Comp. & Maths. with Appls. Vol. 9, No. 6, pp. 747–754, 1983
Printed in Great Britain.

0097-4943/83 $3.00 + .00

THE SYMMETRIC ALL-FURTHEST-NEIGHBOR PROBLEM

GODFRIED T. TOUSSAINT
School of Computer Science, McGill University, 805 Sherbrooke St. West, Montreal, Quebec, Canada H3A 2K6

(*Received June* 1982)

Communicated by E. Y. Rodin

Abstract—Given a set $\mathcal{P}$ of n points on the plane, a *symmetric furthest-neighbor* (SFN) pair of points p, q is one such that both p and q are furthest from each other among the points in $\mathcal{P}$. A pair of points is *antipodal* if it admits parallel lines of support. In this paper it is shown that a SFN pair of $\mathcal{P}$ is both a set of *extreme* points of $\mathcal{P}$ and an *antipodal* pair of $\mathcal{P}$. It is also shown that an *asymmetric furthest-neighbor* (ASFN) pair is not necessarily *antipodal*. Furthermore, if $\mathcal{P}$ is such that no two distances are equal, it is shown that as many as, and no more than, $\lfloor n/2 \rfloor$ pairs of points are SFN pairs. A polygon is *unimodal* if for each vertex p_k, $k = 1, \ldots, n$ the distance function defined by the euclidean distance between p_k and the remaining vertices (traversed in order) contains only one local maximum. The fastest existing algorithms for computing all the ASFN or SFN pairs of either a set of points, a simple polygon, or a convex polygon, require $0(n \log n)$ running time. It is shown that the above results lead to an $0(n)$ algorithm for computing all the SFN pairs of vertices of a *unimodal* polygon.

1. INTRODUCTION

The dual of the *all-furthest-neighbor* problem is the *all-nearest-neighbor* problem which consists of finding the nearest point to every point in a set. The latter problem has received considerable attention recently in computational geometry. Shamos and Hoey[1] have shown that $\Omega(n \log n)$ is a lower bound to this problem and they suggest an $0(n \log n)$ algorithm using the closest-point Voronoi diagram (CPVD). A property of the CPVD is that the perpendicular bisector of any point p_i and its nearest neighbor p_i' coincides with an edge of the Voronoi polygon V_i associated with p_i. Thus it is sufficient to examine each V_i once and find the Voronoi edge closest to p_i for all i. Since there are no more than $3n - 6$ edges to be considered, $0(n)$ time suffices once the CPVD has been obtained.

The $\Omega(n \log n)$ lower bound does not apply when the input is a convex polygon rather than an arbitrary set of points. If the CPVD of a convex polygon could be computed in less than $0(n \log n)$ time one could use the approach of Shamos and Hoey[1] to solve the convex polygon problem. However, no such algorithm is known. In a completely different approach, Lee and Preparata[2] show that the convexity property is sufficient to obtain an $0(n)$ algorithm and they offer an algorithm that makes use of the diameter of the polygon. Yang and Lee[3] propose a simpler $0(n)$ algorithm that does not require the computation of the diameter, to which Fournier and Kedem[4] add a caveat.

In this note we assume that the set of points $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ is given in terms of the cartesian coordinates of the p_i. It is further assumed that the points are in *general* position in the sense that no three are collinear and no four are cocircular. When the set forms a simple polygon it will be denoted by $P = (p_1, p_2, \ldots, p_n)$ where the vertices are given in order in terms of their Cartesian coordinates and are in general position. All indices are taken modulo n.

The *all-furthest-neighbor* problem for a set of points $\mathcal{P}$ (vertices in the case of a polygon) is to find for each point $p_i \in \mathcal{P}$ another point $p_j \in \mathcal{P}$, $j \neq i$ such that

$$d(p_i, p_j) = \max_k \{d(p_i, p_k)\}, \quad k = 1, 2, \ldots, n,$$

where $d(p_i, p_j)$ denotes the Euclidean distance between p_i and p_j. Alternately we can construct the *furthest-neighbor graph* (FNG) by joining two points p_i, p_j with an edge if at least one of p_i, p_j is the *furthest neighbor* of the other. The obvious approach to computing the FNG($\mathscr{P}$) leads to an $0(n^2)$ algorithm. Under the assumption that the furthest neighbor of a point p_i must be a furthest-point-Voronoi-diagram (FPVD) neighbor of p_i Shamos[5] proposed an $0(n \log n)$ algorithm to solve this problem that mimicks the dual closest-point problem. However, Toussaint and Bhattacharya[6] exhibit a counterexample to the above assumption which invalidates this algorithm. They then go on to propose two new algorithms to solve this problem. Algorithm FNG-1 always runs in $0(n \log n)$ time but is complicated. Algorithm FNG-2 is very simple and runs in $0(n)$ *expected* time for a wide range of distributions of the points, but has the drawback of an $0(n^2)$ *worst-case* running time. No $0(n)$ *worst-case* algorithm is known for the all-furthest-neighbor problem for *convex* or *simple* polygons. In [15] it is shown that a linear time complexity can be obtained for *convex unimodal* polygons. A simple polygon is *unimodal* if for each vertex p_k, $k = 1, \ldots, n$ the distance function defined by the Euclidean distance between p_k and the remaining vertices (traversed in order) contains only one local maximum. Note that *unimodal* polygons need not be convex although, by definition, we only consider simple unimodal polygons.

In this paper we are concerned with computing all the *symmetric furthest-neighbor* (SFN) pairs of a *unimodal* polygon P. A pair of vertices $p_i, p_j \in P$ is a *symmetric furthest-neighbor* (SFN) pair if, and only if,

$$d(p_i, p_j) = \max_k \{d(p_i, p_k)\} \quad \text{and} \quad d(p_i, p_j) = \max_k \{d(p_k, p_j)\}$$

for $k = 1, 2, \ldots, n$. When only one of these conditions holds for a pair of points it will be referred to as an *asymmetric furthest-neighbor* (ASFN) pair. Clearly the SFN graph is a subgraph of the ASFN graph and the diameter of the set is contained as an edge in the SFN graph. However, alghough a SFN pair appears to be a close relative of the diameter, quite a number of pairs of points can have this property. As is shown in Section 2, as many as $0(n)$ pairs of vertices of $\mathscr{P}$ can be SFN pairs even when no two distances in $\mathscr{P}$ are equal.

In Section 3 we show how all the SFN pairs of a *unimodal* polygon can be found in $0(n)$ time which is optimal to within a multiplicative constant.

2. THE NUMBER OF SYMMETRIC FURTHEST-NEIGHBOUR PAIRS

The number of distances that can be realized in a finite planar set is a topic which has been of interest in *combinatorial geometry* for some time. The diameter of a set $\mathscr{P}$, denoted by $D(\mathscr{P})$, is defined as follows:

$$D(\mathscr{P}) = \max_{i,j} \{d(p_i, p_j)\}$$

for $i, j = 1, 2, \ldots, n$.

For an arbitrary set $\mathscr{P}$, no more than n pairs of points can realize $D(\mathscr{P})$, and this is achievable[8, 9]. A more accessible proof of this result is given by Erdös[10]. An example of such a set is illustrated in Fig. 1. The triangle $p_1 p_2 p_n$ is equilateral with sides equal to unity. Points $p_3, \ldots, p_{n-1}$ lie on the arc with center at p_1 and radius equal to 1. It is clear that (p_1, p_i), $i = 2, 3, \ldots, n$ yields $n - 1$ pairs and (p_2, p_n) forms the nth pair. Avis[7] has shown that the number of SFN pairs also does not exceed n. Based on the above facts there does not appear to be much difference between the diameter of a set and a furthest-neighbor pair as far as the density of their graphs is concerned. Note however that in Fig. 1 we allow as many pairs of distances to be equal as we wish. Consider now a set $\mathscr{P}$ in which no two distances are equal. Then clearly only *one* pair of points realizes the diameter. We now prove the following theorem.

Theorem 1

For n points, such that no two distances are equal, there are no more than $\lfloor n/2 \rfloor$ pairs of symmetric furthest-neighbor pairs, and this bound is achievable.

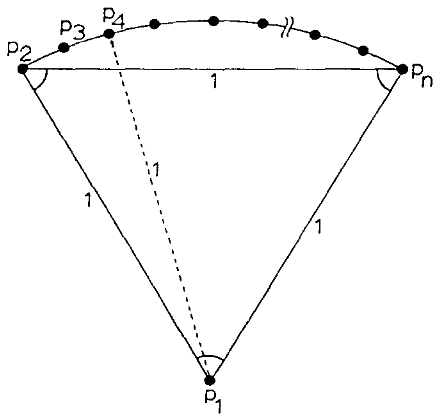

Fig. 1.

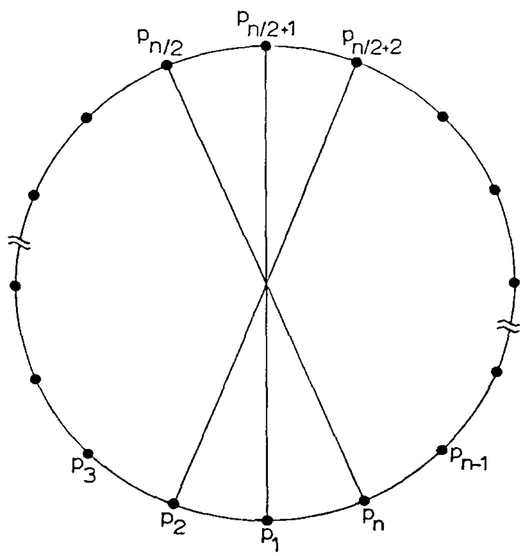

Fig. 2.

Fig. 1. Illustrating the fact that n pairs or points can realize the diameter of a set.

Fig. 2. Illustrating that the bound of Theorem 1 is achievable.

Proof. Construct the SFN graph by joining two points p, q with an edge if both p and q are furthest neighbors of each other. Consider any vertex r of degree at least two in this graph. Then at least two vertices are furthest neighbors of r. This can only occur if the distances from r to its neighbors are equal which contradicts the assumption. Therefore the degree of each vertex is at most one. Hence the SFN graph is a subgraph of a *matching* and thus has no more than $\lfloor n/2 \rfloor$ edges. To show that this bound is achievable consider the case when n is even and the SFN graph must have $n/2$ edges. Place n points on a circle an equal distance apart and refer to Fig. 2. Clearly the $n/2$ diametrically opposite pairs, such as $(p_1, p_{n/2+1})$, are SFN pairs, since the diameter of the circle is greater than any other chord. We must now perturb all the points so that (a) no two distances are equal and (b) previous diametrically opposite pairs, such as $(p_1, p_{n/2+1})$, remain further apart than other pairs. Condition (a) is easy to satisfy: move point p_i, $i = 1, \ldots, n$ to a new location p_i^* such that p_i^* is chosen at random from a uniform distribution over a disk of radius δ centered at p_i. The probability that two distances are equal is then zero. Let $d(p_{n/2+1}, p_1) - d(p_{n/2}, p_1) = \epsilon$. It is straight forward to verify that condition (b) is satisfied if $\delta < \epsilon/4$. Q.E.D.

Thus we see that when points are such that no two distances are equal only *one* pair realizes the diameter whereas $0(n)$ SFN pairs may exist. In any case since the size of the output is linear, it makes sense to look for sub-quadratic algorithms to solve this problem—a topic to which we now turn.

3. ALGORITHMS

Consider first the problem of computing all the SFN pairs of a finite planar set $\mathscr{P}$. A straightforward approach leads to an $0(n^2)$ algorithm. Since a SFN pair is an ASFN pair, one approach is to first compute the ASFN pairs and subsequently select the SFN pairs from among the ASFN pairs.

Algorithm SFN-1

Step 1. Compute all ASFN pairs.

Step 2. Select the SFN pairs from among the ASFN pairs.

Step 1 can be computed in $0(n \log n)$ time using any one of several algorithms presented in [6]. It is a simple matter to go through the list of $0(n)$ ASFN pairs and in linear time select the SFN pairs. Thus the algorithm is dominated by step 1 and runs in $0(n \log n)$ time.

If we are given a simple or convex polygon P no algorithms faster than SFN-1 are available. However, if P is both *convex* and *unimodal*, it is shown in [15] that step 1 of SFN-1 can be done in $0(n)$ time thus yielding a linear algorithm for the SFN problem. We now show that all the SFN pairs can be computed in $0(n)$ time for arbitrary *unimodal* polygons.

Definition. A pair of points is an *antipodal* pair if it admits parallel lines of support.

THEOREM 2

A SFN pair is *antipodal.*

Proof. Let $a, b \in \mathscr{P}$ be a SFN pair. Let LUNE(a, b) denote the intersection of two circles each with radius equal to $d(a, b)$ one centered at a and the other at b, and refer to Fig. 3. Since b is the furthest point from a and a is the furthest point from b, it follows that no points lie outside LUNE(a, b). Construct two parallel lines L_a and L_b passing through a and b and tangential to LUNE(a, b). Since LUNE(a, b) is contained in the infinite slab determined by L_a and L_b, so is $\mathscr{P}$. Therefore L_a and L_b are parallel lines of support and (a, b) is an *antipodal* pair. Q.E.D.

Note that the converse is not necessarily true. For consider three points a, b, c that form an isoceles triangle with base (a, b) and $d(a, b) < d(a, c) = d(b, c)$. Clearly (a, b) is an *antipodal* pair but the furthest neighbor of both a and b is c. Thus this example shows that an *antipodal* pair need not be even an ASFN pair.

THEOREM 3

As ASFN pair need not be *antipodal.*

Proof. Consider a parallelogram $abcd$ composed of the union of two right-angled triangles abc and acd and refer to Fig. 4. Let the sides ab and cd have length x and let the diagonal ac have length y, where $y \gg x$. Let the circle of radius x centered at a intersect the line through ac at f. Now place four points as follows: three at a, c and d, and the fourth e on arc bf such that $0° < \theta < 90°$. Clearly d is the furthest point of a while e is the furthest point of d and thus (a, d) is an ASFN pair. But (a, d) is not an *antipodal* pair as it does not admit parallel lines of support. Q.E.D.

Definition. An *extreme* point p of a convex polygon P (or set of points $\mathscr{P}$) is a point such that there does not exist a line segment $(a, b) \in P$ (or $(a, b) \in CH(\mathscr{P})$, where CH denotes convex hull) with p lying in the interior of (a, b).

Let $VCH(\mathscr{P})$ denote the set of *extreme* points of the convex hull of $\mathscr{P}$. We than have the following theorem.

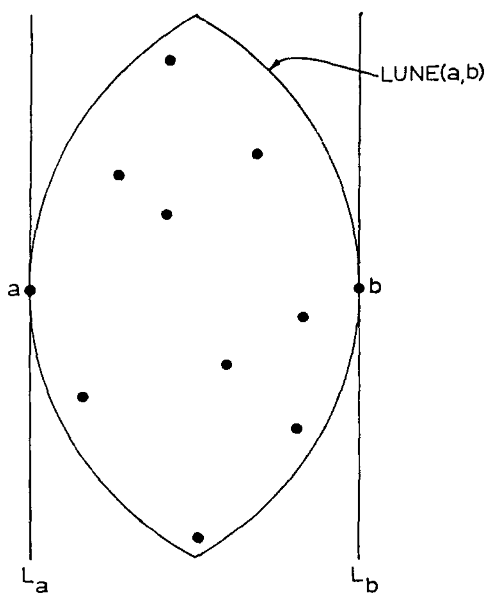

Fig. 3. Illustrating the fact that a SFN pair is an *antipodal* pair.

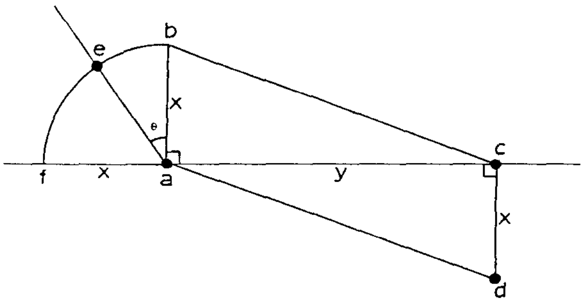

Fig. 4. Illustrating the fact that an ASFN pair need not be *antipodal*.

THEOREM 4

If (p, q) is a SFN pair then $p, q \in VCH(\mathscr{P})$.

Proof. Assume that $q \notin VCH(\mathscr{P})$. Extend the line from p through q to intersect an edge, say (p_i, p_j) of $CH(\mathscr{P})$ at x and refer to Fig. 5. Either px is perpendicular to p_ip_j or it is not. If it is then both p_i and p_j are further from p than q is. Therefore (p, q) is not a SFN pair, a contradiction. If px is not perpendicular to p_ip_j then one of the angles at x is greater than 90°. Let angle $pxp_j > 90°$. Then $d(p, p_j) > d(p, x) \geq d(p, q)$, a contradiction. Similar arguments hold if both p and q are not extreme points. Therefore $(p, q) \in VCH(\mathscr{P})$.

Q.E.D.

Let $CH(P)$ denote the convex polygon determined by the convex hull of P. We then have the following theorem.

THEOREM 5

If a simple polygon P is *unimodal*, then $CH(P)$ is also *unimodal*.

Proof. From Jordan's Curve Theorem it follows that given a simple polygon P the convex hull vertices of P occur in the same order in $CH(P)$ as they do in P. Therefore for each vertex $p_k \in CH(P)$ the distance function is defined for an ordered subset of the arguments of the corresponding distance function for $p_k \in P$. Therefore if the latter distance function is unimodal, so is the former. Q.E.D.

Theorems 2 and 4, together with the fact that a convex polygon has only $0(n)$ *antipodal* pairs[11, 12], yields the following linear algorithm for finding all the SFN pairs of a *unimodal* polygon P. For simplicity it is assumed in the description below that no two distances between the vertices of P are equal. Modifications can be made to the algorithm, without affecting the linear time complexity, to handle the case of equal distances but their inclusion drowns the core of the algorithm in irrelevant details.

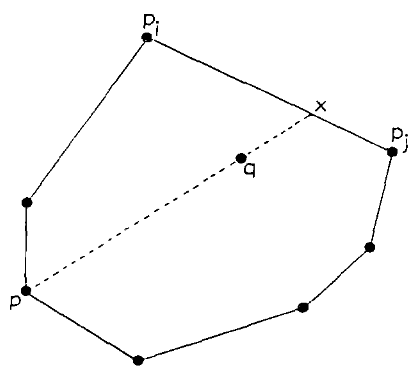

Fig. 5. Illustrating the fact aht if (p, q) is a SFN pair of $\mathscr{P}$ then p, q are extreme points of $\mathscr{P}$.

Algorithm SFN–UPOL

Input. A unimodal polygon $P = (p_1, p_2, \ldots, p_n)$.

Output. All SFN pairs of vertices.

Step 1. Find $CH(P)$.

Step 2. Generate all antipodal pairs of $CH(P)$.

Step 3. For each antipodal pair $\{p_i, p_j\} \in CH(P)$ test whether vertices $\{p_{i-1}, p_{i+1}, p_{j-1}, p_{j+1}\} \in P$ lie in the interior of LUNE(p_i, p_j); if all four are included in the interior of the lune then $\{p_i, p_j\}$ is a SFN pair; otherwise not.

THEOREM 6

Algorithm SFN–UPOL computes all the SFN pairs of a *unimodal* polygon in $0(n)$ time.

Proof. The correctness of step 1 follows from theorem 4, i.e. we can neglect vertices of P which are not convex hull vertices. In addition $CH(P)$ can be computed in $0(n)$ time even for an arbitrary simple polygon[14]. The correctness of step 2, in further reducing the pairs of vertices to be searched, follows from theorem 2. Shamos[11] and Brown[12] give two $0(h)$ algorithms for generating all the *antipodal* pairs of $CH(P)$ where h is the number of vertices on $CH(P)$. In the worst case step 2 runs in $0(n)$ time. Finally consider step 3. Since there are $0(n)$ pairs of antipodal vertices and for each of these the lune inclusion tests require only $0(1)$ time, step 3 runs in $0(n)$ time. The correctness of step 3 follows from the *unimodality* of P, for if $d(p_i, p_{j-1}) < d(p_i, p_j)$ and $d(p_i, p_{j+1}) < d(p_i, p_j)$ then p_j is the furthest vertex from p_i. Q.E.D.

Another linear algorithm can be obtained using the ASFN algorithm of [15] for *convex unimodal* polygons. First compute $CH(P)$. Now, from theorem 5 it follows that $CH(P)$ is a *convex unimodal* polygon. Thus with the algorithm of [15] we can solve the ASFN and SFN problems for the $CH(P)$. While this does not solve the ASFN problem for P, it does so for the SFN case due to theorem 4.

As stated in the proof of theorem 6 above, step 1 of algorithm SFN–UPOL can always be computed in $0(n)$ time using the algorithm of McCallum and Avis[14] which will work for arbitrary *simple* polygons. However, since *unimodal* polygons have additional structure and since the algorithm in [14] is relatively complex compared to steps 2 and 3 of algorithm SFN–UPOL, one wonders whether a much simpler convex hull algorithm than that in [14] will work for *unimodal* polygons. In [16] it is shown that an exceedingly simple convex hull algorithm due to Sklansky[17] works for a class of polygons known as *weakly externally visible* polygons.

Let $bd(P)$ denote the boundary of a simple polygon P. Let ray(x) denote an infinite half-line starting at point x and proceeding in any direction. A simple polygon is said to be *weakly externally visible* if, and only if, for every $x \in bd(P)$ there exists a ray(x) such that $P \cap \text{ray}(x) = x$. Intuitively, consider a polygon P to be completely surrounded by a circle. If P is weakly externally visible then the entire boundary of P is visible at one time or another as a guard patrols along the circle. We now show that *unimodal* polygons are *weakly externally visible* (w.e.v.) and thus the simple algorithm of [17] can be used in step 1 of SFN–UPOL.

Let $\overline{p_i p_j}$ be an edge of the convex hull of a polygon P such that it is not an edge of P itself. Then p_i and p_j, $i < j$, are the vertices of $\overline{p_i p_j}$ and they determine two polygonal chains: the left chain $LC(p_i, p_j)$ and the right chain $RC(p_i, p_j)$. Let $HL(p_i, p_j)$ denote the half-line from p_i in the direction of p_j. We then have the following lemma.

LEMMA 1

Given any vertex $p_k \in LC(p_i, p_j)$ and one of the vertices of $\overline{p_i p_j}$, say p_i, there exists a vertex $p_m \in RC(p_i, p_j)$ such that $d(p_i, p_m) > d(p_i, p_k)$.

Proof. Construct the half-line $L = HL(p_i, p_k)$ and refer to Fig. 6. From Jordan's curve theorem it follows that L must intersect P beyond p_k. Furthermore L must intersect at least one point in $RC(p_i, p_j)$. Let x denote the first such intersection of L with some edge $\overline{p_l p_{l+1}} \in RC(p_i, p_j)$. If x is a vertex of p we are done, for $d(p_i, x) > d(p_i, p_k)$. If not we have three cases: (a) $\angle p_i x p_{l+1} = 90°$, (b) $\angle p_i x p_{l+1} < 90°$, and (c) $\angle p_i x p_{l+1} > 90°$. In case (a) it follows from elementary geometry that both p_l and p_{l+1} are further from p_i than p_k is and thus $m = l$ or $l + 1$. In case (b) $m = l$ and in case (c) $m = l + 1$. Q.E.D.

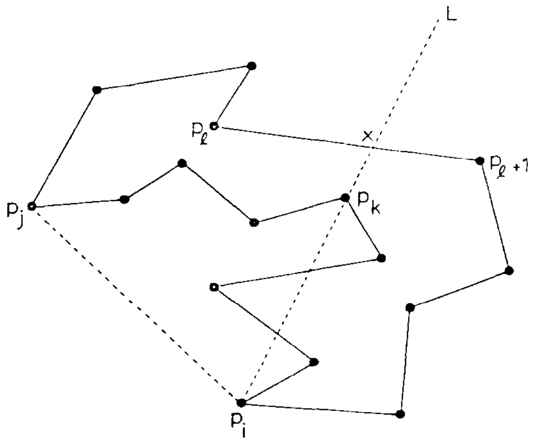

Fig. 6. Illustrating the proof of lemma 1.

Definition. A polygon P is *weakly visible* from an edge $\overline{uv}$ if for every point $x \in P$ there exists a point $y \in \overline{uv}$ such that the interior of $\overline{xy}$ lies in the interior of P.

LEMMA 2

A polygon is *weakly visible* from $\overline{uv}$ if, and only if, every vertex of P is visible from some point on $\overline{uv}$.

Proof. The proof of this lemma is given in [18].

Definition. A *deficiency polygon* of P is a polygon determined by the union of an edge such as $\overline{p_ip_j}$, $i < j$, with the polygonal chain $CL(p_i, p_j)$. Note that a polygon is w.e.v. if all its deficiency polygons are *weakly visible* from their corresponding convex hull edges.

THEOREM 7

A *unimodal* polygon is *weakly externally visible.*

Proof. (By contradiction.) Assume that we have a unimodal polygon and it is not w.e.v., and refer to Fig. 7. Then (by lemma 2) there must exist a deficiency polygon determined by some edge $\overline{p_ip_j}$ of $CH(P)$ such that $LC(p_i, p_j)$ contains at least one vertex not visible from any point on $\overline{p_ip_j}$. Let p_k be the first such vertex encountered in traversing $RC(p_j, p_i)$. By Jordan's curve theorem it follows that, since p_k is connected to p_i via $LC(p_i, p_k)$, the half-line $HL(p_i, p_{k+1})$ must intersect $LC(p_i, p_k)$ beyond p_{k+1}. Let $y \in \overline{p_lp_{l+1}}$ be one such intersection point. From arguments similar to those used in the proof of lemma

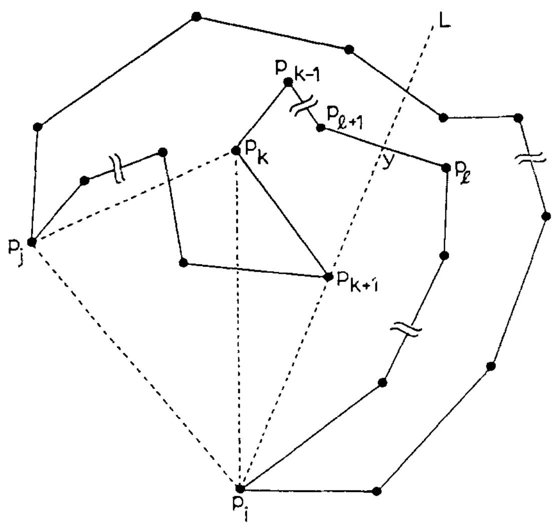

Fig. 7. Illustrating the proof of Theorem 7.

1 it follows that either p_l or p_{l+1} is further from p_i than p_{k+1} is. Therefore the distance function for p_i obtains at least one local maximum in traversing $LC(p_i, p_{k+1})$. From lemma 1 it follows that there exists a vertex $p_m \in RC(p_i, p_j)$ such that $d(p_i, p_m) > \max\{d(p_i, p_l), d(p_i, p_{l+1})\}$. Therefore the distance function for p_i obtains at least one local maximum on $LC(p_{k+1}, p_i)$. Therefore in traversing the entire polygon the distance function for p_i obtains at least two local maxima which is a contradiction since P is unimodal. Q.E.D.

4. CONCLUDING REMARKS

An $0(n)$ algorithm, based on searching only the *antipodal* pairs, has been presented for finding all the SFN pairs of vertices of a *unimodal* n-vertex polygon. One open problem that remains is an $0(n)$ algorithm for computing all the ASFN pairs of a *unimodal* polygon. Another open problem is an $0(n)$ algorithm for finding all the ASFN or SFN pairs of a *convex* polygon. The ASFN problem cannot be solved by searching only the *antipodal* pairs since as theorem 3 demonstrates not all ASFN pairs are *antipodal*.

Acknowledgement—The author is grateful to David Avis and Binay Bhattacharya for several useful discussions on this topic.

REFERENCES

1. M. I. Shamos and D. Hoey, Closest-point problems. *Sixteenth Ann. IEEE Symp. on Foundations of Computer Science*, pp. 151–162. October (1975).
2. D. T. Lee and F. P. Preparata, The all nearest-neighbor problem for convex polygons. *Inform. Proc. Letters* **7**, 189–192 (1978).
3. C. C. Yang and D. T. Lee, A note on the all nearest-neighbor problem for convex polygons. *Inform. Proc. Letters* **8**, 193–194 (1979).
4. A. Fournier and Z. Kedem, Comments on the all nearest-neighbor problem for convex polygons. *Inform. Proc. Letters* **9**, 105–107 (1979).
5. M. I. Shamos, *Problems in Computational Geometry*. Carnegie-Mellon University (1977).
6. G. T. Toussaint and B. K. Bhattacharya, On geometric algorithms that use the furthest-point Voronoi diagram. Tech. Rept. No. SOCS-81.3, School of Computer Science, McGill University, January (1981).
7. D. Avis, The number of furthest-neighbor pairs of a finite planar set. Tech. Rept. No. SOCS-81.8, School of Computer Science, McGill University, February 1981.
8. H. Hopf and D. Pannwitz, Jahresbericht der Deutsch. *Math. Vereinigung* **43**, 2 Abt., p. 114, problem 167 (1934).
9. J. W. Sutherland, Jahresbericht der Deutsch. *Math. Vereinigung* **45**, 2 Abt., p. 33 (1935).
10. P. Erdös, On sets of distances of *n* points. *Am. Math. Monthly* **53**, 248–250 (1946).
11. M. I. Shamos, Computational geometry. Ph.D. thesis, Yale University, May (1978).
12. K. Q. Brown, Geometric transforms for fast geometric algorithms. Ph.D. thesis, Carnegie-Mellon University, December (1979).
13. G. T. Toussaint, Pattern recognition and geometrical complexity. *Proc. Fifth International Conf. on Pattern Recognition*, pp. 1324–1347. Miami Beach, December (1980).
14. D. McCallum and D. Avis, A linear algorithm for finding the convex hull of a simple polygon. *Inform Proc. Letters* **9**, 201–206 (1979).
15. G. T. Toussaint, Complexity, convexity, and unimodality. Tech. Rept. S.O.C.S.-81.22, School of Computer Science, McGill University, July 1981.
16. G. T. Toussaint and D. Avis, On a convex hull algorithm for polygons and its application to triangulation problems. *Pattern Recognition* **15**(1), 23–29 (1982).
17. J. Sklansky, Measuring concavity on a rectangular mosaic. *IEEE Trans. Comput.* **C-21**, 1355–1364 (1972).
18. D. Avis and G. T. Toussaint, An optimal algorithm for determining the visibility of a polygon from an edge. *IEEE Trans. Comput.* **C-30**(12), 910–914 (1981).