Computational Geometry: Theory and Applications

Stabbers of line segments in the plane

Mercè Claverol [a,1], Delia Garijo [b,2], Clara I. Grima [b,2], Alberto Márquez [b,2], Carlos Seara [c,*,1]

[a] Dpt. de Matemàtica Aplicada IV, Universitat Politècnica de Catalunya, Spain
[b] Dpto. de Matemática Aplicada I, Universidad de Sevilla, Sevilla, Spain
[c] Dpt. de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Spain

ABSTRACT

The problem of computing a representation of the stabbing lines of a set S of segments in the plane was solved by Edelsbrunner et al. We provide efficient algorithms for the following problems: computing the stabbing wedges for S, finding a stabbing wedge for a set of parallel segments with equal length, and computing other stabbers for S such as a double-wedge and a zigzag. The time and space complexities of the algorithms depend on the number of combinatorially different extreme lines, critical lines, and the number of different slopes that appear in S.

Keywords:
Stabbers
Transversal
Line segments
Red-blue separability

1. Introduction

Let $S=\{s_1,\ldots,s_n\}$ be a set of n not necessarily disjoint line segments (or segments) in the plane. If p and q are the endpoints of a segment, for convenience, we require that $p\neq q$ and neither of them are at the infinity; consequently, lines, rays, and points are not considered to be segments, not even degenerate ones. In order to avoid tedious case analysis, we assume that the set of endpoints of the segments is a point set in the plane in general position, i.e., no three endpoints are collinear. Since we are interesting on stabbing the segments of S, we can assume that three segments can meet, or one segment can be contained into another, or more generally, two segments can have a (non-input) segment as their intersection (the two segments are reduced to the intersection segment).

A line is a *transversal* of (or *stabs*) S if it intersects each segment of S even when it meets the segment only at an endpoint, and even when it contains the segment. Edelsbrunner et al. [10] presented a $\Theta(n\log n)$ time and $O(n)$ space algorithm for solving the problem of constructing a representation of all transversal lines or stabbing lines of S. See Edelsbrunner [8] for an analysis of this problem from both a combinatorial and computational point of view. The lower bound from Edelsbrunner et al. [10] does not apply to the decision problem: *determining if there exists a line stabber for S*. Avis et al. [2] presented an $\Omega(n\log n)$ time lower bound in the fixed order algebraic decision tree model to determine the existence of a line stabber for S. For a set of n vertical segments, a stabbing line can be computed in $\Theta(n)$ time.

A stabbing line ℓ for S classifies the endpoints of the segments in two classes: endpoints above ℓ, say red points; and endpoints below ℓ, say blue points. The endpoint on ℓ is classified according to the other endpoint. Thus, we can see the problem of stabbing S as a problem of classifying the endpoints of the segments into disjoint monochromatic red and blue regions defined by the stabber, i.e., as a separability problem. Following this line of research, in this paper we deal with the

* Corresponding author.
E-mail addresses: merce@ma4.upc.edu (M. Claverol), dgarijo@us.es (D. Garijo), grima@us.es (C.I. Grima), almar@us.es (A. Márquez), carlos.seara@upc.edu (C. Seara).

[1] Supported by project MTM2009-07242, and Gen. Cat. DGR2009GR1040.
[2] Supported by project MTM2008-05866-C03-01.

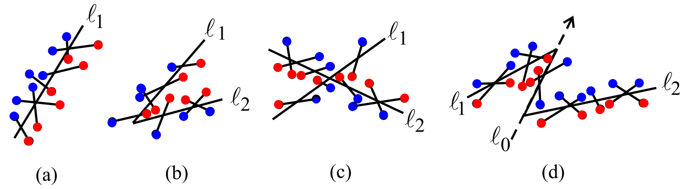

Fig. 1. Stabbers from (a) to (d): line, wedge, double-wedge, zigzag.

problem of finding stabbers for S satisfying the condition that there is no segment stabbed by more than one element of the stabber (Fig. 1). We call this condition the *separability condition*. So we look for stabbers for S such that we can assign red and blue colors to the endpoints of the segments and split the plane into disjoint monochromatic regions. Hurtado et al. [14] classified red and blue points with separators which are similar to our stabbers.

If there is no stabbing line for S it is natural to ask for a stabbing *wedge* (two rays with a common endpoint) because it can be a good approximation to a stabbing line if its aperture angle is close to π. Thus, we consider the problem of computing the stabbing wedges for S. As a particular problem, we study the problem of finding a stabbing wedge for a set of parallel segments with equal length. We also look for other stabbers for S such as: a double-wedge stabber formed by two intersecting lines, and a zigzag stabber formed by two non-intersecting rays and a segment joining the origins of the rays. Fig. 1 illustrates the kind of stabbers we will consider in this paper. The general goal is to design efficient algorithms for computing these stabbers for S depending on whether they satisfy the separability condition above, i.e., we ask for stabbers *with the separability condition* or stabbers *without the separability condition*. The time and space complexities of the algorithms are sensitive to some parameters of S, more concretely, they depend on the number of combinatorially different extreme lines, critical lines, and the number of different slopes that appear in S.

Outline of the paper. In Section 2 we study the problem of computing a representation of the set of *extreme lines* of S. The extreme lines are a key tool for solving the problem of computing the set of stabbing wedges for S in Section 3. The problem of finding a stabbing wedge for a set of parallel segments with equal length is considered in Section 4. Finally, in Section 5 we show efficient algorithms for the problem of computing a stabbing double-wedge for S and for the problem of finding a stabbing zigzag for S.[3]

Related works. Stabbing problems have since been widely investigated and arise in many diverse problems in computational geometry. Claverol [6] as a part of her PhD thesis initiated the study here developed. In this paper we improve the complexities she obtained and some other stabbing problems are also considered. Atallah and Bajaj [1] presented an algorithm for line stabbing simple objects in the plane, where a simple object is an object which has constant store description. Edelsbrunner, Guibas and Sharir [11] showed how to construct a representation of the line stabbers of convex polygons. O'Rourke [19] presented an algorithm for finding a stabbing line of vertical line segments. Goodrich and Snoeyink [12] presented a natural variant considering another type of stabbers different from the lines by solving the problem of computing a transversal convex polygon for a set of parallel segments. Bhattacharya et al. [4] worked on the problem of computing the shortest transversal segment for a set of lines in the plane and also for a set of convex polygons. Lyons et al. [15] studied the problem of computing the minimum perimeter convex polygon which stabs a set of isothetic line segments. Rappaport [20] considered the problem of computing a simple polygon with minimum perimeter which stabs or contains a set of line segments. Mukhopadhyay et al. [16–18] considered the problem of computing the minimum area convex polygon which stabs a set of parallel line segments.

2. Extreme lines

This section is devoted to the study of the set of extreme lines for S from a computational and from a combinatorial point of view, since we shall design algorithms for determining stabbers for S whose complexities depend on the computation of this set of lines. We also introduce some terminology that shall be used throughout this paper.

A line ℓ is an *extreme line* for S if ℓ stabs a subset of segments $S_1 \subseteq S$ with $S_1 \neq \emptyset$, and the remaining segments $S_2 = S \setminus S_1$ lie in only one of the open half-planes defined by ℓ. Otherwise, ℓ is a *non-extreme line* for S. We denote by ℓ^+ (ℓ^-) the left (right) open half-plane defined by a directed line ℓ. If the segments of S_2 lie in ℓ^- (ℓ^+), we say that ℓ is a *left-extreme* (*right-extreme*) line for S. Thus, ℓ is an extreme line for S if and only if ℓ is either left-extreme or right-extreme for S. There are two types of non-extreme lines ℓ for S: either ℓ does not intersect the convex hull of S, denoted by $CH(S)$, or the line ℓ intersects $CH(S)$, and both half-planes ℓ^+ and ℓ^- contain at least one segment of S.

2.1. Properties of extreme lines

Next we show some properties about extreme lines. All omitted proofs are straightforward.

[3] A preliminary version of this paper was presented in EuroCG'09 [7].

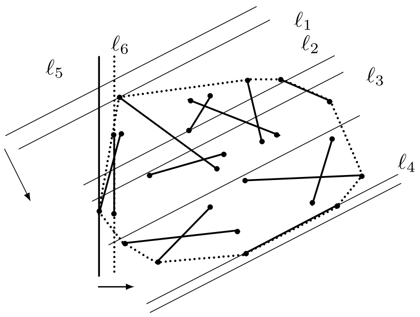

Fig. 2. Extreme lines and non-extreme lines with the same slope.

Lemma 1. *There always exist extreme lines for S with a given slope. Sweeping with a line of a given slope, the two types of non-extreme lines for S can appear. An endpoint of a segment or the segment itself determines the change from extreme to non-extreme line.*

Fig. 2 illustrates an instance of extreme lines and non-extreme lines with the same slope just sweeping with a line parallel to ℓ_1 or to ℓ_5.

Lemma 2. *Every extreme line for S intersects at least one segment of S with an endpoint in $CH(S)$, but the converse is not true. Moreover, for every segment $s \in S$ with at least one endpoint in $CH(S)$ there exists an extreme line that only intersects s.*

Proof. Suppose on the contrary that ℓ is an extreme line for S which intersects a subset of segments of S, but none of the endpoints of those segments are located in $CH(S)$. Then the segments stabbed by ℓ belong to the interior of $CH(S)$, which implies that there is at least one segment of S located in each of the half-planes, ℓ^+ and ℓ^-, with an endpoint in $CH(S)$. Hence ℓ is a non-extreme line, which leads to a contradiction. The line ℓ_3 in Fig. 2 proves that the converse is not true.

Consider a segment $s \in S$ with an endpoint q in $CH(S)$. To prove that there exists an extreme line that only intersects s, it suffices to consider the supporting line of $CH(S)$ passing through q. $\square$

Lemma 3. *A line ℓ is a stabbing line for S if and only if ℓ is both a left-extreme and a right-extreme line for S.*

A key consequence of Lemma 3 is the following. Suppose that S is not stabbed by a line, and consider a directed line ℓ with any orientation. Sweeping S with ℓ, it is not possible to go from a left-extreme to a right-extreme line. Necessarily, a non-extreme line for S is obtained between the two types of lines.

Definition 1. An endpoint q of a segment of S is a *critical point* of S for a given orientation m, if any sweeping directed line ℓ with orientation m is either left-extreme or right-extreme and becomes non-extreme once it crosses q.

Note that Lemma 1 implies that there is a critical point for each orientation. Extreme lines ℓ_2 and ℓ_4 in Fig. 2 define critical points.

Definition 2. An extreme line ℓ is a *critical line* for S if ℓ passes through a critical point.

Critical lines and critical points play an important role in the design of our algorithms. The following rotating process allow us to visualize the sequences of critical points and critical lines for S (Fig. 3). Consider a directed critical line ℓ which rotates clockwise anchored at critical points of S. The first critical point corresponds to the vertical-upwards directed critical line. When ℓ bumps a new critical point, then ℓ changes the anchored point to the new critical point, and both critical points are consecutive in the sequence of critical points. Denote by q the critical point in which ℓ is anchored at some moment of the process. Let ℓ_q^r and ℓ_q^l be the right and left ray, respectively, from q contained in ℓ according to the orientation of ℓ. The following result is straightforward.

Lemma 4. *The directed critical line ℓ only can become non-extreme line for S after the ray ℓ_q^r passes through an endpoint p of a segment $s \in S$ strictly contained in ℓ^+. In this case, p becomes the next critical point for S. If p is the endpoint of the segment $\overline{pq} \in S$ and the ray ℓ_q^l passes through p, then p becomes the next critical point for S.*

Consider the sequence F_1 of critical points obtained when a directed critical line follows the rotating process above-described, starting from a vertical-upwards directed critical line to a vertical-downwards directed critical line. A sequence F_2 is obtained by interchanging the roles of vertical-upwards and vertical-downwards. Denote by D_1 and D_2 the respective

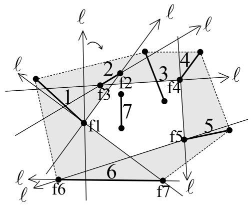

Fig. 3. Critical points (f_i) and directed critical lines (ℓ) of S.

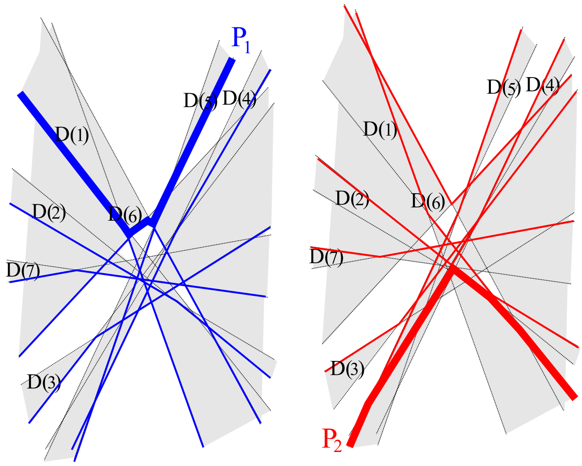

Fig. 4. (Left) Upper envelope of the lower rays, (right) lower envelope of the upper rays of the segment set in Fig. 3.

sequences of directed critical lines passing through two consecutive critical points. In order to compute D_1 and D_2 we first recall the following tool.

A standard geometric tool which will be used throughout this work is *duality* [10]: the geometric transform denoted by $\mathcal{D}$ which maps a point into a non-vertical line and vice versa. Thus, the dual transform $\mathcal{D}$ maps a point $p = (a, b)$ to the non-vertical line $\mathcal{D}(p) : y = 2ax - b$ and vice versa, that is, it maps a non-vertical line $\ell : y = cx + d$ to the point $\mathcal{D}(\ell) = (c/2, -d)$. A segment $s_i \in S$ is determined by its endpoints. The endpoints are transformed by $\mathcal{D}$ into two lines. If s_i is not vertical, $\mathcal{D}(s_i)$ is a double-wedge which does not contain a vertical line in its interior. Thus, the double-wedge is formed by two upper rays and two lower rays. If s_i is a vertical segment, $\mathcal{D}(s_i)$ is a *strip*. The set of endpoints of the segments in S is transformed by $\mathcal{D}$ into an arrangement of $2n$ lines denoted by $\mathcal{A}(S)$.

The transform $\mathcal{D}$ satisfies the following properties: (i) $\mathcal{D}$ maintains the relative position (above/below) of points and lines; (ii) a line ℓ intersects a segment s_i if and only if the point $\mathcal{D}(\ell)$ lies in the double-wedge $\mathcal{D}(s_i)$; (iii) the stabbing lines of S stand in one-to-one correspondence with the intersection points of their double-wedges, i.e., $\bigcap_{s_i \in S} \mathcal{D}(s_i)$.

Rappaport [20] used duality to design an $O(n \log n)$ time algorithm for computing D_1 and D_2, although his goal was to use these sequences for a different problem. His algorithm is essentially based on the following lemma for which it is assumed, without loss of generality, that S contains no vertical segments. Thus for every $s \in S$, the double-wedge $\mathcal{D}(s)$ determines two upper rays and two lower rays. This result is also the key tool used by Edelsbrunner et al. [10] to design an algorithm for computing a representation of the stabbing lines of S.

Lemma 5. *(See [10,20].) The sequence D_1 (D_2) corresponds in $\mathcal{A}(S)$ to the sequence of vertices of the lower (upper) envelope of the set of upper (lower) rays of $\mathcal{D}(s)$. Both sequences D_1 and D_2 have linear complexity.*

The lower (upper) envelope of the upper (lower) rays of $\mathcal{D}(s)$, forms a (not necessarily convex) x-monotone polygonal chain P_1 (P_2) with a linear number of edges (see Fig. 4 for an example). Using Lemma 5, Rappaport [20] and Edelsbrunner et al. [10] presented a divide and conquer algorithm to obtain the following result.

Theorem 1. *(See [10,20].) The sequences D_1 and D_2, and the polygonal chains P_1 and P_2 can be computed in $O(n \log n)$ time and $O(n)$ space.*

Since a stabbing line for S is both left-extreme and right-extreme, Edelsbrunner et al. [10] computed the intersection of both polygonal chains P_1 and P_2 to obtain the cells in $\mathcal{A}(S)$ which define the locus of stabbing lines for S. Thus if S is not stabbed by a line, then the polygonal chains P_1 and P_2 do not intersect.

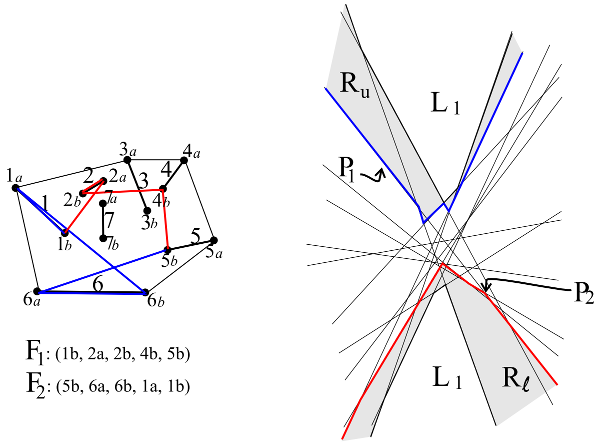

Fig. 5. (Left) Segment set S and sequences (F_1 and F_2) of critical points, (right) the shaded regions are the locus L of extreme lines for S. Both endpoints of segments 1 and 2 are critical points. Segment 6 is in $CH(S)$ and so its endpoints become critical points. The polygonal chain P_1 is not convex since part of the double-wedge containing segments 1 and 6 belongs to P_1. Analogously for P_2 due to segment 2.

2.2. Computing a representation of extreme lines

Two lines, ℓ_1 and ℓ_2, are said to be *combinatorially different* with respect to S if either the subsets S_1 and S_2 of segments stabbed by ℓ_1 and ℓ_2, respectively, are different; or if $S_1 = S_2$ then the subsets of endpoints of segments above (left of) ℓ_1 and ℓ_2 are different.

Denote by h_S and g_S the numbers of combinatorially different extreme lines and non-extreme-lines, respectively, for S. Observe that two extreme (or non-extreme) lines for S, ℓ_1 and ℓ_2, are combinatorially different with respect to S if and only if the points $\mathcal{D}(\ell_1)$ and $\mathcal{D}(\ell_2)$ lie in different cells of $\mathcal{A}(S)$. Thus, we shall present an algorithm which computes a representation of all the extreme lines for S in $\mathcal{A}(S)$. This representation is the locus L of points in the plane (union of cells in $\mathcal{A}(S)$) which correspond in the primal to extreme lines for S. The number of cells in L is exactly equal to h_S. Our algorithm shall compute the boundary of L and the cells inside L. Because there is no stabbing line for S, the locus L is formed by two disjoint regions. We call the top and bottom boundaries of the union of cells in L the *exterior* boundary of L, and the two boundaries (polygonal chains) which separate the two disjoint regions of the union of cells in L, the *interior* boundary of L (see Fig. 5).

Let L_1 and L_2 be the sequences of edges of the exterior boundary and the interior boundary of L, respectively. By Lemmas 1, 2, 3, and 4 we have the following result.

Lemma 6. *The dual of the directed supporting lines of $CH(S)$ forms the exterior boundary L_1. Analogously, the dual of the directed critical lines of S forms the interior boundary L_2.*

The sequence L_1 is formed by the upper and lower envelopes of $\mathcal{A}(S)$, the edges of L_2 come from the dual of the sequence of critical points of S, and L_2 is the boundary of the set of cells of non-extreme lines for S. Moreover, since L_2 is formed by P_1 and P_2 the following result is a straightforward consequence of Lemmas 5 and 6 and Theorem 1.

Lemma 7. *The boundary L_2 of L can be computed in $O(n \log n)$ time and $O(n)$ space.*

If there is no stabbing line for S, i.e., P_1 and P_2 do not intersect, then L is the union of two disjoint regions in $\mathcal{A}(S)$: the upper region R_u delimited by P_1, and the lower region R_l delimited by P_2 (Fig. 5). If P_1 and P_2 intersect, the representation in $\mathcal{A}(S)$ of the set of stabbing lines for S is $R_u \cap R_l$. Next we show how to construct R_u and R_l.

Construction of R_u and R_l. (a) Properties: Since R_u is bounded by P_1 (the upper envelope of the lower rays), the apices of the double-wedges (dual to segments of S) are all below or on P_1 (analogously they are all above or on P_2 because P_2 is the lower envelope of the upper rays). For each ray of a double-wedge, we record each side with $+$ or $-$ depending on which side of the double-wedge lies on (later we will used it to determine the segments of S stabbed by extreme lines). Recall that P_1 and P_2 are x-monotone polygonal chains, starting and ending with rays. We focus on the construction of R_u (the construction of R_l is analogous). Let m_1 be the complexity of P_1 which is linear in n.

(b) Preprocess: The $2n$ upper rays of the double-wedges contribute to the cells of R_u if and only if they intersect P_1. The origin of each upper ray is below or on P_1. We form from P_1 two bounded simple polygons, by intersecting it with a large bounding box enclosing all of its vertices; one polygon lies below P_1 and one above. Now, we use the ray-shooting algorithm from Hershberger and Suri [13], which can preprocess the two polygons in $O(m_1)$ time to support $O(\log m_1)$-time ray-shooting queries. Now, for every upper ray, we repeatedly use the data structure to identify all of its intersections

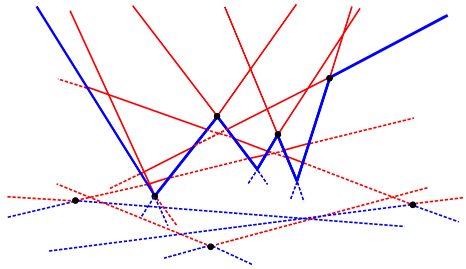

Fig. 6. Constructing R_u.

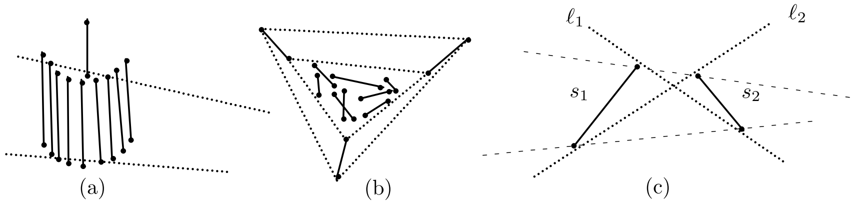

Fig. 7. (a) $h_S = O(n^2)$, (b) $h_S = 6$, (c) non-intersecting segments.

with P_1, at the cost of $O(\log m_1)$ per intersection. If the upper ray has more that one intersection point we compute the segments (with endpoints on P_1) contained *inside* R_u formed by consecutive intersection points. All those segments will contribute to the arrangement of cells in R_u. Let m_2 (m_3) be the number of rays (segments) obtained in this way. These sets of rays and segments can be computed in $O((m_2 + m_3)\log m_1)$ time. Note that m_1 and m_2 are linear in n, but m_3 can be quadratic since an upper ray can intersects P_1 many times (see Fig. 6).

(c) Computation: The arrangement of cells in R_u is *simple*, that is, no three lines pass through the same point (which is true by the assumption that the endpoints of the segments in S are in general position) and no two lines are parallel (assuming that no two endpoints of segments in S share the same vertical line: in $O(n\log n)$ time we can rotate the coordinate system to achieve this). From the set of $m_1 + m_2 + m_3$ rays and segments we can compute the arrangement of the h_S cells in R_u using topological sweep [9] (see also Balaban's algorithm [3]) in $O((m_1 + m_2 + m_3)\log(m_1 + m_2 + m_3) + h_S)$ time and $O(h_S + n)$ space. The time complexity is slightly better than $O(h_S \log h_S)$ as h_S is between $m_1 + m_2 + m_3$ and $\Theta((m_1 + m_2 + m_3)^2)$. The cells in R_l are constructed similarly. We now describe the algorithm which computes a representation of all the extreme lines for S.

Algorithm 1. Representation-extreme-lines
Input: Set S of n segments in the plane.
Output: L_1, L_2, and the set of cells in L.

1. In $O(n\log n)$ time, compute L_1 as the upper and lower envelopes of $\mathcal{A}(S)$.
2. Use Theorem 1 to compute L_2 in $O(n\log n)$ time. L_2 is formed by P_1 and P_2. If P_1 and P_2 do not intersect, then L is formed by the regions R_u and R_l. The complexities of P_1, P_2, and L_2 are $O(n)$.
3. In $O(h_S \log h_S)$ time and $O(h_S)$ space, compute the set of h_S cells in R_u and R_l which forms L, i.e., the set of all the combinatorially different extreme lines for S.

Theorem 2. *A representation of the combinatorially different extreme lines for S can be computed in $O(h_S \log h_S + n\log n)$ time and $O(h_S + n)$ space.*

Notice that when h_S is constant (as in Fig. 7(b)) or linear, the complexities of Algorithm 1 are $O(n\log n)$ time and $O(n)$ space.

2.3. Bounds for h_S and g_S

Clearly, $|CH(S)| \leqslant h_S$ and the number h_S is at most $O(n^2)$. A nice instance, showing that the upper bound is tight, is the segment set proposed by Claverol [6] (Fig. 7(a)). On the other hand, there are at least two combinatorially different non-extreme lines for S, the line ℓ which does not intersect $CH(S)$ and $CH(S)$ is either above or below ℓ. Thus, $2 \leqslant g_S$. Consider now the intersection graph $G = (V, E)$ whose vertex set is $V = S$, and two segments s_1 and s_2 are adjacent if and only if $s_1 \cap s_2 \neq \emptyset$. Let $G^c = (V, E^c)$ be the complementary graph of G. If S is formed by pairwise non-intersecting segments,

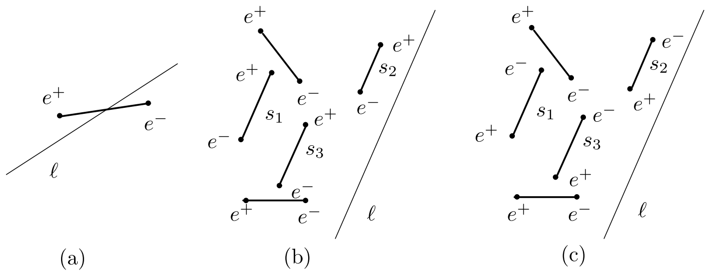

Fig. 8. (a) Classifying the endpoints of a segment with respect to ℓ, (b) and (c) the two possible classifications of a given segment set where ℓ is parallel to segments s_1, s_2 and s_3.

then every pair of segments gives rise to at least one non-extreme line (Fig. 7(c)) and hence $g_S \leqslant |E^c| = \binom{n}{2} = O(n^2)$. Thus, $2 \leqslant g_S \leqslant O(n^2)$ and the bounds are tight.

2.4. Computing a representation of critical lines

As mentioned earlier, among extreme lines, critical lines play an important role in our algorithms. These lines are points in the cells of $\mathcal{A}(S)$, called *critical cells*, which have one edge in either P_1 or P_2. Our next aim is to compute a representation of the combinatorially different critical lines.

All the critical cells in R_u have an edge in P_1 (analogously, critical cells in R_l have an edge in P_2), and two of them are adjacent if they share a segment (or ray) with endpoint in P_1. Since P_1 is an x-monotone polygonal chain, then we can walk on P_1 from one critical cell to an adjacent critical cell with the unique change of the shared edge according to the labels of the edge: crossing from $+$ to $-$ or from $-$ to $+$. To do that we need to compute the intersection points of the upper rays above with P_1 and to sort them by increasing x-coordinate (because P_1 is x-monotone). Recall from above that there are $O(m_2+m_3)$ intersection points, and they can be sorted in $O((m_2+m_3)\log(m_2+m_3))$ time and $O(m_2+m_3)$ space (see Fig. 6). As initial stage we need to compute the leftmost critical cell, or in other words, the set of endpoints on the left of the vertical-upwards directed critical line which is anchored in the first critical point. This can be done in $O(n\log n)$ time. Now, walking from left to right along P_1 we visit all critical cells (the dual graph of the critical cells is a tree and we can cross a segment at most twice). Then computing a point from all the combinatorially different critical cells can be done in $O((m_2+m_3)\log(m_2+m_3)+n\log n)$ time and $O((m_2+m_3)+n)$ space. Note that $m_2+m_3 < c_S$ because a ray or a segment contributes to at least one critical cell. Thus, $O((m_2+m_3)\log(m_2+m_3))$ is slightly better than $O(c_S \log c_S)$, and a representation of all the combinatorially different critical lines can be computed in $O(c_S\log c_S+n\log n)$ time and $O(c_S+n)$ space.

Theorem 3. *A representation of the combinatorially different critical lines for S can be computed in $O(c_S\log c_S+n\log n)$ time and $O(c_S+n)$ space.*

Note that c_S can be quadratic, one can constructs examples where many upper rays intersect P_1 many times (see Fig. 6).

3. Stabbing wedge

Let $W=\{\ell_1,\ell_2\}$ be a stabbing wedge for S, where ℓ_1 and ℓ_2 are the two rays of W. The line containing ℓ_i is denoted by ℓ'_i for $i=1,2$. The half-planes defined by ℓ'_i are written as $\ell_i'^{+}$ and $\ell_i'^{-}$. We can assume that S is not stabbed by a line what implies that ℓ_1 stabs a subset of segments $S_1 \subsetneq S$ with $S_1 \neq \emptyset$, and the set $S_2 = S\setminus S_1$ is stabbed by ℓ_2. The aim of this section is to study the problem of computing stabbing wedges for S with or without the *separability condition*.

Denote by m_i the slope of (the line containing) the segment $s_i \in S$, and by k_S the number of different slopes of the segments of S. Given a line ℓ and a segment s, we can classify the endpoints of s with respect to ℓ whenever ℓ and the line containing s are not parallel. It suffices to do a parallel sweeping by a line ℓ until it crosses s, leaving one endpoint in ℓ^+, and the other one in ℓ^-. These endpoints are denoted by e^+ and e^-, respectively (see Fig. 8(a)). If ℓ and the line containing s are parallel, the endpoints of s cannot be classified. Indeed, there are two possible classifications of the endpoints of those segments: the endpoint with bigger y-coordinate of each segment is classified into e^+ and the other one into e^-, or vice versa (Figs. 8(b) and 8(c)).

Lemma 8. *Any parallel sweeping by a line ℓ with a fixed slope gives rise to at most* 2 *classifications of the endpoints of the segments of S.*

Proof. Consider a line ℓ with slope m. If $m \neq m_i$ for $i=1,\dots,k_S$, then the endpoints of each segment can be classified into e^+ and e^- giving rise to exactly one classification. Assume now that $m=m_i$ and that there are some segments of S with slope m_i. Since these segments are parallel to ℓ, there are two possible classifications of the endpoints of those segments

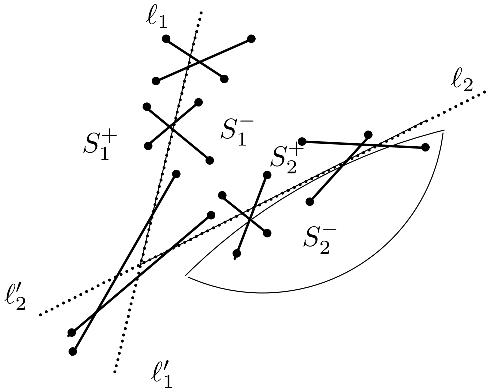

Fig. 9. A stabbing wedge for S and the point sets S_1^+, S_1^-, S_2^+, and S_2^-.

as we said above. For horizontal segments, consider the endpoint with bigger x-coordinate in order to distinguish the two possible assignments. □

3.1. Stabbing wedges with the separability condition

Let $W = \{\ell_1, \ell_2\}$ be a stabbing wedge for S with the separability condition, i.e., no segment can be stabbed by both rays. Let S_i^+ (S_i^-) for $i = 1, 2$, be the set of endpoints of the segments of S_i classified as e^+ (e^-) with respect to ℓ'_i. Thus, S_1^- and S_2^+ are contained inside the wedge W, and S_1^+ and S_2^- are located outside the wedge W (Fig. 9). These assignments $\{+, -\}$ depend on the relative position of the lines ℓ'_1 and ℓ'_2. We shall mainly concentrate on a general wedge as the one shown in Fig. 9. The following lemma is straightforward.

Lemma 9. *If $W = \{\ell_1, \ell_2\}$ is a stabbing wedge for S, then ℓ'_1 and ℓ'_2 are extreme lines for S, and at least half of the segments of S are stabbed by either ℓ_1 or ℓ_2.*

The next two results use the following notation. The line ℓ'_1 is an extreme line for S. The set $S_1 \subsetneq S$ with $S_1 \neq \emptyset$, is the segment subset stabbed by ℓ'_1, and $S_2 = S \setminus S_1$. Denote by S_1^+ and S_1^- the classification of the endpoints of the segments of S_1 given by ℓ'_1 according to Lemma 8. The sets S_2^+ and S_2^- are the classification of the endpoints of the segments of S_2 obtained by sweeping with a line, say ℓ'_2, with fixed slope m according to Lemma 8.

Lemma 10. *There exists a stabbing wedge $W = \{\ell_1, \ell_2\}$ for S with ℓ_1 contained in ℓ'_1 if and only if S_2^- is line separable from $S_1^- \cup S_2^+$ by the line ℓ'_2.*

Proof. Given a stabbing wedge $W = \{\ell_1, \ell_2\}$ for S with ℓ_1 stabbing S_1 and ℓ_2 stabbing $S_2 = S \setminus S_1$, the line ℓ'_2 separates S_2^- from $S_1^- \cup S_2^+$ (see Fig. 9). Denote by m the slope of the sweeping line ℓ'_2 according to Lemma 8. Since ℓ'_1 is an extreme line, we can assume that S_2 is contained in ${\ell'_1}^-$. Observe that $S_1^- \cup S_2^+$ is contained in both half-planes ${\ell'_2}^+$ and ${\ell'_1}^-$. Thus, the segments of S are all stabbed by ℓ'_1 and by ℓ'_2. The intersection point p of ℓ'_1 and ℓ'_2 is the apex of a stabbing wedge $W = \{\ell_1, \ell_2\}$ for S such that ℓ'_i contains the half-line ℓ_i for $i = 1, 2$. Moreover, since $S_1^- \cup S_2^+ \subset {\ell'_2}^+ \cap {\ell'_1}^-$ then no segment of S is stabbed by the two rays ℓ_1 and ℓ_2. □

Lemma 11. *The locus of apices of stabbing wedges W for S with respect to the classification of endpoints into S_i^+ and S_i^- for $i = 1, 2$, can be computed in $O(n \log n)$ time. The four interior supporting lines between $CH(S_1^+)$ and $CH(S_1^- \cup S_2^+)$, and between $CH(S_1^- \cup S_2^+)$ and $CH(S_2^-)$ define at most two (possible unbounded and degenerate) convex quadrilateral Q whose interior points are apices of stabbing wedges for S.*

Proof. Clearly $CH(S_1^- \cup S_2^+)$ is contained inside W. Thus, W separates the endpoints in $CH(S_1^- \cup S_2^+)$ from the remaining endpoints. Hurtado et al. [14] show how to compute the locus of apices of all these separating wedges W in $O(n \log n)$ time. □

Definition 3. Two stabbing wedges W_1 and W_2 for S are *combinatorially different* if the sets of endpoints of segments inside the wedges W_1 and W_2 are different.

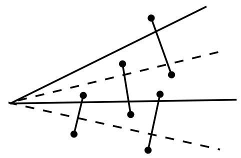

Fig. 10. Two combinatorially different stabbing wedges with the same apex.

It is easy to construct a segment set having two combinatorially different stabbing wedges with the same apex (Fig. 10). Thus, there is not a one-to-one correspondence between apices and combinatorially different stabbing wedges.

Lemmas 9, 10, and 11 are the key tools to design the next algorithm for computing the set $\mathcal{W}$ of combinatorially different stabbing wedges for S and a set $\mathcal{Q}_{\mathcal{W}}$ of convex quadrilaterals Q which are a representation of those stabbing wedges.

Algorithm 2. STABBING-WEDGES-WITH-SEPARABILITY-CONDITION
Input: Set S of n segments in the plane.
Output: $\mathcal{W}$ and $\mathcal{Q}_{\mathcal{W}}$.

1. *Preprocess*:
   (a) In $O(n \log k_S)$ time, sort the k_S different slopes of the segments by increasing angular order. Denote them by $m_1, \ldots, m_{k_S}$.
   (b) Use ALGORITHM 1 to compute the set of h_S cells of L in $O(h_S \log h_S + n \log n)$ time and $O(h_S + n)$ space. In $O(h_S)$ time compute the dual graph G of L and do a traversal of G obtaining a tree. Following the tree-traversal (visiting each edge of the tree at most twice) we can get a sequence of $h'_S = O(h_S)$ of nodes of the tree, i.e., a sequence $C = (C_1, \ldots, C_{h'_S})$ of adjacent cells of L: a sequence for the cells in R_u, and a sequence for the cells in R_l.
   (c) Select a point $p_{(1,1)} \in C_1$. Let $\mathcal{D}(p_{(1,1)}) := \ell'_{1,(1,1)}$. Compute the set of segments stabbed by $\ell'_{1,(1,1)}$, written as $S_{1,(1,1)}$, and $S_{2,(1,1)} = S \setminus S_{1,(1,1)}$. Classify the endpoints of the segments of $S_{1,(1,1)}$ with respect to $\ell'_{1,(1,1)}$ obtaining the sets $S^+_{1,(1,1)}$ and $S^-_{1,(1,1)}$. Compute $CH(S^+_{1,(1,1)})$ and $CH(S^-_{1,(1,1)})$. With the slope m_1 determine the classification of the endpoints of the segments in $S_{2,(1,1)}$, i.e., at most two pairs $(S^+_{2,(1,1)}, S^-_{2,(1,1)})$. Compute $CH(S^+_{2,(1,1)})$ and $CH(S^-_{2,(1,1)})$.
2. **For** $i = 1, \ldots, k_S$ **do**
   For $j = 1, \ldots, h'_S$ if i is odd and for $j = h'_S, \ldots, 1$ if i is even **do**
   (a) If $i = j = 1$, use the data computed in Step 1c and go to Step 2d.
   (b) Select a point $p_{(i,j)} \in C_j$. Let $\mathcal{D}(p_{(i,j)}) := \ell'_{1,(i,j)}$. The difference between the segment sets stabbed by $\ell'_{1,(i,j-1)}$ and $\ell'_{1,(i,j)}$ is at most one segment s since the cells C_{j-1} and C_j are adjacent. This segment s can be stabbed by $\ell'_{1,(i,j-1)}$ and not by $\ell'_{1,(i,j)}$ or vice versa.
   (c) From s and the sets $S^+_{1,(i,j-1)}$, $S^-_{1,(i,j-1)}$, $S^+_{2,(i,j-1)}$, and $S^-_{2,(i,j-1)}$, in constant time update: (1) $S_{1,(i,j)}$ is the set of segments stabbed by $\ell'_{1,(i,j)}$ and $S_{2,(i,j)} = S \setminus S_{1,(i,j)}$; (2) the sets of endpoints $S^+_{1,(i,j)}$, $S^-_{1,(i,j)}$ classified with respect to $\ell'_{1,(i,j)}$; and (3) the (at most two) pairs of sets of endpoints $(S^+_{2,(i,j)}, S^-_{2,(i,j)})$ classified with respect the slope m_i. For each pair proceed as follows.
   (d) In $O(\log n)$ time, update $CH(S^+_{1,(i,j)})$, $CH(S^-_{1,(i,j)} \cup S^+_{2,(i,j)})$, and $CH(S^-_{2,(i,j)})$ and check whether $CH(S^-_{1,(i,j)} \cup S^+_{2,(i,j)})$ and $CH(S^-_{2,(i,j)})$ are line separable by a line, say $\ell'_{2,(i,j)}$. If so, by Lemma 10, there exists a stabbing wedge $W_{(i,j)}$ for S with apex in the intersection point of $\ell'_{1,(i,j)}$ and $\ell'_{2,(i,j)}$. Compute this wedge $W_{(i,j)} = \{\ell_{1,(i,j)}, \ell_{2,(i,j)}\}$. Add $W_{(i,j)}$ to $\mathcal{W}$.
   Compute the quadrilateral $Q_{(i,j)}$ defined by the interior supported lines between $CH(S^+_{1,(i,j)})$ and $CH(S^-_{1,(i,j)} \cup S^+_{2,(i,j)})$ and the interior supported lines between $CH(S^-_{1,(i,j)} \cup S^+_{2,(i,j)})$ and $CH(S^-_{2,(i,j)})$. Add $Q_{(i,j)}$ to $\mathcal{Q}_{\mathcal{W}}$.

Theorem 4. *The set $\mathcal{W}$ of combinatorially different stabbing wedges for S with the separability condition and the set $\mathcal{Q}_{\mathcal{W}}$ can be computed in $O(h_S k_S \log n + n \log n)$ time and $O(h_S + n)$ space.*

Proof. Step 1 of ALGORITHM 2 can be done in $O(h_S \log h_S + n \log n)$ time and $O(h_S + n)$ space since its complexity is dominated by the complexity of ALGORITHM 1. The time complexity of Step 2 is $O(k_S h_S \log n)$ since convex hulls can be updated by insertions/deletions in $O(\log n)$ time [5]. The total cost is $O(h_S k_S \log n + n \log n)$ time and $O(h_S + n)$ space. There are at most h_S combinatorially different stabbing wedges for S since each one can be defined by an extreme line. □

Corollary 1. *The set $\mathcal{W}$ of combinatorially different stabbing wedges for a set of n parallel segments with the separability condition and the set $\mathcal{Q}_{\mathcal{W}}$ can be computed in $O(h_S \log n + n \log n)$ time and $O(h_S + n)$ space.*

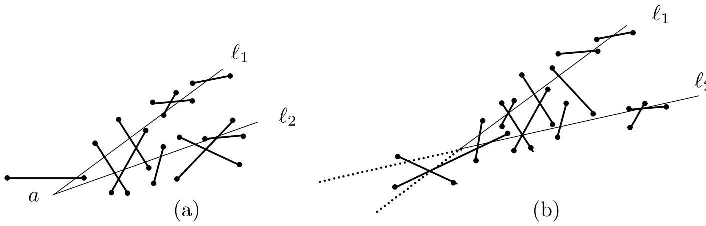

Fig. 11. (a) Stabbing wedge without the separability condition, (b) non-stabbing wedge.

Remarks. Fig. 7(a) shows that the number of combinatorially different stabbing wedges for a set of (parallel) segments can be quadratic. To decide whether S is stabbed by two parallel lines with the separability condition can be done using Algorithm 2 and checking if at least one of the quadrilaterals is unbounded. The stabbing wedge with maximum and minimum aperture angle can also be computed by checking in constant time the aperture angle of the stabbing wedges with apices in the vertices of the quadrilaterals in $\mathcal{Q}_{\mathcal{W}}$ and maintaining the maximum and minimum aperture angle.

3.2. Stabbing wedges without the separability condition

Let $W = \{\ell_1, \ell_2\}$ be a stabbing wedge for S without the separability condition. The same terminology than in the previous subsection is used for ℓ'_i, S_1, $S_2 = S \setminus S_1$, S_i^+ and S_i^- for $i = 1, 2$. We start by proving some fundamental properties of this type of wedge.

Lemma 12. *If S is stabbed by a wedge $W = \{\ell_1, \ell_2\}$ without the separability condition, then we can create a combinatorially equivalent stabbing wedge W' such that the line supporting one of the rays of W' becomes a critical line for S and the line supporting the other ray of W' becomes one of the two interior supporting lines of $CH(S_2^+)$ and $CH(S_2^-)$.*

Proof. The ray ℓ_1 can be rotated anchored at the apex of W decreasing the aperture angle of W until ℓ_1 passes through a critical point for S before reaching the ray ℓ_2, since otherwise there exists a stabbing line for S. Thus, the new ℓ'_1 becomes critical line for S (Fig. 11(a)). Note that a segment might be stabbed by the two rays. Moreover, the line ℓ'_2 can be moved decreasing the aperture angle of W and maintaining the apex on ℓ'_1, until the new ℓ'_2 becomes an interior supporting line of $CH(S_2^+)$ and $CH(S_2^-)$. $\square$

By Lemma 12, we can conclude that the candidates to be the ray ℓ_1 of W are critical lines. Moreover the ray ℓ_2 is determined by computing the interior supporting lines of $CH(S_2^+)$ and $CH(S_2^-)$. Notice that Lemma 10 does not hold for stabbing wedges without the separability condition. Thus, we have designed a process to determine the apex of the possible stabbing wedge. Once a wedge W has been computed, we have to check whether W with apex a stabs all the segments in S_1 (Fig. 11(b)). An algorithm similar to Algorithm 2 with some changes can be used to compute a set of stabbing wedges for S violating the separability condition. The changes are the following:

Step 1(b). Compute a representation of the set of c_S critical cells in $O(c_S \log c_S + n \log n)$ time and $O(c_S + n)$ space and compute a sequence $C = (C_1, \ldots, C_{c'_S})$ of $c'_S = O(c_S)$ adjacent critical cells: a sequence of the cells in R_u and a sequence of the cells in R_l.

Step 2. Use c'_S instead h'_S and write Step 2(d) as follows: In $O(\log n)$ time, compute the interior supporting lines between $CH(S^+_{2,(i,j)})$ and $CH(S^-_{2,(i,j)})$, assign $\ell'_{2,(i,j)}$ to each of these lines (two possibilities), let $a_{(i,j)}$ be the intersection point of $\ell'_{1,(i,j)}$ and $\ell'_{2,(i,j)}$, let $W_{(i,j)}$ be the corresponding wedge with apex in $a_{(i,j)}$. In $O(n)$ time check whether $W_{(i,j)}$ stabs all the segments in $S_{1,(i,j)}$, and if it is so add $W_{(i,j)}$ to $\mathcal{W}$.

Theorem 5. *A set $\mathcal{W}$ of stabbing wedges for S without the separability condition can be computed in $O(c_S k_S n + n \log n)$ time and $O(c_S + n)$ space.*

Proof. Step 1 can be done in $O(c_S \log c_S + n \log n)$ time and $O(c_S + n)$ space. Because Step 2(d) uses $O(n)$ time, Step 2 can be done in $O(c_S k_S n)$ time. $\square$

Corollary 2. *A set $\mathcal{W}$ of stabbing wedges for a set of n parallel segments without the separability condition can be computed in $O(c_S n + n \log n)$ time and $O(c_S + n)$ space.*

3.3. Lower bound for stabbing wedges

Now, we show an $\Omega(n \log n)$ lower bound for the decision problem of the existence of a stabbing wedge for an arbitrary set of segments S.

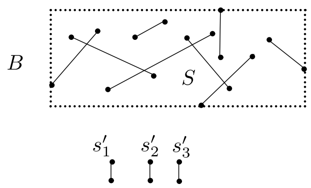

Fig. 12. Lower bound construction for the segment set $S' = S \cup \{s'_1, s'_2, s'_3\}$.

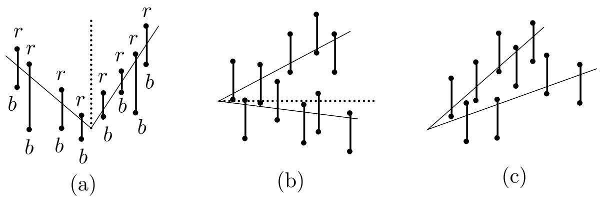

Fig. 13. The three types of stabbing wedges.

Theorem 6. *Deciding whether there exists a stabbing wedge for S requires $\Omega(n \log n)$ time in the fixed order algebraic decision tree model.*

Proof. We reduce the decision of the stabbing wedge problem to the problem of deciding whether there exists a stabbing line for a segment set, which has an $\Omega(n \log n)$ time lower bound in the fixed order algebraic decision tree model [2].

Let S be an arbitrary segment set. In $O(n)$ time, compute the minimum orthogonal box B containing S defined by the endpoints of the segments in S with the biggest and smallest y-coordinates, and the biggest and smallest x-coordinates. Let s'_1, s'_2, and s'_3 be three "small" vertical segments out of the box B such that any stabbing line for the three of them does not intersect B (Fig. 12). Consider the set $S' = S \cup \{s'_1, s'_2, s'_3\}$. By construction, there exists a stabbing wedge for S' if and only if there exists a stabbing line for S. Observe that our construction does not work with two segments s'_1 and s'_2 since there might exist a stabbing wedge for $S \cup \{s'_1, s'_2\}$ with no stabbing line for S. □

4. Stabbing wedges for parallel segments with equal length

Let $S = \{s_1, \ldots, s_n\}$ be a set of n parallel and vertical segments with *equal length* in the plane. To check whether S is stabbed by a line takes $O(n)$ time, e.g., color red (blue) the endpoints of the segments with bigger (smaller) y-coordinate, and decide whether the red points are line separable from the blue points (see [8]).

We consider the problem of computing a stabbing wedge for S with the separability condition, assuming that S is not stabbed by a line. Corollary 1 says that such a stabbing wedge can be computed in $O(h_S \log n + n \log n)$ time and $O(h_S + n)$ space, which in the worst case is $O(n^2 \log n)$ time and $O(n^2)$ space (the segments in Fig. 7(a) might be parallel and with equal length). In this section, we improve the running time and the space assuming that the segments have equal length.

Up to symmetry, we distinguish three types of possible stabbing wedges $W = \{\ell_1, \ell_2\}$ for S according to the relative position of the rays ℓ_1 and ℓ_2. Denote by α_W the *aperture angle* or *interval direction* defined by ℓ_1 and ℓ_2. The three types are the following: (a) α_W contains the vertical direction; (b) α_W contains the horizontal direction; and (c) both rays ℓ_1 and ℓ_2 have positive slope (Fig. 13).

4.1. Type (a)

We describe an $O(n \log n)$-time algorithm for computing a stabbing wedge for S when α_W contains the vertical direction. It uses an $O(n \log n)$ time and $O(n)$ space algorithm for computing the separating wedges of n red and blue points in the plane [14].

1. Classify into red/blue the endpoints of the segments in S as follows. Let R and B be the sets of endpoints of segments of S with bigger and smaller y-coordinate, respectively (Fig. 13(a)).
2. In $O(n \log n)$ time, compute a separating wedge of R and B.

Observe that there exists a stabbing wedge for S if and only if R and B are wedge separable. The assumption of equal length is not used in this algorithm. All the possible stabbing wedges are combinatorially equal and by Lemma 11 the locus of their apices can be computed in $O(n \log n)$ time.

Theorem 7. *A stabbing wedge for S such that α_W contains the vertical direction can be computed in $O(n \log n)$ time and $O(n)$ space.*

Remark. If the slopes m and m' of the rays of a stabbing wedge are known, we can apply the following $O(n)$ time algorithm to compute a stabbing wedge for S: Take the median M of the x-coordinates of the endpoints of the segments, let S_1 (S_2) be the subset of segments which endpoints have x-coordinate on the left (right) side of M, in $O(n/2)$ time check whether the red endpoints of S_1 (S_2) are line separable from the blue endpoints of S_1 (S_2). In the affirmative answer, compute the first and last endpoints (witnesses) of S_1 (S_2) according to a sweep line with slope m (m'). Proceed computing the median on the left or right subset in case of one negative answer, and update the "best" witnesses considering the stored old witnesses compared with the computed for the next subsets. The time needed is $O(n/2) + O(n/4) + O(n/8) + \cdots = O(n)$. Notice that also in $O(n)$ time we can check whether there exists a stabbing wedge for S with apex on a given point.

4.2. Type (b)

We now describe an $O(n\log n)$-time algorithm for computing a stabbing wedge for S when α_W contains the horizontal direction. For each segment $s_i \in S$, consider its midpoint ρ_i. In $O(n\log n)$ time, sort these midpoints according to a sweep with a horizontal line (decreasing y-coordinate), let $\preccurlyeq_h$ denote this order. Because of the equal length and the separability conditions, the following lemma is straightforward.

Lemma 13. *If there exists a stabbing wedge $W = \{\ell_1, \ell_2\}$ for S such that α_W contains the horizontal direction, then the midpoints of the segments stabbed by ℓ_1 appear first in the $\preccurlyeq_h$ order than the midpoints of the segments stabbed by ℓ_2.*

The following result is a straightforward consequence of Lemma 10.

Lemma 14. *There exists a stabbing wedge $W = \{\ell_1, \ell_2\}$ for S if and only if both S_1^+ and S_2^- are line separable from $S_1^- \cup S_2^+$.*

Lemmas 13 and 14 are the key tools to design the following algorithm for computing a set $\mathcal{W}$ of stabbing wedges for S such that α_W contains the horizontal direction.

Algorithm 3.
Input: Set S of n vertical segments in the plane with equal length.
Output: $\mathcal{W}$.

1. In $O(n\log n)$ time, sort the segments of S according to the $\preccurlyeq_h$ order obtaining the segment sequence $S^* = (s_1^*, \ldots, s_n^*)$. For $i = 1, \ldots, n-1$ we define $S_{1,i} = (s_1^*, \ldots, s_i^*)$ and $S_{2,i} = (s_{i+1}^*, \ldots, s_n^*)$.
   In $O(n)$ time, assign e^+ (e^-) to the endpoint of $s \in S^*$ with bigger (smaller) y-coordinate. Denote by $S_{1,i}^+$ ($S_{1,i}^-$) the sets of endpoints of $S_{1,i}$ classified into e^+ (e^-). The sets $S_{2,i}^+$ and $S_{2,i}^-$ are denoted analogously.
2. In $O(n\log n)$ time, compute $CH(S_{1,1}^+)$, $CH(S_{1,1}^- \cup S_{2,1}^+)$, and $CH(S_{2,1}^-)$.
3. **For** $i := 1$ to $n-1$ **do**
   (a) In $O(\log n)$ time, check whether $CH(S_{1,i}^+)$ and $CH(S_{1,i}^- \cup S_{2,i}^+)$ are line separable, and whether $CH(S_{1,i}^- \cup S_{2,i}^+)$ is separable from $CH(S_{2,i}^-)$.
   If the answer is affirmative in both cases, there exists a stabbing wedge for S. In $O(\log n)$ time, compute a line $\ell'_{1,i}$ ($\ell'_{2,i}$) separating $CH(S_{1,i}^+)$ and $CH(S_{1,i}^- \cup S_{2,i}^+)$ ($CH(S_{1,i}^- \cup S_{2,i}^+)$ and $CH(S_{2,i}^-)$). Let $W_i = \{\ell_{1,i}, \ell_{2,i}\}$. Add W_i to $\mathcal{W}$.
   (b) In constant time, update the sets $S_{1,i}$, and $S_{2,i}$, obtaining the sets $S_{1,i+1}$ and $S_{2,i+1}$ (only one segment changes from one set to other set). In $O(\log n)$ time, update $CH(S_{1,i+1}^+)$, $CH(S_{1,i+1}^- \cup S_{2,i+1}^+)$, and $CH(S_{2,i+1}^-)$.

Theorem 8. *A stabbing wedge for S such that α_W contains the horizontal direction can be computed in $O(n\log n)$ time and $O(n)$ space.*

4.3. Type (c)

First, we show how to obtain a consistent classification of the midpoints of the segments according to the possible stabbing wedge of type (c). Denote by d the length of the segments of S. Assume that there exists a stabbing wedge $W = \{\ell_1, \ell_2\}$ of type (c) for S. Suppose also that its aperture angle α_W is known. Consider the line ℓ'_i containing the ray ℓ_i for $i = 1, 2$. Denote by ℓ''_1 (ℓ''_2) the line below (above) and parallel to ℓ'_1 (ℓ'_2) such that the *vertical distance* between the two lines is exactly $d/2$ (see Fig. 14).

Obviously, the angle defined by ℓ''_1 and ℓ''_2 is α_W. Let ℓ be the bisector line of α_W. In fact, any line whose slope is within the slope interval defined by α_W can play the role of ℓ. Consider the upper rays and the lower rays of the double-wedge formed by the lines ℓ''_1 and ℓ''_2. By construction of ℓ''_i, all the midpoints of the segments of S stabbed by ℓ_1 (ℓ_2) are above (below) or over these upper rays (lower rays). Let $\preccurlyeq_\ell$ be the order of the midpoints of the segments of S according to a sweeping by ℓ. The following lemma is straightforward.

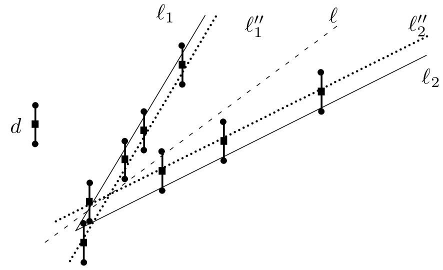

Fig. 14. Both rays ℓ_1 and ℓ_2 of a stabbing wedge have positive slope.

Lemma 15. *If there exists a stabbing wedge $W = \{\ell_1, \ell_2\}$ of type* (c) *for S with aperture angle α_W, then all the midpoints of the segments stabbed by ℓ_1 appear first in the $\preccurlyeq_\ell$ order than all the midpoints of the segments stabbed by ℓ_2.*

Using Lemmas 14 and 15, we design the following algorithm for computing a set $\mathcal{W}$ of stabbing wedges W of type (c) with aperture angle $\alpha_W \geqslant \alpha$.

Algorithm 4.
Input: Set S of n vertical segments in the plane with equal length and a value $\alpha > 0$.
Output: $\mathcal{W}$.

1. *Preprocess*. Let $\lceil \frac{\pi}{2\alpha} \rceil := t$ be a constant. Let $\{\alpha_1, \dots, \alpha_t\}$ be the set of t different aperture angles obtained by splitting the first quadrant orientations into t consecutive aperture angles. Let ℓ_j be the bisector of α_j.
2. **For** $j := 1$ to t **do**: Run Algorithm 3 on S using the $\preccurlyeq_{\ell_j}$ order instead of $\preccurlyeq_h$.

Since Step 1 takes constant time and Step 2 repeats Algorithm 3 a constant number t of times, we can state the following result.

Theorem 9. *A stabbing wedge for S of type* (c) *with aperture angle $\alpha_W \geqslant \alpha > 0$ can be computed in $O(\frac{n}{\alpha}\log n)$ time and $O(n)$ space.*

Remarks. There is no relationship between α_W and the length of the segments. Indeed, α_W might be very small and the length be very large. Fig. 7(a) shows that there can be $O(n^2)$ combinatorially different stabbing wedges of types (b) and (c) for a set S, nevertheless our algorithms find one of them. *A still open question is whether the algorithms for the stabbing wedges of types (a), (b) and (c) are optimal.*

5. Other stabbers

In this section we consider the problem of computing other simple stabbers for S such as a double-wedge and a zigzag, both with or without the separability condition.

5.1. Stabbing double-wedge

A *stabbing double-wedge* $DW = \{\ell_1, \ell_2\}$ for a set of segments S is formed by two intersecting lines ℓ_1 and ℓ_2, where ℓ_1 stabs the subset $S_1 \subsetneq S$ with $S_1 \neq \emptyset$, and ℓ_2 stabs $S_2 = S \setminus S_1$.

5.1.1. Stabbing double-wedges without the separability condition

The main property of a stabbing double-wedge $DW = \{\ell_1, \ell_2\}$ without the separability condition is that every segment can be stabbed by both lines ℓ_1 and ℓ_2. Consider the sets S_2^+ and S_2^- of endpoints of $S_2 = S \setminus S_1$ classified according to a sweeping line of slope m as in Lemma 8. We use the following lemma to design the algorithm for computing a set $\mathcal{DW}$ of stabbing double-wedges for S. Its proof is straightforward.

Lemma 16. *There exists a stabbing double-wedge $DW = \{\ell_1, \ell_2\}$ for S without the separability condition if and only if $CH(S_2^+)$ and $CH(S_2^-)$ are line separable.*

Algorithm 5. Stabbing-double-wedges-without-separability-condition
Input: Set S of n segments in the plane.
Output: $\mathcal{DW}$.

1. *Preprocess*
   (a) In $O(n \log k_S)$ time sort the k_S different slopes of the segments by increasing angular order. Denote them by $m_1, \dots, m_{k_S}$.
   (a) In $O(n^2)$ time compute $\mathcal{A}(S)$ which divides the plane into $O(n^2)$ cells. Compute the dual graph G of $\mathcal{A}(S)$, do a traversal of G, and (following the tree-traversal of the dual graph as above) obtain the sequence $C = (C_1, \dots, C_{r_S})$ of $r_S = O(n^2)$ adjacent cells.
   (c) Select a point $p_{(1,1)} \in C_1$. Let $\mathcal{D}(p_{(1,1)}) := \ell_{1,(1,1)}$. Let $S_{1,(1,1)}$ be the subset of segments stabbed by $\ell_{1,(1,1)}$ and $S_{2,(1,1)} = S \setminus S_{1,(1,1)}$. Classify the endpoints of the segments of $S_{1,(1,1)}$ with respect to $\ell_{1,(1,1)}$ obtaining the sets $S^+_{1,(1,1)}$ and $S^-_{1,(1,1)}$. Compute $CH(S^+_{1,(1,1)})$, and $CH(S^-_{1,(1,1)})$. With slope m_1, determine the classification of the endpoints of the segments of $S_{2,(1,1)}$, i.e., at most two pairs $(S^+_{2,(1,1)}, S^-_{2,(1,1)})$. Compute $CH(S^+_{2,(1,1)})$, and $CH(S^-_{2,(1,1)})$.
2. **For** $i = 1$ to k_S **do**
   For $j = 1, \dots, r_S$ **do**
   (a) If $i = j = 1$, use the data computed in Step 1c and go to Step 2d.
   (b) Select a point $p_{(i,j)} \in C_j$. Let $\mathcal{D}(p_{(i,j)}) := \ell_{1,(i,j)}$. The difference between the segment sets stabbed by $\ell_{1,(i,j)}$ and $\ell_{1,(i,j-1)}$ is at most a segment s since the cells C_{j-1} and C_j are adjacent. This segment s can be stabbed by $\ell_{1,(i,j-1)}$ and not by $\ell_{1,(i,j)}$ or vice versa.
   (c) From s and the sets $S^+_{1,(i,j-1)}$, $S^-_{1,(i,j-1)}$, $S^+_{2,(i,j-1)}$, and $S^-_{2,(i,j-1)}$, in constant time update: (1) $S_{1,(i,j)}$ is the set of segments stabbed by $\ell_{1,(i,j)}$ and $S_{2,(i,j)} = S \setminus S_{1,(i,j)}$; (2) the sets of endpoints $S^+_{1,(i,j)}$, $S^-_{1,(i,j)}$ classified with respect to $\ell_{1,(i,j)}$; and (3) the pair of sets of endpoints $(S^+_{2,(i,j)}, S^-_{2,(i,j)})$ classified with respect the slope m_i (at most two pairs). For each pair proceed as follows.
   (d) In $O(\log n)$ time update $CH(S^+_{2,(i,j)})$, and $CH(S^-_{2,(i,j)})$ and check if they are line separable by a line $\ell_{2,(i,j)}$. If so, by Lemma 16, there exists a stabbing double-wedge $DW_{(i,j)} = \{\ell_{1,(i,j)}, \ell_{2,(i,j)}\}$ for S. Add $DW_{(i,j)}$ to $\mathcal{W}$.

Notice that Step 2 of the above-described algorithm uses $O(n^2 k_S \log n)$ time. The time complexity of Step 2 dominates the time complexity of Step 1.

Theorem 10. *A stabbing double-wedge for S without the separability condition can be computed in $O(n^2 k_S \log n)$ time and $O(n^2)$ space.*

Corollary 3. *A stabbing double-wedge for a set of n parallel segments without the separability condition can be computed in $O(n^2 \log n)$ time and $O(n^2)$ space.*

5.1.2. Stabbing double-wedges with the separability condition

We now assume the separability condition, i.e., a segment cannot be stabbed by both lines of a stabbing double-wedge $DW = \{\ell_1, \ell_2\}$. The following lemma is straightforward.

Lemma 17. *If $DW = \{\ell_1, \ell_2\}$ is a stabbing double-wedge for S with the separability condition, then ℓ_1 (ℓ_2) gives rise to a disjoint bipartition of the subset S_2 (S_1).*

A simple $O(n^4)$-time and $O(n^2)$-space algorithm for computing a stabbing double-wedge is as follows. Consider a line ℓ_1 corresponding to a cell in $\mathcal{A}(S)$ which stabs the subset $S_1 \subsetneq S$ with $S_1 \neq \emptyset$. In $O(n^2)$ time, check whether there exists a cell in $\mathcal{A}(S)$ corresponding to a line ℓ_2 which stabs exactly the segment subset $S_2 = S \setminus S_1$.

Next we describe an algorithm depending on k_S. It is analogous to ALGORITHM 5 but replacing Step 2d by the following step:

Step 2(d): In $O(n \log n)$ time do: (1) update $CH(S^+_{2,(i,j)})$ and $CH(S^-_{2,(i,j)})$, check whether $CH(S^+_{2,(i,j)})$ and $CH(S^-_{2,(i,j)})$ are separable by a line, say $\ell_{2,(i,j)}$; (2) if so classify the endpoints of $S_{2,(i,j)}$ as follows: color red the endpoints in $\ell^+_{2,(i,j)} \cap \ell^+_{1,(i,j)}$, say R_1, and the endpoints in $\ell^-_{2,(i,j)} \cap \ell^-_{1,(i,j)}$, say R_2; color blue the endpoints of $S_{2,(i,j)}$ in $\ell^-_{2,(i,j)} \cap \ell^+_{1,(i,j)}$, say B_1, and the endpoints in $\ell^+_{2,(i,j)} \cap \ell^-_{1,(i,j)}$, say B_2; (3) sort the intersection points of the segments in $S_{1,(i,j)}$ with the line $\ell_{1,(i,j)}$; for each bipartition of $S_{1,(i,j)}$ according to this order, from bottom to top, classify the endpoints of the segments in $S_{1,(i,j)}$ as follows: color the endpoints of the bottom segments as blue in B_1 (red in R_2) if they are in $\ell^+_{1,(i,j)}$ ($\ell^-_{1,(i,j)}$), and color the endpoints of the top segments as red in R_1 (blue in B_2) if they are in $\ell^+_{1,(i,j)}$ ($\ell^-_{1,(i,j)}$); (4) maintaining the convex hulls $CH(R_1 \cup B_2)$ and $CH(B_1 \cup R_2)$, check whether they are line separable by a line $\ell'_{2,(i,j)}$. If so, by Lemma 17, there exists a stabbing double-wedge $DW_{(i,j)} = \{\ell_{1,(i,j)}, \ell'_{2,(i,j)}\}$ for S. Add $DW_{(i,j)}$ to $\mathcal{W}$.

Theorem 11. *A stabbing double-wedge for S with the separability condition can be computed in* $\min\{O(n^4), O(n^3 k_S \log n)\}$ *time and $O(n^2)$ space.*

Corollary 4. *A stabbing double-wedge for a set of n parallel segments with the separability condition can be computed in $O(n^3 \log n)$ time and $O(n^2)$ space.*

5.2. Stabbing zigzag

A zigzag $ZZ = \{\ell_1, s, \ell_2\}$ is a non-convex simple 3-polygonal chain formed by two non-intersecting rays ℓ_1, ℓ_2 and a segment s joining the origin of both rays.

Since a zigzag $ZZ = \{\ell_1, s, \ell_2\}$ splits the plane into two disjoint regions, it seems natural to consider a stabbing zigzag for S with the separability condition such that a segment is stabbed by only one of the three elements of the zigzag and no by the three of them, so that its endpoints can be classified into red or blue. Next we give a short description of the algorithm for computing a stabbing zigzag for S.

5.2.1. Stabbing zigzag with the separability condition

Consider the $O(n^2)$ possible lines ℓ_0 which contain s and stabs a subset $S_0 \subsetneq S$ with $S_0 \neq \emptyset$. Classify the segments of $S \setminus S_0$ depending on which half-plane ℓ_0^- or ℓ_0^+ they lie on, say S_1 and S_2, respectively. We now apply the classification of the endpoints in S_1 and S_2 by two (equal or different) slopes in an additive way since the processes for S_1 and S_2 are independent. The separating lines ℓ_1' for S_1 and ℓ_2' for S_2 are obtained. We then check the separability condition for the two corresponding stabbing wedges: (1) left of ℓ_0 formed by ℓ_1' and ℓ_0, and (2) right of ℓ_0 formed by ℓ_2' and ℓ_0, again in an independent way by using the updated convex hulls of the colored endpoints as we did for the stabbing wedge. In the two affirmative answers case we can compute the stabbing zigzag formed by the rays ℓ_1 and ℓ_2 contained in ℓ_1' and ℓ_1', respectively, and the segment s they define which is contained in the line ℓ_0. Notice that the algorithm spends $O(n \log n)$ time in a preprocess step for computing the initial convex hulls and then, in the following steps the convex hulls can be updated in $O(\log n)$ time per step. Thus, the complexities of the described algorithm are $O(n^2 k_S \log n)$ time and $O(n^2)$ space.

Theorem 12. *A stabbing zigzag for S with the separability condition can be computed in $O(n^2 k_S \log n)$ time and $O(n^2)$ space.*

5.2.2. Stabbing zigzag without the separability condition

The algorithm for finding a stabbing zigzag without the separability condition is basically the same algorithm above for the stabbing zigzag with the separability condition but with the unique change of checking whether from the separating lines for S_1 and for S_2 we can construct a stabbing zigzag for S without the separability condition. To do this task we proceed as follows: As line ℓ_1' (ℓ_2') for S_1 (S_2) we take the two possible interior supporting lines of $CH(S_1^+)$ and $CH(S_1^-)$ ($CH(S_2^+)$ and $CH(S_2^-)$). Compute the intersection point A (B) of the line ℓ_1' (ℓ_2') with the line ℓ_0 and the corresponding ray ℓ_1 (ℓ_2) contained in ℓ_1' (ℓ_2'). In $O(n)$ time, check whether the zigzag formed by the ray ℓ_1, the segment $\overline{AB}$, and the ray ℓ_2 stab all the segments in S. In the affirmative case, we obtain a stabbing zigzag $ZZ = \{\ell_1, \overline{AB}, \ell_2\}$ for S. Thus, the time complexity is $O(n^2 k_S n)$.

Theorem 13. *A stabbing zigzag for S without the separability condition can be computed in $O(n^3 k_S)$ time and $O(n^2)$ space.*

The following tables summarize the main results obtained in this paper. By sc (nsc) we denote *with* (*without*) *the separability condition*

Stabber	Time	Space
Wedge (sc)	$O(h_S k_S \log n + n \log n)$	$O(h_S + n)$
Wedge (nsc)	$O(c_S k_S n + n \log n)$	$O(c_S + n)$
Double-wedge (sc)	$\min\{O(n^4), O(n^3 k_S \log n)\}$	$O(n^2)$
Double-wedge (nsc)	$O(n^2 k_S \log n)$	$O(n^2)$
Zigzag (sc)	$O(n^2 k_S \log n)$	$O(n^2)$
Zigzag (nsc)	$O(n^3 k_S)$	$O(n^2)$

Parallel segments with equal length	Time	Space
Wedge type (a) and (b)	$O(n \log n)$	$O(n)$
Wedge type (c)	$O((n/\alpha) \log n)$	$O(n)$

Acknowledgements

The first author gives special thanks to M. Abellanas and F. Hurtado for their help in the first study of the stabbing problems which was part of her PhD thesis. E. Arkin and J.S.B. Mitchell deserve our thanks for their useful comments. B. Aronov deserves special thanks for his comments about some questions arose by the referees. We also thank to two anonymous referees for their suggestions.

References

[1] M. Atallah, C. Bajaj, Efficient algorithms for common transversal, Inform. Process. Lett. 25 (1987) 87–91.
[2] D. Avis, J.-M. Robert, R. Wenger, Lower bounds for line stabbing, Inform. Process. Lett. 33 (1989) 59–62.
[3] I.J. Balaban, An optimal algorithm for finding segments intersections, in: 11th Sympos. on Comput. Geom., 1995, pp. 211–219.
[4] B.K. Bhattacharya, J. Czyzowicz, P. Egyed, G. Toussaint, I. Stojmenovic, J. Urrutia, Computing shortest transversals of sets, in: Sympos. Comput. Geom., 1991, pp. 71–80.
[5] G.S. Brodal, R. Jacob, Dynamic planar convex hull, in: Proceedings of the 43rd Annual IEEE Symposium on Foundations on Computer Science, 2002, pp. 617–626.
[6] M. Claverol, Problemas geométricos en morfología computacional, PhD thesis, Universitat Politècnica de Catalunya, 2004.
[7] M. Claverol, D. Garijo, C.I. Grima, A. Márquez, C. Seara, Stabbers of line segments in the plane, in: European Workshop of Computational Geometry, 2009.
[8] H. Edelsbrunner, Algorithms in Combinatorial Geometry, EATCS Monographs on Theoretical Computer Science, vol. 10, Springer-Verlag, 1987.
[9] H. Edelsbrunner, L.J. Guibas, Topologically sweeping an arrangement, J. Comput. System Sci. 38 (1989) 165–195; J. Comput. System Sci. 42 (1991) 249–251, Corrigendum.
[10] H. Edelsbrunner, H.A. Maurer, F.P. Preparata, A.L. Rosenberg, E. Welzl, D. Wood, Stabbing line segments, BIT 22 (1982) 274–281.
[11] H. Edelsbrunner, L.J. Guibas, M. Sharir, The upper envelope of piecewise linear functions: algorithms and applications, Report no. UIUCDCS-R-87-1390, University of Illinois, 1987.
[12] M.T. Goodrich, J.S. Snoeyink, Stabbing parallel segments whit a convex polygon, in: Proc. 1st Workshop Algorithms and Data Structures, in: Lecture Notes in Computer Science, vol. 382, Springer-Verlag, 1989, pp. 231–242.
[13] J. Hershberger, S. Suri, A pedestrian approach to ray shooting: shoot a ray, take a walk, J. Algorithms 18 (1995) 403–431.
[14] F. Hurtado, M. Noy, P.A. Ramos, C. Seara, Separating objects in the plane by wedges and strips, Discrete Appl. Math. 109 (2000) 109–138.
[15] K.A. Lyons, Henk Meijer, David Rappaport, Minimum polygon stabbers of isothetic line segments, Dept. of Computing and Information Science, Queen's University, Canada, 1990.
[16] A. Mukhopadhyay, C. Kumar, B.K. Bhattacharya, Computing an area-optimal convex polygonal stabber of a set of parallel line segments, in: 5th Canad. Conf. Comput. Geometry, 1993, pp. 169–174.
[17] A. Mukhopadhyay, C. Kumar, E. Greene, B. Bhattacharya, On intersecting a set of parallel line segments with a convex polygon of minimum area, Inform. Process. Lett. 105 (2008) 58–64.
[18] A. Mukhopadhyay, E. Greene, S.V. Rao, On intersecting a set of isothetic line segments with a convex polygon of minimum area, in: Workshop on Computational Geometry and Applications, 2007, pp. 41–54.
[19] J. O'Rourke, An on-line algorithm for fitting straight lines between data ranges, Comm. ACM 24 (1981) 574–578.
[20] D. Rappaport, Minimum polygon transversals of line segments, Internat. J. Comput. Geom. Appl. 5 (3) (1995) 243–256.