria by a parameter $\lambda \in [0, 1]$, which specifies the degree of simplification. With increasing λ the enclosed area is increased, while the perimeter is decreased. We show that for the case that Q must not have holes we can reduce the problem to finding a cost-minimal path in a directed acyclic graph that is based on the given set of possible shortcuts. However, especially for the application in geovisualization, where it is about the simplification of spatial structures, we deem the support of holes in the simplification as an essential key feature. For example, in Figure 2d the connections between the lakes are not displayed anymore as they are very narrow, while it is desirable to still show the large lakes. We therefore investigate the case of shortcut hulls with holes in greater detail.

Input Polygon. As input we expect a clockwise-oriented polygon P that is *weakly-simple*, which means that we allow vertices to lie in the interior of edges as well as edges that point in opposite directions to lie on top of each other; see Figure 3. In particular, the edges of P do not cross each other. Such polygons are

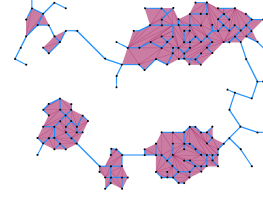


Figure 4: Shortcut hull of a minimum spanning tree.

more general than simple polygons and can be used to describe more complex geometric objects such as the faces of a graph embedded into the plane; see Figure 4 for minimum spanning tree. For the input polygon P we further require that its exterior is one connected region; we say that the exterior of P is *connected*; see Figure 3. Hence, both a simple polygon and the outer face of the plane embedding of a planar graph are possible inputs. Finally, we emphasize that P may have holes. We can handle every hole separately assuming that we have inserted a narrow channel in P connecting it with the exterior of P ; consider the lakes in Figure 2. We can force the algorithm to fill the artificially introduced channel with the interior of Q .

Formal Problem Definition. We are given a weakly-simple polygon P with connected exterior and a set \mathcal{C} of directed edges in the exterior of P such that the endpoints of the edges in \mathcal{C} are vertices of P ; see Figure 5a. We call the elements in \mathcal{C} *shortcuts*. A \mathcal{C} -hull is a weakly-simple polygon whose oriented boundary consists only of directed edges from \mathcal{C} , whose exterior is connected, and that contains P . We allow \mathcal{C} -hulls to have holes. We observe that such holes can only lie in the exterior of P . We are interested in a \mathcal{C} -hull Q that linearly balances the perimeter and enclosed area of Q . Formally, we define the *cost* of a \mathcal{C} -hull Q as

$$c(Q) = \lambda \cdot c_P(Q) + (1 - \lambda) \cdot c_A(Q), \quad (1)$$

where $\lambda \in [0, 1]$ is a given constant balancing the perimeter $c_P(Q)$ and the area $c_A(Q)$ of Q . Further, Q is *optimal* if for every \mathcal{C} -hull Q' of P it holds $c(Q) \leq c(Q')$.

SHORTCUTHULL.

given: weakly-simple polygon P with n vertices and connected exterior, set \mathcal{C} of shortcuts of P , and $\lambda \in [0, 1]$
find: optimal \mathcal{C} -hull Q of P (if it exists)

Further, we observe that it holds $|\mathcal{C}| \in O(n^2)$ as the edges of \mathcal{C} have their endpoints on the boundary of P .

Our Contribution. We first discuss how to construct an optimal \mathcal{C} -hull in $O(|\mathcal{C}|)$ time for the case that it must not have holes (Section 3). Afterwards, we turn our focus to \mathcal{C} -hulls that may have holes (Sections 4–6).

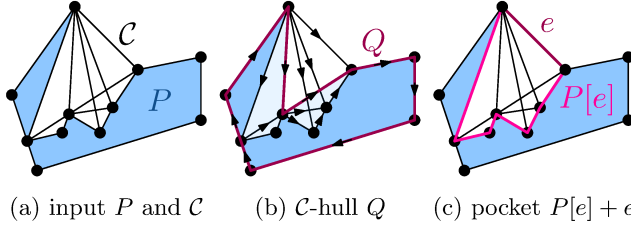


Figure 5: The input, a solution, and a subinstance for an instance of the problem.

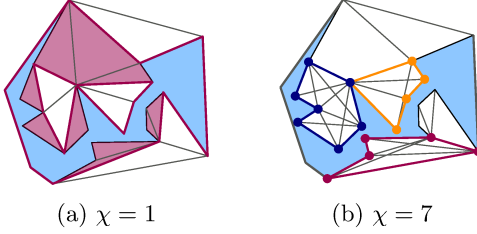


Figure 6: Two examples of set C with different spatial complexities χ . (a) C -triangulation and C -hull. (b) connected components of the crossing graph.

In particular, we show that finding an optimal C -hull Q of P is closely related to finding a triangulation T of the exterior of P such that each triangle $\Delta \in T$ either belongs to the interior or exterior of Q ; see Figure 6a. We present an algorithm that solves SHORTCUTHULL in $O(n^2)$ time if we forbid holes and in $O(n^3)$ time in the general case. Moreover, in the case that the edges of C do not cross each other, it runs in $O(n)$ time. More generally, we analyse the running time based on the structure of C . Let S be the region between P and the convex hull of P . Let G be the *crossing graph* of C , i.e., each node of G corresponds to an edge in C and two nodes of G are adjacent if the corresponding edges in C cross each other. The *spatial complexity* of C is the smallest number $\chi \in \mathbb{N}$ for which every connected component of G can be enclosed by a polygon with χ edges that lies in the exterior of P and only consists of vertices from P ; see Figure 6. We show that the purposed algorithm runs in $O(\chi^3 + n\chi)$ time. We emphasize that $\chi \in O(n)$. Moreover, we present two variants of C -hulls that restrict the number of permitted edges or bends. We further discuss relations of shortcut hulls with respect to problems from application in cartography and computational geometry (Section 7).

2 Related Work

In the following, we consider two major research fields that are closely related to our work. At first, the field of representing geometric objects by less complex and possibly schematized geometric objects and, secondly, the field of constrained and weighted triangulations. Ap-

plication fields for the representation of geometric objects by less-complex and possibly schematized objects are found, for example, in cartography: administrative borders [8, 11, 27, 52], building footprints [29, 54], and metro maps [31, 43, 55]. In particular, we want to point out the generalization of isobathymetric lines in sea charts where the simplified line should lie on the downhill side of the original line to avoid the elimination of shallows [56]. In this context, it is important to find a good balance between the preservation of the information and the legibility of the visualization [12]. Considering a polygon as input geometry, a basic technique for simplification and schematization is the convex hull [6, 18, 26, 45]. An approach for rectilinear input polygons are tight rectilinear hulls [9]. Multiple other approaches for polygonal hulls of polygons exist—some of them can be solved in polynomial time [29], while others are shown to be NP-hard [30]. A closely related field is the topologically correct simplification and schematization of polygonal subdivisions [11, 24, 38, 40, 53]. For the case that multiple geometric objects are the input of the problem, there exist several techniques for combining the aggregation and representation by a more simple geometry. In the case that the input is a set of polygons, a common technique is to use a partition of the plane, such as a triangulation, as basis [17, 32, 36, 37, 46, 50]. In the case that the input is a set of points, we aim at representing this by a polygonal hull. Many approaches such as α -shapes [23] and χ -shapes [22] use a triangulation as their basis. Another approach is based on shortest-paths [19]. Note that there also exists work on combining the aggregation of point sets resulting in schematized polygons [10, 54]. For considering polylines as input there exists work on computing an enclosing simple polygon based on the Delaunay triangulation [3]. The schematization of polylines is also closely related to our approach. On the one hand, there is the schematization of a polyline inside a polygon or between obstacles [2, 35, 41, 49]. Alternatively, there also exists work on the simplification of a polyline based on a Delaunay triangulation [3, 4, 5]. For the general simplification of polylines we also refer to the Douglas-Peucker algorithm, which is most widely applied in cartography [20], and similar approaches [1, 42, 44].

Triangulating a polygon is widely studied in computational geometry. Triangulation of a simple polygon can be done in worst-case linear time [13]. A polygon with h holes, having in total n vertices, can be triangulated in $O(n \log n)$ time [28] or even $O(n + h \log^{1+\epsilon} h)$ time [7]. Our approach is particularly related to minimum-weight triangulations [47] and constrained triangulations [14, 15, 33, 34, 48].

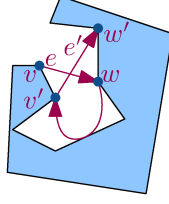


Figure 7: Illustration of proof for Theorem 1. Due to the order of the vertices of P the edges e and e' cannot both be part of S .

3 Computing Optimal Shortcut Hulls without Holes

Let G_C be the graph induced by the edges in C . We call G_C the *geometric graph* of C . If we do not allow the shortcut hull to have holes, we can compute an optimal C -hull Q based on a cost-minimal path in G_C ; see Figure 5b. For each edge e let $P[e]$ be the polyline of P that is enclosed by e . We call the polygon describing the area enclosed by e and $P[e]$ the *pocket* of e ; see Figure 5c. We direct e of G_C such that it starts at the starting point of $P[e]$ and ends at the endpoint of $P[e]$. For each edge e we introduce costs that rate the length $c_P(e)$ of e as well as the area $c_A(P[e])$ of the pocket of e with respect to λ , i.e. $c(e) = \lambda \cdot c_L(e) + (1 - \lambda) \cdot c_A(P[e])$.

Observation 1 *The vertices of the convex hull of P are part of the boundary of any shortcut hull of P .*

Due to Observation 1, any C -hull of P contains the topmost vertex v of P . Hence, G_C does not contain any edge e that contains v in its pocket and when removing v from G_C we obtain a directed acyclic graph. We use this property to prove that a cost-minimal path in G_C corresponds to an optimal C -hull.

Theorem 1 *The problem SHORTCUTHULL without holes can be solved in $O(|C|)$ time. In particular, in the case that the edges in C do not cross each other it can be solved in $O(n)$ time and $O(n^2)$ time otherwise.*

Proof. Let $S = (e_1, \dots, e_l)$ be the sequence of edges of the shortest path in G_C starting at v and ending at v . Let Q be the polygon that we obtain by interpreting S as a polygon. We show that Q is an optimal C -hull. In particular, we need to show that Q is crossing-free. Due to the definition of G_C , the following two properties hold: (i) each edge $e = xy$ of G_C starts and ends on the boundary of P and (ii) e is directed such that x is the starting point of $P[e]$ and y is the end point of $P[e]$. Hence, the vertex x appears before y on the boundary of P when going along P starting at its topmost point. Assume that the edges $e_i = x_i y_i$ and $e_j = x_j y_j$ with $1 \leq i < j \leq l$ cross. Since $i < j$, the start and end points of e_i and e_j appear in the order $x_i y_i x_j y_j$ on S . Due to properties (i) and (ii), x_j lies in the pocket of

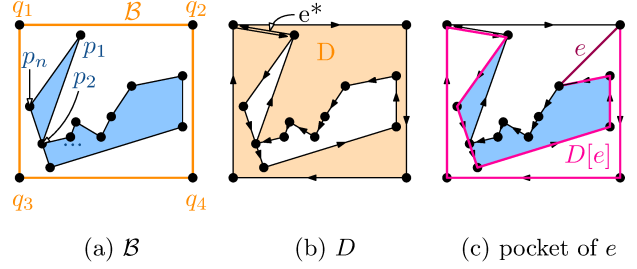


Figure 8: Containing box \mathcal{B} and sliced donut D of P .

e_i . Let $S_{i,j} = (e_{i+1}, \dots, e_{j-1})$. Since properties (i) and (ii) apply for each edge in $S_{i,j}$, this is a contradiction. The computation of a shortest path in a directed acyclic graph with $|C|$ vertices and edges takes $O(|C|)$ time [16]. In particular, when no two edges of C cross, we obtain $O(n)$ running time and otherwise $O(n^2)$. \square

If we allow Q to have holes, we cannot rate the costs for the area of a pocket in advance.

4 Structural Results for Shortcut Hulls with Holes

In this section, we present structural results for SHORTCUTHULL, which we utilize for an algorithm in Section 5. We allow the shortcut hull to have holes.

4.1 Basic Concepts

Let P be a weakly-simple polygon with connected exterior. Let p_1, \dots, p_n be the vertices of P ; see Figure 8a. We assume that the topmost vertex of P is uniquely defined; we always can rotate P such that this is the case. We denote that vertex by p_1 and assume that P is clockwise oriented. Further, let C be a set of shortcuts of P and $\lambda \in [0, 1]$; see Figure 5a. Due to Observation 1, any C -hull of P contains p_1 .

First we introduce concepts for the description of the structural results and the algorithm. Let \mathcal{B} be an axis-aligned rectangle such that it is slightly larger than the bounding box of P ; see Figure 8a. Let q_1, \dots, q_4 be the vertices of \mathcal{B} in clockwise order such that q_1 is the top-left corner of \mathcal{B} . We require that the diagonal edges $q_1 q_3$ and $q_2 q_4$ intersect P , which is always possible. We call \mathcal{B} a *containing box* of P . Let D be the polygon $q_1 \dots q_4 q_1 p_1 p_n \dots p_1 q_1$. We call D a *sliced donut* of P ; see Figure 8b. We observe that D is a weakly-simple polygon whose interior is one connected region. Further, we call $e^* = p_1 q_1$ the *cut edge* of D . For an edge e in the interior of D connecting two vertices of D let $D[e]$ be the polyline of D that connects the same vertices such that e^* is not contained; see Figure 8c. Let $D[e] + e$ be the polygon that we obtain by concatenating $D[e]$ and e such that e^* lies in the exterior of $D[e] + e$. Note that

if $e \in \mathcal{C}$ then $D[e] = P[e]$. We call $D[e] + e$ the *pocket* e . In particular, we define D to be the pocket of e^* .

Observation 2 *The edges of a \mathcal{C} -hull of P are contained in the sliced donut D .*

In the following, we define a set \mathcal{C}^+ of edges in D with $\mathcal{C} \subseteq \mathcal{C}^+$ that we use for constructing triangulations of D , which encode the shortcut hulls. Generally, a *triangulation* of a polygon H is a superset of the edges of H such that they partition the interior of H into triangles. Further, for a given set E of edges an *E -triangulation* of H is a triangulation of H that only consists of edges from E . Moreover, we say that a set E of edges is *part of* a triangulation T if E is a subset of the edges of T . Conversely, we also say that T *contains* E if E is part of T . Note that the edges of H are part of any E -triangulation of H .

We call a set \mathcal{C}^+ of edges with $\mathcal{C} \subseteq \mathcal{C}^+$ an *enrichment* of the shortcuts \mathcal{C} and the sliced donut D if (1) every edge of \mathcal{C}^+ is contained in D , (2) every edge of \mathcal{C}^+ starts and ends at vertices of D , and (3) for every set $\mathcal{C}' \subseteq \mathcal{C}$ of pair-wisely non-crossing edges there is a \mathcal{C}^+ -triangulation T of D such that \mathcal{C}' is part of T . First, we observe that \mathcal{C}^+ is well-defined as every edge in \mathcal{C} satisfies the first two properties. Further, by definition for any \mathcal{C} -hull Q there is a \mathcal{C}^+ -triangulation T of D that contains Q . Hence, as an intermediate step our algorithm for computing an optimal \mathcal{C} -hull Q creates an enrichment of \mathcal{C} and D , and then constructs a \mathcal{C}^+ -triangulation that contains Q . In Section 4.2 we discuss the structural correspondences between \mathcal{C}^+ -triangulations of D and (optimal) \mathcal{C} -hulls. In Section 4.3 we then show how to construct \mathcal{C}^+ . For example a simple approach for an enrichment of \mathcal{C} is the set of all possible shortcuts in D . We observe that any enrichment \mathcal{C}^+ of \mathcal{C} has $O(n^2)$ edges. In general, the size of \mathcal{C}^+ can be described by the spatial complexity of \mathcal{C} , which impacts the running time of our algorithm (Section 5).

4.2 From \mathcal{C}^+ -Triangulations to \mathcal{C} -Hulls

In this section, we assume that we are given an enrichment \mathcal{C}^+ for the set of shortcuts \mathcal{C} and a sliced donut D . Let T be a \mathcal{C}^+ -triangulation of D ; see Figure 9.

Observation 3 *For each enrichment \mathcal{C}^+ of \mathcal{C} and each \mathcal{C} -hull Q there exists a \mathcal{C}^+ -triangulation T of the sliced donut D such that Q is part of T .*

Let T be a \mathcal{C}^+ -triangulation of D such that the \mathcal{C} -hull Q is part of T ; see Figure 9a. We can partition the set of triangles of T in those that are contained in the interior of Q and those that are contained in the exterior of Q . We call the former ones *active* and the latter ones *inactive*; see Figure 9b. Further, we call an edge e of T a *separator* if (1) it is part of P and adjacent to an

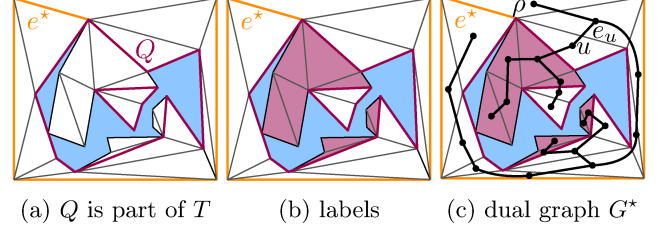


Figure 9: \mathcal{C}^+ -triangulation T . (b) The red triangles are active, while all other triangles are inactive. (c) The restricted dual graph G^* of T forms a tree with root ρ .

inactive triangle, or (2) it is adjacent to both an active and an inactive triangle. Conversely, let $\ell: T \rightarrow \{0, 1\}$ be a labeling of T that assigns to each triangle Δ of T whether it is active ($\ell(\Delta) = 1$) or inactive ($\ell(\Delta) = 0$). We call the pair $\mathbf{T} = (T, \ell)$ a *labeled \mathcal{C}^+ -triangulation*. From Observation 3 we obtain the next observation.

Observation 4 *For each enrichment \mathcal{C}^+ of \mathcal{C} and each \mathcal{C} -hull Q there exists a labeled \mathcal{C}^+ -triangulation such that its separators stem from \mathcal{C} and form Q .*

Let $\mathbf{T} = (T, \ell)$ be a labeled \mathcal{C}^+ -triangulation of the interior of a polygon H . We denote the set of separators of \mathbf{T} by $S_{\mathbf{T}}$. We define

$$c_P(S_{\mathbf{T}}) = \sum_{e \in S_{\mathbf{T}}} c_P(e) \text{ and } c_A(T) = \sum_{\substack{\Delta \in T, \\ \ell(\Delta)=1}} c_A(\Delta),$$

where $c_P(e)$ denotes the length of e and $c_A(\Delta)$ denotes the area of Δ . The *costs* of \mathbf{T} are then defined as

$$c(\mathbf{T}) = \lambda \cdot c_P(S_{\mathbf{T}}) + (1 - \lambda) \cdot c_A(T).$$

For any $e \in \mathcal{C}^+ \setminus \mathcal{C}$ we define $c_P(e) = \infty$. Thus, we have $c(\mathbf{T}) < \infty$ if and only if $S_{\mathbf{T}} \subseteq \mathcal{C}$. We call a labeled \mathcal{C}^+ -triangulation \mathbf{T} of H *optimal* if there is no other labeled \mathcal{C}^+ -triangulation \mathbf{T}' of H with $c(\mathbf{T}') < c(\mathbf{T})$.

Next, we show that a labeled \mathcal{C}^+ -triangulation $\mathbf{T} = (T, \ell)$ that is optimal can be recursively constructed based on optimal sub-triangulations. Let G^* be the restricted dual graph of T , i.e., for each triangle Δ of T G^* has a node and two nodes are adjacent iff the corresponding triangles are adjacent in T ; see Figure 9c.

Lemma 1 *The restricted dual graph G^* of a \mathcal{C}^+ -triangulation T of D is a binary tree.*

Proof. As each edge of T starts and ends at the boundary of D , each edge of T splits D into two disjoint regions. Hence, G^* is a tree. Further, since each node of G^* corresponds to a triangle of T , each node of G^* has at most two child nodes. \square

We call G^* a *decomposition tree* of D . Let ρ be the node of G^* that corresponds to the triangle of T that is adjacent to the cut edge e^* of D ; as e^* is a boundary edge of D , this triangle is uniquely defined. We assume that ρ is the root of G^* ; see Figure 9c. Let G_u^* be an arbitrary sub-tree of G^* that is rooted at a node u of G^* . Further, let e_u be the edge of the triangle Δ_u of u that is not adjacent to the triangles of the child nodes of u ; we call e_u the *base edge* of Δ_u . The triangles of the nodes of G_u^* form a \mathcal{C}^+ -triangulation T_u of the pocket $A_u = D[e_u] + e_u$ of e_u . Thus, G_u^* is a decomposition tree of A_u . A labeled \mathcal{C}^+ -sub-triangulation $\mathbf{T}_u = (T_u, \ell_u)$ consists of the \mathcal{C}^+ -triangulation T_u of A_u with $T_u \subseteq T$ and the labeling ℓ_u with $\ell_u(\Delta) = \ell(\Delta)$ for every $\Delta \in T_u$.

Lemma 2 Let \mathbf{T} be a labeled \mathcal{C}^+ -triangulation of D that is optimal. Let $\mathbf{T}_u = (T_u, \ell_u)$ be the labeled \mathcal{C}^+ -sub-triangulation of \mathbf{T} rooted at the node u and let $\mathbf{T}'_u = (T'_u, \ell'_u)$ be an arbitrary labeled \mathcal{C}^+ -triangulation of the same region. We denote the triangles of \mathbf{T}_u and \mathbf{T}'_u adjacent to e_u by Δ_u and Δ'_u , respectively.

If Δ_u and Δ'_u have the same labels, i.e., $\ell_u(\Delta_u) = \ell'_u(\Delta'_u)$, then $c(\mathbf{T}_u) \leq c(\mathbf{T}'_u)$.

Proof. For the proof we use a simple exchange argument. Assume that there is a labeled \mathcal{C}^+ -triangulation \mathbf{T}'_u of the pocket $D[e_u] + e_u$ with $\ell(\Delta_u) = \ell'(\Delta'_u)$ and $c(\mathbf{T}'_u) < c(\mathbf{T}_u)$. As both \mathbf{T}_u and \mathbf{T}'_u are triangulations of the pocket $D[e_u] + e_u$, we can replace the triangles of T_u with the triangles of T'_u in T obtaining a new triangulation T' of D . Further, we define a new labeling ℓ' such that $\ell'(\Delta) = \ell(\Delta)$ for every $\Delta \in T \setminus T_u$ and $\ell'(\Delta) = \ell'(\Delta)$ for every $\Delta \in T'_u$. Let $\mathbf{T}' = (T', \ell')$ be the corresponding labeled \mathcal{C}^+ -triangulation of D . The following calculation shows $c(\mathbf{T}') < c(\mathbf{T})$, which contradicts the optimality of \mathbf{T} .

$$\begin{aligned} c(\mathbf{T}') &= \lambda \cdot (c_P(S_{\mathbf{T}'} \setminus S_{\mathbf{T}'_u}) + c_P(S_{\mathbf{T}'_u})) + \\ &\quad (1 - \lambda) \cdot (c_A(T' \setminus T'_u) + c_A(T'_u)) \\ &= \lambda \cdot c_P(S_{\mathbf{T}} \setminus S_{\mathbf{T}'_u}) + (1 - \lambda) \cdot c_A(T \setminus T'_u) + \\ &\quad \lambda \cdot c_P(S_{\mathbf{T}'_u}) + (1 - \lambda) \cdot c_A(T'_u) \\ &= \lambda \cdot c_P(S_{\mathbf{T}} \setminus S_{\mathbf{T}'_u}) + (1 - \lambda) \cdot c_A(T \setminus T'_u) + c(\mathbf{T}'_u) \\ &< \lambda \cdot c_P(S_{\mathbf{T}} \setminus S_{\mathbf{T}_u}) + c(\mathbf{T}_u) = c(\mathbf{T}) \end{aligned}$$

Altogether, we obtain the statement of the lemma. \square

We use Lemma 2 for a dynamic programming approach that yields a labeled \mathcal{C}^+ -triangulation \mathbf{T} of D that is optimal.

Lemma 3 Let \mathbf{T} be a labeled \mathcal{C}^+ -triangulation of D that is optimal and has cost $c(\mathbf{T}) < \infty$. The separators of \mathbf{T} form an optimal \mathcal{C} -hull of P .

Proof. We show the following two claims, which proves the lemma. (1) For every \mathcal{C} -hull Q of P there is a labeled

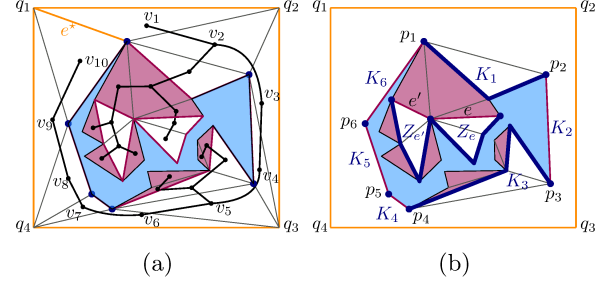


Figure 10: Proof of Lemma 3. (a) The triangles incident to the vertices q_1, q_2, q_3 and q_4 form a path in the dual graph of the labeled triangulation \mathbf{T} . (b) The vertices p_1, \dots, p_5 form a \mathcal{C}^+ -hull of P containing all active triangles (red) of \mathbf{T} .

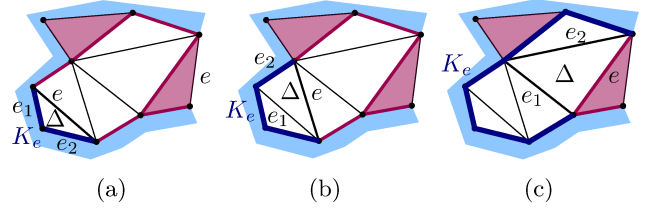


Figure 11: Inductive construction of the boundary path K_e of an edge e that is a base edge of an inactive triangle Δ . (a) Base case. (b) e_1 is a base edge of an inactive triangle, and e_2 is a separator. (c) Both e_1 and e_2 are base edges of inactive triangles.

\mathcal{C}^+ -triangulation \mathbf{T} of D such that the separators of \mathbf{T} form Q and $c(\mathbf{T}) = c(Q)$. (2) For every labeled \mathcal{C}^+ -triangulation \mathbf{T} of D with $c(\mathbf{T}) < \infty$ the separators of \mathbf{T} form a \mathcal{C} -hull Q with $c(\mathbf{T}) = c(Q)$.

Claim 1. Let Q be a \mathcal{C} -hull of P . By the definition of \mathcal{C}^+ there is a \mathcal{C}^+ -triangulation T of D such that Q is part of T . We define the labeling ℓ such that $\ell(\Delta) = 1$ for every triangle $\Delta \in T$ that is contained in the interior of Q and $\ell(\Delta) = 0$ for every other triangle $\Delta \in T$. Hence, the separators of the labeled \mathcal{C}^+ -triangulation $\mathbf{T} = (T, \ell)$ are the edges of Q . Further, by the construction of \mathbf{T} we have $c(\mathbf{T}) = c(Q)$. This proves Claim 1.

Claim 2. Let $\mathbf{T} = (T, \ell)$ be a \mathcal{C}^+ -triangulation of D with $c(\mathbf{T}) < \infty$ and let $S_{\mathbf{T}}$ be the separators of \mathbf{T} . By the definition of the costs of \mathbf{T} we have $S_{\mathbf{T}} \subseteq \mathcal{C}$. Moreover, as T is a triangulation, the edges in $S_{\mathbf{T}}$ do not cross each other. We show that the edges in $S_{\mathbf{T}}$ form a \mathcal{C} -hull Q with $c(Q) = c(\mathbf{T})$. Let G^* be the dual graph of T . As the diagonal edges of the containing box \mathcal{B} intersect P , each triangle of T that is incident to one of the vertices of \mathcal{B} is also incident to a vertex of P ; see Figure 10a. The vertices of the triangles incident to the vertices of \mathcal{B} form a path v_1, \dots, v_k in G^* such that v_1 is the root of G^* and v_k is a leaf. We denote the triangles represented by this path by $\Delta_1, \dots, \Delta_k$, respectively.

Let p_1, \dots, p_l be the vertices of P in the order as they

are incident to the triangles $\Delta_1, \dots, \Delta_k$ in clockwise order; see Figure 10a. We define $p_{l+1} = p_1$. The vertices p_1, \dots, p_l form a weakly-simple polygon Q' that contains P ; if P crossed Q' , this would contradict that the vertices are incident to the disjoint triangles $\Delta_1, \dots, \Delta_k$. We observe that Q' is a \mathcal{C}^+ -hull of P without holes. Let $T' \subseteq T$ be the set of triangles that are contained in Q' and let E' be the edges of these triangles. We first show that for each edge $e \in E'$ that is a base edge of an inactive triangle in \mathbf{T} there is a path K_e in the pocket of e such that (1) K_e only consists of edges from $S_{\mathbf{T}}$, (2) K_e connects the endpoints of e , and (3) the polygon $K_e + e$ only contains inactive triangles of \mathbf{T} . We call K_e the *boundary path* of e ; see Figure 11. Later, we use these boundary paths to assemble Q .

Let Δ be the inactive triangle of which e is the base edge and let e_1 and e_2 be the other two edges of Δ . We do an induction over the number of triangles of \mathbf{T} that are contained in the pocket of e . If the pocket of e only contains Δ , both edges e_1 and e_2 are edges of P ; see Figure 11a. Hence, by definition they are separators. We define K_e as the path $e_1 + e_2$, which satisfies the three requirements above. So assume that the pocket of e contains more than one triangle; see Figure 11b–c. If e_1 is not a separator, then it is the base edge of an inactive triangle. Hence, by induction there is a path K_{e_1} that satisfies the requirements above. If e_1 is a separator, we define $K_{e_1} = e_1$. In the same way we define a path K_{e_2} for the edge e_2 . The concatenation $K_{e_1} + K_{e_2}$ forms a path that satisfies the requirements above, which proves the existence of the boundary path for an edge $e \in E'$.

We now describe the construction of the boundary of Q . For a pair p_i, p_{i+1} with $1 \leq i < l$ the adjacent triangle incident to one of the vertices of \mathcal{B} is inactive. Let $K_i = p_i p_{i+1}$ if $p_i p_{i+1}$ is a separator. Otherwise, $p_i p_{i+1}$ is the base edge of an inactive triangle in \mathbf{T} . Thus, it has a boundary path $K_{p_i p_{i+1}}$ and we define K_i as K_e . The concatenation $K_1 + \dots + K_l$ forms the boundary B of a weakly-simple polygon Q that encloses P ; see Figure 10b. By construction it consists of edges from \mathcal{C} .

Finally, we show how to construct the holes of Q . Let $e \in S_{\mathbf{T}}$ be a separator that is contained in the interior of B and that is a base edge of an inactive triangle; see e and e' in Figure 10b. The polygon Z_e that consists of e and the boundary path K_e only contains inactive triangles of \mathbf{T} and is entirely contained in B . Further, for any pair e and e' of such separators in the interior of B the interiors of the polygons Z_e and $Z_{e'}$ are disjoint. Hence, we set these polygons to be the holes of Q . Thus, we obtain a \mathcal{C} -hull Q of P with holes such that the inactive triangles of \mathbf{T} lie in the exterior of Q , while all active triangles lie in the interior of Q . This implies that $c(Q) = c(\mathbf{T})$, which concludes the proof of Claim 2. \square

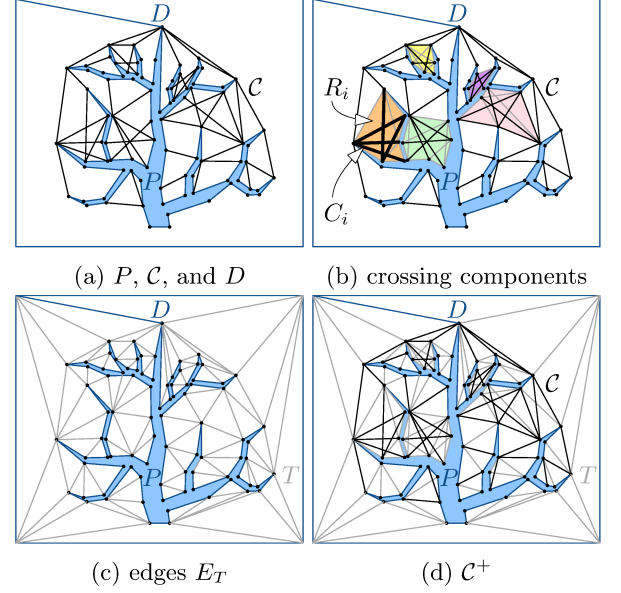


Figure 12: Obtaining the enrichment \mathcal{C}^+ from \mathcal{C} .

4.3 From \mathcal{C} to \mathcal{C}^+

Solving SHORTCUTHULL relies on the considered enrichment \mathcal{C}^+ . For an edge $e \in \mathcal{C}^+$ let δ_e be the number of triangles that can be formed by e and two other edges from \mathcal{C}^+ , and let $\delta(\mathcal{C}^+)$ be the maximum δ_e over all edges e in \mathcal{C}^+ . In Section 5 we show that the problem can be solved in $O(|\mathcal{C}^+| \cdot \delta(\mathcal{C}^+))$ time.

A simple choice for \mathcal{C}^+ is the set of all edges that lie in D and connect vertices of D . It is an enrichment of \mathcal{C} as it contains any choice of \mathcal{C} and any triangulation of D that is based on the vertices of D is a subset of \mathcal{C}^+ .

Observation 5 *There is an enrichment \mathcal{C}^+ of \mathcal{C} with $|\mathcal{C}^+| \in O(n^2)$ and $\delta(\mathcal{C}^+) \in O(n)$.*

If \mathcal{C} has no crossings, we can do much better. We first observe that the edges of any triangulation T of the sliced donut D are an enrichment of \mathcal{C} and D if \mathcal{C} is a subset of these edges. Hence, we can define an enrichment as the set of edges of a triangulation T of D such that the edges of \mathcal{C} are part of T ; for this purpose we can for example utilize constrained Delaunay triangulations, but also other triangulations are possible.

Observation 6 *If the edges in \mathcal{C} do not cross, \mathcal{C} has an enrichment \mathcal{C}^+ with $|\mathcal{C}^+| \in O(n)$ and $\delta(\mathcal{C}^+) \in O(1)$.*

In the following we generalize both constructions of \mathcal{C}^+ and relate $|\mathcal{C}^+|$ and $\delta(\mathcal{C}^+)$ to the number n of vertices of P and the spatial complexity χ of \mathcal{C} . Let $\mathcal{C}_1, \dots, \mathcal{C}_h$ be subsets of \mathcal{C} such that two edges $e \in \mathcal{C}_i$ and $e' \in \mathcal{C}_j$ with $1 \leq i, j \leq h$ cross each other if and only if $i = j$; see Figure 12. We call \mathcal{C}_i a *crossing component* of \mathcal{C} . Let R_i be the polygon in D with fewest edges, that is defined by vertices of P and contains \mathcal{C}_i . We call R_i the *region* of

C_i . Let \mathcal{C}^+ be the set of edges that contains (i) all edges of \mathcal{C} , (ii) the edges E_T of a constrained triangulation for the interior of D , and (iii) for each $1 \leq i \leq h$ the set E_{R_i} of all possible shortcuts of region R_i such that these start and end at vertices of R_i and are contained in D . Hence, an enrichment is of size $O(\chi^2 + n)$ as each region R_i has at most χ vertices.

Theorem 2 *There is an enrichment \mathcal{C}^+ of \mathcal{C} with $|\mathcal{C}^+| \in O(\chi^2 + n)$ and $\delta(\mathcal{C}^+) \in O(\chi)$.*

Proof. Let \mathcal{C}^+ be the set of edges that contains all edges of \mathcal{C} , E_T , and E_{R_1}, \dots, E_{R_h} . We show that \mathcal{C}^+ is an enrichment, by proving that for each set $\mathcal{C}' \subseteq \mathcal{C}$ of pair-wisely non-crossing edges there is a \mathcal{C}^+ -triangulation T of D such that \mathcal{C}' is part of T .

Observe that the regions R_1, \dots, R_h of crossing components induce a partition \mathbf{R} of D that contains R_1, \dots, R_h and regions R'_1, \dots, R'_g partitioning $D \setminus \bigcup_{i=1}^h R_i$. Since an edge $e \in \mathcal{C}^+$ cannot cross the boundary of two regions $R, R' \in \mathbf{R}$, the triangulation of each region $R \in \mathbf{R}$ can be constructed independently.

Let E be the edges of \mathcal{C}' that are contained in region $R \in \mathbf{R}$. If R is a region of a crossing component, \mathcal{C}^+ contains all shortcuts in this region. Since the edges of E are crossing-free, there exists a \mathcal{C}^+ -triangulation of R that is constrained to E . Thus, the edges of E are part of a \mathcal{C}^+ -triangulation of R . If R is not a region of a crossing component, the enrichment \mathcal{C}^+ contains the edges of a triangulation of D constrained to all edges of \mathcal{C} that are contained in R . Since $E \subseteq \mathcal{C}$, this triangulation contains all edges of E . By joining the \mathcal{C}^+ -triangulations for each region of the partition, we obtain a \mathcal{C}^+ -triangulation of D such that \mathcal{C}' is part of it. \square

5 Computing Optimal Shortcut Hulls with Holes

The core of our algorithm is a dynamic programming approach that recursively builds the decomposition tree of T as well as the labeling ℓ using the sliced donut D of the input polygon P and the input set of shortcut \mathcal{C} as guidance utilizing Lemma 2. The algorithm consists of the following steps.

1. Create a containing box \mathcal{B} and the sliced donut D of P and \mathcal{B} . Let e^* be the cut edge of D .
2. Create an enrichment \mathcal{C}^+ of \mathcal{C} and D .
3. Create the geometric graph $G_{\mathcal{C}^+}$ based on \mathcal{C}^+ . Let \mathcal{T} be the set of triangles in $G_{\mathcal{C}^+}$.
4. Determine for each edge e of $G_{\mathcal{C}^+}$ the set $\mathcal{T}_e \subseteq \mathcal{T}$ of all triangles (e, e_1, e_2) in $G_{\mathcal{C}^+}$ such that e_1 and e_2 lie in the pocket of e .
5. Create two tables A and I such that they have an entry for each edge e of $G_{\mathcal{C}^+}$.
 - $A[e]$: minimal cost of a labeled \mathcal{C}^+ -triangulation \mathbf{T} of the pocket $D[e] + e$ s.t. the triangle adjacent to e is active.

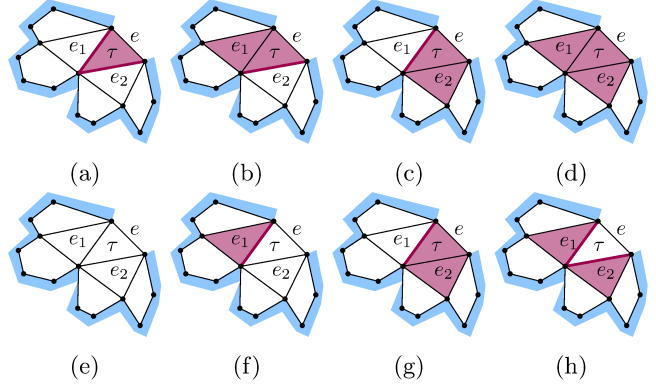


Figure 13: The possible cases for the (a)–(d) active (red) and (e)–(h) inactive cost of a triangle Δ .

- $I[e]$: minimal cost of a labeled \mathcal{C}^+ -triangulation \mathbf{T} of the pocket $D[e] + e$ s.t. the triangle adjacent to e is inactive.
6. Starting at $I[e^*]$ apply a backtracking procedure to create a \mathcal{C}^+ -triangulation \mathbf{T} of D that is optimal. Return \mathbf{T} and the corresponding optimal \mathcal{C} -hull Q of \mathbf{T} (see proof of Lemma 3 for construction of Q).

We now explain Step 5 and Step 6 in greater detail.

Step 5. We compute the table entries of A and I in increasing order of the areas of the edges' pockets. Let e be the currently considered edge of $G_{\mathcal{C}^+}$. For a triangle $\Delta = (e, e_1, e_2) \in \mathcal{T}_e$ of e we define its *active cost* x_Δ as

$$x_\Delta = \sum_{i \in \{1,2\}} \min\{A[e_i], I[e_i] + \lambda \cdot c_P(e_i)\}.$$

Hence, x_Δ is the cost of a labeled \mathcal{C}^+ -triangulation \mathbf{T}_e of the pocket $D[e] + e$ such that Δ is active and the subtriangulations of \mathbf{T}_e restricted to the pockets $D[e_1] + e_1$ and $D[e_2] + e_2$ are optimal, respectively; see Figure 13 for the four possible cases.

$$A[e] = \begin{cases} \infty & e \notin \mathcal{C} \\ \beta \cdot c_A(e) & e \in \mathcal{C}, \mathcal{T}_e = \emptyset, \\ \min\{x_\Delta \mid \Delta \in \mathcal{T}_e\} + \beta \cdot c_A(e) & e \in \mathcal{C}, \mathcal{T}_e \neq \emptyset, \end{cases}$$

where $\beta = (1 - \lambda)$. Analogously, we define for Δ its *inactive cost* y_Δ as

$$y_\Delta = \sum_{i \in \{1,2\}} \min\{A[e_i] + \lambda \cdot c_P(e_i), I[e_i]\}.$$

Hence, y_Δ is the cost of a labeled \mathcal{C}^+ -triangulation \mathbf{T}_e of the pocket $D[e] + e$ such that Δ is inactive and the subtriangulations of \mathbf{T}_e restricted to the pockets $D[e_1] + e_1$ and $D[e_2] + e_2$ are optimal, respectively. We compute the entry $I[e]$ as follows.

$$I[e] = \begin{cases} \infty & e \in \mathcal{C} \text{ and } \mathcal{T}_e = \emptyset, \\ \min\{y_\Delta \mid \Delta \in \mathcal{T}_e\} & \text{otherwise.} \end{cases}$$

By the definition of the tables A and I and Lemma 2 it directly follows, that $I[e^*]$ is the cost of a labeled \mathcal{C}^+ -triangulation of D that is optimal. In particular, by Lemma 3 the entry $I[e^*]$ is the cost of an optimal \mathcal{C} -hull.

Step 6. When filling both tables, we further store for each entry $A[e]$ the triangle $(e, e_1, e_2) \in \mathcal{T}_e$ with minimum active cost. In particular, for the edge e_i (with $i \in \{1, 2\}$) we store a pointer to the entry $A[e_i]$ if $A[e_i] < I[e_i] + \lambda \cdot c_P(e_i)$ and a pointer to the entry $I[e_i]$ otherwise. Similarly, we store for each entry $I[e]$ the triangle $(e, e_1, e_2) \in \mathcal{T}_e$ with minimum inactive cost. In particular, for the edge e_i (with $i \in \{1, 2\}$) we store a pointer to the entry $I[e_i]$ if $I[e_i] < A[e_i] + \lambda \cdot c_P(e_i)$ and a pointer to the entry $A[e_i]$ otherwise. Starting at the entry $I[e^*]$, we follow the pointers and collect for each encountered entry its triangle —if such a triangle does not exist, we terminate the traversal. If the entry belongs to A we label Δ active and if it belongs to I, we label Δ inactive. The set T of collected triangles forms a labeled \mathcal{C}^+ -triangulation \mathbf{T} of D that is optimal. By Lemma 3 the separators of \mathbf{T} form an optimal \mathcal{C} -hull.

Running Time. The first step clearly runs in $O(n)$ time. By Theorem 2 there is an enrichment \mathcal{C}^+ of \mathcal{C} and D that has size $O(\chi^2 + n)$. It can be easily constructed in $O(\chi^3 + \chi n)$ time, which dominates the running times of Step 2, Step 3 and Step 4. Further, for each edge e of $G_{\mathcal{C}^+}$ the set \mathcal{T}_e contains $\delta(\mathcal{C}^+)$ triangles. Hence, filling the tables A and I takes $O(|\mathcal{C}^+| \cdot \delta(\mathcal{C}^+))$ time. Hence, by Theorem 2 we obtain $O(\chi^3 + \chi n)$ running time. The backtracking takes the same time.

Theorem 3 *SHORTCUTHULL can be solved in $O(\chi^3 + n\chi)$ time. In particular, it is solvable in $O(n^3)$ time in general and in $O(n)$ time if the edges in \mathcal{C} do not cross.*

6 Edge and Bend Restricted Shortcut Hulls

In this section, we discuss two variants of SHORTCUTHULL in which we restrict the number of edges and bends of the computed shortcut hull. These restrictions are particularly interesting for the simplification of geometric objects as they additionally allow us to easily control the complexity of the simplification.

6.1 Restricted \mathcal{C} -Hull: Number of Edges

Next, we show how to find a \mathcal{C} -hull Q that balances its enclosed area and perimeter under the restriction that it consists of at most k edges. We say that Q is optimal *restricted to at most k edges*, if there is no other \mathcal{C} -hull Q' with at most k edges and $c(Q') < c(Q)$.

k -EDGESHORTCUTHULL.

given: weakly-simple polygon P with n vertices and connected exterior, set \mathcal{C} of shortcuts of P , $\lambda \in [0, 1]$, and $k \in \mathbb{N}$
find: optimal \mathcal{C} -hull Q of P (if it exists) restricted to at most k edges.

To solve k -EDGESHORTCUTHULL we adapt Step 5 of the algorithm presented in Section 5. We extend the tables A and I by an additional dimension of size k modelling the budget of edges that we have left for the particular instance. For a shortcut $e \in \mathcal{C}^+$ and a *budget* b we interpret the table entries as follows.

- $A[e][b]$: cost of labeled \mathcal{C}^+ -triangulation \mathbf{T} of the pocket of e s.t. \mathbf{T} is optimal, the triangle adjacent to e is active and \mathbf{T} contains at most b separators.
- $I[e][b]$: cost of labeled \mathcal{C}^+ -triangulation \mathbf{T} of the pocket of e s.t. \mathbf{T} is optimal, the triangle adjacent to e is inactive and \mathbf{T} contains at most b separators.

Let e be the currently considered edge of $G_{\mathcal{C}^+}$ when filling the tables. For a triangle $\Delta = (e, e_1, e_2) \in \mathcal{T}_e$ of e its active and inactive costs depend on the given budgets b_1 and b_2 with $1 \leq b_1, b_2 \leq k$ that we intend to use for the sub-instances attached to e_1 and e_2 .

$$x_{\Delta, b_1, b_2} = \sum_{i \in \{1, 2\}} \min\{A[e_i][b_i], I[e_i][b_i - 1] + \lambda \cdot c_P(e_i)\}$$

$$y_{\Delta, b_1, b_2} = \sum_{i \in \{1, 2\}} \min\{A[e_i][b_i - 1] + \lambda \cdot c_P(e_i), I[e_i][b_i]\}$$

Hence, for the case that $e \in \mathcal{C}$ and $\mathcal{T}_e \neq \emptyset$ we define

$$A[e][b] = \min\{x_{\Delta, b_1, b_2} \mid \Delta \in \mathcal{T}_e, b_1 + b_2 = b\} + \beta \cdot c_A(e),$$

where $\beta = (1 - \lambda)$. There are b possible choices of b_1 and b_2 that satisfy $b_1 + b_2 = b$. Thus, we can compute $A[e][b]$ in $O(b)$ time. For the remaining cases we define

$$A[e][b] = \begin{cases} \infty & e \notin \mathcal{C} \\ \beta \cdot c_A(e) & e \in \mathcal{C}, \mathcal{T}_e = \emptyset, \end{cases}$$

which can be computed in $O(1)$ time. Moreover, for the case that $e \notin \mathcal{C}$ or $\mathcal{T}_e = \emptyset$ we define

$$I[e][b] = \min\{y_{\Delta, b_1, b_2} \mid \Delta \in \mathcal{T}_e, b_1 + b_2 = b\}.$$

For the same reasons as before we can compute $I[e][b]$ in $O(1)$ time. For $e \in \mathcal{C}$ or $\mathcal{T}_e \neq \emptyset$ we define $I[e][b] = \infty$. Finally, to cover border cases we set $A[e][0] = \infty$ and $I[e][0] = \infty$. Altogether, the entry $I[e^*][k]$ contains the cost of an optimal \mathcal{C} -hull that is restricted to k edges. Apart from minor changes in Step 6 the other parts of the algorithm remain unchanged.

Running time. Compared to the algorithm of Section 5 the running time of computing a single entry increases by a factor of $O(k)$. Further, there are $O(k)$ times more entries to be computed, which yields that the running time increases by a factor of $O(k^2)$.

Theorem 4 *The problem k -EDGESHORTCUTHULL can be solved in $O(k^2(\chi^3 + n\chi))$ time. In particular, it can be solved in $O(k^2n^3)$ time in general and in $O(k^2n)$ time if the edges in \mathcal{C} do not cross.*

6.2 Restricted \mathcal{C} -Hull: Number of Bends

A slightly stronger constraint than restricting the number of edges is restricting the number of bends of a \mathcal{C} -hull. Formally, we call two consecutive edges of a simply-weakly polygon a *bend* if the enclosed angle is not 180° . We say that Q is *optimal restricted to at most k bends* if there is no other \mathcal{C} -hull Q' with at most k bends and $c(Q') < c(Q)$.

k -BENDSHORTCUTHULL.

- given:** weakly-simple polygon P with n vertices and connected exterior, set \mathcal{C} of shortcuts of P , $\lambda \in [0, 1]$, and $k \in \mathbb{N}$
find: optimal \mathcal{C} -hull Q of P (if it exists) that is restricted to at most k bends.

If the vertices of P are in general position, i.e., no three vertices lie on a common line, a \mathcal{C} -hull Q of P is optimal restricted to at most k bends if and only if it is optimal restricted to k edges. Hence, in that case we can solve k -BENDSHORTCUTHULL using the algorithm presented in Section 6.1. In applications, the case that the vertices of P are not in general position, occurs likely when the input polygon is, e.g., a schematic polygon or a polygon whose vertices lie on a grid. In that case, we add an edge p_1p_h to \mathcal{C} for each sequence p_1, \dots, p_h of at least three vertices of P that lie on a common line; we add p_1p_h only if it lies in the exterior of P . The newly obtained set \mathcal{C}' has $O(n^2)$ edges. Hence, compared to \mathcal{C} it possibly has an increased spatial complexity with $\chi \in O(n)$. From Theorem 4 we obtain the next result.

Theorem 5 *The problem k -BENDSHORTCUTHULL can be solved in $O(k^2 \cdot n^3)$ time.*

7 Relations to other Geometric Problems

We have implemented the algorithm presented in Section 5. For example, computing a shortcut hull for the instance shown in Figure 2 one run of the dynamic programming approach (Step 5) took 400ms on average. This suggests that despite its cubic worst-case running time our algorithm is efficient enough for real-world applications. However, more experiments are needed to substantiate this finding.

Balancing the Costs of Area and Perimeter In Figure 1 we display a series of optimal \mathcal{C} -hulls²; see also Figure 14. We use the same polygon and the set of all

²Figure 1b: $\lambda = 0.906$; Figure 1c: $\lambda = 0.995$; Figure 1e: $\lambda = 0.914$; Figure 1f: $\lambda = 0.975$

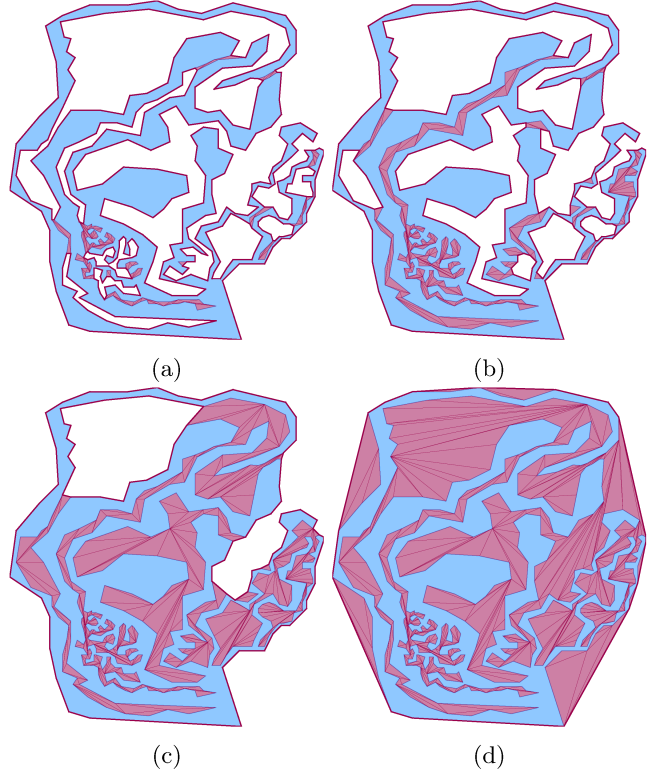


Figure 14: Optimal \mathcal{C} -hulls for increasing values of λ . In particular, for $\lambda = 1$ we only consider the costs for the area (Subfigure (a)) and for $\lambda = 0$ we only consider the costs for the perimeter (Subfigure (d)).

possible shortcuts as input while increasing the parameter λ of the cost function. To find relevant values of λ we implemented a systematic search in the range $[0, 1]$. It uses the simple observation that with monotonically increasing λ the amount of area enclosed by an optimal shortcut hull increases monotonically. More in detail, we compute the optimal shortcut hull for $\lambda = 0$ and $\lambda = 1$. If the area cost c_A of these shortcut hulls differ, we recursively consider the intervals $[0, 0.5]$ and $[0.5, 1]$ for the choice of λ similar to a binary search. Otherwise, we stop the search.

As presented in Equation 1, we consider costs for the area and perimeter in SHORTCUTHULL. The second column of Figure 1 shows a result for a small value of λ , i.e., the costs for the area are weighted higher. As expected the resulting optimal \mathcal{C} -hull is rather close to the input polygon. In contrast, the last column of Figure 1 shows the optimal \mathcal{C} -hull for a larger λ -value. We particularly obtain holes that represent large areas enclosed by the polygon, while small gaps are filled.

Simplification and Schematization of Simple Polygons

In the following, we discuss how our approach relates to typical measures for simplification and schematization. These are the number of edges, the number of bends [20]

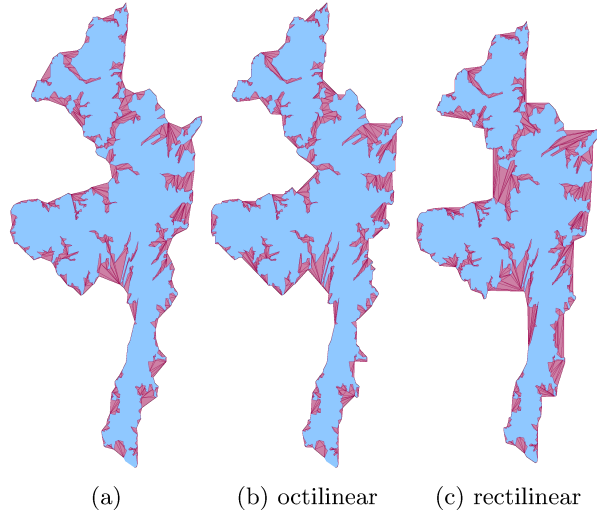


Figure 15: Simplification (a) and schematization (b)–(c) of the main island of Shetland.

or the perimeter [51], which are implemented by shortcut hulls; e.g., Figure 15a shows the simplification of the border of the main island of Shetland by a \mathcal{C} -hull as defined in `SHORTCUTHULL`. The schematization of a polygon is frequently implemented as a hard constraint with respect to a given set O of edge orientations. For schematizing a polygon with \mathcal{C} -hulls, we outline two possibilities: a non-strict and a strict schematization. For the non-strict schematization, we adapt the cost function of the shortcuts such that edges with an orientation similar to an orientation of O are cheap while the others are expensive; see Figure 15b for O consisting of horizontal, vertical, and diagonal orientations and Figure 15c for O consisting of the horizontal and vertical orientations. The strict schematization restricts the set \mathcal{C} of shortcuts, such that each edges' orientation is from O . For example, one can define \mathcal{C} based on an underlying grid that only uses orientations from O . We then need to take special care about the connectivity of \mathcal{C} , e.g., by also having all edges of the input polygon in \mathcal{C} .

Aggregation of Multiple Objects and Clustering We can adapt \mathcal{C} -hulls for multiple geometric objects, e.g. a point set. We suggest to use a geometric graph that contains all vertices of the input geometries, all edges of the input geometries and is connected as input for problem `SHORTCUTHULL`, e.g., a minimum spanning tree of the point set; see Fig 16. With increasing λ -value the regions of the shortcut hull first enclosed are areas with high density. By removing all edges of Q that are not adjacent to the interior of Q , we possibly receive multiple polygons which each can be interpreted as a cluster.

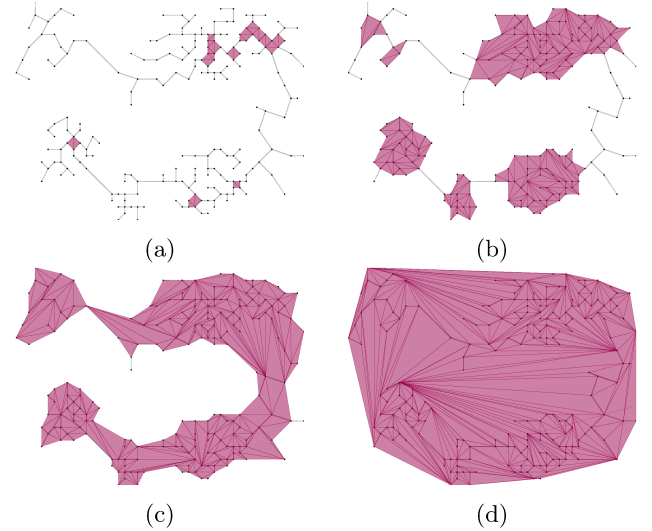


Figure 16: Optimal \mathcal{C} -hulls for increasing values of λ for a point set using a minimum spanning tree as basis.

8 Conclusion

We introduced a simplification technique for polygons that yields shortcut hulls, i.e., crossing-free polygons that are described by shortcuts and that enclose the input polygon. In contrast to other work, we consider the shortcuts as input. We introduced a cost function of a shortcut hull that is a linear combination of the covered area and the perimeter. Computing optimal shortcut hulls without holes takes $O(n^2)$ time. For the case that we permit holes we presented an algorithm based on dynamic programming that runs in $O(n^3)$ time. If the input shortcuts do not cross it runs in $O(n)$ time.

We plan on considering (i) the bends as part of the cost function, (ii) more general shortcuts, e.g. allowing one bend per shortcut, and (iii) optimal spanning trees for the case of multiple input geometries.

Acknowledgements This work has partially been funded by the German Research Foundation under Germany's Excellence Strategy, EXC-2070 - 390732324 - PhenoRob, and by NSF (Mitchell, CCF-2007275).

References

- [1] M. A. Abam, M. de Berg, P. Hachenberger, and A. Zarei. Streaming algorithms for line simplification. *Discret. Comput. Geom.*, 43(3):497–515, 2010.
- [2] J. Adegeest, M. H. Overmars, and J. Snoeyink. Minimum-link c -oriented paths: Single-source queries. *Int. J. Comput. Geom. Appl.*, 4(1):39–51, 1994.
- [3] T. Ai, S. Ke, M. Yang, and J. Li. Envelope generation and simplification of polylines using Delaunay triangulation. *Int. J. Geogr. Inf. Sci.*, 31(2):297–319, 2017.

- [4] T. Ai, Y. Liu, and J. Chen. The hierarchical watershed partitioning and data simplification of river network. In *Progress in spatial data handling*, pages 617–632. Springer, 2006.
- [5] T. Ai, Q. Zhou, X. Zhang, Y. Huang, and M. Zhou. A simplification of ria coastline with geomorphologic characteristics preserved. *Marine Geodesy*, 37(2):167–186, 2014.
- [6] C. Alegría, D. Orden, C. Seara, and J. Urrutia. Efficient computation of minimum-area rectilinear convex hull under rotation and generalizations. *J. Glob. Optim.*, 79(3):687–714, 2021.
- [7] R. Bar-Yehuda and B. Chazelle. Triangulating disjoint jordan chains. *Int. J. Comput. Geom. Appl.*, 4(4):475–481, 1994.
- [8] T. Barkowsky, L. J. Latecki, and K. Richter. Schema-tizing maps: Simplification of geographic shape by discrete curve evolution. In *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, volume 1849 of *Lecture Notes in Computer Science*, pages 41–53. Springer, 2000.
- [9] A. Bonerath, J.-H. Haunert, and B. Niedermann. Tight Rectilinear Hulls of Simple Polygons. In *Proc. of the 36th European Workshop on Computational Geometry, EuroCG 2020*.
- [10] A. Bonerath, B. Niedermann, and J. Haunert. Retrieving α -shapes and schematic polygonal approximations for sets of points within queried temporal ranges. In *Proc. of 27th Int. Conf. on Advances in Geographic Information Systems, SIGSPATIAL 2019*, pages 249–258. ACM, 2019.
- [11] K. Buchin, W. Meulemans, A. van Renssen, and B. Speckmann. Area-preserving simplification and schematization of polygonal subdivisions. *ACM Trans. Spatial Algorithms Syst.*, 2(1):2:1–2:36, 2016.
- [12] D. Burghardt, S. Schmid, and J. Stoter. Investigations on cartographic constraint formalisation. In *10th ICA Workshop on Generalization and Multiple Representation*, volume 19, page 2, 2007.
- [13] B. Chazelle. Triangulating a simple polygon in linear time. *Discret. Comput. Geom.*, 6:485–524, 1991.
- [14] L. P. Chew. Constrained Delaunay triangulations. *Algorithmica*, 4(1):97–108, 1989.
- [15] F. Y. L. Chin and C. A. Wang. Finding the constrained Delaunay triangulation and constrained voronoi diagram of a simple polygon in linear time. *SIAM J. Comput.*, 28(2):471–486, 1998.
- [16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009.
- [17] J. Damen, M. van Kreveld, and B. Spaan. High quality building generalization by extending the morphological operators. In *11th ICA Workshop on Generalization and Multiple Representation*, pages 1–12, 2008.
- [18] J. J. Daymude, R. Gmyr, K. Hinnenthal, I. Kostitsyna, C. Scheideler, and A. W. Richa. Convex hull formation for programmable matter. In *Proc. of 21st Int. Conf. on Distributed Computing and Networking, ICDCN 2020*, pages 2:1–2:10. ACM, 2020.
- [19] M. de Berg, W. Meulemans, and B. Speckmann. Delineating imprecise regions via shortest-path graphs. In *Proc. of 19th Int. Conf. on Advances in Geographic Information Systems, SIGSPATIAL 2011*, pages 271–280. ACM, 2011.
- [20] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: Int. J. for Geographic Information and Geovisualization*, 10(2):112–122, 1973.
- [21] A. Driemel and S. Har-Peled. Jaywalking your dog: Computing the fréchet distance with shortcuts. *SIAM J. Comput.*, 42(5):1830–1866, 2013.
- [22] M. Duckham, L. Kulik, M. F. Worboys, and A. Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognit.*, 41(10):3224–3236, 2008.
- [23] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Trans. Information Theory*, 29(4):551–558, 1983.
- [24] R. Estkowski and J. S. B. Mitchell. Simplifying a polygonal subdivision while keeping it simple. In *Proc. of 17th Symp. on Computational Geometry, SOCG 2001*, pages 40–49. ACM, 2001.
- [25] A. Filtser and O. Filtser. Static and streaming data structures for fréchet distance queries. In *Proc. of Symp. on Discrete Algorithms, SODA 2021*, pages 1150–1170. SIAM, 2021.
- [26] E. Fink and D. Wood. *Restricted-Oriented Convexity*. Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2004.
- [27] J. García and J. Fdez-Valdivia. Boundary simplification in cartography preserving the characteristics of the shape features. *Computers & Geosciences*, 20(3):349–368, 1994.
- [28] M. R. Garey, D. S. Johnson, F. P. Preparata, and R. E. Tarjan. Triangulating a simple polygon. *Inf. Process. Lett.*, 7(4):175–179, 1978.
- [29] J. Haunert and A. Wolff. Optimal and topologically safe simplification of building footprints. In *Proc. of 18th Int. Symp. on Advances in Geographic Information Systems, SIGSPATIAL 2010*, pages 192–201. ACM, 2010.
- [30] J.-H. Haunert, A. Wolff, et al. Optimal simplification of building ground plans. In *Proc. of 21st ISPRS Congress*, pages 372–378, 2008.
- [31] B. Jacobsen, M. Wallinger, S. G. Kobourov, and M. Nöllenburg. Metrosets: Visualizing sets as metro maps. *IEEE Trans. Vis. Comput. Graph.*, 27(2):1257–1267, 2021.
- [32] C. B. Jones, G. L. Bundy, and M. J. Ware. Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, 22(4):317–331, 1995.

-
- [33] T. C. Kao and D. M. Mount. Incremental construction and dynamic maintenance of constrained Delaunay triangulations. In *Proc. of 4th Canadian Conf. on Computational Geometry, CCCG 1992*, pages 170–175, 1992.
 - [34] D.-T. Lee and A. K. Lin. Generalized Delaunay triangulation for planar graphs. *Discrete & Computational Geometry*, 1(3):201–217, 1986.
 - [35] D. T. Lee, C. Yang, and C. K. Wong. Rectilinear paths among rectilinear obstacles. *Discret. Appl. Math.*, 70(3):185–215, 1996.
 - [36] C. Li, Y. Yin, X. Liu, and P. Wu. An automated processing method for agglomeration areas. *ISPRS Int. J. Geo Inf.*, 7(6):204, 2018.
 - [37] J. Li and T. Ai. A triangulated spatial model for detection of spatial characteristics of GIS data. In *Proc. of Int. Conf. on Progress in Informatics and Computing, PIC 2010*, volume 1, pages 155–159. IEEE, 2010.
 - [38] T. Mendel. Area-preserving subdivision simplification with topology constraints: Exactly and in practice. In *Proc. of 20th Workshop on Algorithm Engineering and Experiments, ALENEX 2018*, pages 117–128. SIAM, 2018.
 - [39] W. Meulemans. *Similarity measures and algorithms for cartographic schematization*. PhD thesis, Mathematics and Computer Science, 2014.
 - [40] W. Meulemans, A. van Renssen, and B. Speckmann. Area-preserving subdivision schematization. In *Proc. of 6th Int. Conf. on Geographic Information Science, GIScience 2010*, volume 6292 of *Lecture Notes in Computer Science*, pages 160–174. Springer, 2010.
 - [41] J. S. B. Mitchell, V. Polishchuk, and M. Sysikaski. Minimum-link paths revisited. *Comput. Geom.*, 47(6):651–667, 2014.
 - [42] G. Neyer. Line simplification with restricted orientations. In *Proc. of 6th Workshop on Algorithms and Data Structures, WADS '99*, volume 1663 of *Lecture Notes in Computer Science*, pages 13–24. Springer, 1999.
 - [43] M. Nöllenburg. A survey on automated metro map layout methods. In *Schematic Mapping Workshop 2014*, 2014.
 - [44] J. L. G. Pallero. Robust line simplification on the plane. *Comput. Geosci.*, 61:152–159, 2013.
 - [45] G. J. E. Rawlins and D. Wood. Optimal computation of finitely oriented convex hulls. *Inf. Comput.*, 72(2):150–166, 1987.
 - [46] A. Sayidov and R. Weibel. Generalization of geological maps: Aggregation and typification of polygon groups. 2019.
 - [47] M. I. Shamos and D. Hoey. Closest-point problems. In *Proc. of 16th Symp. on Foundations of Computer Science, FOCS 1975*, pages 151–162. IEEE Computer Society, 1975.
 - [48] J. R. Shewchuk and B. C. Brown. Fast segment insertion and incremental construction of constrained Delaunay triangulations. *Comput. Geom.*, 48(8):554–574, 2015.
 - [49] B. Speckmann and K. Verbeek. Homotopic c-oriented routing with few links and thick edges. *Comput. Geom.*, 67:11–28, 2018.
 - [50] S. Steiniger, D. Burghardt, and R. Weibel. Recognition of island structures for map generalization. In *Proc. of 14th Int. Symp. on Geographic Information Systems, SIGSPATIAL 2006*, pages 67–74. ACM, 2006.
 - [51] E. R. Tufte. The visual display of quantitative information. *The Journal for Healthcare Quality (JHQ)*, 7(3):15, 1985.
 - [52] T. C. van Dijk, A. van Goethem, J. Haunert, W. Meulemans, and B. Speckmann. Map schematization with circular arcs. In *Proc. of 8th Int. Conf. on Geographic Information Science, GIScience 2014*, volume 8728 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2014.
 - [53] A. van Goethem, W. Meulemans, B. Speckmann, and J. Wood. Exploring curved schematization of territorial outlines. *IEEE Trans. Vis. Comput. Graph.*, 21(8):889–902, 2015.
 - [54] M. van Kreveld, T. van Lankveld, and M. de Rie. (α, δ) -sleeves for reconstruction of rectilinear building facets. In *Progress and New Trends in 3D Geoinformation Sciences*, pages 231–247. Springer, 2013.
 - [55] H. Wu, B. Niedermann, S. Takahashi, M. J. Roberts, and M. Nöllenburg. A survey on transit map layout - from design, machine, and human perspectives. volume 39, pages 619–646, 2020.
 - [56] X. Zhang and E. Guilbert. A multi-agent system approach for feature-driven generalization of isobathymetric line. In *Advances in Cartography and GIScience. Volume 1*, pages 477–495. Springer, 2011.