A QUADRATIC TIME ALGORITHM FOR THE MINMAX LENGTH TRIANGULATION

HERBERT EDELSBRUNNER* AND TIOW SENG TAN†

Abstract. We show that a triangulation of a set of n points in the plane that minimizes the maximum edge length can be computed in time $O(n^2)$. The algorithm is reasonably easy to implement and is based on the theorem that there is a triangulation with minmax edge length that contains the relative neighborhood graph of the points as a subgraph. With minor modifications the algorithm works for arbitrary normed metrics.

Key words. Computational geometry, point sets, triangulations, two dimensions, minmax edge length, normed metrics

AMS(MOS) subject classifications. 68U05, 68Q25, 65D05

Appear in: *SIAM Journal on Computing*, **22 (3)**, 527–551, (1993)

1. Introduction. A *triangulation* of a (finite) point set S in $\Re^2$ is a maximally connected straight line plane graph whose vertices are the points of S. Maximality implies that with the exception of the unbounded face each face of the graph is a triangle. The number of different triangulations of S depends on $n = |S|$ as well as the relative location of the points. As implied by a result in [ACNS82], 10^{13n} is an upper bound on the number of triangulations of any set of n points in $\Re^2$. Furthermore, if S is in convex position then it admits $\frac{1}{n-1}\binom{2n-4}{n-2} \geq 2^{n-3}$ different triangulations. In order to choose an optimal triangulation, under some criterion, it is thus not feasible to exhaustively search the set of all triangulations.

Indeed, except for a handful of particular optimality criteria, the problem of finding an optimal triangulation for a given point set is hard, that is, no polynomial time algorithms are known. Among these exceptions are the maxmin angle criterion [Sibs78], the minmax angle criterion [ETW92], the minmax smallest enclosing circle criterion [Raja91], and the minmax circumscribed circle criterion. The optimum under the first, third and fourth criterion is achieved by the Delaunay triangulation which can be constructed in time $O(n \log n)$ [Del34], [PrSh85], [Edel87].

In this paper we study the complexity of minimizing the maximum edge length. A triangulation that minimizes the length of its longest edge is called a *minmax length triangulation*. It is related to the so-called *minimum length* (or *minimum weight*) *triangulation* that minimizes the sum of the edge lengths. The latter problem has been studied by Plaisted and Hong [PlHo87], Lingas [Ling87], and others. In spite of the lack of a proof that the problem is NP-hard, no polynomial time algorithm for constructing a minimum length triangulation is currently known. Even more annoying is the lack of a constant approximation scheme, that is, an algorithm that in polynomial time constructs a triangulation guaranteed to have total edge length at most some constant times the optimum. The currently best approximation scheme is described in [PlHo87] and guarantees a factor of $O(\log n)$.

---

* Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801. The work of this author was supported by the National Science Foundation grants CCR-8921421.

† Department of Information Systems and Computer Science, National University of Singapore, Republic of Singapore. The work of this author was done during his study leave at the University of Illinois at Urbana-Champaign.

In view of the apparent difficulty to compute minimum length triangulations, it is somewhat surprising that we are able to give a polynomial, in fact quadratic time algorithm for constructing a minmax length triangulation. It is the first polynomial time algorithm for this problem. There is evidence for the potential usefulness of such a triangulation (see [BrZl70], [WGS90]). Still, the authors of this paper consider the additional insight into optimum triangulations under edge length criteria as the main contribution of this paper.

The reader might find it instructive to rule out seemingly promising approaches to computing minmax length triangulations before diving into the occasionally involved developments of the forthcoming sections. Note first that the Delaunay triangulation does not minimize the maximum edge length (see also Section 2). Second, the incremental greedy method, that repeatedly adds the shortest edge that does not intersect any previously added edge, also fails to minimize the maximum edge length. Third, let us take a brief look at the decremental greedy method that throws away edges in the order of decreasing length. It stops the deletion process if another deletion would render the set of edges so that it does not contain any triangulating subset (see Wismath [Wism80, page 81]). The trouble with this approach is that it is not clear how to efficiently decide whether the evolving edge set is still sufficient to triangulate the point set. Indeed, Lloyd [Lloy77] proves that the general version of this problem (decide whether a given edge set contains a triangulation) is NP-hard. Finally, the iterative methods that use the edge-flip [Laws77] or the more general edge-insertion operation [ETW92] can get caught in local optima. The approach taken in this paper is entirely different from the above paradigms.

The organization of this paper is as follows. Section 2 reviews a few results on relative neighborhood graphs and other subgraphs of the Delaunay triangulation. Section 3 formulates the global algorithm; its straightforward implementation using dynamic programming takes time $O(n^3)$. The only intricate part of this algorithm is the proof of correctness provided in Section 4. Sections 5 and 6 present a specialized polygon triangulation algorithm that can be used to speed up the general algorithm to time $O(n^2)$. While Sections 2 through 6 assume that the Euclidean metric is used to measure length, Section 7 demonstrates that all results extend to general normed metrics. Indeed, the arguments in Sections 2 through 6 are axiomatically derived from a few basic lemmas in order to minimize the number of changes necessary to generalize the results. Finally, Section 8 briefly discusses the contributions of this paper and states some related open problems.

2. Subgraphs of the Delaunay Triangulation. The approach to constructing a minmax length triangulation taken in this paper first adds enough edges to decompose the plane into simple polygonal regions and then (optimally) triangulates these regions. Both Plaisted and Hong [PlHo87] and Lingas [Ling87] used this approach to compute approximations of the minimum length triangulation. In our case, the initial set of edges is provided by the (boundary of the) convex hull and the relative neighborhood graph of the point set S. The remainder of this section formally introduces these graphs, along with the Delaunay triangulation and the minimum spanning tree of S, and reviews some basic facts about their relationships. If x, y, z are three points in $\Re^2$ then xy denotes the relatively open line segment with endpoints x and y, $|xy|$ is its length, and xyz denotes the open triangle with vertices x, y, z.

The *Delaunay triangulation* of S, denoted by $dt(S)$, contains an edge ab, $a, b \in S$, if there is a circle through a and b so that all other points lie outside the circle. If the points are in general position then $dt(S)$ is indeed a triangulation.

As mentioned in the introduction, the Delaunay triangulation does not minimize the length of the longest edge. Take for example the points $a = (-2, 0)$, $b = (1, \sqrt{3})$, $c = (1, -\sqrt{3})$, $d = (2 - \epsilon, 0)$, with $0 < \epsilon < 1$. They form a convex quadrilateral $abdc$ and the Delaunay triangulation uses ad as the fifth edge. As ϵ approaches 0 the length of ad approaches $\frac{2}{\sqrt{3}}$ times the length of the longest edge in the alternative triangulation. Indeed, $\frac{2}{\sqrt{3}}$ is the worst possible ratio as can be shown using the result of [Raja91] that the Delaunay triangulation minimizes the radius of the maximum smallest enclosing circle, where the maximum is taken over all triangles. If the radius of this circle is 1 then the longest edge of the Delaunay triangulation has length at most 2. By the optimality result every minmax length triangulation has a smallest enclosing circle of radius at least 1 and therefore an edge of length at least $\sqrt{3}$ (see also [WGS90]).

The *convex hull* of S is the smallest convex polygon that contains S. We define $ch(S)$ as the graph defined by the edges of this polygon. In the (degenerate) case where three or more collinear points lie on the boundary of this polygon we think of each such point as a vertex of the polygon. Thus, edges are taken only between adjacent collinear points. Each convex hull edge is an edge of *every* triangulation of S, and therefore also of every minmax length triangulation.

An edge ab belongs to the *relative neighborhood graph* of S, denoted by $rng(S)$, if

$$|ab| \leq \min_{x \in S - \{a,b\}} \max\{|xa|, |xb|\}.$$

This definition goes back to Toussaint [Tous80] who modified a similar definition by Lankford [Lank69] for use in pattern recognition. Alternatively, we can define the *lune* of ab as the set $\{x \in \Re^2 : \max\{|xa|, |xb|\} < |ab|\}$, and define $rng(S)$ as the set of edges ab whose lunes have empty intersection with S.

A *minimum spanning tree* of S, $mst(S)$, is a spanning tree of S that minimizes the total edge length; it also minimizes the maximum edge length.

All four graphs, $dt(S), ch(S), rng(S), mst(S)$, are plane and connected, and with the exception of $ch(S)$, they span S. Where convenient we will interpret these graphs as edge sets. Plainly, $ch(S) \subseteq dt(S)$, and as observed by Toussaint [Tous80], we also have $mst(S) \subseteq rng(S) \subseteq dt(S)$. Obviously, $ch(S) \subseteq mlt(S)$, for every minmax length triangulation $mlt(S)$, and we will show in Section 4 that there exists an $mlt(S)$ so that $rng(S) \subseteq mlt(S)$.

3. The Global Algorithm. As mentioned above there exists a minmax length triangulation $mlt(S)$ that contains all edges of $ch(S)$ and $rng(S)$. Because $ch(S) \cup rng(S)$ is a connected graph, it decomposes the convex hull of S into simple *polygonal regions*, which we define as open sets, that contain no points of S. It is thus natural to construct $mlt(S)$ by computing $ch(S) \cup rng(S)$ and then (optimally) triangulating each polygonal region.

Strictly speaking, however, the polygonal regions are not necessarily simple polygons in the usual sense of the term, although their interiors *are* simply connected. The difference is that the interior of the closure of a polygonal region is not necessarily the same as the region itself; it may contain edges of the region and it may be non-simply connected. The most effective way to deal computationally with this minor difficulty is to represent each edge by a pair of oppositely directed edges, and to represent the boundary of each region by the collection of directed edges for which the region lies on their left hand side. In effect, this means that we interpret each polygonal region as a genuine simple polygon, simply by pretending that its zero-width cracks are opened up a tiny amount. In most cases, this is a convenient interpretation and the notation will be adjusted accordingly. Only occasionally, the difference between a simple polygonal region and a simple polygon will be uncovered.

Let us now formally specify the algorithm and give a preliminary analysis.

Input. A set S of n points in $\Re^2$.

Output. A minmax length triangulation of S.

Algorithm. 1. Construct $ch(S)$ and $rng(S)$.
2. Determine the polygonal regions defined by $ch(S) \cup rng(S)$.
3. Find a minmax length triangulation for each such polygonal region.

Step 1 can be carried out in time $O(n \log n)$ using results documented in [PrSh85] and [Supo83] (see also [JKY90]). Using the standard quad-edge data structure of [GuSt85] for storing the plane graph $ch(S) \cup rng(S)$, step 2 can be accomplished in time $O(n)$. Finally, we can use dynamic programming to compute an optimal triangulation for each polygon in time cubic and storage quadratic in the number of its vertices (see [Klin80], [Gilb79]). This adds up to time $O(n^3)$ and storage $O(n^2)$. The correctness of the algorithm will be established in the next section. Sections 5 and 6 will show how to speed up the algorithm to time $O(n^2)$ using a specialized polygon triangulation algorithm.

4. The Subgraph Theorem. The main result of this section is what we call the Subgraph Theorem which was announced earlier. We begin with two elementary geometry lemmas about distances between four points in convex and in non-convex position.

$\square$-LEMMA. *For a convex quadrilateral abcd, we have* $|ab| + |cd| < |ac| + |bd|$.

Proof. Let x be the intersection point of the two diagonals, ac and bd. Clearly, $|ab| + |cd| < (|ax| + |xb|) + (|cx| + |xd|) = |ac| + |bd|$. $\square$

In words, the total length of the two diagonals of a convex quadrilateral always exceeds the total length of two opposite sides. This is true even if three of the four vertices are collinear. It implies that if one diagonal is no longer than one of the edges then the other diagonal is longer than the opposite edge.

Δ-LEMMA. *Let a, b, c, d be four distinct points so that the closure of the triangle abc contains d. Then* $|ad| < \max\{|ab|, |ac|\}$.

Proof. If a, b, c, d are collinear the result is obvious. Otherwise, let d' be the intersection of the edge bc with the line passing through a and d, and note that $|ad| \leq |ad'|$. Of all points on bc only the endpoints can possibly maximize the distance to a. The assertion follows because if d' is an endpoint of bc then $d \neq d'$ and therefore ad is strictly shorter than ad'. $\square$

Note that the length of the longest edge of any minimum spanning tree is no longer than the longest edge of any triangulation of S. This follows trivially from the fact that every triangulation contains a spanning tree. It is not very difficult to prove that the same is true for the relative neighborhood graph of S. First we need some notation. The circle with center x and radius ρ is denoted by (x, ρ), and the *bisector* of two points p and q is the set of points equidistant to both.

LENGTH LEMMA. *Every triangulation of S contains an edge that is at least as long as the longest edge of $rng(S)$.*

Proof. Let pq be the longest edge of $rng(S)$ and let $t(S)$ be an arbitrary triangulation of S. If $pq \in t(S)$ there is nothing to prove. Otherwise, pq intersects edges $r_1s_1, r_2s_2, \ldots, r_ks_k$ of $t(S)$, sorted from p to q, with all r_i on one side of the line through p and q and all s_i on the other. If pq is longer than all edges in $t(S)$ then r_1 and s_1 are both inside the circle $C_p = (p, |pq|)$, because pr_1 and ps_1 are both edges of $t(S)$. By the definition of $rng(S)$, r_1 and s_1 are thus outside or on the circle $C_q = (q, |pq|)$. Therefore, r_1 and s_1 lie in the half-plane of points closer to p than to q. Symmetrically, r_k and s_k lie inside C_q and outside or on C_p and therefore in the half-plane of points closer to q than to p. For each $1 \leq i \leq k-1$ we have either $r_i = r_{i+1}$ or $s_i = s_{i+1}$, which implies that there is an index j so that r_j and s_j do not lie on the same side of the bisector of pq. But then the $\Box$-Lemma implies that $|r_js_j| > |pq|$, because $|pq|$ is no longer than each of two opposite edges of the convex quadrilateral pr_jqs_j, a contradiction. $\Box$

The proof of the Subgraph Theorem is similar to that of the Length Lemma, although considerably more involved. The basic idea is to assume an extreme counterexample and to contradict its existence by retriangulating parts of it using no long edges. In the following, we first develop three facts showing the possibilities of retriangulations, and then prove the theorem.

Let $t(S)$ be a minmax length triangulation of S that does not contain some edge pq of $rng(S)$. Suppose pq intersects the triangles $t_1, t_2, \ldots, t_k$ of $t(S)$, sorted from p to q (see Fig. 1 left). The deletion of the edges that intersect pq would result in a simply

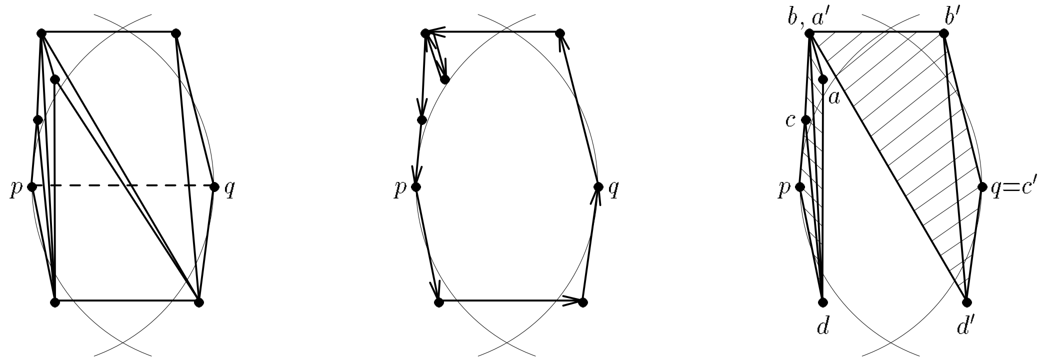

FIG. 1. *To the left we see the triangles of $t(S)$ that intersect pq. If we remove the edges intersecting pq we get a polygon whose boundary is oriented in a counterclockwise order. The prefix P and the suffix Q defined for this configuration are illustrated to the right. Although b and a' are the same point, they refer to different angles of this point.*

connected region, which can be interpreted (as in Section 3) as a polygonal region — we treat each edge in its boundary as a pair of edges with opposite direction, and to trace the boundary of the region we traverse all directed edges that have the region on their left side. Any two consecutive (directed) edges define an *angle* (see Fig. 1 middle). Note that a vertex can correspond to many angles, although the common situation is that it corresponds only to one. We will therefore sometimes ignore the difference between vertices and corresponding angles. Points p and q correspond to only one angle each. An angle is *convex* if the two defining edges form a left-turn. Call the sequence of edges from p to q the *lower chain* and the sequence from q to p the *upper chain*. Each chain contains at least one convex angle different from p and q.

A *prefix* is an initial subsequence of $t_1, t_2, \ldots, t_k$, and a *suffix* is a terminal subsequence of $t_1, t_2, \ldots, t_k$. We say that a prefix (suffix) *covers* an angle of the polygon if it contains all triangles incident to this angle. Let i be minimal so that the prefix $P = t_1, t_2, \ldots, t_i$ covers a convex angle other than p, and let j be maximal so that the suffix $Q = t_j, t_{j+1}, \ldots, t_k$ covers a convex angle other than q. P and Q consist of at least two triangles each. We let b be the convex angle (vertex) covered by P — it is incident to both t_i and t_{i-1} — and d be the other vertex common to t_i and t_{i-1}. Furthermore, c is the third vertex of t_{i-1} and a is the third vertex of t_i (see Fig. 1 right). Symmetrically, define vertices b', d', c', a' of Q. We say that P (Q) is *type* 1 if the last (first) two triangles of P (Q) are the only ones incident to b (b'), and it is *type* 2, otherwise (see Fig. 2). If P is type 1 then a, b, c belong to the same chain and d belongs to the other chain (this includes the case that $c = p$), and if P is type 2 then a, b belong to one chain and c, d to the other.

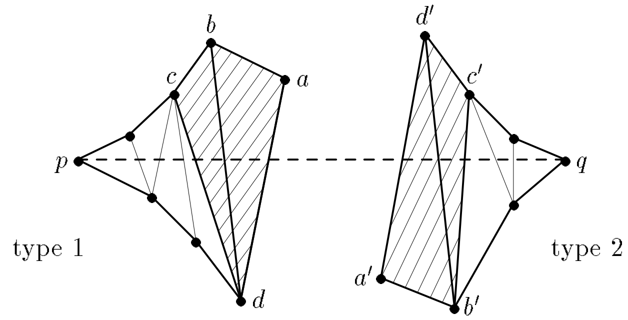

FIG. 2. *The prefix P with vertices a, b, c, d and the suffix Q with vertices a', b', c', d' are defined depending on pq. P is type 1 and Q is type 2. For illustration purposes the constraint that all vertices must lie outside the lune of pq has been ignored.*

FACT 1. *$P = t_1, t_2, \ldots, t_i$ and $Q = t_j, t_{j+1}, \ldots, t_k$ share at most two triangles, that is, $i - 1 \leq j$.*

Proof. We show that the suffix $R = t_{i-1}, t_i, \ldots, t_k$ covers at least one convex angle other than q, so Q cannot be bigger than R. If P is type 1 then R covers b, which is convex. Otherwise, R covers all angles between d and q, d included. Since all angles between p and d, p and d excluded, are non-convex, at least one angle between d and q must be convex, and this angle is covered by R. ☐

It should be clear that $abcd$ and $a'b'c'd'$ are both convex quadrilaterals by the choice of their vertices. The next two facts imply that either $abcd$, or $a'b'c'd'$, or both have alternate triangulations using ac or $a'c'$, while maintaining the maximum edge length of $t(S)$. In other words, bd, or $b'd'$, or both can be switched. Formally, we call bd ($b'd'$) *switchable* if ac ($a'c'$) is no longer than the longest edge of $t(S)$. Fact 2 shows strong locality constraints for a and d (a' and d') if bd ($b'd'$) is not switchable. Define

$$A = \{x \in \Re^2 : |xp| > |pq| \text{ and } |xp| > |xq|\} \text{ and}$$

$$D = \{x \in \Re^2 : |xp| \geq |pq| \text{ and } |xq| < |pq|\},$$

with the understanding that A and a belong to one half-plane defined by the line passing through p and q, and D and d belong to the other (see Fig. 3).

FACT 2. *If bd is not switchable then $a \in A$ and $d \in D$.*

Proof. Since bd is not switchable ac must be longer than the other five edges defined by a, b, c, d, and, by the Length Lemma, it must be longer than pq. We first

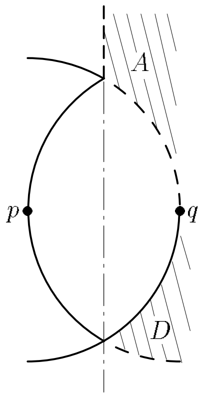

FIG. 3. *The regions A and D as defined for the case when a is on the upper chain.*

show that $|ac| \le |ap|$ and then derive the four inequalities needed to establish the claim.

(1) $|ac| \le |ap|$. We can assume that $c \ne p$. Note that c is contained in the closure of triangle bdp. Since the line passing through b and d separates a from p, the closures of the two triangles abp and adp cover bdp completely, and therefore one of them contains c. If c lies in the closure of abp the claim follows from $|ab| < |ac|$ and the Δ-Lemma for abp, and if c lies in adp it follows from $|ad| < |ac|$ and the Δ-Lemma for adp.

(2) $|ap| > |pq|$. From the Length Lemma we get $|pq| < |ac|$ and from (1) we get $|ac| \le |ap|$.

(3) $|dq| < |pq|$. Assume $|dq| \ge |pq|$. The $\square$-Lemma for $paqd$ implies $|ad| > |ap|$ and thus $|ad| > |ac|$ because of (1), a contradiction.

(4) $|dp| \ge |pq|$. This is immediate from (3) because pq is an edge of $rng(S)$.

(5) $|ap| > |aq|$. Assume $|ap| \le |aq|$ and recall $|dp| \ge |pq|$ from (4). By the $\square$-Lemma for $paqd$ we get $|ad| > |aq|$, which implies $|ad| > |ap|$ by assumption, and $|ad| > |ac|$ by (1), a contradiction.

The proof of Fact 2 is now complete because (2) and (5) are equivalent to $a \in A$ and (3) and (4) are equivalent to $d \in D$. $\square$

Symmetrically, we define regions A' and D' which is where a' and d' must lie if $b'd'$ is not switchable. Using Facts 1 and 2 we can now show that there is always an edge that can be switched.

FACT 3. *It is not possible that both bd and $b'd'$ are non-switchable.*

Proof. If bd and $b'd'$ are both non-switchable, then ad lies on q's side of the bisector of pq and $a'd'$ lies on p's side, by Fact 2. Because of Fact 1 and because ad is the last edge of P and $a'd'$ is the first edge of Q we have $\{a, d, a', d'\} = \{a, b, c, d\} = \{a', b', c', d'\}$. Furthermore, the fact that bd and $b'd'$ are both edges of $t(S)$ implies that they are the same and thus $b = d'$, $d = b'$, $a = c'$, $c = a'$ (see Fig. 4). It follows that the polygonal region has the shape of a diamond with p, b, q, d as the only convex angles. This contradicts the locality constraints for a, b, c, d stated in Fact 2. In particular, the chain from p to $d \in D$ is concave or straight and therefore enclosed by the circle $(q, |pq|)$. It follows that this chain is disjoint from A', which is where $c = a'$, the predecessor of d in this chain, is supposed to lie. $\square$

With the above results and notations, we now choose an extremal counterexample to prove the main result of this section.

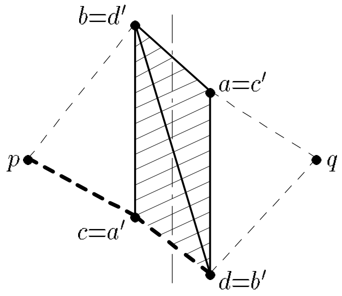

FIG. 4. *If bd and $b'd'$ are both non-switchable then b and d are the only convex angles besides p and q.*

SUBGRAPH THEOREM. *Every finite point set S in $\Re^2$ has a minmax length triangulation $mlt(S)$ so that $rng(S) \subseteq mlt(S)$.*

Proof. We assume there is a set S so that no minmax length triangulation contains $rng(S)$. Let $t(S)$ be a minmax length triangulation of S that satisfies the following extremal properties, where later properties are contingent upon earlier ones.

(i) $t(S)$ minimizes the number of edges that intersect pq.

(ii) $t(S)$ minimizes the number of edges incident to b that intersect pq.

(iii) $t(S)$ minimizes the number of edges incident to b' that intersect pq.

It is conceivable that $t(S)$ is not unique, but it will be sufficient to assume that $t(S)$ is any one of the remaining triangulations.

By Fact 3, either bd, or $b'd'$, or both are switchable. If bd is switchable and P is type 1 then the number of edges that intersect pq decreases when bd is switched. This contradicts property (i). Thus, P must be type 2 if bd is switchable, and, similarly, Q must be type 2 if $b'd'$ is switchable. When we switch bd the degree of b decreases, which contradicts property (ii). Thus, it must be that bd is not switchable and $b'd'$ is. But switching $b'd'$ decreases the degree of b', which would contradict property (iii), unless the degree of b increases at the same time. Remember that (iii) is contingent upon (ii), so if (ii) is not satisfied any more then we cannot draw any conclusion. Thus, the configuration left for analysis is as shown in Fig. 5.

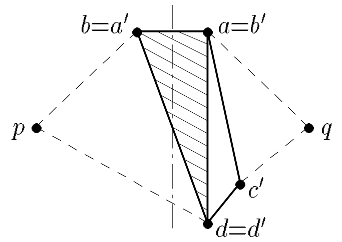

FIG. 5. *In the final configuration bd is non-switchable, so $a \in A$ and $d \in D$, and $b'd'$ is switchable, so Q is type 2. Furthermore, switching $b'd'$ to $a'c'$ increases the degree of b, so $a' = b$ and therefore P and Q overlap in exactly one triangle. The figure ignores that by rights all points should lie outside the lune of pq.*

To reach the final contradiction, we switch $b'd'$ and redefine Q based on the new configuration. Since all angles from (the old) d' to q are non-convex, the new points b' and a' are the same as before, and the new d' is the old c'. Thus, we can again switch $b'd'$, and so on, until Q is type 1 or $c' = q$ at which point the next switch decreases the number of edges intersecting pq. This finally contradicts property (i). □

Remark. A natural extension of minimizing the length of the longest edge in a triangulation is to also minimize the length of the second longest edge, and so on. Let $mvt(S)$ be a triangulation that minimizes the entire vector of edge-lengths in this fashion. If the points of S are in general position then $mvt(S)$ is unique. Curiously, it is not always true that (there is an) $mvt(S)$ (that) contains $rng(S)$ as a subgraph. The smallest example that illustrates this observation consists of four points a, b, c, d so that c and d lie fairly close to b, ab and cd intersect, and c and d both lie outside the circle $(a, |ab|)$.

5. Triangulating rng**-Polygons.** The goal of this section and the next is to improve the cubic time algorithm of Section 3 to quadratic time. This is done using a specialized polygon triangulation algorithm. The main part of the algorithm, and the structural properties of minmax length triangulations that guarantee its correctness, are developed in this section.

Recall that the first two steps of the algorithm in Section 3 decompose the convex hull of S into polygonal regions by drawing all edges of $ch(S)$ and $rng(S)$; these steps remain unaltered. Each region is represented by a cyclic chain of directed edges that trace its boundary in a counterclockwise order around the region. Because $rng(S)$ is a connected graph that spans S, any polygonal region is bounded by at most one edge not in $rng(S)$; this edge is in $ch(S) - rng(S)$. We call a polygonal region a *complete rng-polygon* if all its edges belong to $rng(S)$, and an *incomplete rng-polygon*, otherwise.

Obviously, rng-polygons are not as general as arbitrary polygonal regions because for each edge ab, except possibly for one, the lune of ab, $\lambda_{ab} = \{x \in \Re^2 : \max\{|ax|, |bx|\} < |ab|\}$, is free of points of S. We call pq a *diagonal* of a polygonal region if it lies in the region entirely. For each diagonal pq of an rng-polygon it must be that λ_{pq} contains at least one point of S. We further distinguish between the cases where λ_{pq} contains points of S on both sides of pq and where it does not.

For a directed edge $\vec{pq}$ let $h_{\vec{pq}}$ be the set of points to the left of or on the directed line that passes through p and q in this order. Define the *half-lune* of $\vec{pq}$ as

$$\eta_{\vec{pq}} = \lambda_{pq} \cap h_{\vec{pq}}.$$

By definition, $\lambda_{pq} = \eta_{\vec{pq}} \cup \eta_{\vec{qp}}$, and we have $pq \in rng(S)$ iff $\eta_{\vec{pq}} \cap S = \eta_{\vec{qp}} \cap S = \emptyset$. We call pq a 2-*edge* if both half-lunes contain points of S, and we call it a 1-*edge* if only one half-lune contains points of S. For a 1-edge pq, we say the side where the half-lune contains points of S is *beyond* pq, and the other side is *beneath* pq. Note for example that if pq is a 1-edge bounding an incomplete rng-polygon R then $pq \in ch(S)$ and therefore R is beyond pq. We will see later that 1-edges are useful in triangulating rng-polygons.

The first lemma of this section shows that when we triangulate an rng-polygon R, whether complete or incomplete, we can ignore all points outside R. More specifically, it shows that the type of any diagonal or edge of R remains unchanged when we remove all points of S that are not vertices of R.

REDUCTION LEMMA. *Let pq be a diagonal or edge of an rng-polygon R. If $\eta_{\vec{pq}}$ contains points of S then it also contains vertices of R.*

Proof. Assuming $\eta_{\vec{pq}}$ contains points of S but no vertices of R, it must intersect edges of R without containing their endpoints. Let yy' be the edge closest to p and q, and let x be a point in $\eta_{\vec{pq}} \cap S$. Since x is not a vertex of R it must lie on the other side of yy', as seen from p and q. So $yy' \in rng(S) - ch(S)$, and therefore

$\max\{|xy|, |xy'|\} \geq |yy'|$. Assume without loss of generality that $|xy| \geq |yy'|$. If y' lies outside or on the circle $(p, |pq|)$ we consider the convex quadrilateral $pyxy'$. Otherwise, y' lies outside or on $(q, |pq|)$ in which case we consider the convex quadrilateral $qyxy'$. But now we have $|xy| \geq |yy'|$ and either $|py'| > |px|$ or $|qy'| > |qx|$, a contradiction to the $\Box$-Lemma in both cases. ∎

Using the Reduction Lemma we now address vertices visible from both endpoints of an edge. We need some notation. Two points x, y inside or on the boundary of a polygonal region are *visible* from each other if xy is contained in the region. The *distance* of a point x to an edge pq is defined as the infimum, over all points $z \in pq$, of $|xz|$. If $|pq| > \max\{|px|, |qx|\}$ then this distance is referred to as the *height* of the triangle pqx.

VISIBILITY LEMMA. *Let pq be a diagonal or edge of an rng-polygon R, and let x be a vertex of R that lies in $\eta_{\vec{pq}}$ and minimizes the distance from pq. Then x is visible from p and also from q.*

Proof. Consider the triangle pqx, let $x' \in pq$ be the point with minimum distance from x, and assume without loss of generality that x is not visible from q. Let yy' be an edge of R that intersects qx. The proof of the Reduction Lemma implies that at least one endpoint of yy' lies in $\eta_{\vec{pq}}$, say $y \in \eta_{\vec{pq}}$. In addition, y and y' lie outside the triangle pqx because x is closest to pq (see Fig. 6). Hence, yy' intersects xp, xq

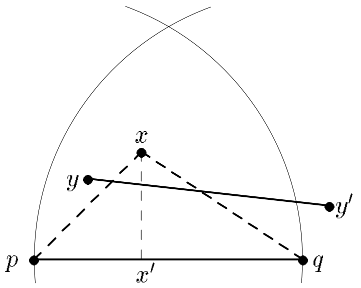

FIG. 6. *The quadrilateral $xyx'y'$ is convex because $x' \in pq$ and $y, y' \notin pqx$.*

and all edges xz with $z \in pq$. Thus, $xyx'y'$ is a convex quadrilateral, and because of $|yx'| \geq |xx'|$ by the choice of x, we have $|yy'| > |y'x|$ from the $\Box$-Lemma. By symmetry, if y' lies in $\eta_{\vec{pq}}$ we have $|yy'| > |xy|$, which implies $yy' \notin rng(S)$. This is a contradiction because $yy' \notin ch(S)$. Thus, y' must lie outside $\eta_{\vec{pq}}$. If y' lies outside or on the circle $(p, |pq|)$ then $|py'| > |px|$ and therefore $|xy| < |yy'|$ by the $\Box$-Lemma for $py'xy$. Symmetrically we get $|xy| < |yy'|$ from the $\Box$-Lemma for $qy'xy$ if y' lies outside or on the circle $(q, |pq|)$. Together with $|xy'| < |yy'|$ this contradicts $yy' \in rng(S)$. ∎

We need one more elementary lemma.

CONTAINMENT LEMMA. *If $x \in \eta_{\vec{pq}}$ then $\eta_{\vec{xp}} \subseteq \lambda_{pq}$.*

Proof. Take a point $z \in \eta_{\vec{xp}}$ and consider the four points p, q, x, z. If $z \in pq$ there is nothing to prove. Otherwise, $pzqx$ or $pqzx$ is a convex quadrilateral (possibly with three of the four vertices collinear) or $z \in pqx$. In each case $|qz| < |pq|$ can be shown using the $\Box$- or the Δ-Lemma. This implies $z \in \lambda_{pq}$. ∎

The following lemma is of fundamental importance to the quadratic time triangulation algorithm.

1-EDGE LEMMA. *Let pq be a 1-edge of an rng-polygon R, and let x be a vertex of R that lies in $\eta_{\vec{pq}}$ and minimizes the distance from pq. Then px is either an edge of R or a 1-edge with pqx beneath px, and the same is true for qx.*

Proof. We have $\eta_{\vec{xp}} \subseteq \lambda_{pq}$ by the Containment Lemma. The part of $\eta_{\vec{xp}}$ in $\eta_{\vec{qp}}$ contains no point of S because $\eta_{\vec{qp}} \cap S = \emptyset$ by assumption. For a different reason also the part of $\eta_{\vec{xp}}$ in $\eta_{\vec{pq}}$ contains no point of S. This is because a point $y \in \eta_{\vec{xp}} \cap \eta_{\vec{pq}}$ would be closer to pq than x is, as can be shown using the $\square$-Lemma for $px'yx$ (see Fig. 7). So px is an edge of R if $\eta_{\vec{px}}$ contains no point of S either, and it is a 1-edge with triangle pqx on its beneath side, otherwise. The argument for qx is symmetric. $\square$

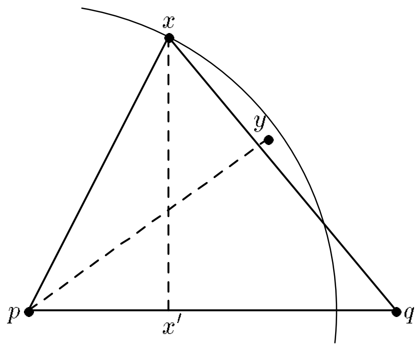

FIG. 7. *Vertex x is visible from p and from q, so pqx is empty. It follows that if $y \in \eta_{\vec{xp}} \cap \eta_{\vec{pq}}$ then $pqyx$ is a convex quadrilateral.*

5.1. Incomplete *rng*-Polygons. The above lemmas are sufficient for efficiently triangulating an incomplete *rng*-polygon. As defined earlier, all edges of an incomplete *rng*-polygon R are *rng*-edges, except for one 1-edge, $pq \in ch(S) - rng(S)$, which has R on its beyond side. The algorithm below can triangulate more general incomplete *rng*-polygons, that is, it is not necessary that $pq \in ch(S)$, but it must be that pq is a 1-edge and R lies beyond pq.

Input. An incomplete *rng*-polygon R that lies beyond its 1-edge pq.

Output. A minmax length triangulation of R.

Algorithm.
1. Find a vertex x in λ_{pq} that minimizes the distance from pq.
2. Draw edges px and qx. This decomposes R into the triangle pqx, and two possibly empty incomplete *rng*-polygons R_1 and R_2.
3. Recursively triangulate R_1 and R_2.

The correctness of this algorithm follows from the 1-Edge Lemma. Indeed, it implies that if R_1 is non-empty then it lies beyond px, which is the only 1-edge of R_1. Similarly, R_2 lies beyond its 1-edge qx, provided R_2 is non-empty. Thus, the input invariant is maintained all the way through the recursion. This implies that the algorithm successfully triangulates. By the choice of point x, the edges px and qx are both shorter than pq. It follows that the diagonals are monotonely decreasing in length, down a single branch of the recursion, and therefore all diagonals constructed by the algorithm are shorter than pq. A straightforward implementation of the algorithm takes time quadratic in the number of vertices of R.

Remark. Instead of choosing a vertex x that minimizes the distance to pq, step 1 of the algorithm could also choose other vertices as long as they are visible from p and q and lie in their lune. An interesting choice among these vertices is the vertex y

that minimizes $\max\{|yp|, |yq|\}$. As long as y is unique, which is the non-degenerate case, this choice leads to a triangulation of the polygon R that lexicographically minimizes the sorted vector of edge lengths. Another possible choice is the vertex z that minimizes $|zp| + |zq|$. This vertex is automatically visible from p and from q and might be useful in actual implementations because it is often considerably less expensive to compute the distance between two points than between a point and a line segment.

5.2. A Lemma on Polygon Retriangulation. This subsection presents a technical lemma on retriangulating a polygonal region. It will find application in Sections 5.3 and 6, and is also of independent interest. In order to conveniently distinguish between boundary and non-boundary edges of a triangulation, we call a non-boundary edge a *diagonal*. Let X be a polygonal region, $t(X)$ a triangulation of X, and xx' a diagonal of X that is not in $t(X)$. We say that xx' *generates* $t(X)$ if it intersects every diagonal of $t(X)$. We give an algorithmic description of a particular triangulation of X, called the *fan-out triangulation* $f_x(X)$ with *(fan-out) center* x. The triangulation is illustrated in Fig. 8.

1. Connect x to all vertices of X that are visible from x. Call these vertices and also the two vertices connected to x by edges of X *neighbors* of x.
2. Two neighbors of x are said to be *adjacent* if they are consecutive in the angular order around x. Connect any two adjacent neighbors u, v of x, unless uv is an edge of X.
3. Every edge uv created in step 2 decomposes X into two parts, and the part that does not contain x is called the *pocket* X_{uv} of uv. Assume that u is the endpoint of uv so that the other incident edge of the pocket, uw, is partially visible from x. Recursively construct the fan-out triangulation of X_{uv} with center v.

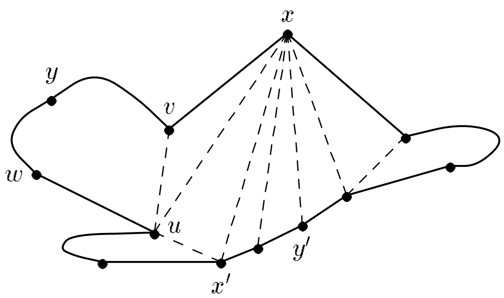

FIG. 8. *The polygonal region X is triangulated by fanning out from x, connecting adjacent neighbors of x, and recursing in the thus created pockets. The illustration of this process is schematic and ignores some of the inherent shape constraints for X.*

We introduce some terminology. Among the diagonals of $f_x(X)$ we distinguish between *fan-out* edges constructed in step 1 and *cut-off* edges constructed in step 2 of the above algorithm. Each call of the algorithm triangulates part of a pocket and recurses in each component (pocket) of the remainder. We call a pocket V a *child* of another pocket Z if $V \subset Z$ and V is maximal. The original polygonal region, X, is also called a pocket and forms the root of the tree defined by the child relation. This tree is exactly the recursion tree of the algorithm. Each pocket Z is associated with a fan-out center z. The maximum distance between z and any other vertex of Z is called the *width* of Z.

The lengths of the diagonals of $f_x(X)$ are constrained by the length of the longest edge of X, the length of the longest diagonal of $t(X)$, and the width of X. More specifically, we prove the following result.

FAN-OUT LEMMA. *Let X be a polygonal region, with δ_1 the length of its longest edge, let $t(X)$ be a triangulation of X, with δ_2 the length of its longest diagonal, let xx' be a generator of $t(X)$, and let δ_3 exceed the maximum distance of x from any vertex of X. Then $|ab| < \max\{\delta_1, \delta_2, \delta_3\}$ for every diagonal ab of $f_x(X)$.*

Proof. Note that the assertion follows if we prove that $\max\{\delta_1, \delta_2, \delta_3\}$ exceeds the width of every pocket Z created during the algorithm. To see this notice that the width of Z is an upper bound on the length of any fan-out edge emanating from the center of Z. Each cut-off edge uv that creates a child pocket V of Z is incident to the fan-out center of V which implies that the width of V is an upper bound on its length.

The proof of the upper bound on the widths of all pockets proceeds inductively, from the top to the bottom of the tree. The width of X is less than δ_3, by assumption, and therefore also less than $\max\{\delta_1, \delta_2, \delta_3\}$. For the inductive step consider a pocket Z and a child V of Z. We show that the bound on the width of Z is inherited by V, with some environmental influence from X and $t(X)$. Let z be the fan-out center of Z, δ the width of Z, v the fan-out center of V, uv the cut-off edge that creates V, and w the other vertex of V adjacent to u.

First, we prove $|uv| < \max\{\delta_1, \delta\}$. By definition of fan-out center v lies inside the triangle uwz. The Δ-Lemma thus implies $|uv| < \max\{|uw|, |uz|\}$, and we get the claimed inequality because $|uw| \leq \delta_1$ and $|uz| \leq \delta$. Second, we show that $\max\{\delta_2, \delta\}$ exceeds the maximum distance between v and any vertex of V other than u. Let $y \neq v, u$ be such a vertex and let yy' be a diagonal of $t(X)$ that intersects xx'. Such a diagonal exists because xx' generates $t(X)$. It follows that yy' intersects uv and that therefore v lies inside the triangle $yy'z$. Using the Δ-Lemma we get $|yv| < \max\{|yy'|, |yz|\} \leq \max\{\delta_2, \delta\}$ because $|yy'| \leq \delta_2$ and $|yz| \leq \delta$. The two bounds together imply that the width of V is less than $\max\{\delta_1, \delta_2, \delta\}$, and induction shows that it is less than $\max\{\delta_1, \delta_2, \delta_3\}$. $\square$

In Section 6 we will need a result as given in the Fan-Out Lemma, but restricted to the fan-out triangulation on one side of the generator. More specifically we need the following corollary whose proof is almost the same as the one of the Fan-Out Lemma.

FAN-OUT COROLLARY. Suppose W is a polygonal region, $t(W)$ a triangulation of W, xx' a generator of $t(W)$, and X the part of W on one side of xx'. Let δ_1 be the length of the longest edge of X, δ_2 the length of the longest diagonal of $t(W)$, and let δ_3 exceed the maximum distance of x from any vertex of X. Then $|ab| < \max\{\delta_1, \delta_2, \delta_3\}$ for every diagonal ab of $f_x(X)$.

Remark. The Fan-Out Lemma can also be formulated without the assumption of an initial triangulation. The condition on the diagonal xx' is now that each vertex of X must be visible from some point of xx'. The parameter δ_2 needs to be redefined as the maximum, over all vertices y of X, of the infimum, over all points a of xx' visible from y, of the distance between y and a.

5.3. Complete *rng*-Polygons. It will be convenient to assume that no two diagonals and edges of the *rng*-polygon R are equally long. With this assumption we can show that every triangulation of R, and therefore also every minmax length triangulation, contains a 2-edge. To see this take the longest edge pq of a triangulation. It is not an edge of R because the third vertex of the incident triangle lies in its lune

λ_{pq}. It is therefore a diagonal with incident triangles pqr and pqs, and we have $r, s \in \lambda_{pq}$ by maximality of pq. Since r and s lie on different sides of pq it follows that pq is a 2-edge.

We prove below that there is a minmax length triangulation $mlt(R)$ of R that contains only one 2-edge pq. By the argument above pq is the longest edge of $mlt(R)$. We call pq *expandable* if there are vertices r and s in λ_{pq}, on different sides of pq and both visible from p and q, so that $E = \{pr, qr, ps, qs\}$ is a set of rng- and 1-edges and the quadrilateral $prqs$ lies beneath the 1-edges in E. It should be clear that once we draw an expandable 2-edge we can complete the triangulation using the algorithm for incomplete rng-polygons (Section 5.1). The resulting triangulation uses no 2-edge other than pq which is thus the longest edge of the triangulation.

We first present the algorithm and then prove its correctness by showing that every complete rng-polygon R has a minmax length triangulation that contains an expandable 2-edge. This, however, assumes that no two diagonals or edges of R have equal length. If this non-degeneracy constraint is not satisfied it is necessary to run the algorithm with a simulation of non-degeneracy, see [EdMü90]. The side-effects of this simulation and how they can be undone will be discussed in Section 5.4.

Input. A complete rng-polygon R.

Output. A minmax length triangulation of R.

Algorithm.
1. Find the shortest expandable 2-edge pq, together with corresponding rng- and 1-edges pr, qr, ps, qs.
2. Triangulate the incomplete rng-polygons defined by pr, qr, ps, qs.

As mentioned in Section 5.1, step 2 takes time that is only quadratic in the number of vertices of R. In Section 6 we will see how step 1 can be implemented so it runs in quadratic time too. We now formulate and prove the lemma that implies the correctness of the algorithm.

2-EDGE LEMMA. *Let R be a complete rng-polygon with no two diagonals or edges of the same length. Then there exists a minmax length triangulation $mlt(R)$ of R that contains an expandable 2-edge.*

Proof. We assume there is no minmax length triangulation of R that contains an expandable 2-edge. A contradiction to this assumption will be derived using an extreme minmax length triangulation $t(R)$ defined as follows. Let pq be the longest edge of $t(R)$ and let pqr and pqs be the incident triangles. By the non-degeneracy assumption, pq is the longest edge of *every* minmax length triangulation of R. Choose $t(R)$ so that the sum of heights of pqr and pqs (that is, the distance of r from pq plus the distance of s from pq) is a minimum. We prove below that pq is expandable and that r and s are witnesses thereof, that is, the quadrilateral $prqs$ lies beneath every 1-edge in $E = \{pr, qr, ps, qs\}$.

Case 1. Assume that $prqs$ lies beyond at least one 1-edge in E, say beyond pr. Then we can retriangulate R on this side of pr using the algorithm for incomplete rng-polygons. Among others, this algorithm removes edge pq, and all new edges are shorter than pr, which itself is shorter than pq. This contradicts the assumption that $t(R)$ is a minmax length triangulation.

Case 2. Assume that one of the edges of E, say pr, is a 2-edge, and assume without loss of generality that $r \in \eta_{p\vec{q}}$. Thus, there is a non-empty set of vertices z of R contained in the half-lune $\eta_{r\vec{p}}$. By the Containment Lemma these vertices z lie in λ_{pq}, and by the Visibility Lemma a non-empty subset S' of the z are visible from both p and r.

If a vertex z is in S' then either $pz \cap rq \neq \emptyset$ or $rz \cap pq \neq \emptyset$, see Fig. 9. Let S'_p

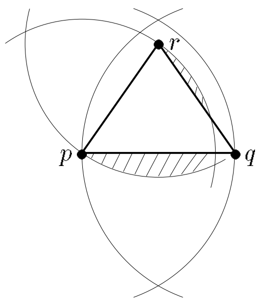

FIG. 9. *The points z lie in the interior of $\eta_{\vec{rp}} - pqr$, which consists of one or two connected components depending on whether or not the angle at r in the triangle pqr is non-acute.*

be the subset of vertices z of the first kind, and let S'_r be the subset of vertices of the second kind. If $S'_p \neq \emptyset$ choose $x \in S'_p$ so that the number of edges of $t(R)$ that intersect px is a minimum. Next, remove all edges from $t(R)$ that intersect px and denote by X the polygonal region thus generated. If, on the other hand, $S'_p = \emptyset$, then choose $x \in S'_r \neq \emptyset$ so that the number of edges in $t(R)$ that intersect rx is a minimum, again remove all edges from $t(R)$ that intersect rx, and denote the resulting polygonal region by X. For convenient reference we set $x' = p$ in the first case and $x' = r$ in the second. In either case, we construct a retriangulation $f_x(X)$ of X by fanning out from x, as described in Section 5.2.

We show below that the new triangulation of R has properties that contradict the assumptions of case 2. Most importantly, the Fan-Out Lemma of Section 5.2, together with a few claims which we are about to prove, imply that the edges of $f_x(X)$ do not exceed pq in length.

CLAIM 2.1. *Except for x, all vertices of X lie outside the half-lune $\eta_{\vec{rp}}$.*

Proof (of Claim 2.1). Let $y_1y_2, y_3y_4, \ldots, y_{m-1}y_m$ be the edges, sorted from x' to x, that are removed from $t(R)$ when X is constructed. Suppose the claim is not true. Then there is a smallest index $j \leq m-1$ with $y_{j+1} \in \eta_{\vec{rp}}$. Consider the polygonal region X_j of $t(R)$ that is created by removing the edges $y_1y_2, y_3y_4, \ldots, y_{j-1}y_j$ from $t(R)$. Since y_{j+1} is the only vertex of X_j that lies in $\eta_{\vec{rp}}$ it is visible from p *and* from r, inside X_j. But this means that $y_{j+1}x'$ intersects fewer edges of $t(R)$ than xx'. This contradicts the choice of x and completes the proof of Claim 2.1.

CLAIM 2.2. *For each vertex y of X we have $|xy| < |pq|$.*

Proof (of Claim 2.2). Clearly, both px and rx are shorter than pq. So let y be any vertex different from p, r, x, and let yy' be an edge of $t(R)$ that intersects $x'x$. Because of Claim 2.1, x is visible within X from p and also from r, so $pyxy'$ and $ryxy'$ are convex quadrilaterals. Since y' lies outside $\eta_{\vec{rp}}$ it cannot lie inside both of the circles $(p, |pr|)$ and $(r, |pr|)$. If y' lies inside $(r, |pr|)$ then $|py'| > |px|$ which implies $|yy'| > |xy|$ by the $\square$-Lemma for $pyxy'$. Otherwise, we have $|ry'| > |rx|$ which implies $|yy'| > |xy|$ by the $\square$-Lemma for $ryxy'$. This concludes the proof of Claim 2.2 because yy' is an edge of $t(R)$ and is therefore no longer than pq.

Claim 2.2 and the Fan-Out Lemma imply that all diagonals of $f_x(X)$ are shorter than pq. In the case where $pq \cap rx \neq \emptyset$ we now have a contradiction, because the retriangulating process of X eliminates pq and all edges of the resulting new triangulation of R are shorter than pq. In the case where $rq \cap px \neq \emptyset$ the new triangulation still includes pq. We show below that the height of the new triangle incident to pq is smaller than the height of pqr and thus arrive at a contradiction.

So assume $rq \cap px \neq \emptyset$; in this case pq is an edge of the boundary of X and p is visible from x. If q is also visible from x then the new triangle incident to pq is pqx with height $|xx'|$, where $x' \in pq$ minimizes the distance to x. Analogously define $r' \in pq$ that minimizes the distance to r. Since $|pr| > |px|$ we have $|rr'| > |xr'|$ by the $\square$-Lemma for $prxr'$. Together with $|xr'| \geq |xx'|$ this implies $|rr'| > |xx'|$. If q is not visible from x then pq belongs to the pocket X_{uv} defined by a cut-off edge uv. We have $u = p$, $w = q$, and the center v of X_{uv} lies inside pqx. So again, either pqv is a triangle, and its height is less than that of pqx and therefore that of pqr, or q is not visible from v, in which case the argument can be repeated. Eventually, we arrive at a triangle incident to pq whose height is less than that of pqr. $\square$

Remark. Recall that the assertion of the 2-Edge Lemma is made under the condition that no two diagonals or edges of the complete rng-polygon R are equally long. Indeed, the assertion is false without this condition. Take, for example, two equilateral triangles abc and abd and move d slightly towards the common edge ab. For $S = \{a, b, c, d\}$ we have $rng(S) = \{ac, cb, bd, da\}$, ab is a 1-edge, and cd is a 2-edge. So $acbd$ is a complete rng-polygon. There is only one minmax length triangulation of $acbd$, namely the one obtained by drawing the diagonal ab. But ab is not a 2-edge.

5.4. Undoing the Simulated Perturbation. For every finite point set S in $\Re^2$ there is an arbitrarily small perturbation S' so that S' satisfies convenient non-degeneracy assumptions (see [EdMü90]). For a point $p \in S$ we denote its perturbed version by p'. In the case of relative neighborhood graphs and minmax length triangulations this means that no two pairs of points in S' define the same distance. Because the perturbation is arbitrarily small, the non-degenerate properties of S are maintained, that is, for four not necessarily distinct points $p, q, r, s \in S$ with $|pq| < |rs|$ we have $|p'q'| < |r's'|$.

Let us consider the effect of the perturbation on the computation of a minmax length triangulation. Clearly, if $p'q' \in rng(S')$ then $pq \in rng(S)$, but not vice versa. The fact that in the perturbed setting the relative neighborhood graph has potentially fewer edges than in the unperturbed setting does not adversely influence the triangulation algorithm since $rng(S')$ is still connected and spans S'. When the edges of $ch(S')$ are added and the polygonal regions defined by $ch(S') \cup rng(S')$ are triangulated, it can happen that triangles $a'b'c'$ are constructed whose unperturbed counterparts abc are flat, that is, a, b, c are collinear. Although this is not a problem for the algorithm, it is somewhat distressing when this triangulation is interpreted as a triangulation of S. The remainder of this section shows how to remedy this deficiency.

Let $t(S')$ be a minmax length triangulation of S', and consider its unperturbed version $t(S)$, that is, $pq \in t(S)$ iff $p'q' \in t(S')$. A longest edge of $t(S)$ is no longer than a longest edge of any minmax length triangulation $mlt(S)$ of S, since $mlt(S')$, the perturbed version of $mlt(S)$, is a valid triangulation of S' and would otherwise contradict that $t(S')$ is a minmax length triangulation of S'. The reverse is also true, namely a longest edge of $t(S)$ is no shorter than a longest edge of $mlt(S)$. We show this by converting $t(S)$ into a minmax length triangulation of S.

Consider the dual graph $t^*(S')$ of $t(S')$ and call a node $a'b'c'$ *flat* if a, b, c, are collinear. Determine the connected components of the subgraph of $t^*(S')$ induced by the set of all flat nodes. Each component corresponds to a collection of collinear points in S, interconnected by flat triangles, see Fig. 10. Carry out the following steps

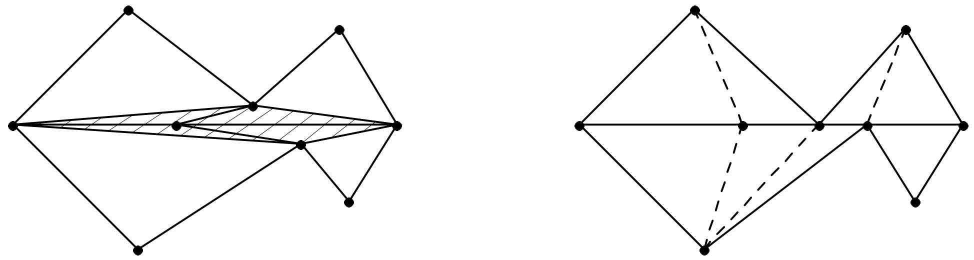

FIG. 10. *The five points in the middle of the left triangulation are the perturbed versions of five collinear points in the right triangulation.*

for one component at a time. Remove all edges of the flat triangles of the component, sort the corresponding points along the supporting line, and add edges connecting points that are adjacent in the sorted order. This produces regions bounded by more than three edges, as shown in Fig. 10. All vertices x of such a region are collinear, except for one vertex y which is connected to the first and last of the vertices x. Triangulate this region by connecting y to all other vertices x. By the Δ-Lemma the newly introduced edges are no longer than the longer of the two original edges incident to y.

6. Finding the Shortest Expandable 2-Edge. This section shows how the first step of the algorithm for triangulating a complete *rng*-polygon R can be made to run in time $O(n^2)$, where n is the number of vertices of R. As in Section 5.3, we assume that no two diagonals or edges of R are equally long; so the shortest expandable 2-edge is unique. For convenience we also assume that no three vertices of R are collinear.

Input. A complete *rng*-polygon R.

Output. The shortest expandable 2-edge of R.

Algorithm.
1. Determine the type of each diagonal pq of R.
2. For each 2-edge pq find vertices p', p'', q', q'' that minimize the counterclockwise angles $\angle p'pq, \angle qpp'', \angle q'qp, \angle pqq''$, contingent upon pp', pp'', qq', qq'' being *rng*-edges or 1-edges with pq on their beneath sides (see Fig. 11).
3. Return the shortest 2-edge pq for which pp', qq', pp'', qq'' are such that $p' = q''$ or $pp' \cap qq'' \neq \emptyset$, and $p'' = q'$ or $pp'' \cap qq' \neq \emptyset$.

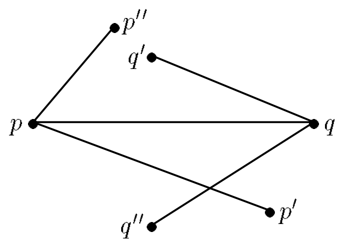

FIG. 11. *By the choice of p' the counterclockwise angle $\angle p'pq$ contains no 1-edge with pq on its beneath side. Symmetric statements hold for p'', q', and q''.*

Below we give the algorithmic details of the above steps.

STEP 1, *classifying diagonals*. For each vertex p of R, we compute all incident diagonals pq and their angular order around p. Furthermore, we determine whether or not the half-lune $\eta_{\vec{pq}}$ contains any vertex of R. Recall that by the Visibility Lemma $\eta_{\vec{pq}}$ contains a vertex visible from p if it contains a vertex of R at all. We can thus base the decision whether or not $\eta_{\vec{pq}}$ is empty of vertices solely on the vertices visible from p. As defined earlier, pq is a 2-edge if both half-lunes of pq contain vertices of R. Otherwise, pq is a 1-edge and its beyond side is where the half-lune contains vertices of R. We now show that the computation for p can be done in time $O(n)$. It follows that $O(n^2)$ time suffices for step 1.

Computing the sorted sequence of diagonals $pp_1, pp_2, \ldots, pp_m$ incident to p is a standard operation for simple polygons and can be done in time $O(n)$, see e.g. [ElAv81], [JoSi87], [Lee83]. Let pp_0 and pp_{m+1} be the two edges of R incident to p and assume that $p_0, p_1, p_2, \ldots, p_m, p_{m+1}$ is in a counterclockwise order around p. To determine whether there is a vertex of R in the half-lune $\eta_{\vec{pp_i}}$ for $1 \leq i \leq m$, we scan the list $p_0, p_1, \ldots, p_{m+1}$ once, from smallest index to largest. During the scan we maintain a stack of diagonals pp_l whose half-lunes $\eta_{\vec{pp_l}}$ are not yet found to contain any vertex of R. Before pushing pp_i onto the stack, we remove all diagonals pp_l whose half-lunes contain p_i. Using a straightforward extension of the Containment Lemma we can show that the order of processing implies that the edges whose half-lunes contain p_i lie on top of the ones whose half-lunes do not contain p_i. Thus, the former can be removed simply by repeatedly popping the topmost diagonal. When the scan is complete, the stack contains exactly all diagonals pp_l whose half-lunes contain no vertex of R. Since a diagonal can be pushed and popped only once each, the entire process takes constant time per diagonal.

STEP 2, *finding rng- and 1-edges*. For each vertex p, we scan $pp_1, pp_2, \ldots, pp_m$ in this order. In the process we keep track of the most recent *rng*-edge or 1-edge $p\bar{p}$ whose beneath side is in the direction of the scan. Initially, $p\bar{p} = pp_0$. When a 2-edge pq is encountered then $p\bar{p}$ is the edge pp' that belongs to pq. A symmetric scan is carried out to find the edge pp'' that belongs to pq. The total time, for all vertices p of R, is clearly $O(n^2)$.

STEP 3, *returning the solution*. Step 3 is computationally trivial. It takes time $O(n^2)$ since constant time suffices to test whether or not pp', pp'', qq', qq'' satisfy the conditions of step 3. However, it is not trivial to see that the edge pq returned in step 3 is also the shortest expandable 2-edge. First note that the shortest expandable 2-edge is no shorter than pq. This is because all 2-edges shorter than pq fail the test of step 3. The following straightforward topological lemma implies that these 2-edges are not expandable.

CROSSING LEMMA. *Let $v_1, v_2, \ldots, v_n$ be the sequence of vertices of a simple polygon, and let v_1v_i and v_jv_n be two diagonals. Then $v_1v_i \cap v_jv_n \neq \emptyset$ iff $j < i$.*

Proof. The edge v_jv_n decomposes the polygon into two disjoint polygons with vertex sequences $v_1, v_2, \ldots, v_j, v_n$ and $v_j, v_{j+1}, \ldots, v_n$. If $j < i$ then neither of the two polygons has v_1 *and* v_i on its boundary. It follows that v_1v_i crosses from one polygon into the other, and because v_1v_i is a diagonal, this is only possible by crossing v_jv_n. To prove the other direction we assume $v_1v_i \cap v_jv_n \neq \emptyset$ and observe that v_1 and v_i belong to different polygons because there is no way that v_1v_i can enter the second polygon and leave it again. Thus, $j < i$. □

So it remains to show that the edge pq computed in step 3 is indeed expandable.

Expandability Lemma. *The shortest 2-edge pq of R that satisfies the conditions of step 3 is also expandable.*

Proof. We show below that R can be triangulated on both sides of pq using only edges shorter than pq. If we now assume that pq is not expandable we get a contradiction to the 2-Edge Lemma, because pq is the longest edge of the triangulation and all expandable 2-edges are longer than pq.

We describe how to triangulate the part of R to the right of $\vec{pq}$; the other part is symmetric.

Case 1. $p' = q''$. Assume $|qq''| > |pp'|$. Then $|qq''| < |pq|$ for otherwise $p \in \eta_{q\vec{n}_q}$ and qq'' would neither be an rng-edge nor a 1-edge with pq on its beneath side. If we apply the triangulation algorithm for incomplete rng-polygons (Section 5.1), once for pp' and once for qq'', we get a triangulation with the desired properties.

Case 2. $pp' \cap qq'' \neq \emptyset$. In this case pp' and qq'' are 1-edges. Because pp' and qq'' intersect, it must be that p' is closer to q than to p or that q'' is closer to p than to q. Assume without loss of generality that $|q''p| < |q''q|$. As in case 1 we also have $|q''q| < |pq|$, but note that we do not necessarily have $|pp'| < |pq|$.

We now describe the triangulation process. It takes three steps illustrated in Fig. 12 and 13.

1. Construct the triangulation $t_{qq''}$ of R beyond qq'', using the algorithm for incomplete rng-polygons (see Fig. 12).
2. Find the subset V of vertices of R that lie inside the triangle pqq'' and compute the convex hull C of $V \cup \{p, q''\}$. Add the edges of C that are diagonals of R to the triangulation, and connect q to all vertices of C (see Fig. 12).
3. Step 2 creates untriangulated pockets Y_{uv}, one for each edge uv of C that is a diagonal of R. Assume that u precedes v on the clockwise path from p to q'' on the boundary of C. The pocket Y_{uv} is triangulated as follows.
   3.1 Set $u_L := v$ if uv is a 1-edge and pq lies on the beneath side of uv. Otherwise, find a vertex u_L so that $|uu_L| < |pq|$, uu_L is a 1-edge, pq lies beneath uu_L, and uu_L does not intersect C. (The existence of such a vertex u_L will be established shortly.)
   3.2 Construct the triangulation t_{uu_L} of R beyond uu_L, again using the algorithm for incomplete rng-polygons, but retain only the triangles that lie completely inside the pocket Y_{uv}. Let X_{uv} denote the untriangulated part of Y_{uv}.
   3.3 Construct the fan-out triangulation $f_v(X_{uv})$.

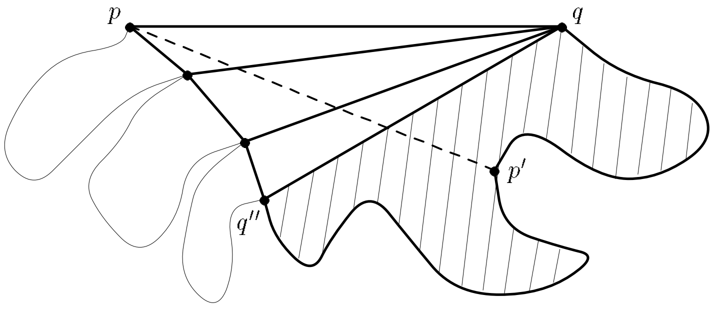

Fig. 12. *The shaded portion represents the triangulation beyond qq''; it forms part of the final triangulation.*

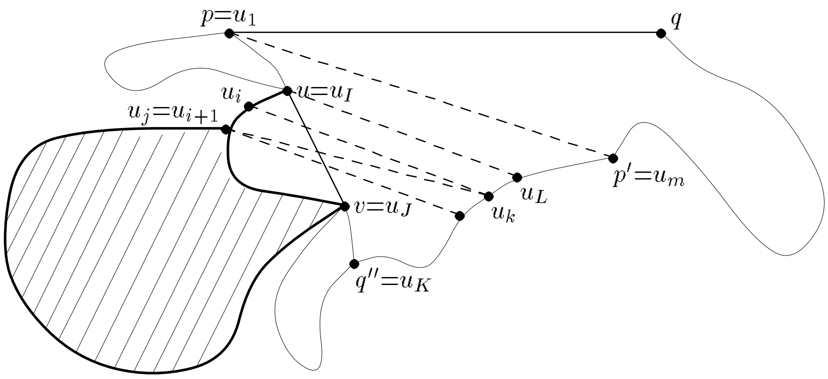

FIG. 13. *The shaded portion of the pocket Y_{uv} represents the part of the triangulation t_{uu_L} beyond uu_L that is retained for the final triangulation. The remaining portion is triangulated by fanning out from v.*

The remainder of the proof establishes that all diagonals of the thus constructed triangulation are shorter than pq. This is indeed obvious for $t_{qq''}$, as constructed in step 1. We now prove an easy extension of the Δ-Lemma which implies that all edges created in step 2 are shorter than pq.

CLAIM 1. *Let abc be a triangle and d, e two points inside abc. Then $|de| < \max\{|ab|, |ac|, |bc|\}$.*

Proof (of Claim 1). Assume without loss of generality that e lies inside abd. The Δ-Lemma for abd implies $|de| < \max\{|ad|, |bd|\}$, and the same lemma for abc implies $\max\{|ad|, |bd|\} < \max\{|ab|, |ac|, |bc|\}$. This completes the proof of Claim 1.

If $uu_L = uv$ then $|uu_L| < |pq|$ which implies that all edges of t_{uu_L}, as constructed in step 3.2, are shorter than pq. In this case the proof is complete as $X_{uv} = \emptyset$ and no edges are added to Y_{uv} in step 3.3. For the remainder of the proof we thus assume that $u_L \neq v$ which is the case only if $\eta_{\vec{uv}}$ contains at least one vertex of R. We show that a vertex u_L satisfying the conditions of step 3.1 indeed exists, and that all edges of the fan-out triangulation $f_v(X_{uv})$ are shorter than pq. Assume the sequence of vertices of the part of R beyond pp' is $p = u_1, u_2, \ldots, q'' = u_K, \ldots, u_m = p'$ (see Fig. 13).

CLAIM 2. *There exists a 1-edge uu_L that satisfies the conditions of step 3.1.*

Proof (of Claim 2). Construct a triangulation $t_{pp'}$ of R beyond pp' using the algorithm for incomplete *rng*-polygons. This triangulation contains at least one edge uu_l disjoint from C. The main invariant of the algorithm (described in Section 5.1) implies that uu_l is a 1-edge and pq lies on its beneath side. If $|uu_l| < |pq|$ then u_l satisfies the conditions for u_L and we are done.

So assume $|uu_l| > |pq|$. Similar to the Containment Lemma we can show that the part of $\eta_{\vec{uv}}$ to the left of $\vec{uu_l}$ is contained in $\eta_{\vec{uu_l}}$ and thus contains no vertex of R. It follows that the vertices in $\eta_{\vec{uv}}$ must be among $u_{K+1}, u_{K+2}, \ldots, u_{l-1}$. By the Visibility Lemma at least one of these vertices is visible from u. Let U be the subset of vertices that are visible from u (including the ones outside $\eta_{\vec{uv}}$), and let $u_L \in U$ minimize the distance to u. We have $|uu_L| < |uv| < |uu_l|$ and, as above, the part of $\eta_{\vec{uu_L}}$ to the left of $\vec{uu_l}$ is contained in $\eta_{\vec{uu_l}}$. Therefore, this part contains no vertex of R. The part of $\eta_{\vec{uu_L}}$ to the right of $\vec{uu_l}$ contains no vertex of R by the choice of u_L. It follows that uu_L is a diagonal that satisfies the conditions of step 3.1, which completes the proof of Claim 2.

We now show two easy facts about t_{uu_L} before examining the edges constructed by step 3.3.

CLAIM 3. *If $u_i u_j u_k$, with $i < j < k$, is a triangle of t_{uu_L} then $u_i u_k$ is its longest edge.*

Proof (of Claim 3). The first triangle constructed is $u_I u_l u_L$, for some $I < l < L$, and its longest edge is $u_I u_L$ because $u_l \in \lambda_{u_I u_L}$. The general assertion follows by induction, which completes the proof of Claim 3.

CLAIM 4. *The edges of t_{uu_L} that intersect uv, sorted from u to v, are monotonely decreasing in length.*

Proof (of Claim 4). If $u_i u_j u_k$, with $i < j < k$, intersects uv, $u = u_I$ and $v = u_J$, then either $I \le i < j = i + 1 \le J < k$ or $I < i < J \le j < k$ (see Fig. 13). In both cases $u_i u_k$ intersects uv closer to u than the other intersecting edge, $u_j u_k$ or $u_i u_j$. By Claim 3, $u_i u_k$ is longer than both, which implies the assertion.

Note that if we delete edges from t_{uu_L} that intersect uv, then we get a polygonal region, say W_{uv}, of which X_{uv} is the part on one side of uv. We can thus interpret uv as a generator of t_{uu_L} restricted to W_{uv}. Since the edges of X_{uv} and t_{uu_L} are shorter than $|pq|$, we just need to show that all vertices of X_{uv} are closer to v than $|pq|$, and the rest follows from the Fan-Out Corollary. Indeed, we prove a stronger bound on the maximum distance from v to a vertex of X_{uv}.

CLAIM 5. *For each vertex x of X_{uv} we have $|vx| \le |vu|$.*

Proof (of Claim 5). Consider the vertices of X_{uv} in turn from $u = u_I$ to $v = u_J$, and assume inductively that $|vu_i| \le |vu|$, for all $I \le i < j$. Consider u_j and the triangle $u_{j-1} u_j u_k$ in t_{uu_L}. By Claim 4, we have $|u_{j-1} u_k| > |u_j u_k|$. If $u_{j-1} u_j v u_k$ is a convex quadrilateral then the $\square$-Lemma implies $|vu_{j-1}| > |vu_j|$, as desired. Otherwise, u_j is contained in $vu_k u_{j-1}$ and therefore also in vuu_{j-1}. The Δ-Lemma implies $|vu_j| < \max\{|vu|, |vu_{j-1}|\}$ which completes the proof of Claim 5.

This also completes the proof of the lemma. $\square$

The following theorem summarizes the algorithmic implications of all of this.

MINMAX LENGTH THEOREM. *A minmax length triangulation of a set of n points in $\Re^2$ can be constructed in time $O(n^2)$.*

The algorithm that constructs a minmax length triangulation in the claimed amount of time is a combination of the algorithms given in Sections 3, 5.1, 5.3, and 6. Its correctness has been demonstrated in Sections 4, 5.3, and 6.

7. Arbitrary Normed Metrics. An open convex region $D \subseteq \Re^2$ that is symmetric with respect to the origin can be used to impose a *norm* on $\Re^2$: for a point $x \in \Re^2$ define $\|x\| = \|x\|_D = \alpha$ if x lies on the boundary of $\alpha D = \{\alpha y \in \Re^2 : y \in D\}$. The norm can then be used to impose a (*normed*) *metric* on $\Re^2$: for two points $x, y \in \Re^2$ define $|xy| = |xy|_D = \|y - x\|_D$. D is the *unit-disk* of the metric and the boundary of D is its *unit-circle*. Notice that the three requirements for a metric are indeed satisfied. First, $|ab| = 0$ iff $a = b$ because $\|x\| = 0$ iff x is the origin. Second, $|ab| = |ba|$ because D is centrally symmetric and therefore $\|x\| = \|-x\|$. Third, the triangle inequality, $|ac| \le |ab| + |bc|$, follows from the convexity of D. Examples of normed metrics are the l_p-metrics, for $1 \le p \le \infty$, and the so-called A-metric discussed in [WWW85] for its applications to VLSI.

In this section we assume that the triangle inequality is strict unless a, b, c lie on a line in this order. This is the case iff the defining convex region D is strictly convex, that is, no line intersects the boundary of D in more than two points. This assumption is convenient and in fact without loss of generality as every convex but not strictly convex region D' can be approximated arbitrarily closely by a strictly convex region D. Computationally, this approximation can be simulated by defining

$$\|x\|_D = \|x\|_{D'} + \epsilon \|x\|_2,$$

where $\|x\|_2$ is the Euclidean or l_2-norm and ϵ is an arbitrarily small but positive real number. Clearly, if ϵ is sufficiently small then a minmax length triangulation under D is also a minmax length triangulation under D'.

In the remainder of this section we point out where the developments in Sections 2 through 6 need to be adjusted when the Euclidean metric is replaced by an arbitrary normed metric. Most importantly, the graphs defined in Section 2 can be extended in a natural way. More specifically, the definition of $ch(S)$ remains unchanged as it makes no reference to any distance notion. If we now stipulate that "circle" means a homothetic copy of the unit-circle as defined above and "$|ab|$" means the distance under the normed metric defined by D then the definitions of $mlt(S)$, $dt(S)$, $rng(S)$, and $mst(S)$ can be taken verbatim. The minimum spanning tree, $mst(S)$, is connected and spans S, and the Delaunay triangulation, $dt(S)$, is plane because any two circles intersect in at most two points. Since we still have $mst(S) \subseteq rng(S) \subseteq dt(S)$ we conclude that all three graphs are connected and plane and they span S. We remark that these three graphs are not necessarily plane if D is not strictly convex.

As mentioned in the introduction, the developments in Sections 2 through 6 are all based on a small number of basic facts, namely the distance relations expressed by the $\Box$-Lemma and the $\triangle$-Lemma, the convexity of the lune of an edge, and the straightness of the bisector of two points. The $\Box$-Lemma and the $\triangle$-Lemma are direct consequences of the triangle inequality and hold in the stated form (with strict inequality) for arbitrary normed metrics as long as D is strictly convex. The lune of two points is clearly convex as it is the intersection of two homothetic copies of D. Unfortunately, the bisector of two points $p \neq q$, $\ell_{pq} = \{x : |xp| = |xq|\}$, is not necessarily straight. Nevertheless, ℓ_{pq} is still a simple curve that partitions $\Re^2$ into two unbounded regions, called *half-planes*, one containing p and the other q. The two half-planes are star-shaped with respect to p and q, that is, any line through p or q intersects ℓ_{pq} in at most one point. In addition, ℓ_{pq} is symmetric with respect to $\frac{p+q}{2}$ because D is centrally symmetric.

There is only one place where the straightness of the bisector is used in a substantial way, and that is in the proof of Fact 3 in Section 4. We restate this fact and show how to prove it without the use of the straightness of the bisector. We suggest the reader go back to Section 4 and remind him- or herself of Facts 1 and 2. Recall in particular that bd ($b'd'$) is said to be *switchable* if ac ($a'c'$) is no longer than the longest edge of the triangulation $t(S)$.

FACT 3. *It is not possible that both bd and $b'd'$ are non-switchable.*

Proof. As established in Fact 2, if bd is non-switchable then a and d are contained in the open half-plane defined by ℓ_{pq} that contains q. Symmetrically, if $b'd'$ is not switchable then a' and d' are contained in the other open half-plane. Unlike in the Euclidean case, it is possible that ad and $a'd'$ intersect ℓ_{pq}. It is thus also possible that ad precedes $a'd'$ in the order of edges sorted from p to q by their intersections with pq (see Fig. 14). Below we argue that if this is the case then ad (and symmetrically $a'd'$) is switchable. In particular, we show $|ad| > |ap|$ which, together with $|ap| > |ac|$ from Fact 2, implies that ad is switchable.

One characteristic of the described situation is that ad intersects ℓ_{pq} in at least one point inside the lune of pq. Let x be such an intersection point closest to a. If $pq \cap dx \neq \emptyset$ then $pdqx$ is a convex quadrilateral with $|pd| \geq |pq|$ by construction.

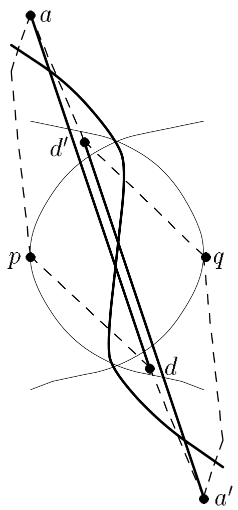

FIG. 14. *Although a and d lie on q's side of the bisector and a' and d' lie on p's side, ad intersects pq closer to p than $a'd'$ does. This is not possible if the bisector is a line as for the Euclidean metric, see Fig. 4.*

The □-Lemma thus implies $|dx| > |qx| = |px|$. It follows that $|ad| = |ax| + |dx| > |ax| + |px| > |ap|$. On the other hand, if $pq \cap dx = \emptyset$ then consider the point $y = ad \cap pq$ and note that $|py| \leq |qy|$. We derive $|dy| > |py|$ from $|py| + |dy| > |pd| \geq |pq| \geq 2|py|$. Therefore, $|ad| = |ay| + |dy| > |ay| + |py| > |ap|$ as desired. □

All other steps of the proof of the Subgraph Theorem go through unchanged for arbitrary normed metrics. We thus get the following generalization.

GENERAL SUBGRAPH THEOREM. *Let S be a finite point set in $\Re^2$ equipped with a normed metric with strictly convex unit-disk. Then S has a minmax length triangulation $mlt(S)$ so that $rng(S) \subseteq mlt(S)$.*

So the algorithm for computing a minmax length triangulation is clear – it is the same as for the Euclidean metric, only that the length of edges is now measured in terms of a normed metric possibly different from the Euclidean metric. We assume that the length of an edge in this metric can be computed in constant time. A careful reexamination of Sections 5 and 6 shows that the specialized polygon triangulation algorithm works also in the context of arbitrary normed metrics. We remark, however, that it includes the distance computation between a point and a line segment. Although it is certainly reasonable to assume that this can be done in constant time too, the observation in the remark at the end of Section 5.1 can be used to avoid this computation. We thus have the following algorithmic result which generalizes the MinMax Length Theorem of Section 6.

GENERAL MINMAX LENGTH THEOREM. *Let S be a set of n points in $\Re^2$ equipped with a normed metric with strictly convex unit-disk. Given the relative neighborhood graph, a minmax length triangulation of S can be constructed in time $O(n^2)$.*

The algorithmic result extends to arbitrary normed metrics. As mentioned above, a norm with non-strictly convex unit-disk can be simulated by one with strictly convex unit-disk. It follows that the quadratic time bound also holds for arbitrary normed metrics. The result stated in the General MinMax Length Theorem raises the question of how fast $rng(S)$ can be constructed. The trivial algorithm tests all $\binom{n}{2}$ edges, each in time $O(n)$, and therefore takes time $O(n^3)$. Faster algorithms are known for the

l_p-metrics where $\mathrm{O}(n \log n)$ time suffices (see [JKY90] and [Lee85]).

8. Discussion. The main contribution of this paper is the first polynomial time algorithm for computing a minmax length triangulation of a set S of n points in $\Re^2$. Given the relative neighborhood graph of S, the algorithm takes time $\mathrm{O}(n^2)$. The algorithm works for arbitrary normed metrics. The polynomial time bound follows because the relative neighborhood graph of S can be found in polynomial time. The question remains whether or not a minmax length triangulation can be computed in less than quadratic time.

The results of this paper are an out-growth of our general efforts to understand triangulations that optimize length criteria. There are, however, still many related problems whose complexities remain open. These include the problem of minimizing the entire vector of edge-lengths, the minimum length triangulation problem, and the maxmin length triangulation problem.

Acknowledgment. The authors thank an anonymous referee for suggestions on the organization of this paper.

REFERENCES

[ACNS82] M. Ajtai, V. Chvátal, M. M. Newborn and E. Szemerédi, *Crossing-free subgraphs*, Ann. Discrete Math., **12** (1982), pp. 9–12.

[BrZl70] J. Bramble and M. Zlámal, *Triangular elments in the finite element method*, Math. Computation, **24** (1970), pp. 809–820.

[Del34] B. Delaunay, *Sur la sphère vide*, Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk, **7** (1934), pp. 793–800.

[Edel87] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, Germany, 1987.

[EdMü90] H. Edelsbrunner and E. P. Mücke, *Simulation of Simplicity: a technique to cope with degenerate cases in geometric algorithms*, ACM Trans. Graphics, **9** (1990), pp. 66–104.

[ETW92] H. Edelsbrunner, T. S. Tan and R. Waupotitsch, *An $O(n^2 \log n)$ time algorithm for the minmax angle triangulation*, SIAM J. Sci. Stat. Comp., **13** (1992), pp. 994–1008.

[ElAv81] H. ElGindy and D. Avis, *A linear algorithm for computing the visibility polygon from a point*, J. Algorithms, **2** (1981), pp. 186–197.

[Gilb79] P. D. Gilbert, *New results in planar triangulations*, M. Sc. Thesis, Dept. Comput. Sci., Univ. Illinois, Urbana, Illinois, 1979.

[GuSt85] L. J. Guibas and J. Stolfi, *Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams*, ACM Trans. Graphics, **4** (1985), pp. 74–123.

[JKY90] J. W. Jaromczyk, M. Kowaluk and F. F. Yao, *An optimal algorithm for constructing β-skeletons in L_p metric*, To appear in *SIAM J. Comput.*

[JoSi87] B. Joe and R. B. Simpson, *Corrections to Lee's visibility polygon algorithm*, BIT, **27** (1987), pp. 458–473.

[Klin80] G. T. Klincsek, *Minimal triangulations of polygonal domains*, Ann. Discrete Math., **9** (1980), pp. 121–123.

[Lank69] P. M. Lankford, *Regionalization: theory and alternative algorithms*, Geographical Analysis, **1** (1969), pp. 196–212.

[Laws77] C. L. Lawson, *Software for C^1 surface interpolation*, in Math. Software III, J. R. Rice, ed., Academic Press, 1977, pp. 161–194.

[Lee83] D. T. Lee, *Visibility of a simple polygon*, Computer Vision, Graphics, and Image Process., **22** (1983), pp. 207–221.

[Lee85] D. T. Lee, *Relative neighborhood graphs in the L_1-metric*, Pattern Recognition, **18** (1985), pp. 327–332.

[Ling87] A. Lingas, *A new heuristic for minimum weight triangulation*, SIAM J. Algebraic Discrete Methods, **8** (1987), pp. 646–658.

[Lloy77] E. L. Lloyd, *On triangulations of a set of points in the plane*, in Proc. 18th Ann. IEEE Sympos. Found. Comput. Sci. (1977), pp. 228–240.

[PlHo87] D. A. Plaisted and J. Hong, *A heuristic triangulation algorithm*, J. Algorithms, **8** (1987), pp. 405–437.

[PrSh85] F. P. Preparata and M. I. Shamos, *Computational Geometry — an Introduction*, Springer-Verlag, New York, 1985.

[Raja91] V. T. Rajan, *Optimality of the Delaunay triangulation in* $\Re^d$, in Proc. 7th Ann. Sympos. Comput. Geom. (1991), pp. 357–363.

[Sibs78] R. Sibson, *Locally equiangular triangulations*, Comput. J., **21** (1978), pp. 243–245.

[Supo83] K. J. Supowit, *The relative neighborhood graph, with an application to minimum spanning trees*, J. Assoc. Comput. Mach., **30** (1983), pp. 428–448.

[Tous80] G. T. Toussaint, *The relative neighbourhood graph of a finite planar set*, Pattern Recognition, **12** (1980), pp. 261–268.

[WGS90] F. W. Wilson, R. K. Goodrich and W. Spratte, *Lawson's triangulation is nearly optimal for controlling error bounds*, SIAM J. Numer. Anal., **27** (1990), pp. 190–197.

[Wism80] S. K. Wismath, *Triangulations: an algorithmic study*, Techn. Rep. 80-106, M. Sc. Thesis, Dept. Comput. Sci., Queen's Univ., Kingston, Ontario, 1980.

[WWW85] P. Widmayer, Y. F. Wu and C. K. Wong, *Distance problems in computational geometry with fixed orientation*, in Proc. 1st Ann. Sympos. Comput. Geom. (1985), pp. 186–195.