Capturing the Shape of a Point Set with a Line Segment

Nathan van Beusekom[1], Marc van Kreveld[2], Max van Mulken[1], Marcel Roeloffzen[1], Bettina Speckmann[1], and Jules Wulms[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven**
`[n.a.c.v.beusekom | m.j.m.v.mulken | m.j.m.roeloffzen | b.speckmann | j.j.h.m.wulms]@tue.nl`

2 **Department of Information and Computing Sciences, Utrecht University**
`m.j.vankreveld@uu.nl`

Abstract

Detecting groups or clusters in point sets is an important task in a wide variety of application areas. In addition to detecting such groups, the group's shape carries meaning. In this paper, we aim to represent a group's shape using a simple geometric object: a line segment. Specifically, given a radius r, we say a line segment represents the shape of a point set P if it is within Hausdorff distance r from each point $p \in P$. Finding the shortest such line segment is equivalent to stabbing a set of circles of radius r using the shortest line segment. We describe an algorithm for this task that runs in $O(n \log h + h^2)$ time, where n is the size of the point set and h is the size of its convex hull.

Related Version A full version of the paper is available at `arxiv.org/abs/2402.12285`

1 Introduction

Studying groups or clusters in point sets is an important task in a wide variety of application areas. There are many algorithms and approaches to find such groups; examples include the well-known *k-means clustering* [11] or *DBSCAN* [9]. In addition to the mere existence of such groups, the group's characteristics can carry important information as well. In wildlife ecology, for example, the perceived shape of herds of prey animals contains information about the behavioral state of animals within the herd [15]. Since shape is an abstract concept that can get arbitrarily complex, it is often useful to have a simplified representation of group shape that can efficiently be computed.

In this paper, we use a simple geometric object, a line segment, as a shape descriptor of a group of entities in a point-location data set. Specifically, our input is a set P of n points in $\mathbb{R}^2$, and a radius r. Our goal is to find an (oriented) line segment q_1q_2 that lies within Hausdorff distance r from each point $p \in P$. We call such a line segment a *shape-representing* line segment of P. We propose an algorithm that finds the shortest shape-representing line segment in $O(n \log h + h^2)$ time, where h is the size of the convex hull of P.

For a line segment q_1q_2 to be within Hausdorff distance r from a point p, it must intersect the circle of radius r centered at p. Thus, we reformulate the problem: given a set of circles C_P of radius r centered at points in P, we must find the shortest line segment q_1q_2 that intersects all circles in C_P (see Figure 1). We assume that the set P is in general position; no three points of P lie on a line, and that at most two circles of C_P intersect in a point.

Due to space constraints, most proofs are omitted; they can be found in the full version.

40th European Workshop on Computational Geometry, Ioannina, Greece, March 13–15, 2024.
This is an extended abstract of a presentation given at EuroCG'24. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

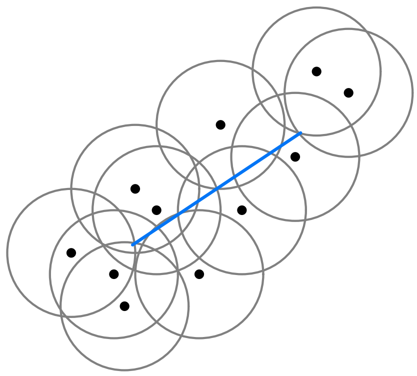

Figure 1 The line segment (blue) must hit every circle of radius r, centered at the points in P.

Related work. A number of shape descriptors have been proposed over the years. A few popular ones are the *alpha shape* of a point set [7] or the *characteristic shape* [6], both of which generate shape-representing polygons. Another way to generate the shape of a point set is to fit a function to the point set [3, 10, 17]. The set of problems of finding one or more geometric objects that intersect a different set of geometric objects is known as the set of *stabbing* problems [8], and several variants have been studied [2, 5, 14]. To our knowledge, stabbing a set of circles with the shortest line segment has not been studied. However, inverse variants that stab line segments with one or more circles have been studied [4, 13].

2 Computing the Shortest Shape-Representing Line Segment

Our algorithm is similar to the rotating calipers algorithm [16]. We start by finding the shortest shape-representing line segment for fixed orientation α, after which we rotate by π while maintaining the line segment, and return the shortest one. Note that, even though a shape-representing line segment does not exist for every orientation, we can easily find an initial orientation α for which it does exist using rotating calipers; these are the orientations at which the rotating calipers have width $\leq 2r$. Although, our input point set P can be of any shape, the following lemma shows that it suffices to consider only its convex hull $\mathrm{CH}(P)$.

▶ **Lemma 2.1.** *If a line segment q_1q_2 intersects all circles defined by the vertices of the convex hull CH(P), then q_1q_2 also intersects all circles defined by the points in P.*

Proof. Since q_1q_2 crosses each circle defined by $\mathrm{CH}(P)$, each vertex of $\mathrm{CH}(P)$ has a distance of at most r to q_1q_2. Any point on the edges of the convex hull are also at most r to q_1q_2, by definition. All other points in P are inside the convex hull and thus each point in P must have a distance of at most r to q_1q_2. ◀

We can compute $\mathrm{CH}(P)$ in $O(n \log h)$ time, where h is the size of the convex hull [1, 12]. Observe that, if rotating calipers initially finds no orientation with width $> 2r$, then point set P can be enclosed by a circle of radius at most r, and the shortest shape-representing line segment is the center point of this enclosing circle. Hence, in the rest of this paper we assume that the shortest shape-representing line segment has non-zero length.

Fixed orientation. We describe how to find the shortest shape-representing line segment with fixed orientation α. Using rotating calipers [16], we can find all orientations in which a shape-representing line segment exists. We pick α such that such a solution exists; for ease

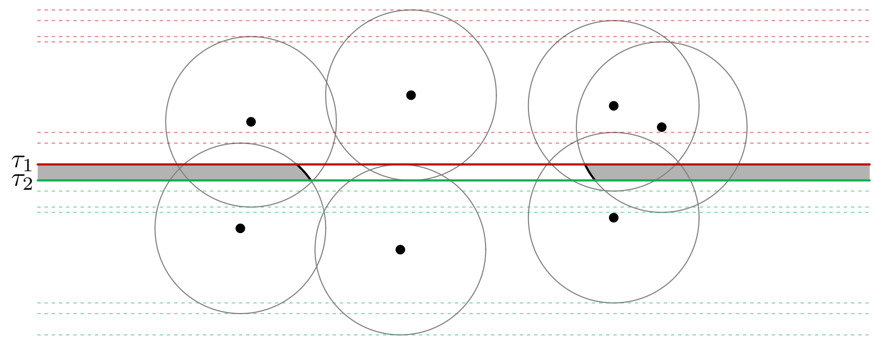

Figure 2 Two extremal tangents τ_1 and τ_2 for horizontal orientation α. The shortest line segment of orientation α that intersects all circles, ends at the boundary of the gray regions.

of exposition and without loss of generality, we assume α to be horizontal. Let the *left/right half-circle* of a circle c be the half-circle between $\pi/2$ and $3\pi/2$ and between $3\pi/2$ and $5\pi/2$, respectively. Lemma 2.1 permits us to consider only points of P on the convex hull, thus for the remainder of this paper we use C_P to indicate the set of circles of radius r centered at the vertices of $\mathrm{CH}(P)$. We use $\mathcal{A}(C_P)$ to denote the circle arrangement of C_P.

Observe that every horizontal line that lies below the bottom-most top horizontal tangent τ_1 and above the top-most bottom horizontal tangent τ_2 of all circles crosses all circles (see Figure 2). If τ_1 lies below τ_2, then there exists no horizontal line that crosses all circles.

To place q_1q_2 in the strip between τ_1 and τ_2, we can define two regions R_1, R_2 in which endpoints q_1 and q_2 must be placed such that the line segment between q_1 and q_2 intersects all circles (see Figure 2). The boundaries of R_1 and R_2 are defined by a *convex sequence* S_1 and S_2 of (in horizontal orientation) the right-most left arcs and left-most right arcs, delimited by the two tangents τ_1, τ_2, as well as these tangents themselves. If R_1 and R_2 intersect, then we can place a single point in their intersection at distance at most r from all points in P. Note that q_1 and q_2 must be on the convex sequences S_1 and S_2, respectively; otherwise, we can move the endpoint onto the convex sequence, shortening q_1q_2 and still intersecting all circles.

▶ **Lemma 2.2.** *For fixed orientation α, we can compute S_1 and S_2 in $O(h^2)$.*

Next, we must place q_1 and q_2 on S_1 and S_2, respectively, such that q_1q_2 is shortest. We show that q_1q_2 is the shortest line segment of orientation α when the tangents of S_1 at q_1 and S_2 at q_2 have equal slope. Vertices on S_1 and S_2 have a range of tangents (see Figure 3).

▶ **Lemma 2.3.** *Let S_1 and S_2 be two convex sequences of circular arcs, and let q_1 and q_2 be points on S_1 and S_2, respectively, such that line segment q_1q_2 has orientation α. If the tangent on S_1 at q_1 and the tangent on S_2 at q_2 have equal slope, then q_1q_2 is minimal.*

Observe that the length of q_1q_2 is unimodal between τ_1 and τ_2. We can hence binary search in $O(\log h)$ time for the optimal placement of q_1 and q_2. By Lemmata 2.2 and 2.3 we can compute the shortest shape-representing line segment of orientation α in $O(h^2)$ time.

Rotation. After finding the shortest line segment for a fixed orientation α, as described in the previous section, we sweep through all orientations α while maintaining τ_1, τ_2, S_1, S_2, and the shortest shape-representing line segment q_1q_2 of orientation α. We allow all of these maintained structures to change continuously as the orientation changes, and store the shortest shape-representing line segment found. Any time a discontinuous change would

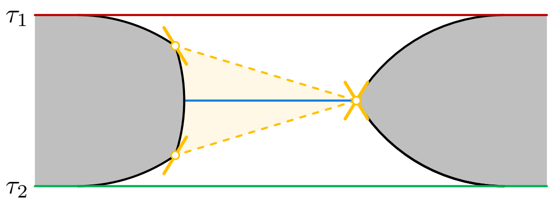

Figure 3 Two convex sequences between τ_1 and τ_2. There are multiple points on the left convex sequence that have the same tangent as the right yellow vertex. Still, there is only one line segment in horizontal orientation for which the tangents of its endpoints are equal (blue).

happen, we trigger an *event* to reflect these changes. We pre-compute and maintain a number of *certificates* in an event queue, which indicate at which orientation the next event occurs. This way we can perform the continuous motion until the first certificate is violated, recompute the maintained structures, repair the event queue, and continue rotation. We distinguish four types of events:

1. q_1 or q_2 moves onto/off of a vertex of S_1 or S_2;
2. τ_1 or τ_2 is a bi-tangent with the next circle on the convex hull;
3. τ_1 or τ_2 hits a (prospective) vertex of S_1 or S_2;
4. τ_1 and τ_2 are the same line.

Since the shortest line segment q_1q_2 in orientation α is completely determined by τ_1, τ_2, S_1, and S_2, the above list forms a complete description of all possible events. Thus, we maintain at most two certificates for events of type 1, 2, and 3, and only a single certificate for type-4 events, which are stored in a constant-size event queue Q, ordered by appearance orientation. Insert, remove, and search operations on Q can hence be performed in $O(1)$ time.

Event handling. In the following descriptions, we assume that an event happens at orientation α, and that ε is picked such that no other events occur between $\alpha - \varepsilon$ and $\alpha + \varepsilon$. Some event may occur in two symmetric cases; one of each is omitted here.

(1) q_1/q_2 moves onto/off of a vertex of S_1/S_2. We describe, without loss of generality, how to handle the event involving q_1 and S_1; the case for q_2 and S_2 are analogous. See Figure 4 for an example of this event. Observe that, since vertices of S_1 cover a range of tangents, there are intervals of orientations at which q_1 remains at a vertex of S_1. As such, we describe two different cases for this event: q_1 moves *onto* or *off* a vertex of S_1.

If q_1 was moving over an arc of S_1 at $\alpha - \varepsilon$ and encounters a vertex at α, then the movement path of q_1 is then updated to remain on the encountered vertex. Additionally, we

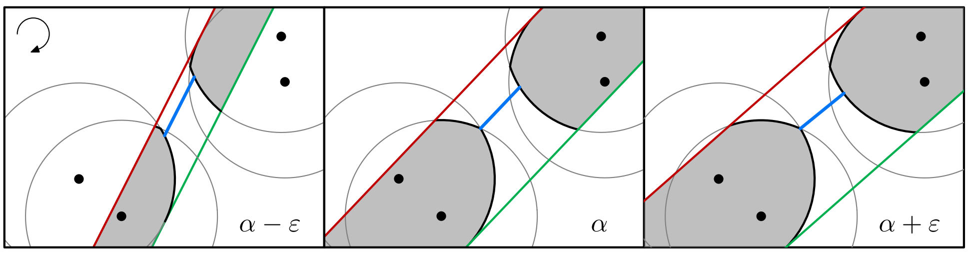

Figure 4 When q_1/q_2 is at a vertex of S_1/S_2, it stops moving.

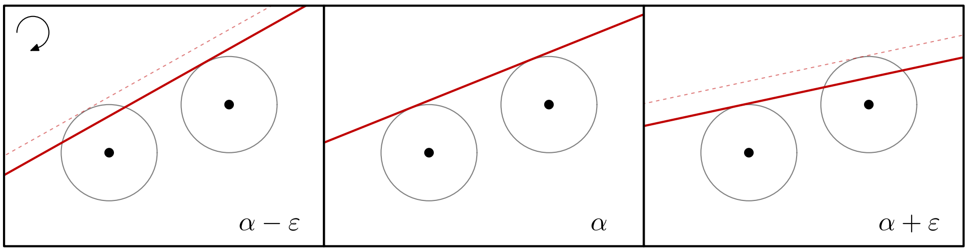

Figure 5 When the defining circle of τ_1/τ_2 changes, τ_1/τ_2 is parallel to a convex hull edge.

place a new type-1 certificate into the event queue that is violated when q_1 should move off of the vertex, e.g. when the final orientation covered by the vertex is reached.

▶ **Lemma 2.4.** *Throughout the full π rotation, q_1/q_2 moves onto/off of a vertex of S_1/S_2 at most $O(h^2)$ times, and we can resolve each occurrence of such an event in $O(1)$ time.*

(2) τ_1 or τ_2 is bi-tangent with the next circle on the convex hull. We describe, without loss of generality, how to handle the event involving τ_1; the case for τ_2 is analogous. See Figure 5 for an example of this event. When τ_1 is a bi-tangent of two circles defined by their centers $u, v \in P$ then, by definition of τ_1, u and v must both be the extremal points in the direction θ perpendicular to α. Therefore, (u, v) must be an edge on the convex hull. Suppose that, without loss of generality, u was the previous extremal vertex in direction $\theta - \varepsilon$, then v is extremal in direction $\theta + \varepsilon$. As such, τ_1 belongs to u at $\alpha - \varepsilon$, and to v at $\alpha + \varepsilon$. When this happens, we insert a new type-2 certificate into the event queue that is violated at the orientation of the next convex hull edge. Additionally, we recompute the certificates of type 3 and 4 that are currently in the event queue to reflect the updated τ_1.

▶ **Lemma 2.5.** *Throughout the full π rotation, τ_1 or τ_2 are bi-tangent with another circle at most $O(h)$ times. The two circles that define such a bi-tangent are adjacent in the convex hull, and we can resolve each occurrence of such an event in $O(1)$ time.*

(3) τ_1 or τ_2 hits a (prospective) vertex of S_1 or S_2. We describe, without loss of generality, how to handle the event involving τ_1 and S_1; the case for τ_2 and S_2 is analogous. Additionally, we can distinguish between the case where τ_1 does not belong to an arc on S (Figure 6, event 3.1) and the case where it does (Figure 7, event 3.2). We will describe the prior case here. The latter is similar, but has different certificates; the differences are described below. Let vertex v be a vertex on convex sequence S_1 that is intersected by τ_1 at orientation α. Then either vertex v is on S_1 at orientation $\alpha - \varepsilon$ but no longer on S_1 at $\alpha + \varepsilon$, or vice versa.

If the arc of S_1 intersecting τ_1 is shrinking, then at orientation α, that arc is completely removed from S_1; vertex v becomes the endpoint of S_1 and starts moving along the next arc of S_1 with the intersection point between τ_1 and S_1. If the affected arc or vertex appeared in a type-1 certificate in the event queue, it is updated to reflect the removal of the arc and the new movement of the vertex. Additionally, we place a new type-3 certificate into the event queue that is violated when τ_1 intersects the next vertex on S_1 (event 3.1), or when τ_1 hits an intersection point between S_1 and the defining circle of τ_1 (event 3.2).

▶ **Lemma 2.6.** *Throughout the full π rotation, τ_1 or τ_2 hits a vertex of S_1 or S_2 at most $O(h^2)$ times, and we can resolve each occurrence of such an event in $O(1)$ time.*

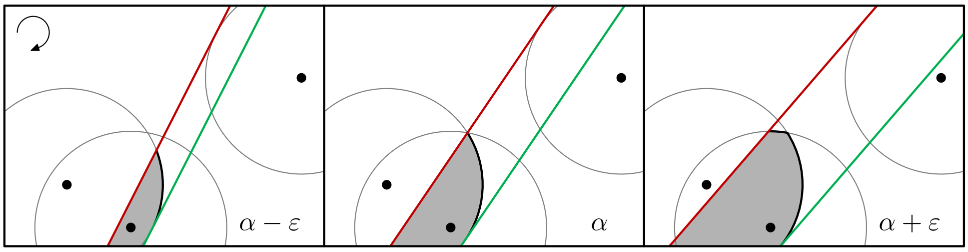

Figure 6 When τ_1/τ_2 hits a vertex of $\mathcal{A}(C_P)$, an arc may need to be added to S_1/S_2.

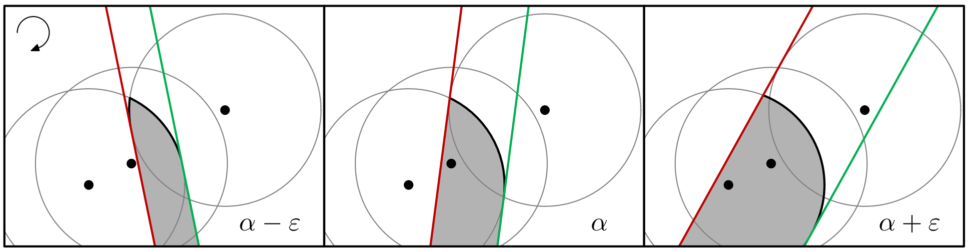

Figure 7 When τ_1/τ_2 hits an intersection of its defining circle, the composition of S_1/S_2 changes.

(4) τ_1 and τ_2 are the same line. When this event takes place, then τ_1 and τ_2 are the inner bi-tangent of their two respective defining circles. See Figure 8 for an example. We distinguish two different cases for this event: either there is a solution at $\alpha - \varepsilon$ and no solution at $\alpha + \varepsilon$, or vice versa.

If there was a solution at $\alpha - \varepsilon$ and there is none at $\alpha + \varepsilon$, we simply stop maintaining q_1q_2, S_1 and S_2 until there exists a solution again. As such, we remove all type-1 and type-3 certificates from the event queue and place a new type-4 certificate into the event queue that is violated at the next orientation where τ_1 and τ_2 are the same line.

▶ **Lemma 2.7.** *Throughout the full π rotation, τ_1 and τ_2 become the same line at most $O(h)$ times, and we can resolve each occurrence of such an event in $O(h)$ time.*

▶ **Theorem 2.8.** *Given a point set P consisting of n points and a radius r, we can find the shortest shape-representing line segment in $O(n \log h + h^2)$ time.*

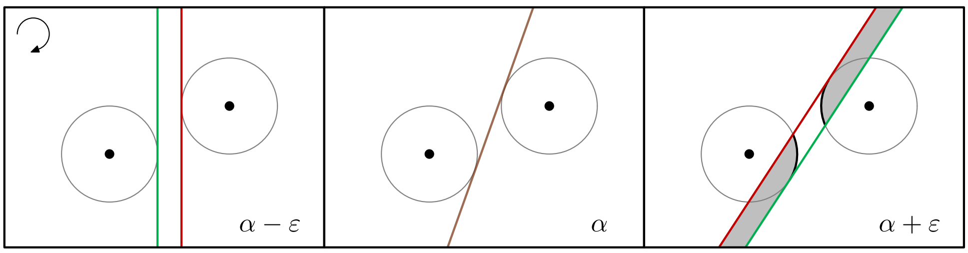

Figure 8 When τ_1 and τ_2 are the same line, they are an inner bi-tangent of their two defining circles.

3 Discussion

An obvious open question is whether the shortest shape-representing line segment can be computed in $O(n \log h)$. In the full version of this abstract, we show that we can actually compute the convex sequence in linear time, given the convex hull. The question is then whether the convex sequence can be traversed efficiently, without using the full circle arrangement $\mathcal{A}(C_P)$. We expect that this may be possible, yet, even this is not sufficient: Observe that in a regular k-gon with a diameter $2r + \varepsilon$, a solution appears/disappears $O(n)$ times. Then, a linear-time convex sequence construction is not sufficient. Furthermore, the circles contributing to a convex sequence may be as far as $n/2$ vertices apart on the convex hull, making amortization difficult. In the full version of this abstract we show that we can compute a $(1 + \varepsilon)$-approximation in $O(n \log h + h/\varepsilon)$ time by sampling orientations and applying the fixed orientation algorithm.

References

1 Timothy M. Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discret. Comput. Geom.*, 16(4):361–368, 1996.

2 Timothy M. Chan, Thomas C. van Dijk, Krzysztof Fleszar, Joachim Spoerhase, and Alexander Wolff. Stabbing rectangles by line segments–how decomposition reduces the shallow-cell complexity. In *29th International Symposium on Algorithms and Computation*, 2018.

3 Danny Z. Chen and Haitao Wang. Approximating points by a piecewise linear function. *Algorithmica*, 66(3):682–713, 2013.

4 Merce Claverol, Elena Khramtcova, Evanthia Papadopoulou, Maria Saumell, and Carlos Seara. Stabbing circles for sets of segments in the plane. *Algorithmica*, 80:849–884, 2018.

5 José Miguel Díaz-Báñez, Matias Korman, Pablo Pérez-Lantero, Alexander Pilz, Carlos Seara, and Rodrigo I Silveira. New results on stabbing segments with a polygon. *Computational Geometry*, 48(1):14–29, 2015.

6 Matt Duckham, Lars Kulik, Mike Worboys, and Antony Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern recognition*, 41(10):3224–3236, 2008.

7 Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983.

8 Herbert Edelsbrunner, Hermann A. Maurer, Franco P. Preparata, Arnold L. Rosenberg, Emo Welzl, and Derick Wood. Stabbing line segments. *BIT Numerical Mathematics*, 22:274–281, 1982.

9 Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.

10 S. Louis Hakimi and Edward F. Schmeichel. Fitting polygonal functions to a set of points in the plane. *CVGIP: Graphical Models and Image Processing*, 53(2):132–136, 1991.

11 John A. Hartigan and Manchek A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society.*, 28(1):100–108, 1979.

12 David G Kirkpatrick and Raimund Seidel. The ultimate planar convex hull algorithm? *SIAM journal on computing*, 15(1):287–299, 1986.

13 Konstantin Kobylkin. Stabbing line segments with disks: complexity and approximation algorithms. In *International Conference on Analysis of Images, Social Networks and Texts*, pages 356–367. Springer, 2017.

14 Lena Marie Schlipf. *Stabbing and Covering Geometric Objects in the Plane*. PhD thesis, FU Berlin, 2014.

15 Ariana Strandburg-Peshkin, Damien R. Farine, Margaret C. Crofoot, and Iain D. Couzin. Habitat and social factors shape individual decisions and emergent group structure during baboon collective movement. *eLife*, 6:e19505, 2016.

16 Godfried T. Toussaint. Solving geometric problems with the rotating calipers. In *IEEE Melecon*, volume 83, page A10, 1983.

17 Der Perng Wang. A new algorithm for fitting a rectilinear x-monotone curve to a set of points in the plane. *Pattern Recognition Letters*, 23(1-3):329–334, 2002.