Computational Geometry 43 (2010) 234–242

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Delaunay triangulation of imprecise points in linear time after preprocessing

Maarten Löffler [a,*], Jack Snoeyink [b]

[a] Utrecht University, Department of Information and Computing Sciences, Padualaan 14, 3584 CH Utrecht, The Netherlands
[b] UNC Computer Science, CB 3175 Sitterson Hall, Chapel Hill, NC 27599-3175, USA

ARTICLE INFO

Article history:
Received 1 September 2008
Received in revised form 29 November 2008
Accepted 23 December 2008
Available online 24 April 2009
Communicated by M. Teillaud

Keywords:
Data imprecision
Delaunay triangulation
Minimum spanning tree

ABSTRACT

An assumption of nearly all algorithms in computational geometry is that the input points are given precisely, so it is interesting to ask what is the value of imprecise information about points. We show how to preprocess a set of n disjoint unit disks in the plane in $O(n \log n)$ time so that if one point per disk is specified with precise coordinates, the Delaunay triangulation can be computed in linear time. From the Delaunay, one can obtain the Gabriel graph and a Euclidean minimum spanning tree; it is interesting to note the roles that these two structures play in our algorithm to quickly compute the Delaunay.

1. Introduction

A fundamental assumption of most algorithms in computational geometry is that the input data given is exact. There are actually two good justifications for this assumption: First, by carefully studying the predicates to perform exact computation on the data given, computational geometers can compute a result that is guaranteed to terminate, be self-consistent, and correct on the given input, which is at least close to the input desired. Second, we geometers don't really know what else to do when someone gives us numbers or coordinates but to believe them. Somehow, these justifications are not reassuring to the application practitioners who know that their data is inexact before they throw it over the wall into the geometer's realm.

In this paper we wanted to explore the question, "What is the value of imprecise information given to an algorithm?" To give a particular direction to our query, we answer a question posed by Marc van Kreveld: Suppose that we are given a set of n disjoint unit disks, which represent imprecise information about the coordinates of corresponding points. Can we preprocess these disks so that if we are given m point sets $P_1 \dots P_m$, with each P_i consisting of exactly one point from each disk, then we can compute their m Delaunay triangulations in $o(mn \log n)$ total time? We show that after $O(n \log n)$ time processing the disks using $O(n)$ memory, one can compute each Delaunay triangulation in $O(n)$ time. And once the Delaunay triangulation is obtained, one can compute other structures from it, including the convex hull, the Gabriel graph, or a Euclidean minimum spanning tree.

Our solution actually uses Gabriel graphs and Euclidean minimum spanning trees for the disk centers to allow us to compute, in linear time, a connected subset of Delaunay edges for the specified points, from which the Delaunay computation can be completed by the algorithm of Chin and Wang [10]. Unfortunately for our algorithm's practicality, this last

* Corresponding author.
E-mail addresses: loffler@cs.uu.nl (M. Löffler), snoeyink@cs.unc.edu (J. Snoeyink).

0925-7721/$ – see front matter
doi:10.1016/j.comgeo.2008.12.007

step involves rather heavy machinery. Some of our worst-case constants are over 100, meaning that our result is primarily theoretical, but it does demonstrate that an algorithm can benefit from imprecise information about the location of points.

2. Related work

Problems of exact computation with imprecise geometric operations or data are being attacked from several directions in computational geometry, often with notable success. To set our work in context, we briefly survey imprecise geometric computations, and remind the reader of several standard graphs and computations involving points and disks. We focus on geometric computation, even though ad-hoc and wireless networking applications have stimulated renewed interest in graphs defined by disks, as models of broadcast reachability or of uncertainty [6].

2.1. Geometric computation on imprecise points

Two different but related issues have dominated the research in robust geometric algorithms [33]: closing the gap between the precise mathematics of Euclidean geometry and the inexact primitives offered by the computer, and handling degenerate cases.

Both are important to the practical application of the theory of computational geometry: advances in exact geometric computing (including interval arithmetic, floating point filters, lazy evaluation, root bounds, and real number types) make correct practical code possible for software like Triangle [26] and libraries like LEDA [24] and CGAL [7]; techniques like simulation of simplicity [11] make correct handling of special cases easier. Both, however, assume that the input is exact—even when it is acknowledged that input coordinates represented in floating point are imprecise, it is assumed that the result of predicate on that input is desired, and that degeneracies can be correctly detected and need only be consistently handled.

One way to think about imprecise input is to say that the predicates may return incorrect responses. An early model of inexact predicates is *epsilon geometry* [17,18]: a predicate on a tuple of points would return *true* or *false* if every tuple within a specified ε either was true or was false; it would return *unknown* if both true and false values could be found within ε of the given tuple. This models the uncertainty of a computation by saying that each point lies in an imprecision region—a disk of radius ε centered at the input coordinates.

For some computations, epsilon geometry can bound the accuracy of the output as a function of the ε bound on the input. To obtain such results, it is usually necessary to impose a restriction that the points are separated by 2ε (i.e. the regions are disjoint) or that at most a constant number of imprecise regions contain any point in the plane, because otherwise the information in the input may not constrain the order type of points chosen in the imprecise regions. (The order type of a point set is just the record whether each ordered triple is colinear or forms a left or right turn.)

Various regions have been used to bound the imprecision for some questions in pattern matching [15], but in these cases the output is a simple pairing of points, and less geometric. There are many examples of using hierarchical structures (quadtrees or octrees, for example) that approximate objects to calculate simple combinatorial or metric properties, such as intersection or distance [5]. Van Kreveld and Löffler [23,30] consider a variety of problems such as determining the largest and smallest convex hulls possible given regions that contain the imprecise points—note that convex hull of the regions is typically larger than the largest hull that can be obtained by selecting one point from each region.

For the *Voronoi diagram*, which is the decomposition of the plane by finite number of *sites* induced by labeling each point in the plane by its set of closest sites, and its dual *Delaunay triangulation*, Fortune analyzed the numerical precision of the predicates [13], and pointed out that *geometric rounding*—rounding the output back down to the precision of the input—is an important step in geometric algorithms that is often not explicitly considered. Sugihara and Iri [28,29] advocated designing algorithms to guarantee topological properties even if the primitives are faulty.

Abellanas et al. [1] and Weller [32] have considered the smallest perturbation of sites that can change the combinatorial structure of Delaunay or Voronoi diagrams. Bandyopadhyay and Snoeyink [3,4] compute the set of “almost-Delaunay simplices,” which are the tuples of points that could define a Delaunay simplex under some perturbation of the entire point set by at most $\varepsilon > 0$. (These simplices overlap and do not form a space-filling diagram, but they are useful in a protein analysis application that depends upon identifying potential neighboring atoms as coordinates are perturbed.) The algorithms to identify almost-Delaunay simplices were relatively brute-force.

Ely and Leclerc [12] and Khanban and Edalat [21] consider the epsilon geometry versions of the In-Circle predicate for Delaunay triangulation with imprecise points modeled as disks or rectangles, respectively. Khanban and co-authors [20,22] developed a theory for returning partial Delaunay or Voronoi diagrams, consisting of the portion of the diagram that is certain.

Van Kreveld’s question was motivated by the desire to statistically sample the possible triangulations given n regions that model imprecision. Our aim in solving this question is not to compute a partial Delaunay diagram, but to compute enough structure that we can recover a connected set of Delaunay edges for a given sample, then complete the Delaunay triangulation in linear time. To explain further, we need some more definitions.

2.2. Disks, graphs, and algorithms

We remind the reader of some standard graphs defined by finite sets of points and disks, and the geometric algorithms to compute them. We also define some notation and observe properties that we will use in subsequent sections.

We follow the idea of epsilon geometry, and model input points as unit-radius disks: Let $\mathcal{R}$ be a set of n disjoint open unit disks in the plane, and let $P = \{p_1, p_2, \ldots, p_n\}$ be their center points. An exact sample for $\mathcal{R}$ is a set of points $\hat{P} = \{\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n\}$ drawn one from each disk: i.e., for all $1 \leqslant i \leqslant n$, the length $|p_i \hat{p}_i| < 1$.

The Delaunay triangulation of P can be defined directly as the graph in which an edge joins two sites $p, q \in P$ if and only if there exists a circumcircle for edge pq that has all other sites of P outside. (For this to give a triangulation, one must make a general position assumption: that no four points are co-circular. This assumption can be removed by symbolic perturbation if desired [11].)

In general, computing the Delaunay triangulation of n points requires $\Theta(n \log n)$ time. This lower bound implies that we cannot completely eliminate the disjointness condition and allow all disks to have a common intersection. The lower bound holds even if the points are sorted along x and y coordinated directions [25]; therefore we also cannot hope to do anything if we replace the unit disks by general convex regions, since for a set of vertical lines we would not know anything more than the sorted order.

Aggarwal et al. [2] gave a clever linear-time algorithm to compute the Voronoi diagram or Delaunay triangulation of points in convex position in the plane. Chin and Wang extended this to compute the constrained Delaunay triangulation of a simple polygon [10]. (See [9] for an exposition of similar ideas applied to compute the medial axis.) Rather than define the constrained Delaunay triangulation here, we simply note that if all edges of the simple polygon satisfy the Delaunay empty circle criterion, then the constrained Delaunay is the Delaunay. As a simple consequence, the Delaunay triangulation of a point set can be computed using Chin and Wang's algorithm, if a spanning tree consisting of Delaunay edges is already known. The algorithm does require that the polygon is decomposed into trapezoids, which can theoretically be done in linear time by Chazelle's algorithm [8].

Gabriel and Sokal [14] defined the Gabriel graph for points sites in a similar manner. First, for two sites p and q, let C_{pq} denote the circle with diameter pq. Sites p and q are joined by edge pq if and only if the circumscribing circle C_{pq} has all other sites outside. It is well known, and obvious from this definition, that the Gabriel graph is a subgraph of the Delaunay triangulation.

Any Euclidean minimum spanning tree (EMST) for P is a subgraph of the Gabriel graph of P. This fact is also well-known, and easy to observe: removing a tree edge uv partitions the EMST into two connected components; no vertex in the component of u can lie strictly inside the circle of radius $|uv|$ around v, and vice versa, so the interior of the lune that is the intersection of both circles is empty. This lune contains C_{uv} except for u and v, so all other points must be outside C_{uv}. One consequence is that any EMST has maximum vertex degree 6.

Since the conference version of this paper, there have been results on computing some (not-necessarily Delaunay) triangulation on points drawn from disjoint disks: Held and Mitchell [19] can preprocess a set of n disjoint unit disks in $O(n \log n)$ time, such that when one point in each disk is given, *some* triangulation of the point set can be computed in linear time. They give a simple and practical solution, and their result can be extended to overlapping regions of different shapes, provided that the regions do not overlap more than a constant number of other regions, the regions are fat, and the sizes do not vary by more than a constant. Van Kreveld et al. [31] improve this result to work for arbitrary disjoint regions.

3. Expanded Gabriel circles and EMST edges

Define the expanded Gabriel circle, C^+_{pq}, as the circle with center $(p+q)/2$ and radius $|pq|/2 + 2$. The expanded Gabriel circle contains the centers of disks that could, in an exact sample, prevent $\hat{p}\hat{q}$ from being a Delaunay edge.

Observation 1. *For disk centers $p, q \in P$, if no point $r \in P$ lies in the expanded Gabriel circle C^+_{pq}, then in any exact sample $\hat{P}$, the edge $\hat{p}\hat{q}$ is Delaunay in $\hat{P}$.*

Proof. Consider the smallest circle D enclosing the unit disks centered at p and q; specifically, the circle D centered at the midpoint $(p+q)/2$ with radius $|pq|/2+1$, whose boundary is drawn dotted in Fig. 1(a). There is another circle inside D that has the samples $\hat{p}$ and $\hat{q}$ on its boundary: shrink D about its center until the first point, say $\hat{p}$, lies on the boundary, then continue to shrink about $\hat{p}$ until $\hat{q}$ is also on the boundary. An exact sample $\hat{r}$ can lie inside D only if the corresponding unit disk center satisfies $r \in C^+_{pq}$. □

Let $T = (P, E)$ be the Euclidean minimum spanning tree (EMST) of P.

We will now show that each point in the plane (and therefore also the sites of P) can lie in at most a constant number of the expanded Gabriel circles defined by the edges in E. We use this in later sections to bound the amount of repair work necessary to find a spanning tree of Delaunay edges for a particular sample from the unit disks centered at P. An example of a minimum spanning tree and the expanded Gabriel circles of its edges is depicted in Fig. 4.

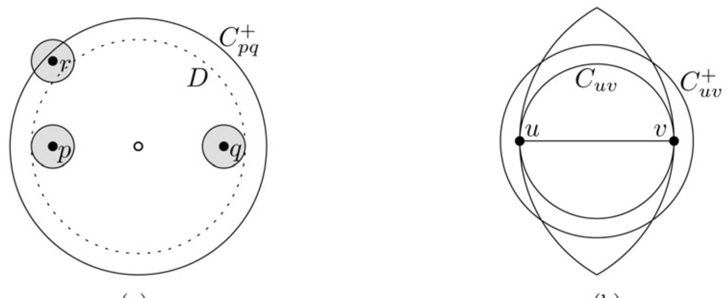

Fig. 1. (a) Expanded Gabriel circle C^+_{pq} contains centers of any disks r such that, in the exact sample $\hat{P}$, the sample point $\hat{r}$ can prevent the edge $\hat{p}\hat{q}$ from being Delaunay (i.e., from having an empty circle). (b) The empty lune for EMST edge uv.

We do an initial partitioning of spanning tree edges into *long* and *short*, depending on whether an edge's length is greater than, or at most, $L = 2 + 2\sqrt{3} \approx 5.464$. This threshold value is chosen so that, if we remove a long edge from the EMST, we can locally identify the connected components that remain.

Lemma 1. *Let uv be a long edge of the EMST of P, and consider any point $w \in P \cap C^+_{uv}$ for which $|uw| \leqslant |vw|$. Then length $|uv| \in (|uw|, |vw|]$, and there is a path from u to w in the EMST that does not use edge uv.*

Proof. Recall that when uv is an edge of the Euclidean minimum spanning tree, the lune that is enclosed by the circles of radius $|uv|$ centered at u and at v has no sites in its interior. When $|uv| > L$, this lune pokes outside the expanded Gabriel circle C^+_{uv}, as in Fig. 1(b). Since the portion of the perpendicular bisector of uv inside the lune cuts the circle C^+_{uv}, we can partition $P \cap C^+_{uv}$ into the sets U and V, closer to u and v, with no ambiguity.

The distance from u to $w \in U$ is maximized if w is at the intersection of the lune boundary with C^+_{uv}. If we let $\ell = |uv|/2$, then because $\ell > L/2$ we know that $\ell + 2 < \ell\sqrt{3}$, and the triangle uvw cannot be equilateral, but must have $|uw| < |uv|$. Now, if we remove uv from the EMST w and u must belong to the same component, since otherwise we could choose uw instead of uv to obtain a spanning tree with lower total distance. □

For a given point p in the plane (possibly a site from P), let E_p denote the set of edges of the EMST whose expanded Gabriel circles enclose p, that is, $E_p = \{uv \in E \mid p \text{ inside } C^+_{uv}\}$. We partition E_p into two groups: the *near* edges, for which both endpoints are at most $L + 2$ away from p, and the *far* edges, for which at least one endpoint is $L + 2$ or more away from p. Note that every far edge must necessarily be long, and that a near edge can be either short or long. We separately bound the numbers of near edges and far edges in E_p.

An easy packing argument bounds the set of near edges for p, which includes all short EMST edges.

Lemma 2. *For any point p in the plane E_p contains at most* 70 *near edges*; *i.e., p is in at most* 70 *expanded Gabriel circles of the edges of the EMST of P that have both endpoints within distance $L + 2$ of p.*

Proof. If a center from P is within $L + 2$ of p, the corresponding disk from $\mathcal{R}$ is within $L + 3$. At most $\lfloor (L+3)^2 \rfloor = 71$ unit disks from $\mathcal{R}$ can fit into this area, inducing at most 70 edges of the minimum spanning tree. □

The constant of 70 is rather pessimistic. The best penny packing known for a circle of radius $L + 3$ has only 57 disks [16, 27], and even then it seems hard to draw many spanning tree edges between them that actually have p in their expanded Gabriel circle.

An angle packing argument in the next lemma shows that an input point $p \in P$ has few far edges.

Lemma 3. *For any point $p \in P$, E_p contains at most* 8 *far edges.*

Proof. We consider far edges $F \subset E_p$ in order of decreasing length, removing them from the EMST of P, and keeping track of the connected component containing p. We assume, without loss of generality, that each far edge is labeled so that the first endpoint is the closer to p; e.g., for uv, we have $|pu| \leqslant |pv|$.

Let T be the current EMST component, which is partitioned into $\{T_u, uv, T_v\}$ by removing uv, the longest edge of $F \cap T$. By Lemma 1, we know that p remains in the component of u, namely T_u, and that $|pu| < |uv| \leqslant |pv|$. We claim that all other edges of F in T belong to T_u: consider another edge $u'v' \in F \cap T$, as illustrated in Fig. 2(a). Since $u'v'$ is long, Lemma 1 gives $|pu'| < |u'v'| \leqslant |pv'|$, and ordering by length gives $|u'v'| \leqslant |uv|$. But uv was chosen as the EMST edge joining T_u and T_v, and the shorter edge pu' was not; therefore u' must be in T_u with p, and v' too since $u'v'$ is an edge of T.

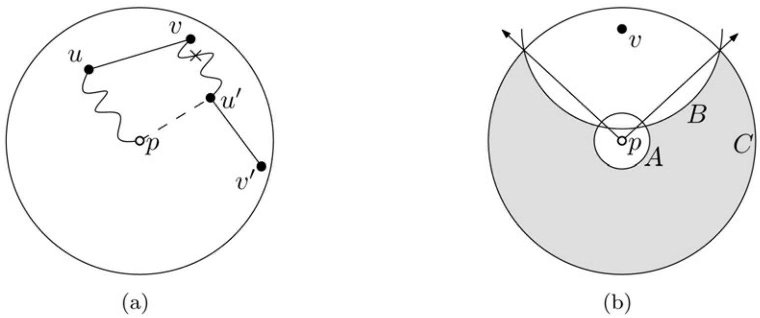

Fig. 2. Illustrating arguments used to show there are few *far* edges in Lemma 3. (a) Removing a long edge uv cannot disconnect another long edge from p, since EMST edge uv is longer than pu'. (b) Using definitions in the text, all far vertices of F lie inside circle C and outside circles A and B, giving an empty sector of angle at least $2\pi/9$ viewed from p.

Next, we show that v indicates a sector of the plane as seen from p that contains no other second endpoints of edges of F—no other *far vertices* of F. By definition, we know that the circle A of radius $L+2$ around p contains no such vertices, and by the previous paragraph we know that the circle B of radius $|uv|$ around v contains no such vertices, since otherwise there would be a shorter possible edge than uv to connect T_u and T_v. Now consider the farthest vertex v' among all vertices in F, so all remaining vertices are inside a circle C of radius $|pv'|$ around p. This vertex must be part of an edge $u'v'$ of length at least $|pv'|-2$, otherwise it would not be in E_p. Therefore, also $|uv| \geqslant |u'v'| \geqslant |pv'|-2$. Now, all remaining far vertices of F must be in the region $C\backslash(A \cup B)$, see Fig. 2(b).

To define the free sector, consider now the angle that pv makes with the intersections between A and B, and the angle it makes with the intersections between B and C. The smaller of those two angles bounds the sector.

Thus, we consider triangles of side lengths $L+2$, $|pv|$, and $|uv|$ and of $|pv|$, $|pv|$, and $|uv|$. We know that $|pv| < |uv|+2$ and $|uv| > L$. The angle at p is minimized as $|pv|$ approaches $L+2$, which would give, in both cases, the isosceles triangle with angle

$$2\arcsin\left(\frac{L/2}{L+2}\right) > 0.7494 > 2\pi/9.$$

Thus, inside the empty circle around v we find sectors of angle $> \pi/4$ on either side of $\overrightarrow{pv}$; each sector contains no far points closer to p than v. At most two empty sectors can overlap—one from the clockwise (CW) and one from the counter-clockwise (CCW) direction around p, which implies that there are at most 8 far edges. □

We can summarize:

Theorem 1. *Let $T=(P,E)$ be the Euclidean minimum spanning tree on the points P. The total number of these points in the expanded circles for all edges is linear in n. That is,*

$$\sum_{uv\in E} \left|C^+_{uv} \cap P\right| = O(n).$$

We can extend the proof of Lemma 3 to bound the number of far edges for an arbitrary point p in the plane, albeit with a large (and overly-pessimistic) constant factor. Since this bound is used only to shorten the description of preprocessing, and not for the algorithm itself, we have not tried to minimize the constant. The next lemma implies that the arrangement of all expanded Gabriel circles has linear complexity.

Lemma 4. *For any point p in the plane $|E_p|$ is constant.*

Proof. The disk packing argument in Lemma 2 shows that there are at most 71 disk centers within distance $L+2$ of any point p. As these are vertices in a Euclidean minimum spanning tree (EMST), for which each vertex has degree at most 6, at most 426 edges of E_p can have a vertex within $L+2$ of p.

We therefore consider only the subset $F \in E_p$ of far edges for p that have both endpoints farther than $L+2$ from p. We show that the edges of F can be organized into a binary tree whose maximum depth is 8 by the angle packing argument used in Lemma 3. Since such a binary tree has at most $2^9-1=511$ nodes, F has at most $511+426=937$ edges.

We build this tree from the root at depth 0. Each node ν is associated with a subset of edges, $F_\nu \subset F$, as well as an edge of F_ν. The root is associated with F, and an edge uv having one endpoint v farthest from p. Removing uv from the EMST partitions the remaining edges of F into two groups, F_u and F_v, such that the edges in F_u are in the same component as u of the EMST after removing uv, and the edges in F_v are in the same component as v. where the first remain connected

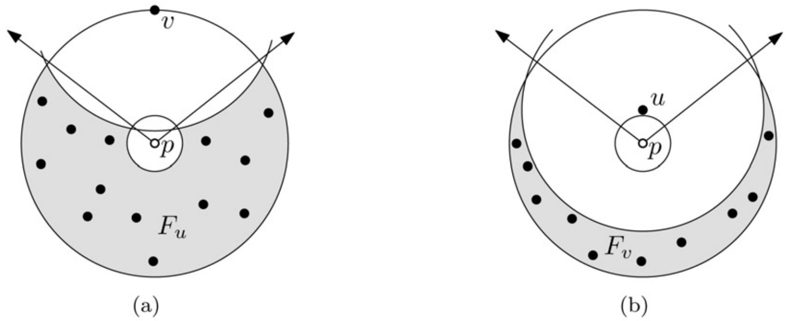

Fig. 3. (a) The point in the plane p and the furthest endpoint v ensure that all endpoints of F_u lie within the shaded area. (b) The point u can be closer to p, but there is a circle with radius $|uv|$ around it that contains no endpoints of F_v.

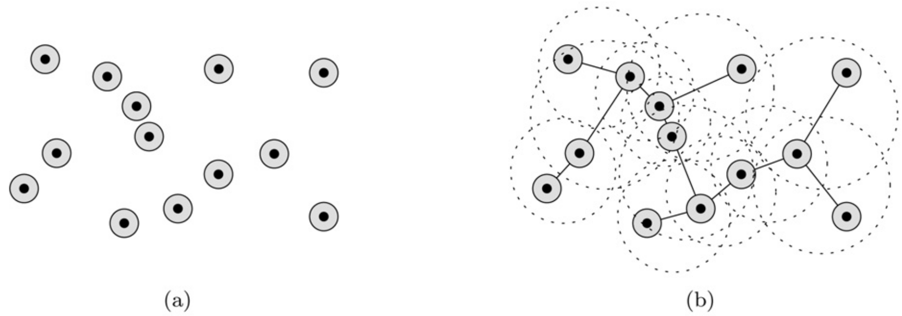

Fig. 4. (a) A set of imprecise points. (b) The edges of a minimum spanning tree, and the disks that intersect their expanded Gabriel circles.

to u and the second connected to v by Lemma 1. These are the edge sets associated with the children of the root. (In determining connectedness, we include EMST edges and vertices within $L+2$ of p, even though they are not in $F \setminus \{uv\}$.)

In general, at node v, the edges F_v are edges of a connected component of the EMST minus the edges associated with the ancestors of v, the associated edge $uv \in F_v$ is chosen so that the endpoint v is farthest from p, and removing uv partitions the edges of the EMST component into F_u and F_v.

For the edges in F_u, we know that no endpoint can lie within a circle of radius $|uv|$ centered at v. We also know that all endpoints lie within a circle of radius $|pv|$ centered at p, and that none lie within $L+2$ of p. These constraints on F_u are depicted in Fig. 3(a). As in the proof of Lemma 3, there is a sector with angle greater than $2\pi/9$, as seen from p, that contains no endpoints from F_u.

The other endpoint of uv can lie closer to p, as shown in Fig. 3(b). Still, in this case there is also a sector with an angle of at least $2\pi/9$ that contains no endpoints of F_v. The point u must still be outside the circle of radius $L+2$ around p, and the circle of radius uv around u intersects the circles around p of radii $L+2$ and pv according to the same restrictions as in the case of v.

This implies that the tree has depth at most 8, and completes the proof of the lemma. □

4. Delaunay computation

Let $\mathcal{R}$ be a set of n disjoint unit disks in the plane that represent the imprecise regions for P, which are the disk center points. Section 4.1 details how to preprocess $\mathcal{R}$ in $O(n \log n)$ time into a linear-size data structure $H(\mathcal{R})$. Section 4.2 shows that given an exact sample $\hat{P}$ consisting of a point inside each disk of $\mathcal{R}$, we can compute the Delaunay triangulation of $\hat{P}$ in linear time using $H(\mathcal{R})$.

4.1. Preprocessing

Let P be the set of center points of the n disjoint unit disks of $\mathcal{R}$. For $H(\mathcal{R})$, we compute a Euclidean minimum spanning tree of P, a list of its edges sorted by increasing length, and for each edge uv the list of points of P that fall inside the expanded Gabriel circle C^+_{uv}. Fig. 4 shows an example.

By Theorem 1 we know that each point of P can fall into at most a constant number of expanded Gabriel circles. Thus, the total size of $H(\mathcal{R})$ is linear.

A minimum spanning tree is easy to compute in $O(n \log n)$ time, since the Delaunay triangulation is a linear-size set of edges that contains all candidates. The sorted list of EMST edges is even easier. Finally, a simple sweep of the arrangement

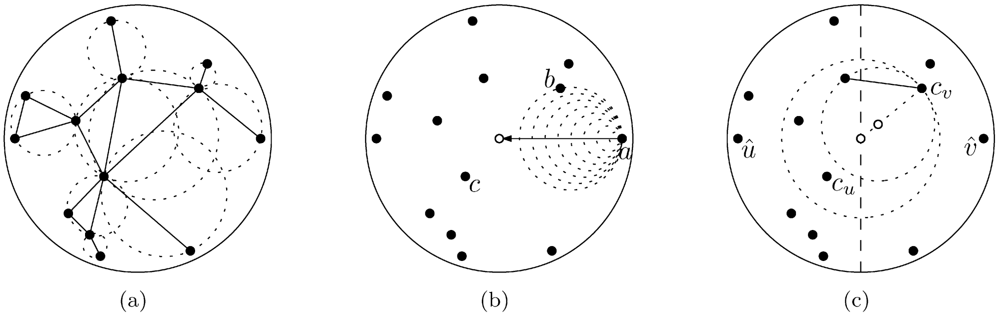

Fig. 5. (a) The Delaunay edges certified by (dotted) empty circles within a bigger circle form a connected graph. (b) Growing a circle from p towards the center. (c) The closest point to the center can be connected to at least one of the points in the other group.

of the expanded Gabriel circles of the EMST edges and the points P can locate all points in their circles; because Lemma 4 says that this arrangement has linear size, the sweep can be carried out in $O(n \log n)$ time.

Lemma 5. *Preprocessing the n disjoint unit disks $\mathcal{R}$ produces a linear size data structure $H(\mathcal{R})$ in $O(n \log n)$ time.*

Denote the list of EMST edges, sorted by increasing length, by $e_1, \ldots, e_{n-1}$. We define notation for the connected components of the graph consisting of the first k edges of this list: Let $\mathcal{I}_k$ be the partition of the index set $\{1, \ldots, n\}$ induced by the connected components of these first k edges: that is, $i, j \in I$ for some $I \in \mathcal{I}_k$ if and only if p_i and p_j can be joined by edges from $\{e_1, \ldots, e_k\}$. We can associate these connected components with $H(\mathcal{R})$ (conceptually, not computationally, as they are needed only for a proof), because our algorithm creates the components (or supersets of them) for points $\hat{P} = \{\hat{p}_1, \ldots, \hat{p}_n\}$ drawn from each disk in $\mathcal{R}$.

4.2. Computing the Delaunay triangulation

Now, given an exact sample $\hat{P} = \{\hat{p}_1, \ldots, \hat{p}_n\}$ of $\mathcal{R}$, and the data structure $H(\mathcal{R})$, we show how to compute in linear time a connected subgraph of the Delaunay triangulation of $\hat{P}$. Chin and Wang's algorithm [10] then completes the Delaunay triangulation of $\hat{P}$ in linear time.

In order to construct such a connected subgraph, we process the edges of the EMST of P by increasing length. For each such edge e, we find a path in the Delaunay triangulation that connects the same components that e connects in the graph composed of all EMST edges shorter than e. We begin by making an observation, illustrated in Fig. 5(a), on the portion of a Delaunay triangulation bounded by a circle.

Lemma 6. *Let P be a set of points in general position in the plane, C be a circle that encloses a subset $Q = P \cap \operatorname{int}(C)$, and E be the set of Delaunay edges of P that have empty circles contained inside $\operatorname{int}(C)$. The graph (Q, E) is connected.*

Proof. Let c be the point of Q closest to the center of C; we show that any vertex $p \in Q$ is connected to c. Initially, let $a = p$, and, as depicted in Fig. 5(b), grow a circle from a towards the center of C, keeping a pinned on the boundary; stop when the circle hits any point $b \in Q$. The edge ab is discovered to be a Delaunay edge in E, and the point b is closer to the center of C than a was. Since P is finite, by setting $a = b$ and repeating this procedure, we eventually construct a path from p to c in the graph (Q, E). $\square$

Suppose now that EMST edge e_k joins $u, v \in P$, and consider the expanded Gabriel circle C^+_{uv}. Lemma 6 says there exists a path of Delaunay edges certified inside C^+_{uv} that joins the corresponding exact samples $\hat{u}, \hat{v} \in \hat{P}$; our task is to compute one efficiently, or at least to compute a subgraph of the Delaunay triangulation of $\hat{P}$ that contains one or more paths.

Theorem 2. *Given n disjoint unit disks $\mathcal{R}$, and the structure $H(\mathcal{R})$ from Lemma 5, the Delaunay triangulation of an exact sample $\hat{P}$ chosen from these disks can be computed in $O(n)$ time.*

Proof. To reconstruct the Delaunay triangulation, we first want to build up the components of the EMST by adding edges in order; the essential task is to find a path of Delaunay edges joining the exact samples $\hat{u}, \hat{v} \in \hat{P}$ for two centers $u, v \in P$ that form an edge e_k in the EMST. We will do this in C^+_{uv}, although we could do it in the smaller $C_{\hat{u}\hat{v}}$ with a slightly longer description of the procedure.

Let $Q \subset \hat{P}$ denote the points inside circle C^+_{uv} and $\mathcal{K} \subset \mathcal{R}$ denote the unit disks with centers inside C^+_{uv}. When e_k is short, penny packing says there are at most a constant number of disks in $\mathcal{K}$, so we can process e_k by computing the Delaunay triangulation of the points Q and discarding edges that are not certified by an empty circle inside C^+_{uv}.

When e_k is long, Lemma 1 says that there are two components that are separated by the perpendicular bisector of uv. Let Q_u and Q_v be the partition of Q by this bisector. It suffices to find a Delaunay edge of $\hat{P}$ from $Q_u \times Q_v$ since the points within Q_u (and within Q_v) have already been connected earlier in the algorithm.

Let $c_u \in Q_u$ and $c_v \in Q_v$ be the closest points to the center of C^+_{uv}, as illustrated in Fig. 5(c), and assume that the distance to c_v is greater, meaning the circle concentric with C^+_{uv} through c_v contains at least one point of Q_u. Shrink this circle with c_v on the boundary by moving its center toward c_v until the last point of Q_u leaves its interior—this point defines the desired Delaunay edge with c_v. Both steps can be carried out in time proportional to $|Q|$.

We spend constant time with each short edge and, by Theorem 1, a total of linear time with the long edges. For each edge we find a path of Delaunay edges of $\hat{P}$ that joins the vertices $\hat{u}$ and $\hat{v}$, so the connected components induced by the sequence of edges found will be supersets of the components of $\mathcal{I}_k$ of the first k edges of the EMST of $\hat{P}$. Thus, we obtain a connected graph after processing all EMST edges, and can invoke Chin and Wang [10] to complete the Delaunay triangulation. □

5. Extensions

Our algorithm works for a very specific class of imprecise regions: disjoint disks of equal radius. In practice, this may be a rather strong assumption. In this section, we show how to extend the result to less restricted regions.

5.1. Overlapping disks

If we allow the regions to be arbitrarily overlapping disks, then there is little we can hope to prove. In the worst case, all disks could coincide, allowing the constructions that establish the $\Omega(n \log n)$ lower bounds for general Delaunay triangulation [25]. If we limit the depth of overlap, however, our result still holds with the algorithm unchanged.

We say a set of disks is *k-overlapping* if no point in the plane is contained in more than k disks. In this case, the number of short edges that can contain a point p increases. Clearly, there cannot be more than $k(r+2)^2$ disks touching a circle of radius r. This means the constant grows linearly in k. The arguments involving long edges do not depend on the disjointness of the disks.

5.2. Other extensions

If we allow the disks to have different radii, then in general the problem is open. However, when there is a constant fraction $c = \frac{R}{r}$ between the largest radius R and the smallest radius r, then we can just increase radii until all disks have radius R. Since we know that the sample points lie inside the input disks, they certainly also lie in the grown disks. Of course the disks start overlapping, but not too much: at most $(c+1)^2$ grown disks contain any given point in the plane.

If the input regions are not disks but squares, then we can grow them to the smallest disks containing them, which are 3-overlapping. If the regions are *fat* in the sense that they contain circles of radius r but are contained in circles of radius R (the same radii for all regions), with $c = \frac{R}{r}$, then we can again replace them by disks of radius R that are at most $(c+1)^2$-overlapping.

Finally, we can also handle combinations of the above (partially overlapping fat regions of restricted different radii) at the expense of an increased constant in the time bound.

6. Discussion

Our result proves that imprecise information about point coordinates has value: after $O(n \log n)$ time spent preprocessing the n regions of imprecision for the points, we can obtain a Delaunay triangulation of exact points sampled from the imprecise regions in linear time.

In our solution, we collect enough structure of the output Delaunay triangulation to obtain a connected subgraph. The Delaunay triangulation can then be completed in linear time by Chazelle's and Chin and Wang's algorithms; however, these algorithms are complicated and make the result not useable in practice. Roughly speaking, the reason why these algorithms are complicated is that no information is known about other vertices "near" a given vertex: in order to (Delaunay) triangulate a polygon in linear time, you have to discover vertices to connect to, that can be far away along the polygon boundary, without spending too much time looking for them. Since our points lie in unit disks, and we are allowed to preprocess them, it is not unlikely that we could compute more structure than just a connected graph, and use this to considerably simplify the algorithms in the last step, and make the algorithm more practical.

This work is one of the first studies about preprocessing imprecise points to allow faster computation of well-known geometric structures, any many directions for further research are possible. We modeled our points as circular regions, and have considered only a couple of the many possible extensions. It may be interesting to study the problem for more general regions, or more restricted regions, or in the presence of correlated imprecision. On the other hand, the question naturally extends to geometric structures other than triangulations, such as spanning trees, planar tours, or geometric matchings, to name just a few. Finally, it would be interesting to know whether the ideas of this paper can be extended to higher dimensions.

Acknowledgements

We thank Marc van Kreveld for the question and discussions and for arranging that the first author could visit UNC Chapel Hill. This research was partially funded by grants from NSF and NGA/Darpa, and by the Netherlands Organization for Scientific Research (NWO) through the project GOGO.

References

[1] M. Abellanas, F. Hurtado, P.A. Ramos, Structural tolerance and Delaunay triangulation, Inf. Process. Lett. 71 (5–6) (1999) 221–227.
[2] A. Aggarwal, L.J. Guibas, J. Saxe, P.W. Shor, A linear-time algorithm for computing the Voronoi diagram of a convex polygon, Discrete Comput. Geom. 4 (6) (1989) 591–604.
[3] D. Bandyopadhyay, J. Snoeyink, Almost-Delaunay simplices: Nearest neighbor relations for imprecise points, in: ACM–SIAM Symp. on Discrete Algorithms, 2004, pp. 403–412.
[4] D. Bandyopadhyay, J. Snoeyink, Almost-Delaunay simplices: Nearest neighbor relations for imprecise 3D points using CGAL, Comput. Geom. 38 (1–2) (2007) 4–15.
[5] K. Bühler, E. Dyllong, W. Luther, Reliable distance and intersection computation using finite precision geometry, in: Numerical Software with Result Verification, in: LNCS, vol. 2991, Springer Verlag, 2004, pp. 160–190.
[6] J. Cartigny, F. Ingelrest, D. Simplot-Ryl, I. Stojmenovic, Localized LMST and RNG based minimum-energy broadcast protocols in ad hoc networks, Ad Hoc Networks 3 (1) (2005) 1–16.
[7] CGAL, Computational Geometry Algorithms Library, http://www.cgal.org.
[8] B. Chazelle, Triangulating a simple polygon in linear time, Discrete Comput. Geom. 6 (5) (1991) 485–524.
[9] F. Chin, J. Snoeyink, C.A. Wang, Finding the medial axis of a simple polygon in linear time, Discrete Comput. Geom. 21 (3) (1999) 405–420.
[10] F.Y.L. Chin, C.A. Wang, Finding the constrained Delaunay triangulation and constrained Voronoi diagram of a simple polygon in linear time, SIAM J. Comput. 28 (2) (1998) 471–486.
[11] H. Edelsbrunner, E.P. Mücke, Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms, ACM Trans. Graph. 9 (1) (1990) 66–104.
[12] J.S. Ely, A.P. Leclerc, Correct Delaunay triangulation in the presence of inexact inputs and arithmetic, Reliable Computing 6 (2000) 23–38.
[13] S. Fortune, Numerical stability of algorithms for 2-d Delaunay triangulations, Internat. J. Comput. Geom. Appl. 5 (1–2) (1995) 193–213.
[14] K.R. Gabriel, R.R. Sokal, A new statistical approach to geographic variation analysis, Systematic Zoology 18 (1969) 259–278.
[15] M.T. Goodrich, J.S.B. Mitchell, M.W. Orletsky, Practical methods for approximate geometric pattern matching under rigid motion, in: Proc. 10th Annu. ACM Sympos. Comput. Geom., 1994, pp. 103–112.
[16] R. Graham, B. Lubachevsky, K. Nurmela, P. Östergård, Dense packings of congruent circles in a circle, Disc. Math. 181 (1998) 139–154.
[17] L.J. Guibas, D. Salesin, J. Stolfi, Epsilon geometry: building robust algorithms from imprecise computations, in: Proc. 5th Annu. ACM Sympos. Comput. Geom., 1989, pp. 208–217.
[18] L.J. Guibas, D. Salesin, J. Stolfi, Constructing strongly convex approximate hulls with inaccurate primitives, Algorithmica 9 (1993) 534–560.
[19] M. Held, J.S.B. Mitchell, Triangulating input-constrained planar point sets, Inf. Process. Lett. 109 (1) (2008) 54–56.
[20] A.A. Khanban, Basic algorithms of computational geometry with imprecise input, PhD thesis, Imperial College, London, 2005.
[21] A.A. Khanban, A. Edalat, Computing Delaunay triangulation with imprecise input data, in: Proc. 15th Canad. Conf. Comput. Geom., 2003, pp. 94–97.
[22] A.A. Khanban, A. Edalat, A. Lieutier, Computability of partial Delaunay triangulation and Voronoi diagram, in: V. Brattka, M. Schröder, K. Weihrauch (Eds.), Electronic Notes in Theoretical Computer Science, vol. 66, Elsevier, 2002.
[23] M. Löffler, M. van Kreveld, Largest and smallest convex hulls for imprecise points, Algorithmica (2008), doi:10.1007/s00453-008-9174-2.
[24] K. Mehlhorn, S. Näher, LEDA: A Platform for Combinatorial and Geometric Computing, Cambridge University Press, Cambridge, UK, 2000.
[25] R. Seidel, A method for proving lower bounds for certain geometric problems, in: G.T. Toussaint (Ed.), Computational Geometry, North-Holland, Amsterdam, Netherlands, 1985, pp. 319–334.
[26] J.R. Shewchuk, Delaunay refinement algorithms for triangular mesh generation, Comput. Geom. 22 (1–3) (2002) 21–74.
[27] E. Specht, The best known packings of equal circles in the unit circle (up to $n = 500$), http://hydra.nat.uni-magdeburg.de/packing/cci/cci.html, July 2007.
[28] K. Sugihara, M. Iri, Two design principles of geometric algorithms in finite-precision arithmetic, Appl. Math. Lett. 2 (2) (1989) 203–206.
[29] K. Sugihara, M. Iri, Construction of the Voronoi diagram for 'one million' generators in single-precision arithmetic, Proc. IEEE 80 (9) (1992) 1471–1484.
[30] M. van Kreveld, M. Löffler, Largest bounding box, smallest diameter, and related problems on imprecise points, in: Proc. 10th Workshop on Algorithms and Data Structures, in: LNCS, vol. 4619, 2007, pp. 447–458.
[31] M. van Kreveld, M. Löffler, J.S.B. Mitchell, Preprocessing imprecise points and splitting triangulations, in: Proc. 19th International Symposium on Algorithms and Computation, 2008, pp. 544–555, http://dx.doi.org/10.1007/978-3-540-92182-0_49.
[32] F. Weller, Stability of Voronoi neighborship under perturbations of the sites, in: Proc. 9th Canad. Conf. Comput. Geom., 1997, pp. 251–256.
[33] C. Yap, Robust geometric computation, in: J.E. Goodman, J. O'Rourke (Eds.), Handbook of Discrete and Computational Geometry, second ed., CRC Press, 2004, pp. 927–952.