Approximating Largest Convex Hulls for Imprecise Points

Maarten Löffler

Marc van Kreveld

Department of Information and Computing Sciences, Utrecht University

Technical Report UU-CS-2007-038

www.cs.uu.nl

ISSN: 0924-3275

Approximating Largest Convex Hulls for Imprecise Points [*]

Marc van Kreveld
marc@cs.uu.nl

Maarten Löffler
loffler@cs.uu.nl

Department of Information and Computing Sciences
Utrecht University, the Netherlands

Abstract

Assume that a set of imprecise points is given, where each point is specified by a region in which the point will lie. Such a region can be modelled as a circle, square, line segment, etc. We study the problem of maximising the area of the convex hull of such a set. We prove NP-hardness when the imprecise points are modelled as line segments, and give linear time approximation schemes for a variety of models, based on the *core-set* paradigm.

1 Introduction

In computational geometry, many fundamental problems take a point set as input on which some computation is done, for example to determine the convex hull, the Voronoi diagram, or a travelling sales route. These problems have been studied for decades. The vast majority of research assumes the locations of the input points to be known exactly. In practice, however, this is often not the case. Coordinates of the points may have been obtained from the real world, using equipment that has some error interval, or they may have been stored as floating points with a limited number of decimals. In real applications, it is important to be able to deal with such imprecise points.

When considering imprecise points, various interesting questions arise. Sometimes it is sufficient to know just some possible solution, which can be achieved by applying existing algorithms to some point set that is possibly the true point set. More information about the outcome can be obtained by computing a probability distribution over all possibilities, for example using Monte Carlo methods. In many applications it is also important to know concrete lower and upper bounds on some measure on the outcome, given concrete bounds on the input.

1.1 Results

There are a number of basic geometric measures on point sets, such as the diameter, the size of the smallest enclosing circle, the area of the convex hull, etc. For most of these measures the lower and upper bounds can be computed exactly in an efficient way [18], as is summarised in Table 1. However, there are a few problems for which no efficient exact algorithm is known.

In [16, 17], we studied a wide range of problems concerning the convex hull of imprecise points. We varied the imprecision model (line segment, square, circle), the objective function (area, perimeter), the goal (maximisation, minimisation), and the restrictions on the input (equal size, disjoint, no

[*]This research was partially supported by the Netherlands Organisation for Scientific Research (NWO) under BRICKS/FOCUS grant number 642.065.503 and through the project GOGO.

Table 1: Exact algorithms for basic geometric measures on imprecise point sets, when the imprecision of a point is modelled as a square region. The $O(n^7)$ result for the largest area convex hull only applies for disjoint squares, see Table 2.

structure	smallest	largest
diameter	$O(n \log n)$ [18]	$O(n \log n)$ [18]
closest pair	$O(n \log n)$ [18]	NP-hard [8]
smallest enclosing circle	$O(n)$ [13]	$O(n)$ [18]
convex hull (area)	$O(n^2)$ [17]	$O(n^7)$ [17]
minimum spanning tree	NP-hard [17]	open

restrictions). It appeared that the maximisation of area variant (see Figure 1) was one of the hardest, where we found many polynomial time algorithms of rather high degree, and were unable to find any polynomial time algorithm for several variants.

Here we present linear time approximation schemes for all variants of the largest area convex hull problem. The algorithms are all of the form $O(n + f(\eta))$, where n is the input size and $\frac{1}{\eta} = \varepsilon$ is the required precision of the answer. The dependence on n is linear, provided that the ceiling operation can be performed in constant time. The dependence on η is polynomial if we have a polynomial time exact algorithm to solve the problem, and superpolynomial otherwise. Our previous exact results and new approximate results are summarised in Table 2.

1.2 Related Work

A lot of research about imprecision in computational geometry is directed at computational imprecision [19], or at stochastic or fuzzy imprecision models. Recently, however, interest in exact approaches to deal with data imprecision is growing [1, 4, 5, 7, 11, 14, 15]. A more extensive overview of related work in this area is given in [17].

The *core-set* framework, introduced by Agarwal and Har-Peled [2], is a powerful way to obtain approximation algorithms, and still an active research topic [3, 9, 10]. In this framework, a point set P is given, and the problem is to maximise some measure $\mu(P)$. To do this, one constructs a *core-set* $P' \subset P$, such that $\mu(P') > (1 - \varepsilon)\mu(P)$. The size of the core-set must only depend on ε, and not on n (or sublinearly on n, depending on the application). Now the total running time of the algorithm is the time it takes to construct P', and the time it takes to compute $\mu(P')$, where the second step does not depend on n. If the first part can be done in linear time, one obtains a *linear time approximation scheme* (LTAS) [6]. A good survey on core-sets is provided by Agarwal *et al.* [3].

We generalise the concept of core-sets to sets of imprecise points. We are not given a set of points,

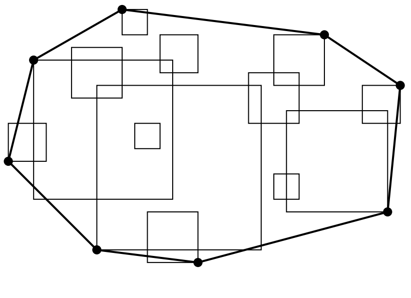


Figure 1: The maximum area convex hull for a set of imprecise points modelled as squares.

Table 2: New and previous results for maximising convex hull area of a set of imprecise points.

model	restrictions	exact algorithm	approximation scheme
line segments	parallel	$O(n^3)$ [17]	$O(n+\eta^3)$
line segments	-	$O(2^n n \log n)$ (NP-hard)	$O(n) + 2^{O(\eta^2)}$
squares	disjoint	$O(n^7)$ [17]	$O(n+\eta^{14})$
squares	equal size	$O(n^5)$ [17]	$O(n+\eta^{12})$
squares	disjoint and equal size	$O(n^3)$ [17]	$O(n+\eta^{12})$
squares	-	$O(4^n n \log n)$	$O(n) + 2^{O(\eta^2)}$
k-gons	disjoint or equal size	$n^{O(k)}$	$O(n) + 2^{O(k \log \eta)}$
k-gons	-	$O(k^n n \log n)$	$O(n) + 2^{O(\eta^2 \log k)}$
circles	disjoint or equal size		$O(n) + 2^{O(\sqrt{\eta} \log \eta)}$
circles	-		$O(n) + 2^{O(\eta^2 \log \eta)}$

but a set of regions. A core-set of such a set is still a subset of bounded size that guarantees a good solution. However, the criteria to include a region in the core-set become more elaborate; in some cases they depend on the size and shape of a region as well as its location, rather than only on its location, as is usually the case. For the classical (precise) convex hull problem it is well known that a small core-set always exists [3]. This immediately implies that a core-set for the imprecise convex hull problem exists as well: take a core-set for the optimal solution. In the remainder of this paper we show how to compute such core-sets efficiently.

2 Preliminaries

Before treating the main results, we first give some small results that are independent of the rest of the text, but are needed in some of the proofs.

2.1 Perfect Matchings in Bipartite Graphs

Let G be a bipartite graph with two sets of vertices V and W, and a set of edges $E \subset V \times W$. Let $M \subset E$ be a maximum matching of G, and let $V' \subset V$ and $W' \subset W$ be the two vertex sets that are used by M. A matching between two subsets $A \subset V$ and $B \subset W$ is called *perfect* if it consists of exactly $\min(|A|, |B|)$ edges.

Lemma 1 *For every subset $Y \subset W$, if there is a perfect matching between V and Y then there is also a perfect matching between V' and Y.*

Proof: Suppose the theorem is false. Let $Y \subset W$ be a subset of W such that there exists a perfect matching between V and Y, but no perfect matching between V' and Y. Let $N \subset E$ be the matching among all perfect matchings between V and Y that uses the largest number of vertices of V'. Let $X \subset V$ be the set of vertices used by N, apart from Y. Then $X \not\subset V'$, so there is a vertex $x \in X$ with $x \notin V'$, see Figure 2.

Now start an augmenting path from x that uses only edges of $M \cup N$. This path takes alternating edges from N and from M, since no two from the same set can use the same vertex. Therefore, this path ends either in a vertex $v \in V' - X$ or in a vertex $w \in Y - W'$. In the first case, we have a perfect matching between $X - \{x\} \cup \{v\}$ and Y, which is in contradiction with the choice of N. In the second case, we have a perfect matching between $V' \cup \{x\}$ and $W' \cup \{w\}$, which contradicts the maximality of M. ⊠

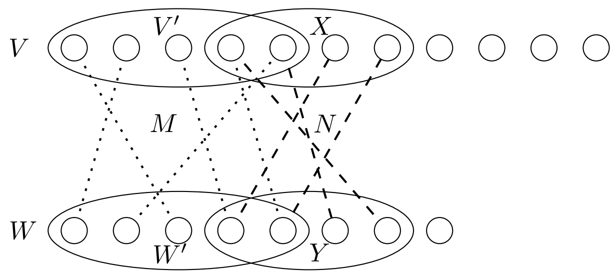


Figure 2: There are vertices in $X - V'$, and from these vertices there is an augmenting path that ends in either $V' - X$ or $Y - W'$.

2.2 Constant Factor Approximation of the Diameter and Width

Lemma 2 *Let P be a set of n points. The diameter of P can be approximated within a factor $\sqrt{2}$ in $O(n)$ time.*

Proof: Compute the axis-parallel bounding box of P, and take the extreme points in the larger direction, so the leftmost and rightmost point if the box is longer in the x direction, otherwise the topmost and bottommost point. In the worst case, the bounding box is a square and the distance between those points is the side of the square, while the real diameter is almost a factor $\sqrt{2}$ larger. ⊠

Lemma 3 *Let P be a set of n points, and let $p, q \in P$ be two points that approximate the diameter within a factor c. Let $\vec{e}_1$ be the direction from p to q and $\vec{e}_2$ a direction perpendicular to it. Let B be the smallest axis-parallel bounding box of P in the $(\vec{e}_1, \vec{e}_2)$ coordinate system. Then the height h of B approximates the width w of P within a factor $2c$, that is, $w \geq \frac{h}{2c}$.*

Proof: Let u and v be the points that realise the real diameter d of P, so the distance between p and q is at least $\frac{d}{c}$, see Figure 3. Now consider the area of the quadrilateral $\Diamond puqv$. We can write this area as $A = \frac{1}{2}|\overline{pq}|h \geq \frac{1}{2}\frac{d}{c}h$. But we can also write it as $A = \frac{1}{2}|\overline{uv}|w'$, where w' is the width of the P in the direction perpendicular to $\overline{uv}$. The distance from p or q to $\overline{uv}$ is smaller than w, so $w' \leq 2w$. Therefore, $\frac{1}{2}\frac{d}{c}h \leq A \leq \frac{1}{2}dw$, and $w \geq \frac{h}{2c}$. Note that if $\Diamond puqv$ is not convex, the area only becomes even smaller than $\frac{1}{2}|\overline{uv}|w'$. ⊠

3 Approximating Largest Convex Hull

We study the problem of finding the largest possible convex hull for a set of imprecise points. Given a set of imprecise points, choose a point in each imprecise point such that the convex hull of the resulting point set is as large as possible. We measure the size of the convex hull by its area.

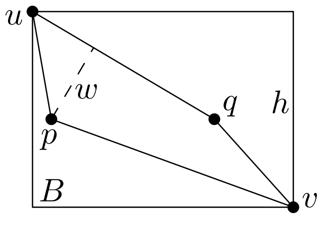


Figure 3: The ratio between w and h is at most $2c$

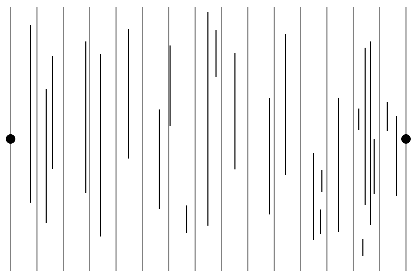


Figure 4: A set of parallel line segments divided in vertical strips.

We are given a set $\mathcal{L} \subset \mathcal{P}(\mathbb{R}^2)$ of imprecise points; that is, $\mathcal{L}$ is a set of subsets of $\mathbb{R}^2$. We want to find a core-set $\mathcal{L}' \subset \mathcal{L}$ with respect to the measure μ, where μ measures the area of the largest possible convex hull. We model the imprecise points as line segments, squares and circles. We are always looking for a $(1-\varepsilon)$-approximation, and we also denote $\eta = \varepsilon^{-1}$.

3.1 Parallel Line Segments

Problem 1 *Given a set of parallel line segments, choose a point on each segment such that the area of the convex hull of the resulting point set is as large as possible.*

We can solve this problem exactly in $O(n^3)$ time, using a dynamic programming solution [17].

3.1.1 Core-Set Construction

Assume that there are a point p_l on the leftmost segment and a point p_r on the rightmost segment that have the same y-coordinate. We can do this without loss of generality, because we can freely skew the problem without changing any areas, keeping the segments vertical.

Let $\mathcal{L}$ be the set of input segments. We will select a core-set $\mathcal{L}' \subset \mathcal{L}$ of at most 16η segments. Let w be the difference in x-coordinates between p_l and p_r, and let h be the (unknown) maximal difference in y-coordinate between any two vertices of the optimal solution. Let δ be $\delta = \frac{1}{4}\varepsilon \cdot w$. We will now divide the plane into 4η vertical strips of width δ, see Figure 4. In each strip, we take the two topmost and the two bottommost endpoints and add the segments they belong to to $\mathcal{L}'$.

3.1.2 Proof of $\mathcal{L}'$ being a Core-Set

Let S^* be the optimal solution for $\mathcal{L}$, the original input, and let S'^* be the optimal solution for $\mathcal{L}'$. First we show that the area of S^* is bounded by a constant factor of the area w times h. Then we prove that the difference in area between S^* and S'^* is only a fraction of this area, dependent on ε.

Lemma 4 *The optimal solution S^* has area at least $\frac{1}{2}wh$.*

Proof: Let the topmost vertex in the optimal solution be h_t higher than p_l, and the bottommost vertex h_b lower, where $h = h_t + h_b$, see Figure 5. The line segment from p_l to p_r has length w, and the topmost vertex is h_t away from this line, since we ensured that it is horizontal. So the area of this upper triangle is $\frac{1}{2}wh_t$. In the same way, the lower triangle has area $\frac{1}{2}wh_b$. This means that the solution S that uses the topmost and bottommost vertices and p_l and p_r has area $\frac{1}{2}wh$, and the optimal solution S^* cannot be smaller than this. ⊠

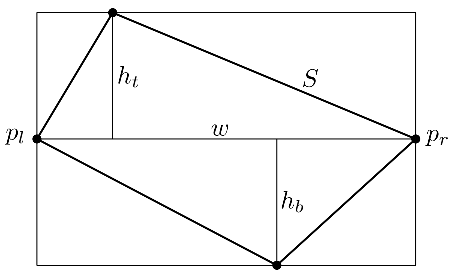


Figure 5: There is a solution S with area $\frac{1}{2}wh$, so S^* is at least as large.

Lemma 5 *There exists a solution S' for $\mathcal{L}'$ such that the difference between S^* and S' is at most $2\delta h$.*

Proof: For each vertical strip, let p_t be the topmost vertex of S^* within that strip and p_b the bottommost vertex. Since p_t and p_b cannot be endpoints of the same segment, there are points p'_t and p'_b in $\mathcal{L}'$ such that p'_t is equal to or above p_t, and p'_b is equal to or below p_b, and they are not endpoints of the same segment, see Figure 6. Use these points in S'. If there are no vertices of S^* in the strip, we just skip it.

We know that S' is a valid solution for $\mathcal{L}$, so $S' \leq S^*$. On the other hand, because of the above, S^* can never be larger than S' with a strip of horizontal width δ around it: the Minkowski sum of S' and the horizontal line segment from $(-\delta, 0)$ to $(\delta, 0)$. So $S^* \leq S' + 2\delta h$. ⊠

Let S'^* be the optimal solution for $\mathcal{L}'$. Now we have:

$$S'^* \geq S' \geq S^* - 2\delta h = S^* - \frac{1}{2}\varepsilon wh \geq S^* - \varepsilon S^* = (1-\varepsilon)S^*$$

3.1.3 Running Time Analysis

As mentioned earlier, we assume that the ceiling operation can be performed in constant time. This is necessary to put the segments into the correct strips in linear time. Without this assumption, the algorithm can be made to run in $O(n \log \eta)$ time.

Theorem 1 *We can compute a core-set of size $O(\eta)$ for Problem 1 in $O(n)$ time.*

This problem can be solved exactly in $O(n^3)$, and therefore approximated in $O(n + \eta^3)$ time.

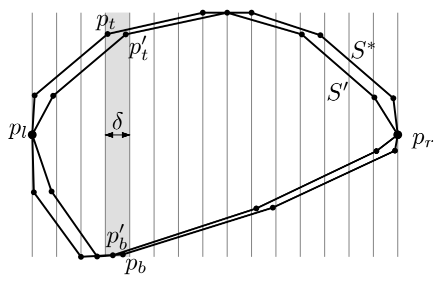


Figure 6: In one strip, the horizontal difference between the points in S^* and S' is at most δ.

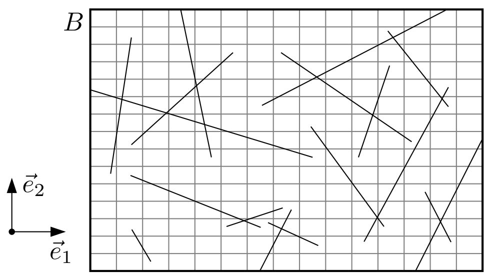


Figure 7: A set of line segments divided according to their cells.

3.2 Arbitrary Line Segments

Problem 2 *Given a set of line segments, choose a point on each segment such that the area of the convex hull of the resulting point set is as large as possible.*

As we proved in [16], this problem is NP-hard. However, for the core-set approach, we do need an exact algorithm. There exists an optimal solution that has every point on an endpoint of its segment. Therefore we can solve the problem in $O(2^n n \log n)$ time by computing the convex hull of every possible set of endpoints. Of course this can be improved slightly.

3.2.1 Core-Set Construction

Firstly, we scale the input to ensure that the width and diameter of the set of endpoints are not too different. Compute a factor $\sqrt{2}$ approximate diameter $\tilde{d}$, and scale the input such that the bounding box aligned with the two points determining $\tilde{d}$ becomes a square, without changing the distance between those poits. Note that this does not influence the relative area of any solution. After scaling, the real diameter may have become a factor $\sqrt{2}$ larger, so $\tilde{d}$ is no longer a factor $\sqrt{2}$ approximation but a factor 2 approximation. If the sides of the square have length b, then by Lemma 3 the width of the scaled input is at least $\frac{1}{4}b$. We also know that the diameter is at most $\sqrt{2}b$, so they differ by at most a factor $4\sqrt{2}$.

Let l_{max} be the longest segment in $\mathcal{L}$. We put l_{max} in $\mathcal{L}'$. Let p and q be two points of the vertex set of $\mathcal{L} - \{l_{max}\}$ that approximate its diameter within a factor 2. Call the direction from p to q $\vec{e}_1$, and the direction perpendicular to this $\vec{e}_2$. Determine the axis-parallel bounding box B of $\mathcal{L} - \{l_{max}\}$ in the $(\vec{e}_1, \vec{e}_2)$ axis system. Let w be the width (the maximum extent in the $\vec{e}_1$ direction) of B, and h the height (the maximum extent in the $\vec{e}_2$ direction) of B. Assume that $w \geq h$, and exchange axes otherwise.

Divide B into 1024η by 1024η grid cells, see Figure 7. The cells are $\delta_1 = \frac{\varepsilon}{1024}w$ long in the $\vec{e}_1$ direction, and $\delta_2 = \frac{\varepsilon}{1024}h$ long in the $\vec{e}_2$ direction. Consider the bipartite graph where one set of nodes corresponds to the set of line segments $\mathcal{L} - \{l_{max}\}$, and the other set corresponds to the cells of the grid. There is an edge between segment l and cell c if one of the endpoints of l is in c. Let M be a maximum matching of this graph, and add all segments that occur in M to $\mathcal{L}'$.

3.2.2 Proof of $\mathcal{L}'$ being a Core-Set

Let S^* be the optimal solution for $\mathcal{L}$, the original input, and let S'^* be the optimal solution for $\mathcal{L}'$, the core-set. Let A denote an amount of area: $A = \mathit{area}(B) + g \cdot w$, where g is the distance from the centre of B to the furthest endpoint of l_{max}. We will first show that the area of S^* is at least a constant fraction of A. Then we will show that the difference in area between S^* and S'^* is only an ε-dependent fraction of A.

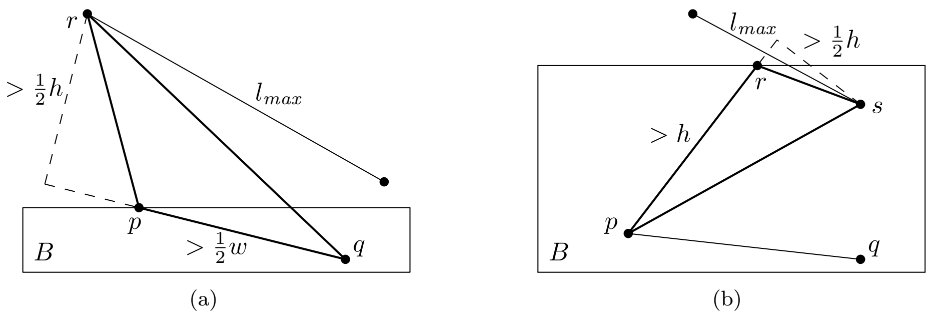


Figure 8: (a) In the narrow case, there is a solution with area at least $\frac{1}{8}wh$. (b) In the non-narrow case, there is a solution with area at least $\frac{1}{128}wh$.

Lemma 6 *The area of S^* is at least $\frac{1}{128}$ times the area of B.*

Proof: We distinguish two cases. If B is narrow, that is, $h < \frac{1}{16}w$, see Figure 8(a), then there exist two points p and q among the endpoints of $\mathcal{L} - \{l_{max}\}$ such that the distance between them is at least $\frac{1}{2}w$, and they are not endpoints of the same segment. There also exists a point r that is an endpoint of l_{max} that is at least $(2\sqrt{2} - 1)h$ away from B, since we know that the width of the original input was at least $\frac{1}{8}\sqrt{2}$ times the diameter. Now r has to be at least $(2\sqrt{2}-2)h > \frac{1}{2}h$ away from the line extending $\overline{pq}$, so the area of $\triangle pqr$ is at least $\frac{1}{8}wh$.

If B is not narrow, that is, $h \geq \frac{1}{16}w$, see Figure 8(b), there exist three points p, q and r among the endpoints of $\mathcal{L} - \{l_{max}\}$ such that the distance between any pair is at least h, otherwise there would be a smaller bounding box. If they are all endpoint of different segments, $\triangle pqr$ is a valid solution of area at least $\frac{1}{4}h^2 > \frac{1}{64}wh$. Otherwise, one of the segments has length at least h, and therefore also l_{max} has length at least h. Suppose p and q are endpoints of the same segment. Now there has to be an endpoint s of l_{max} such that s is at least $\frac{1}{2}h$ away from either the line extending $\overline{pr}$ or the line extending $\overline{qr}$. This means that either $\triangle prs$ or $\triangle qrs$ (or both) has area at least $\frac{1}{8}h^2 > \frac{1}{128}wh$.

In both cases we have a valid solution with an area of at least $\frac{1}{128}wh$, so the area of the optimal solution will also be at least $\frac{1}{128}wh$. ⊠

Lemma 7 *The area of S^* is at least $\frac{1}{128}gw$.*

Proof: There are two points p and q among the endpoints of $\mathcal{L} - \{l_{max}\}$ such that the distance between them is at least $\frac{1}{2}w$, and they are not endpoints of the same segment. Because the width/height ratio of the input was at most $4\sqrt{2}$ and we know there is at least some point more than g away from the centre of B, there must also be a point r that is at least $\frac{1}{8}\sqrt{2}g$ away from the supporting line of $\overline{pq}$. This means that the triangle $\triangle pqr$ has an area of at least $\frac{1}{16}\sqrt{2}gw$. If r is not the other endpoint of the line segment of either p or q, this completes the proof.

Otherwise, suppose p and r belong to the same line segment, and suppose $|pr| < \frac{1}{4}w$. Then we know that l_{max} has length at least $|pr| \geq \frac{1}{8}\sqrt{2}g$, so one of the endpoints of l_{max} is at least $\frac{1}{16}\sqrt{2}g$ away from either $\overline{pq}$ or $\overline{qr}$. Both of these have length at least $\frac{1}{4}w$, so we have a solution of area at least $\frac{1}{64}\sqrt{2}gw$. On the other hand, if $|pr| \geq \frac{1}{4}w$, we have that also $|l_{max}| \geq \frac{1}{4}w$, and of its endpoints is at least $\frac{1}{8}w$ away from either $\overline{pq}$ or $\overline{qr}$, both of which have length at least $\frac{1}{8}\sqrt{2}g$ in this case, and again we have a solution of area at least $\frac{1}{64}\sqrt{2}gw$. ⊠

As a consequence of these lemmata, we now know that the area of S^* is at least a constant fraction of A: $area(S^*) \geq \frac{1}{256}A$.

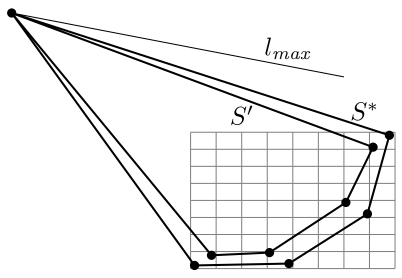


Figure 9: In one cell, the horizontal difference between the points in S^* and S' is at most δ_1, and the vertical difference is at most δ_2.

Lemma 8 *There exists a solution S' for $\mathcal{L}'$ such that the difference between the areas of S' and S^* is at most $\frac{\varepsilon}{256}$ times A.*

Proof: Let Y be the set of grid cells used by the optimal solution S^*. There exists a perfect matching between $\mathcal{L}$ and Y, otherwise S^* would not be possible. By Lemma 1, we know that there is also a perfect matching between $\mathcal{L}'$ and Y. Let S' be the convex hull of the point set that realises this matching, and uses the same endpoint of l_{max} as S^*. Then for each vertex of S^* there is a point of S' in the same grid cell. Going from S^* to S', all vertices can move a distance of δ_1 in the $\vec{e}_1$ direction, and δ_2 in the $\vec{e}_2$ direction, see Figure 9. In the worst case, the transformed solution has a complete 'band' around it. The area of such a band is composed of two triangles incident to l_{max}, which together are smaller than $(\delta_1 + \delta_2)d$, and a part that lies completely within B, which is smaller than $2\delta_1 h + 2\delta_2 w$. In total this is smaller than $\frac{\varepsilon}{256}A$. ⊠

We need that the optimal solution of $\mathcal{L}'$ is at least $(1-\varepsilon)$ times as large as the optimal solution of $\mathcal{L}$. By Lemma 8, there is a solution S' of $\mathcal{L}'$ with an area of at most $\frac{\varepsilon}{256}A$ away from the area of S^*. Furthermore, by Lemmata 6 and 7 we know that the area of S^* is at least $\frac{1}{256}A$. Therefore we have:

$$S'^* \geq S' \geq S^* - \frac{\varepsilon}{256}A \geq S^* - \frac{\varepsilon}{256}256S^* = (1-\varepsilon)S^*$$

3.2.3 Running Time Analysis

To ensure that the input $\mathcal{L}$ is not too skinny, we need to compute an approximate bounding box of the endpoints, which can be done in linear time, according to Lemma 2. Again, we need to perform the ceiling operation to allocate the endpoints of the segments to the right cells of the grid.

To compute a maximum matching, we can use the algorithm by Hopcroft and Karp [12], which runs in $O(\sqrt{|V|}|E|)$ time. In our case, we have $n-1$ nodes on the left side and $2^{20}\eta^2$ nodes on the right side, and every left node has degree 2. When there are more than two left nodes that are connected to the same two right nodes, we will never use more than two of them, so we can reduce the number of left nodes to at most $2 \cdot 2^{40}\eta^4$ by using radix sort. The number of edges is twice the number of left nodes. Now we can compute a maximum matching in $O(\eta^6)$ time. In total this takes $O(n + \eta^6)$ time.

Theorem 2 *We can compute a core-set of size $O(\eta^2)$ for Problem 2 in $O(n + \eta^6)$ time.*

The problem is NP-hard, so unless P=NP there is no polynomial time exact algorithm. A trivial approach takes $O(2^n n \log n)$ time. Using this, we achieve an approximation running time of $O(n) + 2^{O(\eta^2)}$.

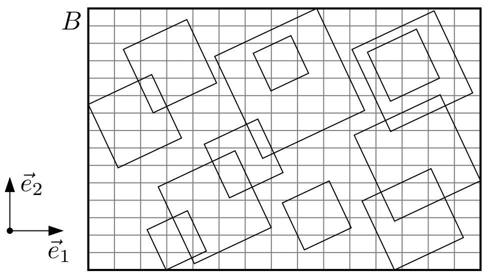


Figure 10: A set of squares divided according to their cells.

3.3 Squares

Problem 3 *Given a set of axis-parallel squares, choose a point in each square such that the area of the convex hull of the resulting point set is as large as possible.*

The status of the general version of this problem is open. In the optimal solution, every point has to be chosen on a corner of its square. Therefore we can solve the problem in $O(4^n n \log n)$ time by computing the convex hull of every possible set of corners.

Under certain conditions, the problem can be solved more efficiently. If the squares are disjoint, we can solve it exactly in $O(n^7)$ time. If the squares all have the same size, we can solve it in $O(n^5)$ time. If the squares are both disjoint and of the same size, we only need $O(n^3)$ time. All of these results can be found in [17].

3.3.1 Core-Set Construction

Let s_{max} be the largest square in $\mathcal{L}$, and s_{max2} the second largest square. Put s_{max} and s_{max2} in $\mathcal{L}'$. Let p and q be two points that approximate the diameter d of the vertices of $\mathcal{L} - \{s_{max}, s_{max2}\}$ by a factor 2. Let $\vec{e}_1$ be the direction from p to q, and $\vec{e}_2$ the direction perpendicular to this. Let B be the smallest bounding box of $\mathcal{L} - \{s_{max}, s_{max2}\}$ in the $(\vec{e}_1, \vec{e}_2)$ axis system, and let w be its width and h its height.

Divide B into $2^{14}\eta$ by $2^{14}\eta$ grid cells, see Figure 10. The cells will be $\delta_1 = 2^{-14}\varepsilon w$ long in the $\vec{e}_1$ direction, and $\delta_2 = 2^{-14}\varepsilon h$ long in the $\vec{e}_2$ direction. Consider the bipartite graph where one set of nodes corresponds to the set of squares $\mathcal{L} - \{s_{max}, s_{max2}\}$, and the other set of nodes corresponds to the cells of the grid. There is an edge between square s and cell c if one of the corners of s is in c. Let M be a maximum matching of this graph, and add all squares that occur in M to $\mathcal{L}'$.

3.3.2 Proof of $\mathcal{L}'$ being a Core-Set

Let S^* be the optimal solution for $\mathcal{L}$, the original input, and let S'^* be the optimal solution for $\mathcal{L}'$, the core-set. First we show that the area of S^* is bounded from below by a constant factor of the area of B. Then we prove that the difference in area between S^* and S'^* is only a fraction of the area of B, dependent on ε.

Lemma 9 *If $n \geq 3$, then the width of S^* is at least $\frac{1}{8}$ times the side length of s_{max2}.*

Proof: Let b be the side length of s_{max2}, and let w^* be the width of S^*. Suppose the lemma is not true, so $w^* < \frac{1}{8}b$. Let p and q be the vertices of S^* that define the diameter d of S^*. Then we know that the area of S^* is $a^* \leq dw^* < \frac{1}{8}db$. Suppose either p or q is not a corner of one of the two largest squares. Then one of the two largest squares has a corner u that is at least $\frac{1}{2}b$ away from the line extending $\overline{pq}$, and there exists a solution of area $\frac{1}{4}db > \frac{1}{8}db$, so in this case S^* would

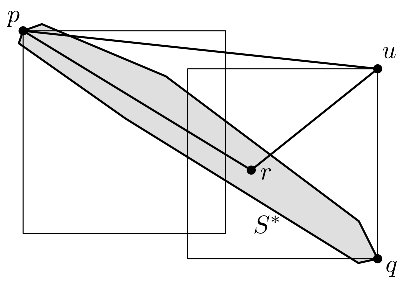


Figure 11: Triangle $\triangle prs$ has a larger area than S^*.

not be optimal, a contradiction. Now suppose that both p and q are corners of the largest two squares, see Figure 11. Let $r \neq p, q$ be an arbitrary vertex of S^*. Now r is at least $\frac{1}{2}d$ away from either p or q, say p without loss of generality. Now the square that has q as a corner has another corner u that is at least $\frac{1}{2}b$ away from the line extending $\overline{pr}$, and there exists a solution of area $\frac{1}{8}db$, so in this case S^* would not be optimal either. Therefore the assumption is false, and the lemma is true. ⊠

This lemma implies that the area of S^* is at least 2^{-8} times the area of s_{max2}.

Lemma 10 *The area of S^* is at least 2^{-12} times the area of B.*

Proof: Let b be the side length of s_{max2}. If $b \geq \frac{1}{4}w$, then this square has area at least $2^{-4}wh$. The optimal solution has area at least 2^{-8} times the second largest square, so at least $2^{-12}wh$.

Next, assume that $b < \frac{1}{4}w$. If p and q, approximating the diameter of the vertices of $\mathcal{L} - \{s_{max}, s_{max2}\}$, would be corners of the same square, then the width of this square would be at least $\frac{1}{2}d \geq \frac{1}{2}w$, which is larger than the diameter of s_{max2}. So p and q are corners of different squares. Let r be the point in P furthest from the line extending $\overline{pq}$, see Figure 12. If r is a corner of yet another square, then the solution pqr has an area of at least $\frac{1}{8}wh$, and so the optimal solution also has at least that area.

If r is a corner of the same square as either p or q, say p, then this means that the width of this square is larger than $\frac{1}{4}\sqrt{2}h$, so $b > \frac{1}{4}h$. The optimal solution S^* uses some corner p'_l of the same square as p_l, the leftmost point in the $\vec{e}_1$ direction, and some corner p'_r of the same square as p_r, the rightmost point in the $\vec{e}_1$ direction and we know that the distance between p'_l and p'_r is at least $w - 3b > \frac{1}{4}w$. We also know that S^* has a width of at least $\frac{1}{8}b > \frac{1}{32}h$, so the area of S^* is at least $2^{-6}wh$. ⊠

Lemma 11 *There exists a solution S' for $\mathcal{L}'$ such that the difference between the areas of S' and S^* is at most $2^{-12}\varepsilon$ times the area of B.*

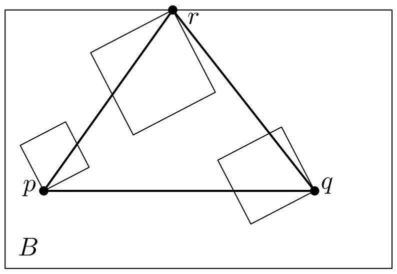


Figure 12: If all squares are small, the triangle $\triangle pqr$ has a large area.

Proof: Let Y be the set of grid cells used by the optimal solution S^*. There exists a perfect matching between $\mathcal{L}$ and Y, otherwise S^* would not be possible. By Lemma 1, we know that there is also a perfect matching between $\mathcal{L}'$ and Y. Let S' be the convex hull of the point set that realises this matching. Then for each vertex of S^* there is a point of S' in the same grid cell. Going from S^* to S', all vertices can move a distance of δ_1 in the $\vec{e}_1$ direction, and δ_2 in the $\vec{e}_2$ direction, see Figure 9. In the worst case, the transformed solution has a complete band around it, that is, it is the Minkowski sum of the old solution and a $2\delta_1$ by $2\delta_2$ rectangle centered at $(0, 0)$. The area of such a band is smaller than $2\delta_1 h + 2\delta_2 w = 2^{-12}\varepsilon wh$. ⊠

Let S'^* be the optimal solution for $\mathcal{L}'$. Then we have:

$$S'^* \geq S' \geq S^* - 2^{-12}\varepsilon wh \geq S^* - 2^{-12}\varepsilon 2^{12} S^* = (1-\varepsilon)S^*$$

3.3.3 Running Time Analysis

The computation of B takes linear time, by Lemma 2. Again, we need to perform the ceiling operation to allocate the corners of the squares to the right cells of the grid.

To compute a maximum matching, we can use the algorithm by Hopcroft and Karp, which runs in $O(\sqrt{|V|}|E|)$ time. In our case, we have $n-1$ nodes on the left side and $2^{28}\eta^2$ nodes on the right side, and every left node has degree 4. When there are more than four left nodes that are connected to the same four right nodes, we will never use more than four of them, so we van reduce the number of left nodes to at most $4 \cdot 2^{112}\eta^8$ by using radix sort. The number of edges is four times the number of left nodes. Now we can compute a maximum matching in $O(\eta^{12})$ time. In total this takes $O(n + \eta^{12})$ time.

Theorem 3 *We can compute a core-set of size $O(\eta^2)$ for Problem 3 in $O(n + \eta^{12})$ time.*

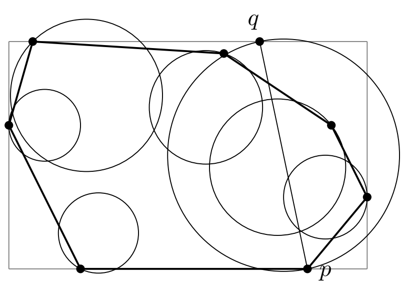
For arbitrary squares, we can solve the problem exactly in $O(4^n n \log n)$ time; therefore we can approximate it in $O(n) + 4^{O(\eta^2)}$ time. For unit size squares, we have an $O(n^5)$ exact algorithm so we get a strong linear time approximation scheme that runs in $O(n + \eta^{12} + \eta^{10}) = O(n + \eta^{12})$. We can solve disjoint squares exactly in $O(n^7)$ so we get an $O(n + \eta^{14})$ LTAS. For squares that are both unit size and disjoint, we have an $O(n^3)$ exact algorithm, but this gives no better result than the general unit size case since the running time is dominated by the term η^{12}.

3.4 Circles

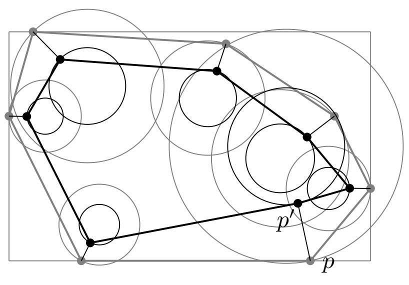
Our exact solution to the convex hull problem for square regions makes use of the four extreme points in the cardinal directions, which makes it impossible to extend to circular regions. A second difficulty is of an algebraic kind. Even if we know which circles have points that contribute to the largest area convex hull, it is not easy to determine where on the circles the points should be. These difficulties remain even for disjoint unit size regions [16].

When we model the points as circular regions (discs), we only need to consider the boundaries, since no vertex of an optimal solution need be chosen in the interior of a region.

Problem 4 *Given a set of circles, choose a point in each circle such that the area of the convex hull of the resulting point set is as large as possible.*



(a)



(b)

Figure 13: (a) The line segment from p to p' cannot go too far outside B. (b) Decreasing the radii of the circles with a factor $(1 - \delta)$ does not decrease the area of the convex hull with more than a factor $(1 - \varepsilon)$.

3.4.1 Approximate Circles by k-gons

Let ε be given. Let $\mathcal{C}$ be the set of circles, and $\mathcal{C}'$ the set of circles with the same centres but radii a factor $(1 - \delta)$ smaller, where $\delta = \frac{1}{8}\varepsilon$.

Lemma 12 *The area of the optimal solution for* $\mathcal{C}'$ *is at least* $(1-\varepsilon)$ *times the area of the optimal solution of* $\mathcal{C}$.

Proof: Let S^* be the optimal solution for $\mathcal{C}$, and B the smallest enclosing bounding box of S^* of width w and height h, let $\vec{e}_1$ and $\vec{e}_2$ be the axes of B. Let S' be the solution for $\mathcal{C}'$ achieved by placing the vertices of S^* on the border of the new smaller circle, but at the same angle with relation to the positive x-axis as they were before, and taking the convex hull of this new point set.

Let p be a vertex of S^*, and let q be the opposite point on the same circle as p, see Figure 13(a). The line segment $\overline{pq}$ cannot be longer than $2w$ in the $\vec{e}_1$ direction and $2h$ in the $\vec{e}_2$ direction, because otherwise choosing q instead of p would yield a better solution. This means that the point p' in C' at the same circle as p is at most δw away from p in the $\vec{e}_1$ direction, and at most δh in the $\vec{e}_2$ direction.

Since this is true for all vertices of S^*, the area of S' is at most $4\delta wh$ smaller than the area of S^*, see Figure 13(b). Since the area of B is at most twice the area of S^*, this means that the area of S' is at least $(1 - 8\delta) = (1 - \varepsilon)$ times as large as the area of S^*. Of course, the optimal solution for $\mathcal{C}'$ can only be larger. $\boxtimes$

We will now approximate the circular imprecise points by k-gons that lie completely within the band between the original circle and the circle with a factor $(1 - \delta)$ smaller radius, see Figure 14.

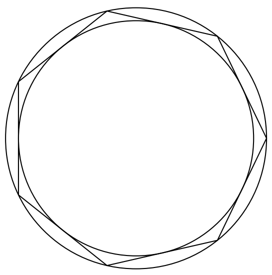


Figure 14: A k-gon between two circles.

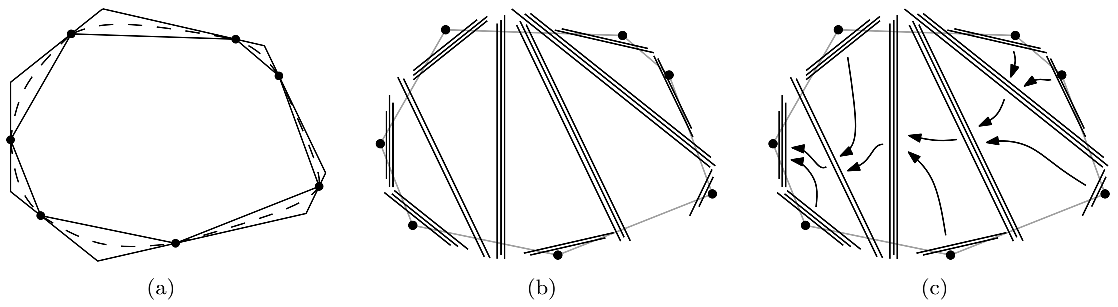


Figure 15: (a) The division of the plane for $k = 7$. (b) There are 11 groups of parallel line segments. (c) The order in which the groups can be combined.

A k-gon fits inside this band when $2k \arccos(1 - \delta) \geq 2\pi$, and this can be estimated by $k \geq 2\pi\sqrt{\eta}$. Let $k = \lceil 2\pi\sqrt{\eta} \rceil$, and $\mathcal{G}$ the set of k-gons (with the same orientation) that have their corners on the circles of C.

Theorem 4 *The area of the optimal solution for $\mathcal{G}$ is a $(1 - \varepsilon)$ approximation for the optimal solution for $\mathcal{C}$.*

Proof: Since all k-gons are contained in the respective circles, the optimal solution for G is a valid solution for C. Since all small circles are contained in the k-gons, the optimal solution for C' is also a valid solution for G, and smaller than the optimal solution for G. $\boxtimes$

3.4.2 Exact Algorithms

The status of the general version of the problem for regular k-gons is open. In the optimal solution, every point has to be chosen on a corner of its k-gon. Therefore we can solve the problem in $O(k^n n \log n)$ time by computing the convex hull of every possible set of endpoints. Of course this can be improved slightly.

As in the case of squares, we can achieve a better algorithm under certain constraints. If the k-gons are either disjoint or unit size, we can solve the problem in $n^{O(k)}$ instead of $k^{O(n)}$ time. We can adapt the algorithm described in [16] in a mostly straightforward manner to the k-gon case. We will briefly discuss the main differences and new ideas that are needed to make these algorithms work.

For both algorithms, we need to know the k extreme points of the solution. These are the vertices of the solution that lie furthest in one of the k directions that are perpendicular to the edges of a k-gon. Trying all possibilities gives a factor $O(n^k)$.

Suppose the k-gons are disjoint. The k extreme points divide the plane into k triangular regions, see Figure 15(a). For each k-gon, we only need to consider the endpoints that are within their respective triangle. Since the k-gons are disjoint, there can be at most $k - 2$ k-gons that intersect more than two of these triangles. For these k-gons, we try every possible combination of their candidate endpoints. This gives a factor $O(k^k)$. [1]

The remaining k-gons can now be represented as line segments. There are at most $2k - 3$ groups of line segments, see Figure 15(b). We can solve the problem in this situation in $O(kn^3)$ time, using a dynamic programming approach as described in [16]. We start with two consecutive groups that pass over only one extreme point, for which there is no group between them. For these two groups, we compute the optimal solution for every pair of points. Then we combine them with the group

[1]In fact, a little more work shows it can only be $O(3^k)$.

that passes over both extreme points. This process is repeated until we have found the optimal solution, see Figure 15(c).

Now suppose the k-gons have equal sizes (but are not necessarily disjoint). The algorithm described in [16] still works exactly as described there, only with k chains instead of four. This gives a running time of $O(n^{k+1})$ instead of $O(n^5)$.

3.4.3 Core-Set Construction

A core-set of a set of regular k-gons can be computed in exactly the same way as with squares, as long as $k \geq 4$. The same proof also applies.

3.4.4 Running Time Analysis

Constructing a core-set of size $O(\eta^2)$ takes $O(\sqrt{|V|}|E|)$ time. In our case, we have $O(\eta^{2k})$ nodes at the left side after removing doubles, and $O(\eta^2)$ nodes at the right side, and each left node has exactly k edges, so $|V| = O(\eta^{2k} + \eta^2)$ and $|E| = O(\eta^{2k+\frac{1}{2}})$. This means that the core-set selection algorithm runs in $O(n + \eta^{3k+\frac{1}{2}}) = O(n) + 2^{O(\sqrt{\eta}\log\eta)}$ time. Again, provided that the ceiling operation takes constant time.

Theorem 5 *We can compute a core-set of size $O(\eta^2)$ for Problem 4 in $O(n) + 2^{O(\sqrt{\eta}\log\eta)}$ time.*

The general problem can be solved exactly in $O(k^n n \log n)$ time. The approximation algorithm then takes $O(n) + 2^{O(\sqrt{\eta}\log\eta)} + O(k^{\eta^2}\eta^2\log\eta) = O(n) + 2^{O(\eta^2\log\eta)}$ time in total.

Under the assumption that the circles are either disjoint or unit size, we have a better exact algorithm, which runs in $n^{O(k)}$ time. The approximation algorithm then takes $O(n) + 2^{O(\sqrt{\eta}\log\eta)} + \eta^{O(\sqrt{\eta})} = O(n) + 2^{O(\sqrt{\eta}\log\eta)}$ time.

4 Conclusions

The *core-set* paradigm has been successfully applied to sets of imprecise points, to obtain approximation algorithms for computationally hard problems. The dependence of the running time on the input size is linear and does not multiply with the dependence on ε, which makes the algorithms suitable for very large sets of imprecise points. On the other hand, the dependence on ε is often highly polynomial or exponential, which limits the achievable precision.

Acknowledgements The authors would like to thank anonymous referees for their detailed and helpful comments.

References

[1] M. Abellanas, F. Hurtado, and P. A. Ramos. Structural tolerance and Delaunay triangulation. *Inf. Proc. Lett.*, 71:221–227, 1999.

[2] P. K. Agarwal and S. Har-Peled. Approximating extent measures of points. In *Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 148–157, 2001.

[3] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via core-sets. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*, volume 52 of *MSRI Publications.* Cambridge University Press, 2005.

[4] D. Bandyopadhyay and J. Snoeyink. Almost-Delaunay simplices: Nearest neighbour relations for imprecise points. In *Proc. 15th ACM-SIAM Symposium on Discrete Algorithms*, pages 410–419, 2004.

[5] C. B. Barber. *Computational geometry with imprecise data and arithmetic.* PhD thesis, Princeton Univ., Princeton, NJ, 1993.

[6] T. M. Chan. Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. *Int. J. Comput. Geometry Appl.*, 12((1-2)):67–85, 2002.

[7] H. Desaulniers and N. F. Stewart. Robustness of numerical methods in geometric computation when problem data is uncertain. *Computer-Aided Design*, 25:539–545, 1993.

[8] J. Fiala, J. Kratochvil, and A. Proskurowski. Systems of distant representatives. *Discrete Applied Mathematics*, 145:306–316, 2005.

[9] G. Frahling and C. Sohler. A fast k-means implementation using coresets. In *Proc. 22nd Annual ACM Symposium on Computational Geometry*, pages 135–143, 2006.

[10] J. Gao, M. Langberg, and L. J. Schulman. Analysis of incomplete data and an intrinsic dimension Helly theorem. In *Proc. 17th Symposium on Discrete Algorithms*, pages 464–473, 2006.

[11] L. J. Guibas, D. Salesin, and J. Stolfi. Constructing strongly convex approximate hulls with inaccurate primitives. *Algorithmica*, 9:534–560, 1993.

[12] J. E. Hopcroft and R. M. Karp. An $n^{\frac{5}{2}}$ algorithm for maximum matching in bipartite graphs. *SIAM Journal on Computing*, 4:225–231, 1973.

[13] S. Jadhav, A. Mukhopadhyay, and B. K. Bhattacharya. An optimal algorithm for the intersection radius of a set of convex polygons. *J. Algorithms*, 20:244–267, 1996.

[14] A. A. Khanban and A. Edalat. Computing Delaunay triangulation with imprecise input data. In *Proc. 15th Canad. Conf. on Comput. Geom.*, pages 94–97, 2003.

[15] Z. Li and V. Milenkovic. Constructing strongly convex hulls using exact or rounded arithmetic. *Algorithmica*, 8:345–364, 1992.

[16] M. Löffler and M. van Kreveld. Largest and smallest convex hulls for imprecise points. Technical Report UU-CS-2006-019, Utrecht University, Institute of Information and Computing Sciences, May 2006.

[17] M. Löffler and M. van Kreveld. Largest and smallest tours and convex hulls for imprecise points. In *Proc. 10th Scandinavian Workshop on Algorithm Theory*, LNCS 4059, pages 375–387, 2006.

[18] M. van Kreveld and M. Löffler. Largest bounding box, smallest diameter, and related problems on imprecise points. In *Proc. 10th Workshop on Algorithms and Data Structures*, LNCS 4619, pages 447–458, 2007.

[19] C.-K. Yap. Robust geometric computation. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 41, pages 927–952. Chapman & Hall/CRC, 2004.