Width and Bounding Box of Imprecise Points

Vahideh Keikha* Maarten Löffler† Ali Mohades* Zahed Rahmati*

Abstract

In this paper we study the following problem: we are given a set $L = \{l_1, ..., l_n\}$ of parallel line segments, and we wish to find a set $P = \{p_1, ..., p_n\}$, where $p_i \in l_i$ such that we maximize/minimize the width of P or the area of the bounding box of P among all possible choices for P. We design an $O(n^2\epsilon^{-4.5})$ approximation algorithm for computing the largest width. We also show that the smallest width and the smallest bounding box can be computed in $O(n^2)$ time. We then proceed to present an $O(n^6)$ time dynamic programming algorithm for computing the largest-area bounding box. We also present an FPTAS for this problem which runs in $O(n^2\epsilon^{-5})$ time.

1 Introduction

Shape fitting is a fundamental problem in computational geometry, computer vision, clustering, data mining and many other areas, which asks the following question: suppose we are given a set P of points in the plane, find a shape that best fits P under some fitting criterion. In computational geometry, many problems fit into the class of shape fitting, e.g., computing the bounding box, the width, the smallest enclosing circle, etc. However, in the real-world, the input is subject to be *imprecise*. Then the question is finding tight bounds on the size of the objective shape.

Imprecise data. Let $P = \{p_1, ..., p_n\}$ be a set of fixed points in the plane. In many applications, each element of P is subject to be computed with some errors, such that we do not know e.g., the exact coordinates of each p_i, or even the existence of p_i. In this situation we call P a set of *imprecise/uncertain points.*

Many studies have focused on solving geometric problems in the presence of imprecise input. Depending on the information we have about the input, different models of imprecision are introduced. Here we briefly mention related models: the *Epsilon-geometry* model, the *Region-based* model, the *Locational* and the *Existential* model, where in these models, it is assumed that there exists a set P_i of points instead of each p_i, but the exact location of p_i in P_i is unknown. See e.g., [3, 5].

*Department of Mathematic and Computer Science, Amirkabir University, [va.keikha,mohades,zrahmati]@aut.ac.ir

†Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, m.loffler@uu.nl

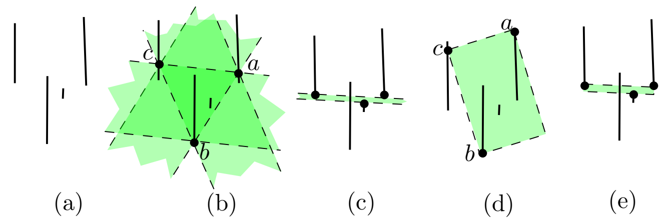

Figure 1: An example. (a) A given set of parallel line segments. (b) The Largest possible width determined by 3 pairs simultaneously. If we move any of a, b or c among their line segments, we reduce at least one of the computed width. (c) The smallest possible width. (d) The largest possible area bounding box. (e) The smallest possible bounding box.

In this paper, we study our problems in the Region-based model. Let R be a set of imprecise points. An *instance* of R is a set P of points selected from distinct regions of R. Then each instance P of R will have different convex hull, width, bounding box, etc. Löffler and van Kreveld introduced a framework for computing some tight lower and upper bounds on the size of such measures, where they modeled the uncertainty of the input by line segments, squares or disks [3].

Contribution. In this paper, we study the following problems: given a set $L = \{l_1, ..., l_n\}$ of parallel line segments, choose a set $P = \{p_1, ..., p_n\}$ of points, where $p_i \in l_i$, such that the size of width or the area of the bounding box of P is as small/large as possible among all possible choices for P (see Figure 1(b-e)). These problems can be interpreted as finding the optimal facilities in the form of a box or a strip which intersects each line-segment-customer.

Preliminaries. Löffler and van Kreveld firstly studied the problem of computing the largest/smallest axis-aligned bounding box of a set of imprecise points modeled as a set of disks or squares in the plane, where their algorithms varied from $O(n \log n)$ to $O(n^2)$ [2]. In the same paper, they proved that computing the largest possible width of a set of imprecise points modeled as a set of arbitrary line segments is NP-hard. The same problem for parallel line segments, squares or disks was posed as open question.

The *axis-aligned bounding box* of a set P of fixed points in the plane is the minimum area bounding box containing P, subject to the constraint that the edges

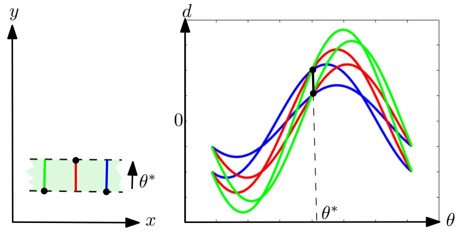

Figure 2: An example. The $\mathcal{A}(\theta, d)$ diagrams of the endpoints of three line segments and two determined widths in direction θ^*. Both the smallest and largest possible width occur in direction θ^*. When we select the lower endpoints of the blue and green line segments, the location of the point on the red line segment determines the width: the lower endpoint of the red segment realizes the smallest possible width, while the upper endpoint realizes the maximum possible width.

of the bounding box are parallel to the $x-y$ coordinate axes. The smallest *oriented bounding box* of P is the minimum area rectangle containing P. From now on, we simply call it *bounding box*. The *width* of a set of points is the narrowest strip containing P. These problems are extensively studied and efficient algorithms are known for them. Once the convex hull of P is known, all these problems can be solved in linear time based on the rotating calipers method [6]. While the convex hull of P is unknown there is an $\Omega(n \log n)$ lower bound for both problems of computing the bounding box and width of P in 2-D.

Results. Let $L = \{l_1, ..., l_n\}$ be a set of parallel line segments. We obtain the following results.

- We show that the largest possible width of L can be approximated within a factor $(1-2\epsilon)$ in $O(n^2\epsilon^{-4.5})$ time (Section 2.1).[1]
- We show that the smallest bounding box of L can be computed in $O(n^2)$ time (Section 3.1).
- We present a more involved $O(n^6)$ time dynamic programming algorithm for computing the largest bounding box of L. We also present an FPTAS for this problem which runs in $O(n^2\epsilon^{-5})$ time (Section 3.2).

We also note that all missing proofs are in Appendix A.1.

2 Width

We start with the *width* problem. Two problems can be considered: finding an instance P on L, so that P maximizes/minimizes the width of P. The minimum width of a set of imprecise points modeled as line segments (or any other convex regions), can be computed in $O(n \log n)$ time [4], in which the problem is so-called *strip transversal*, and the authors studied the problem of computing the thinnest strip that intersects a given set of convex objects. The maximum width problem looks more difficult, because we should find an instance so that our instance maximizes the width of the resulting point set. Since width can be determined by multiple triples of points, and each point can take part in different triples, it looks difficult to find the optimal position of the points (see Figure 1(b)).

Let $L = \{l_1, ..., l_n\}$ be a set of parallel line segments in the plane. Let l_i^- and l_i^+, respectively, denote the lower and upper endpoints of l_i. Let E denote the set of all the endpoints of segments in L. For a point p, let l_p denote the segment that includes p. For each point $p = (x, y)$, we define a $\mathcal{A}(\theta, d)$ diagram to be the plot of the function $d_\theta = x \sin\theta + y \cos\theta$ (1). It is the (signed) distance of p to a line through the origin perpendicular to the ray with angle θ.

$$p_\theta = \begin{bmatrix} x_\theta \\ y_\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}$$

In Figure 2, the $\mathcal{A}(\theta, d)$ diagram of the endpoints of a set of segments is depicted, the region between same color diagrams denotes the $\mathcal{A}(\theta, d)$ diagrams of the remaining points of the segment. From now on, we use $T(P)$ to address the set of $\mathcal{A}(\theta, d)$ diagrams of a set P of points. An example is depicted in Figure 2, where each $T(\{l_i^+, l_i^-\})$ for $i = 1, ..., n$ is assigned a unique color. Note that the $T(\{l_i^+, l_i^-\})$ of a segment l_i intersects any other $T(\{l_j^+, l_j^-\})$ for $j \neq i, j = 1, ..., n$ in a constant number of intersections. Thus there are a quadratic number of intersection points. We call I the set of intersection points, where each element of I is an intersection point between two diagrams with distinct colors.

Notice the smallest possible width of L equals the vertical shortest distance between a point $p \in I$ and a point f_p on another diagram with distinct color, so that at least one color from each diagram is intersected by vertical line segment $\overline{pf_p}$. Since we want to minimize the length of segment $|\overline{pf_p}|$ among all directions θ, it will have one endpoint on an intersection point. Notice that the instances determined in this way will introduce a larger width among all other directions in $\mathcal{A}(\theta, d)$ diagrams of L. Thus this gives a valid width, and further the smallest possible width. But computing the smallest width by this method will cost $O(n^2)$ time.

2.1 Largest width

Let s be a set of $\mathcal{A}(\theta, d)$ diagrams of distinct colors. If s includes exactly one instance of each color, we call s a *complete set*. For a complete set s of diagrams, we define

[1] Our method solves the smallest width problem in $O(n^2)$ time, however, there exists an $O(n \log n)$ time algorithm for this problem [4].

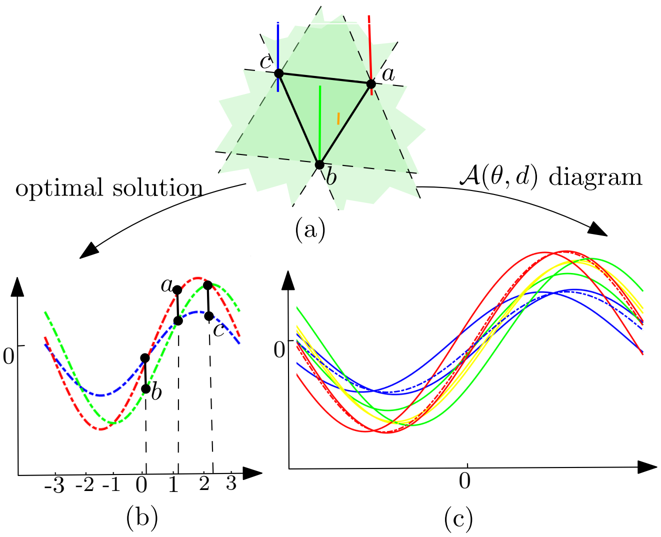

Figure 3: (a) An example. The maximal width is determined by a, b and c. (b) The optimal solution. (c) The $\mathcal{A}(\theta, d)$ diagrams of the endpoints is denoted by solid curves, and the optimal solution is shown in dashed-dotted. The green dashed-dotted is not visible because it is the same as a solid green curve. For better visibility this figure is shown wider.

w_s as the shortest vertical distance between the topmost and bottommost diagrams in s among all values of θ. Obviously w_s determines the width of instances in s.

Observation 1 *Let P be a set of n points from distinct line segments, then $w_{T(P)}$ determines the width of P.*

Proof. The correctness comes from the fact that $w_{T(P)}$ is computed among all values of θ. □

Let P^* denote the set of points which maximizes the width. In the $\mathcal{A}(\theta, d)$ diagram of L, the solution to the largest width problem is equivalent to a complete set $T(P^*)$ of $\mathcal{A}(\theta, d)$ diagrams, so that the value of $w_{T(P^*)}$ is as large as possible among all possible choices of s. In other words, for any other complete set s, $w_{T(P^*)} \geq w_s$. See Figure 3 as an example. As can be seen in Figure 3, the problem probably has algebraic issues. We design an approximation algorithm to solve it. We start stating our results with some observations.

Lemma 1 *Let L be a given set of parallel line segments in the plane, and let E denote the set of all endpoints of L. There exists a solution to the largest width problem so that one of the elements of E is involved in the optimal solution.*

Proof. It is easy to observe that we can translate the set of points realizing the largest width in up or down direction until one of the points determining the strip of width (indeed, the point which has the smallest distance to one of the endpoints of its segment) reaches to the closest endpoint of its segment, or the width misses covering another segment l, which this happens at an endpoint of l again, then the new set realizes the same width, but one element of E is involved in the optimal solution. □

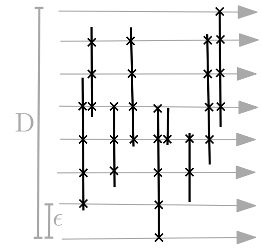

Figure 4: The crosses denote the points approximating the line segments.

Lemma 2 *There exists an ϵ-kernel of size $O(\sqrt{\epsilon^{-1}})$ for the maximum width problem.*

Proof. Agarwal *et al.* [1] proved that for any point set in d-dimensional space, there is an ϵ-kernel of size $O(1/\epsilon^{d-1/2})$ and it is also worst case optimal. Let $opt(L)$ denote the optimal solution to the maximum width problem of L. Then there exists an ϵ-kernel Q_{opt} for $opt(L)$, that is $opt(L) < (1+\epsilon)widt(Q_{opt})$. Let $S(Q_{opt})$ denote the set of segments which share a point on Q_{opt}. Then obviously $opt(S(Q_{opt})) \leq opt(L)$. Also we have $width(Q_{opt}) \leq opt(S(Q_{opt}))$. Then $opt(S(Q_{opt})) \leq opt(L) \leq (1+\epsilon)opt(S(Q_{opt}))$. □

Although we do not know what is our ϵ-kernel, we still can use its size to design a more efficient algorithm.

Let D denote the vertical distance between the highest and smallest y-coordinates of any two endpoints of segments of L, as illustrated in Figure 4. Then we emanate a set ρ of horizontal parallel rays, where the vertical distance between any two consecutive rays is ϵ. For any $l_i \in L$, the intersection points $\rho \cap l_i$ approximate l_i. We will compute the $\mathcal{A}(\theta, d)$ diagrams of $V = \rho \cap L$. Obviously $V \in O(n\epsilon^{-1})$. We postpone the discussions of why this gives us the desired $(1-\epsilon)$ factor, and first discuss the solution on the approximated points.[2]

From Lemma 2 and considering any triple of points which are potentially involved in the maximum width, a naive approach solves the problem in $O(n^{\epsilon^{-1/2}}(\epsilon^{-1/2}\epsilon^{-1})^3)$ time.

Corollary 3 *There exists a PTAS for the maximum width problem which runs in $O(n^{\epsilon^{-1/2}}\epsilon^{-4.5})$ time.*

Observation 2 *If for each $l_i \in L$, $T(\{l_i^+, l_i^-\})$ is always entirely located between two other $\mathcal{A}(\theta, d)$ diagrams in $T(E)$ among all values of θ (shown in yellow in Figure 3(c)), then l_i does not have a role in the constitution of the optimal solution.*

[2] We have supposed $D = 1$, since we are considering the relative error.

The above observation does not necessarily reduce the complexity of the algorithm, but still can reduce the total running time.

2.2 Dynamic programming algorithm

As a consequence of Observation 1 and since we look for the shortest vertical distance between a set of diagrams, the shortest vertical distance will at least use one intersection point of two diagrams with distinct colors. Suppose we have fixed one endpoint b of a segment, and we have computed the $\mathcal{A}(\theta, d)$ diagram of b, $T(\{b\})$ (for simplicity we denote it by $T(b)$). Now the question is how to find a complete set s of diagrams, so that $T(b) \in s$ and other elements of s maximize their vertical distances (w_s) from $T(b)$.

First notice that the number of points in V is in $O(n\epsilon^{-1})$. By considering any triple of points in V which are potentially involved in the optimal solution, obviously the problem can be solved in $O(n^4\epsilon^{-3})$. In the extra $O(n)$ we check whether the strip of triple includes one instance from each segment or not. We will design a DP algorithm which runs in $O(n^2\epsilon^{-4.5})$ time.

For a fix endpoint b, let $w(b)$ denote the length of the shortest vertical segment which is intersected by all the transformations of $\sqrt{\epsilon^{-1}} - 1$ other instances (in the ϵ-kernel) among all directions θ. Let w^* denote the maximum possible width of L. We should maximize the value $w(b)$ for each b. Obviously $w(b)$ in a direction θ can be defined by

$$w(b) = Max_{\forall p_i, d_\theta(T(p_i)) \leq d_\theta(T(b))} \ |d_\theta(T(b)) - d_\theta(T(p_i))| +$$

$$Max_{\forall p_j, d_\theta(T(p_j)) \geq d_\theta(T(b))} \ |d_\theta(T(b)) - d_\theta(T(p_j))|,$$

with $i \neq j$, in which $T(p_i)$ and $T(p_j)$ has the smallest and largest vertical distances from $T(b)$ in direction θ.

$$W(b) = Max \ [w(b)], \ w^* = Max_{b \in E} \ W(b),$$

where $W(b)$ is the maximum over all possible ϵ-kernels on b. There only exist $O(\sqrt{\epsilon^{-1}})$ candidates for each of p_i and p_j, since they belong to an ϵ-kernel of this size. Also we only need to consider the directions θ which is determined by the intersection points of the $\mathcal{A}(\theta, d)$ diagrams of elements in the ϵ-kernel, since the minimum value of $w(b)$ that needs to be maximized happens there. Thus there exist $O((\epsilon^{-1/2}\epsilon^{-1})^2)$ directions θ in total. Also there exist $O(n\epsilon^{-1/2})$ different ϵ-kernels (of size $O(\epsilon^{-1/2})$) to be defined on b. Consequently the dynamic program runs in $(n^2\epsilon^{-4.5})$ time.

Theorem 4 *Let L be a given set of parallel line segments in the plane. The largest possible width of L can be approximated within factor $(1 - 2\epsilon)$ in $O(n^2\epsilon^{-4.5})$ time.*

3 Bounding box

Similarly, for computing the bounding box of L two problems can be considered, neither of these has been studied yet: the smallest area bounding box and the largest area bounding box. Let B^* denote the optimal solution to any of these problems.

3.1 Smallest bounding box

Now we extend our approach for the minimum width problem to design an algorithm for the smallest-area bounding box.

Lemma 5 *Let L be a set of parallel line segments, and let E be set of the endpoints of segments in L. There exists an optimal solution B to the smallest bounding box of L, where each edge of B passes through at least one point of E, and these points belong to distinct segments.*

Proof. Suppose the lemma is false. Then the minimal bounding box B still has an edge e which is determined by a point p_i somewhere on the middle of l_i. Consider a line ℓ through p_i and parallel to e. If we sweep ℓ toward the opposite side of e on B, it will intersect l_i, until it leaves it at an endpoint p'_i (or B misses covering another segment l_j, which happens at an endpoint p_j). Then p'_i (or p_j) can be substituted for p_i to give us a smaller area bounding box. Contradiction. □

Now we have discretized the problem on the endpoints. For any set of fixed points, the smallest bounding box can be determined by five points. Consequently, there exists a naive $O(n^6)$ time algorithm for the smallest bounding box problem, where in the extra $O(n)$ time we should check whether an instance of any segment is included in the solution.[3]

Lemma 6 *Let L be a set of n parallel line segments. Only the directions determined by the intersection points of the elements of $T(E)$ in the $\mathcal{A}(\theta, d)$ diagram of L can be candidates to determine the direction of two parallel edges of B^*.*

Proof. Only the intersection points of $T(E)$ in the $\mathcal{A}(\theta, d)$ diagram of L denote the directions in which a minimum width may exist. Suppose the lemma is false. Then there exists a solution B to the minimum bounding box problem, so that none of the two directions determined by edges of B, are determined by a direction in which a minimal width happens (in an intersection point). Then we find the closest direction $\theta_{l'}$ (to the directions of any of two edges of B) which is a candidate for the smallest width (which happens at an intersection point) (see Figure 5). Let θ_l denote the direction of two parallel edges of B which is closer to $\theta_{l'}$.

[3]Notice that a rotating caliper technique does not look applicable here, since we do not exactly know the convex hull.

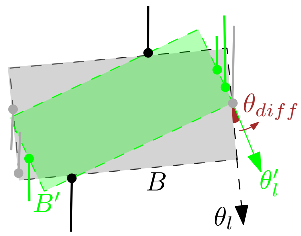

Figure 5: The smallest-area bounding box will be constructed in a direction in which a minimal width exists; if not we still can reduce its area.

Also first suppose the segments determined by direction $\theta_{l'}$ are distinct from the segments involved in the other edges of B'. We define $\theta_{diff} = |\theta_{l'} - \theta_l|$. We substitute the determined width in direction $\theta_{l'}$ for the one in direction θ_l. We then rotate the two other edges of B with θ_{diff} through the previously determined points. Obviously the achieved box B' has a smaller area than B. Contradiction.

Now suppose the determined segments by the width in direction $\theta_{l'}$ are not distinct from the segments involved in the other edges of B'. Then the shared point would be located at a corner of the new box B', while the area of the B' is smaller than B. Contradiction. $\square$

In each intersection point $p \in I$ in direction θ, there may exist a strip with minimal size which includes at least one point from each line segment. The other sides of the potential optimal solution can be determined in direction $\theta + \pi/2$. Since $|I| \in O(n^2)$, the minimum area determined box among the intersection points realizes B^*, and the algorithm works in $O(n^2)$ time.

Theorem 7 *Let L be a set of n parallel line segments. The optimal solution to the smallest bounding box problem of L can be computed in $O(n^2)$ time.*

3.2 Largest bounding box

This problem looks difficult. Even a brute-force algorithm is not straightforward, since we cannot simply expand the edges of a possible box (by using the endpoints of the segments), since collinearity of the points may reduce the size of the optimal box. See Figure 6(a). Let θ^* denote the direction of the largest bounding box. Also notice that at least six points are involved in the optimal solution, since the determined widths in both direction θ^* and $\theta^* + \pi/2$ have the smallest size among all possible directions in which a width can be determined, if not we still can reduce its size, and it is not a valid bounding box. Also as can be seen in Figure 1(d), the largest area bounding box does not necessarily use the orientation of the maximum width.

Lemma 8 *Let L be a set of n parallel line segments, and let E be the set of the endpoints of segments in L. There exists a solution to the largest-area bounding box that uses two points (from distinct segments) of E on its two opposite sides, so that each edge includes one of them.*

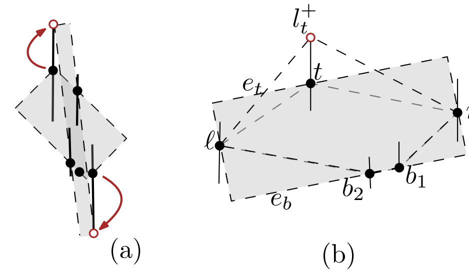

Figure 6: (a) Expanding the edges of a box B to the endpoints of the segments determining the edges of B does not necessarily increase the area of B. (b) The largest-area bounding box at least uses two elements of E on its two parallel sides.

From the $\mathcal{A}(\theta, d)$ diagram of L, the number of different configurations of having two distinct endpoints on two opposite sides of a rectangle is bounded by $O(n^2)$, since in the intersection points the vertical order of two $\mathcal{A}(\theta, d)$ diagrams changes, and there only exists a quadratic number of intersection points. Let t and b denote such endpoints. Also an instance of any other line segment is included in the determined strip by these two endpoints. Notice that four subproblems need to be considered, since we do not know which of the upper or lower endpoints of l_t and l_b are the right ones. With the same argument we had in Lemma 2, there exists an ϵ-kernel of size $O(\epsilon^{-1/2})$ for the largest bounding box. By approximating the set of line segments with a set of parallel rays with ϵ difference between consecutive rays, as discusses in Section 2.1, for any pair of endpoints we need to find a triple of other points to construct a bounding box, where there are $O(\epsilon^{-1/2})$ candidates for each point of triple and $O(\epsilon^{-3})$ possible directions for the optimal box. Thus a DP similar to the one presented in Section 2.2 can solve the problem in $O(n^2\epsilon^{-5})$ time. In the following we try to solve it exactly.

Corollary 9 *Let L be a set of n parallel line segments. There exists an FPTAS for the largest bounding box of L that runs in $O(n^2\epsilon^{-5})$ time.*

Algorithm. Let L be a given set of n parallel line segments. Recall that from $\mathcal{A}(\theta, d)$ diagrams of L we can compute all possible directions which there is a strip $S = d(L, \theta)$, such that S includes at least one instance from each element of L in direction θ. From Lemma 8 we know two distinct segments determine two opposite sides of B^*. As said before, from the $\mathcal{A}(\theta, d)$ diagram of L we understand at most $O(n^2)$ candidates can determine two opposite sides of B^*, since there are $O(n^2)$ intersection points and thus the number of configurations in which all other diagrams are resides between two different diagrams is bounded by $O(n^2)$. Then we should find a valid width with these two points. Let θ denote a direction of such valid width. Notice that there may exist $O(n)$ possible directions for θ. We will consider computing a valid solution from a specific θ, and of course we will repeat it for the remaining possible directions. Then a bounding box B in direction θ can be defined by $B = d(L, \theta)$. $d(L, \theta + \frac{\pi}{2})$ (2), where B is the smallest box which bounds S in $\theta + \frac{\pi}{2}$ direction, so that B is a rectangle. With a bit abusing of the

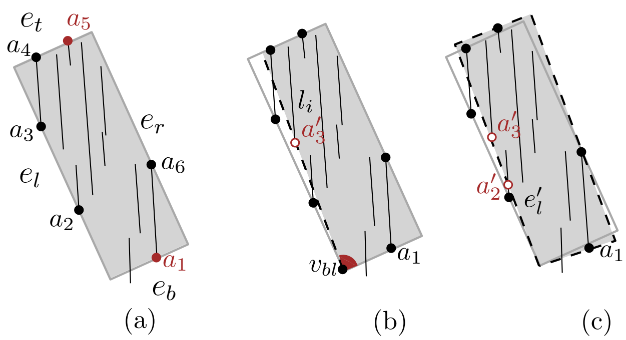

Figure 7: (a) A non-valid bounding box, where it is determined by a_1 and a_5. (b) The maximum inner angle at v_{bl} is determined at a_3'. (c) The corrected edge e_l is denoted by e_l'. $a_1a_2'a_3'a_5a_6$ denote the computed valid box after one step of the DP, which is not completely valid yet. Notice that the hidden set of e_r need to be updated for the next step.

notation, let S and B also denote size of the width and area of the bounding box, respectively.

The above definition of a bounding box B does not necessarily give us a valid box, since some segments may share more than one vertex on the boundary of B, or B may not be the smallest possible box with these instances, so that we still can reduce its area. But correcting this should be done in such away that the removed area from B is minimized. This procedure is called *correcting* B. Also we should do such correction for all possible sub-problems in which a non-valid bounding box is determined by any pair of endpoints. Finally, the largest-area bounding box among all determines the largest-area bounding box of L. Obviously correcting a bounding box B (first computed in direction θ) may also change the direction of B. In the following we show that we still can compute the exact possible rotation of B.

When looking for the largest smallest possible box B, we consider all sets of six points which may define a bounding box in Equation(2), where two of them already determine a valid width and also the first direction of box B, but not necessarily a valid bounding box in that direction. And the other four points are computed accordingly. In other words, for two fixed points, we first determine a valid width through them in a direction θ, and then we compute a valid width in direction $\theta+\pi/2$. Notice that the computed box is a superset for the optimal solution with these instances. Finally in the DP algorithm we try to make the biggest valid box which is determined by these instances. See Figure 7(a) for an example. Let A denote a set consisting of six such points on the boundary of box B. Let e_b, e_l, e_t and e_r denote the edges of B in clockwise direction, and let a_i for $i = 1, ..., 6$ denote such a set A. Also let a_1 be located on edge e_b, a_2, a_3 be located on edge e_l, a_4, a_5 be located on edge e_t, etc. Also let v_{bl} denote the common endpoint of edges e_b and e_l, as can be seen in Figure 7. W.l.o.g suppose B is defined on two endpoints a_1 and a_5.

For each non-distinct element of A, e.g., a_3, a distinct line segment l_i (which is already intersected by B) will share a vertex a_3' on e_l, so that a_3' can be substituted for a_3 to give us a smaller (but a bit more valid) bounding box B'. We call a_3' the *hidden* line segment by e_l. Let $H(A)$ denote all the line segments hidden by the elements of A. In the worst case, correction of B needs to find five hidden vertices by the elements of A, but such substitution should be done in such a way that removed area from B is as small as possible, and the resulting valid B has the largest possible area. (Notice that since $H(A)$ has constant complexity, computing the best configuration can be done in constant time.) Also a hidden line segment l_i might simultaneously be hidden by the points on two edges of B, e.g., in Figure 7, l_i is hidden by both e_l and e_t. We will check both cases in different sub-problems, of which there are constantly many. First suppose l_i should share a vertex on e_l. To do so, we should select a_3' such that the inner angle $a_1v_{bl}a_3'$ has the maximum possible value among all possible choices for a_3'. Further the hidden elements $H(A)$ determine the exact value of the possible rotation of B.

Let Θ denote all possible directions in which a valid possible box B is determined by two points a_1 and a_5, as discussed before, and let $\alpha(A)$ denote the maximum angle of rotation for correcting elements of A. Let A' denote set A, where one distinct element is substituted for one non-distinct element of A. Then we can write our DP as follows:

$$d(L, \theta) = width\ of\ L\ in\ direction\ of\ \theta$$

$$b(L, \theta) = d(L, \theta).d(L, \theta + \frac{\pi}{2}), b(L, \Theta) = Min_{\forall \theta \in \Theta}\, b(L, \theta)$$

Then we have:

$$V(A) = Max\,(Min\, b(A \cup A', [\alpha(A), \alpha(A')]), V(A'))$$

where $V(A)$ denotes the largest bounding box on a possible set A in direction θ. Then the maximum value among all possible $V(A)$ denotes the optimal solution.

The correctness of the algorithm comes from the fact that we consider all possible pairs that can define a bounding box, and then we find the largest possible bounding box on this pair by our dynamic program. Finally, the largest-area corrected box determines the optimal solution. Notice that we may need to rework on a corrected set A', since several elements may need to be corrected. We will correct them clockwise. They only increase a constant number of sub-problems, which at most equals 4×5. Notice that there are constant possible directions in Θ to make a valid box, and it is needed to consider $O(n^2)$ pairs, and for each pair we need to consider $O(n)$ directions, and for each direction θ we should find the width in direction $\theta + \pi/2$. Then we compute the hidden elements of the edges in $O(n)$ time, and we repeat it for any pairs between any two intersection points in $\mathcal{A}(\theta, d)$ of L. Thus the algorithm runs in $O(n^6)$ time and space.

Theorem 10 *Let L be a given set of parallel line segments. The largest bounding box of L can be computed in $O(n^6)$ time and space.*

4 Concluding remarks and open questions

In this paper we present several algorithms for computing an instance P on a set of line segments, so that P maximizes/minimizes the width or the area of the bounding box of P. Solving maximum width problem on a set of squares remained open. We wish to extend our presented algorithms to solve these problems on a set of squares.

References

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

[2] M. Löffler and M. van Kreveld. Largest bounding box, smallest diameter, and related problems on imprecise points. *Computational Geometry*, 43(4):419 – 433, 2010.

[3] M. Löffler. *Data imprecision in computational geometry.* PhD thesis, Utrecht Univesity, 2009.

[4] J.-M. Robert and G. Toussaint. Computational geometry and facility location. In *Proc. International Conference on Operations Research and Management Science*, pages B1–B19, 1990.

[5] D. Salesin, J. Stolfi, and L. Guibas. Epsilon geometry: Building robust algorithms from imprecise computations. In *Proc. 5th Annual Symposium on Computational Geometry*, pages 208–217, 1989.

[6] G. T. Toussaint. Solving geometric problems with the rotating calipers. In *IEEE Melecon*, volume 83, page A10, 1983.

A Appendix

A.1 Omitted proofs

Theorem 4 *Let L be a given set of parallel line segments in the plane. The largest possible width of L can be approximated within factor $(1-2\epsilon)$ in $O(n^2\epsilon^{-4.5})$ time.*

Proof. The only remaining unproved part is the $(1-2\epsilon)$ ratio of approximation. Let $T(P_{app})$ denote a complete set of diagrams which maximizes size of width. Let θ_{app} denote the direction in which $T(P_{app})$ gives the optimal solution. Then on the $\mathcal{A}(\theta, d)$ diagrams, using Equation (1) we have $|w^*| \leq w_{T(P_{app})} + 2\epsilon \cos\theta_{app}$, since $w_{T(P_{app})}$ has the largest value among other complete sets. Obviously $|w^*| \geq \sin\theta_{app}$. But then if $\theta_{app} \geq \pi/4$, $\sin\theta_{app} \geq cos\theta_{app}$ and then, $|w^*|(1-2\epsilon) \leq w_{T(P_{app})}$. In the case where $\theta_{app} < \pi/4$ we obviously have the same width in direction $\theta_{app} + \pi$. The lemma follows. □

Lemma 8 *Let L be a set of n parallel line segments, and let E be the set of the endpoints of segments in L. There exists a solution to the largest-area bounding box that uses two points (from distinct segments) of E on its two opposite sides, so that each edge includes one of them.*

Proof. Like in the case of Lemma 1, there always exists an optimal solution B so that at least one element of E is involved on determining some edge of B. Let e_b denote such edge. In the following, we discuss the existence of another element of E on the opposite side of e_b. Let b_1 denote the endpoint which has determined the edge e_b. W.l.o.g suppose b_1 is located on the bottom side of B.

Suppose the lemma is false. Then there exists a maximal bounding box B for L which is passing through some points ℓ, t, r, b_1 and b_2, so that B has maximal area and B only uses one element of E. Let e_t and e_b denote the top and bottom edges of B. And let t and b_1, b_2, respectively, denote the points that e_t and e_b are passing through them, as illustrated in Figure 6(b). Consider a line l through t and parallel to e_t. If we sweep l away from e_t, it will intersect the segment l_t, so that it leaves it at an endpoint l_t^+. Then obviously the pentagon $\ell l_t^+ r b_1 b_2$ includes the pentagon $\ell t r b_1 b_2$. Since with a fixed length, the new bounding box should now include some new point which previously where located outside the bounding box, it must be expanded from the width. But then the changes of the area of four boxes should be considered, if we substitute l_t^+ for t, the box with an edge through b_1, b_2 will increase its size, and the same argument holds for the boxes with an edge through b_1, r and b_2, l. Thus the collinearity of l_t^+ with existing vertices cannot make a smaller area box. All together, we do not reduce the size of any other possible bounding box of L, and thus any bounding box which is passing through $\ell l_t^+ r b_1$ and b_2 will have a larger area than B. Thus B could not be the largest area bounding box. Contradiction. □