Discrete Comput Geom 4:611-626 (1989)

Discrete & Computational Geometry


# Computing the Geodesic Center of a Simple Polygon*

R. Pollack,[1] M. Sharir,[1,2] and G. Rote[3]

[1] Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA

[2] School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel

[3] Institut für Mathematik, Technische Universität Graz, Austria

**Abstract.** The geodesic center of a simple polygon is a point inside the polygon which minimizes the maximum internal distance to any point in the polygon. We present an algorithm which calculates the geodesic center of a simple polygon with $n$ vertices in time $O(n \log n)$.

## 1. Introduction

The problem addressed in this paper, to locate the point inside a simple polygon $P$ whose maximal internal distance (by a route inside $P$) from any point inside $P$ is minimal, is a generalization of the Euclidean facility location problem which asks for the location of the point (facility) which is least far (in the Euclidean metric) from the furthest of a finite set of points (the community which the facility is to serve) [Me], [Dy2]. Indeed, since the furthest point from a point inside a polygon is always a vertex of the polygon, the geodesic center of the convex hull of the community to be served is the solution to the standard facility location problem. We can consider the problem of finding the geodesic center as another kind of constrained facility location problem where we want, e.g., to locate an

* Work on this paper by the first author has been supported by National Science Foundation Grant No. DMS-8501947. Work on this paper by the second author has been supported by Office of Naval Research Grant No. N00014-82-K-0381, National Science Foundation Grant No. NSF-DCR-83-20085, and by grants from the Digital Equipment Corporation, and the IBM Corporation. Part of the work on this paper by the first two authors has been carried out at the Workshop on Movable Separability of Sets at the Bellairs Research Institute of McGill University, Barbados, February 1986. Work on this paper by the third author has been supported by the Fonds zur Förderung der wissenschaftlichen Forschung (FWF), Project S32/01.

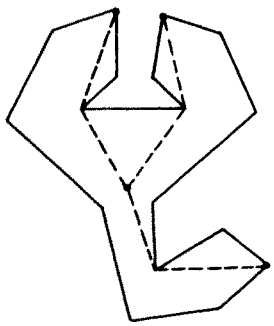

Fig. 1

emergency service on a polygonal island or a nurses station on a polygonal hospital floor. See Fig. 1 for an illustration of the geodesic center problem. The standard Euclidean facility location problem can be solved in time $O(n)$ [Me1], [Dy2], but its extension to the problem of finding the geodesic center of a simple polygon appears to be more difficult.

The problem of computing the geodesic center of a simple polygon has been considered by Asano and Toussaint [AT]. They show that the geodesic center is unique and present an algorithm to compute it in time $O(n^4 \log n)$, where $n$ is the number of vertices in the given polygon. The main idea of their algorithm is to construct the geodesic furthest-point Voronoi diagram of the vertices of the polygon and then to locate the geodesic center at either a vertex of the Voronoi diagram or at the midpoint of a geodesic diameter (i.e., a shortest path inside the polygon joining two vertices which has maximal length over all choices of pairs of vertices). We will also use the term "geodesic diameter" to denote the length of that path. There have been many algorithms to find the geodesic diameter of a simple polygon. The best result at the present time is an $O(n \log n)$-time and $O(n)$-space algorithm due to Suri [Su1].

A related problem is to compute the link diameter and the link center of a simple polygon, where the link distance between two points is the minimum number of edges in a polygonal path joining them inside the polygon and where the link center and diameter are defined in an analogous manner to the definition of geodesic center and diameter. In this case the link center is no longer unique but consists of a polygon which may be as large as the entire given polygon. Suri [Su2] has an $O(n \log n)$-time and $O(n)$-space algorithm which computes the link diameter of a simple polygon and Lenhart *et al.* [Lea] presents an $O(n^2)$ algorithm for computing the link center of a simple polygon. El-Gindy (private communication) also reports similarly efficient algorithms for computing the link center.

Our algorithm proceeds as follows. We start with a triangulation of the polygon $P$, then perform something like a binary search through the diagonals of the triangulation, determining at each tested diagonal, via the algorithm RELCEN to be described in Section 3 below, on which side of that diagonal the geodesic center lies. In this way we locate a triangle which contains the geodesic center.

However, as we move around in this triangle the combinatorial structure of the shortest path to a given vertex can change. We next use a modification of Megiddo's method for solving linear programming problems in linear time [Me1] (see also Dyer [Dy1]) to find a polygon $R$ (containing the geodesic center) within this triangle where the structure of the path from each point $x \in R$ to each vertex $v$ of $P$ remains constant, and consists of the straight segment from $x$ to some, possibly different, vertex followed by a fixed path of known length to $v$. In other words, in our final subregion $R$, the internal distance from a point $x$ to the $i$th vertex of $P$ has a fixed analytic expression of the form $|xu_i| + c_i$, for $i = 1, 2, \ldots, n$ (where $|xu_i|$ denotes the Euclidean distance from $x$ to $u_i$). The problem that remains is equivalent to that of finding a smallest circle that contains a given collection of $n$ circles (where the $i$th circle is centered at $u_i$ and has radius $c_i$). This final task can be accomplished in linear time by a recent algorithm of Megiddo [Me4].

The total time complexity of our algorithm is $O(n \log n)$.

The paper is organized as follows. In Section 2 we present the geometric preliminaries needed to justify the algorithm RELCEN which is presented in Section 3. In Section 4 we present the complete algorithm, and concluding remarks are given in Section 5.

## 2. Geometric Preliminaries

**Definitions.** Let $P$ be a simple polygon with $n$ sides; that is $P$ is a closed and bounded simply connected planar region whose boundary consists of $n$ (nonintersecting) line segments. We assume that $P$ is given as input by the circular sequence of its vertices in their (say, counterclockwise) order along the boundary of $P$. For $x$, $y$ in $P$, $g(x, y)$ is the shortest path inside $P$ between $x$ and $y$, $d(x, y)$ is the length of this path. Furthermore, let $xy$ denote the straight line segment with endpoints $x$ and $y$ and let $|xy|$ denote its length. Finally, we let $\mathbf{u}(x, y)$ denote the unit vector in the direction that the path $g(x, y)$ starts at from $x$, and define the *geodesic angle* $\angle xyz$ as the smaller of the two angles between $\mathbf{u}(y, x)$ and $\mathbf{u}(y, z)$. A subset $C$ of $P$ is called *P-convex* if whenever $x, y$ are in $C$, $g(x, y)$ is a subset of $C$. It immediately follows that the intersection of $P$-convex sets is $P$-convex, and that $P$-convex sets are connected.

As shown in Lee and Preparata [LP], $g(x, y)$ is a polygonal path whose corners are vertices of $P$. For a fixed point $x$ in $P$, the union of $g(x, v)$ over all vertices $v$ of $P$ is a planar tree $Q(x)$ (rooted at $x$), which we call the *shortest path tree* of $P$ (with respect to $x$). This tree has $n$ nodes, namely the vertices of $P$, and its edges are straight segments connecting these nodes. It has been shown by Guibas *et al.* [Gea] that, given a triangulation of $P$, this tree can be computed in linear time.

Given three points $a$, $b$, $c$ in $P$, consider the geodesic paths $g(a, b)$, $g(b, c)$, and $g(c, a)$. There exist points $a'$, $b'$, and $c'$ such that the paths $g(a, b)$ and $g(a, c)$ intersect in the path $g(a, a')$, the paths $g(b, c)$ and $g(b, a)$ intersect in $g(b, b')$, and finally the paths $g(c, a)$ and $g(c, b)$ intersect in $g(c, c')$. Moreover, we see that the geodesic triangle $\triangle a'b'c'$ has only reflex angles along its boundary

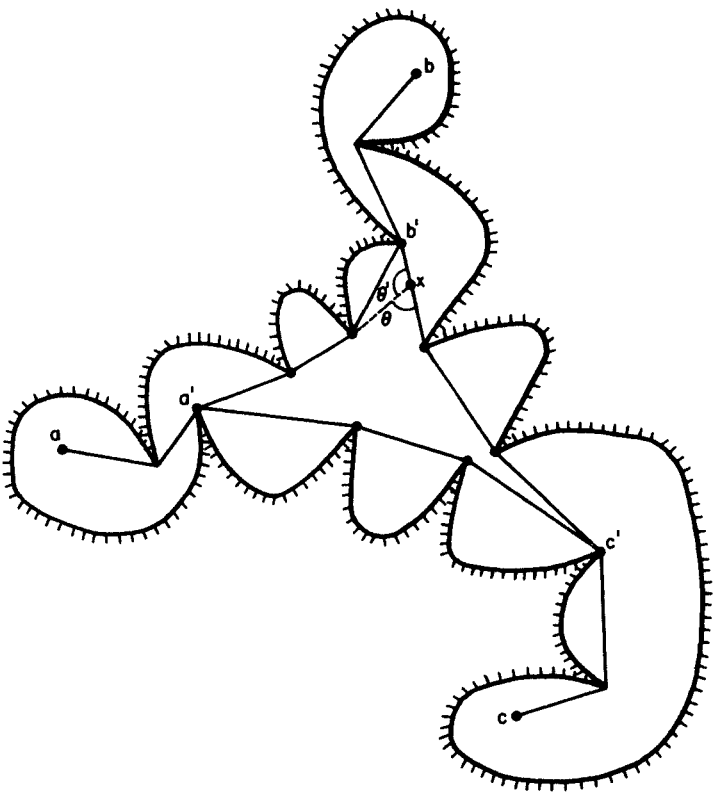


Fig. 2

with the exception of the angles at $a'$, $b'$, and $c'$. This follows from the fact that the sides $g(a', b')$, $g(a', c')$, $g(b', c')$ of this triangle do not intersect, and the interior of this triangle is fully contained in $P$; thus any nonreflex angle along, say, $g(a', b')$ would allow us to shortcut $g(a, b)$ within $P$, thereby contradicting its geodesicity (such a triangle is portrayed in Fig. 2).

**Lemma 1.** *As $x$ varies along $g(b, c)$, $d(a, x)$ is a convex function of $d(b, x)$, and $d(a, x) < \max\{d(a, b), d(a, c)\}$.*

*Proof.* Let $a'$, $b'$, $c'$ be the "bifurcation points" of the paths $g(a, b)$, $g(a, c)$, $g(b, c)$, as defined above. Write $t = d(b, x)$ and $\psi(t) = d(a, x)$, for $x$ in $g(b, c)$. Clearly, $\psi(t)$ is a well-defined continuous and piecewise smooth function of $t$. Moreover, for $x \in g(b, b')$ we have $\psi'(t) = -1$, and for $x \in g(c', c)$ we have $\psi'(t) = +1$. For $x \in g(b', c')$, we can easily check, using the law of cosines, that the right derivative $(\psi')^+(t)$ of $\psi$ at $t = d(b, x)$ is $-\cos\theta(t)$, where $\theta(t)$ is the angle between $\mathbf{u}(x, a)$ and $\mathbf{u}(x, c)$. Similarly, the left derivative $(\psi')^-(t)$ of $\psi$ at $t$ is $+\cos\bar{\theta}(t)$, and $\bar{\theta}(t)$ is the angle between $\mathbf{u}(x, a)$ and $\mathbf{u}(x, b)$. But $\theta(t) + \bar{\theta}(t) \geq \pi$, because $g(b', c')$ has only reflex angles; thus $\cos\bar{\theta}(t) \leq -\cos\theta(t)$, i.e., $(\psi')^-(t) \leq (\psi')^+(t)$.

Moreover, assume that $a$, $b$, $c$ are arranged so that as $x$ varies along $g(b', c')$ from $b'$ to $c'$, the vector $\mathbf{u}(a', x)$ (weakly) turns clockwise (as in Fig. 2). Then, using the fact that the edges of the geodesic triangle $\triangle a'b'c'$ have only reflex angles, it is easy to check that as $x$ varies in the above manner, the vector $\mathbf{u}(x, a)$ strictly turns clockwise, whereas the vector $\mathbf{u}(x, c)$ weakly turns counterclockwise (see Fig. 2). Thus $\theta(t)$ is strictly increasing along $g(b', c')$, and thus $\psi'(t)$ is piecewise continuous and increasing along $g(b, c)$, from which convexity of $\psi$ follows.

Finally, note that $\psi$ is actually strictly convex on $g(b', c')$, whereas it is linear with slope $-1$, $+1$ along the remaining subpaths $g(b, b')$, $g(c', c)$, respectively. This is easily seen to establish the second assertion of the lemma. □

**Corollary 1.** *The* closed ball *of radius r at x,* $B(r, x) = \{y \in P \mid d(x, y) \leq r\}$, *is P-convex.*

The proof of the lemma also implies

**Corollary 2.** *With the notation of the lemma, if the geodesic angle* $\angle ba'c$ *at* $a'$ *is greater than or equal to* $\pi/2$ *then* $d(b, c) > d(a', b)$, $d(a', c)$.

It is obvious that $d(a, x)$ is maximized for $x$ a vertex of $P$.

Let $F(x) = \max_v d(x, v)$, where the maximum is taken over the vertices $v$ of $P$. Clearly, $F$ is continuous and therefore has a minimum value called the *geodesic radius* of $P$ and denoted $\mathrm{gr}(P)$. The set of points

$$G(P) = \{x \in P \mid F(x) = \mathrm{gr}(P)\}$$

is the intersection of the closed balls of radius $\mathrm{gr}(P)$ centered at the vertices of $P$. If $G(P)$ contained two distinct points $x$, $y$ it would have to contain $g(x, y)$ and the geodesic distance from a vertex would have to be constant along each segment of $g(x, y)$, which is clearly impossible by Lemma 1. Hence $G(P)$ consists of a single point, called the *geodesic center* of $P$, and denoted by $\mathrm{geocen}(P)$ (see also [AT]).

Let $c$ be a point of $P$ and let the set of furthest vertices from $c$ be $V(c) = \{v_1, v_2, \ldots, v_k\}$. The following lemma gives sufficient conditions for such a point $c$ to be the geodesic center.

**Lemma 2.** *If the set* $\{\mathbf{u}(c, v) \mid v \in V(c)\}$ *does not lie in an open half-plane through c then* $c = \mathrm{geocen}(P)$. *Furthermore, if c is a vertex of P and for some pair* $v_1, v_2 \in V(c)$ *the geodesic angle* $\angle v_1 c v_2$ *meets the exterior of P in a neighborhood of c then also* $c = \mathrm{geocen}(P)$.

*Proof.* Suppose first $c$ is not a vertex of $P$. For any $v_i$ in $V(c)$, the line through $c$ orthogonal to $\mathbf{u}(c, v_i)$ determines a diagonal $D_i$ of $P$ which separates $P$ into two components (the "sides" of $D_i$). Suppose that $x$ is a point of $P$ lying on the side of $D_i$ which does not contain $v_i$; then $g(x, v_i)$ must cross $D_i$, say at $x'$ and

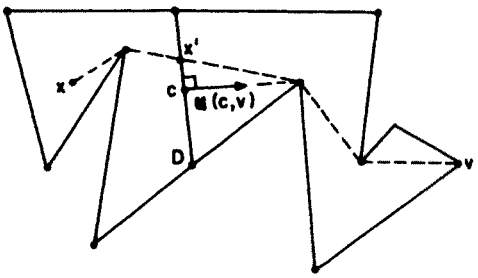


Fig. 3

since either $x' = c$ or the geodesic angle $\angle x'cv_i$ is $\pi/2$ we see, by Corollary 2, that $d(x, v_i) > d(x', v_i) > d(c, v_i) = F(c)$, whence $x$ is not geocen$(P)$ (see Fig. 3). Hence geocen$(P)$ must be contained in the intersection of all (closed) half-polygons $P_1, \ldots, P_k$, where $P_i$ is the side of $D_i$ containing $v_i$, and the assumptions of the lemma imply that this intersection is $\{c\}$. Similar arguments apply if $c$ is a vertex of $P$. □

**Remark.** If two directions $\mathbf{u}(c, v_1)$ and $\mathbf{u}(c, v_2)$ are opposite, then again $c$ is the geodesic center and $g(v_1, v_2)$ is a geodesic diameter (whose midpoint is $c$).

**Definition.** Given a *cut* of $P$, i.e., a segment $C$ which is openly contained in $P$ and has endpoints on the boundary of $P$, thus separating $P$ into two components, the *relative center* of $P$ on $C$ is the unique point $c$ of $C$ which minimizes $F(x)$ for $x$ in $C$ and is denoted RELCEN$(C) = c$.

After having found the relative center $c$ on a cut $C$ we would like to know where to look for the geodesic center. The case that it lies on $C$ (and hence geocen$(P) = c$) has been covered by the previous lemma. In the complementary case, the following lemma enables us to determine on which side of $C$ the geodesic center lies.

**Lemma 3.** *Let* $c = \text{RELCEN}(C)$, *and assume that the cone with apex* $c$ *spanned by all directions* $\mathbf{u}(c, v)$ *for* $v$ *in* $V(c)$ *lies in an open half-plane passing through* $c$, *so that its intersection with a sufficiently small neighborhood of* $c$ *is wholly contained in* $P$. *Let* $\mathbf{u}^*$ *be a unit vector in the direction of the bisector of the angle of this cone. Then* geocen$(P)$ *lies on the side of* $C$ *to which* $\mathbf{u}^*$ *points* (see Fig. 4).

*Proof.* Let $d(c, v) = r$ for each $v$ in $V(c)$, while the next furthest vertex from $c$ has geodesic distance $r' < r$ from $c$. Note that for $c'$ in $B(r - r', c)$ the distance from $c'$ to any vertex not in $V(c)$ is at most $r$. Choose a positive $\delta < \min\{r - r', \rho\}$, where $\rho$ is the length of the smallest initial segment of $g(c, v)$ for $v$ in $V(c)$, sufficiently small so that the boundary of $B = B(\delta, c)$ consists of a single circular arc and possibly a portion of at most two segments of the boundary of $P$ (the latter might occur only in case $c$ is on the boundary of $P$). Let $H$ be the line passing through $c$ and perpendicular to $\mathbf{u}^*$. For each $v \in V(c)$ let $h(v)$ be the

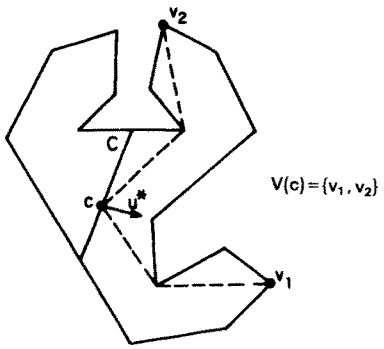


Fig. 4

perpendicular distance from $H$ to $p_v$, where $p_v$ is the intersection of $g(c, v)$ and the boundary of $B(\delta, c)$, and let $h = \min_{v \in V(c)} h(v)$. The point $c + h\mathbf{u}^* = p$ in $B(\delta, c)$ satisfies $d(p, v) < r$ for all vertices $v$ of $P$, since for $v$ in $V(c)$,

$$d(p, v) < d(p, p_v) + d(p_v, v) < d(c, p_v) + d(p_v, v) = r,$$

and we have already seen that $d(p, v) < r$ for the other vertices of $P$. Hence $p$ lies in the intersection of the closed balls of some radius $r'' < r$ at each of the vertices of $P$. Since the geodesic center of $P$ must also lie in this intersection which is connected (since $P$-convex) and disjoint from $C$ (since, by definition of RELCEN($C$), $F \geq r > r''$ on $C$), we see that the geodesic center of $P$ lies on the same side of $C$ as $p$. □

## 3. The Algorithm RELCEN

Suppose a triangulation of $P$ is given. Let $C$ be a cut of $P$, i.e., $C$ is openly contained in $P$ and has endpoints $a$ and $b$ on the boundary of $P$; thus $C$ splits $P$ into two portions $P_1$, $P_2$. For the sake of exposition, the subsequent analysis deals only with that portion of $P$, say $P_1$, which, in a sufficiently small neighborhood of $C$, lies to the left of the directed segment $\overrightarrow{ab}$. Symmetric arguments will apply to the other portion $P_2$ of $P$.

We construct, in linear time [Gea], the tree $Q(a)$ of shortest paths from $a$ to the vertices of $P_1$ and the tree $Q(b)$ of shortest paths from $b$ to the vertices of $P_1$. Let $v$ be any vertex of $P_1$. It is shown [Gea] that $v$ is visible from some point in $C$ if and only if the geodesic triangle $\triangle abv$ has only reflex angles along its two sides $g(a, v)$, $g(b, v)$. When this is the case, $v$ is visible from (each point in) a subinterval $pq$ of $C$, where $p$ and $q$ are the intersections of $C$ with the extensions of the last edges $v_a v$, $v_b v$ along $g(a, v)$, $g(b, v)$, respectively; see Fig. 5. Thus $v$ is visible from $C$ if and only if the parents $v_a$, $v_b$ of $v$ in $Q(a)$, $Q(b)$, respectively, are distinct.

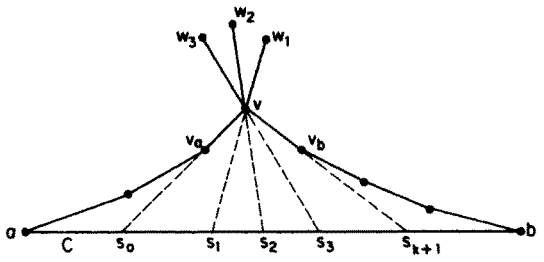


Fig. 5

It also follows easily from Guibas *et al.* [Gea] that if $v$ is visible from $C$ and has a child $w$ in $Q(a)$ (resp. in $Q(b)$) such that $v_a vw$ is a right (resp. a left) turn, then $w$ is also a child of $v$ in the other tree, and is not visible from $C$. Hence each child $w$ of a visible vertex $v$ in $Q(a)$ or in $Q(b)$ is of one of the following three types:

(i) $w$ is invisible from $C$ (and is a child of $v$ in both trees $Q(a)$, $Q(b)$).
(ii) $w$ is visible from $C$, is a child of $v$ only in $Q(a)$, and $v_a vw$ is a left turn.
(iii) $w$ is visible from $C$, is a child of $v$ only in $Q(b)$, and $v_b vw$ is a right turn.

Let $c(v)$ denote the maximum geodesic distance from $v$ to any invisible vertex in the subtree of $Q(a)$ (or of $Q(b)$) rooted at $v$; if no such vertex exists, $c(v) = 0$.

Define a function $d_l(v)$ on the vertices $v$ of $P_1$ recursively as follows:

(a) If $v$ is invisible from $C$, put $d_l(v) = c(v)$.
(b) If $v$ is visible from $C$, let $d_l(v)$ be the maximum of $c(v)$ and of $\max\{d_l(w_i) + |w_i v| : w_i \text{ is a type (ii) child of } v\}$.

Similarly we define a symmetric function $d_r(v)$ in terms of the type (iii) children of $v$.

Let $v$ be a visible vertex of $P_1$, and let $w_1, \ldots, w_k$ be the children of $v$ of type (ii) arranged in counterclockwise order around $v$. Extend the segments $w_1 v, \ldots, w_k v$ past $v$ until they intersect $C$ at the respective points $s_1, \ldots, s_k$. Let $s_0$ be the intersection of $C$ with the extension of $v_a v$, and let $s_{k+1}$ be the intersection of $C$ with the extension of $v_b v$. For each $i = 0, \ldots, k$ define a function

$$f^l_{v,i}(x) = \begin{cases} |xv| + \max(c(v), |vw_j| + d_l(w_j) : j > i), & x \in [s_i, s_{i+1}], \\ -\infty, & \text{otherwise,} \end{cases}$$

and, in a completely symmetric manner, we define a corresponding collection of partial functions $f^r_{v,i}$ induced by type (iii) children in $Q(b)$. We also repeat the entire construction for the other portion $P_2$ of $P$, obtaining similar collections of functions $f^l_{v,i}, f^r_{v,i}$ for vertices $v$ of $P_2$. Enumerate all these functions as $g_1, \ldots, g_m$. Note that $m = O(n)$, because each function is determined uniquely either by a pair of adjacent type (ii) or type (iii) children of some visible vertex $v$ in $Q(a)$

or in $Q(b)$, or by the parent of $v$ in one of these trees and one of the two "extreme" corresponding type (ii) or type (iii) children of $v$ in that tree. Since each of these trees has linear size, the property follows.

Let $F(x) = \max_w d(x, w)$ for $x \in C$, where the maximum is taken over all vertices $w$ of $P$.

**Lemma 4.** *$F(x) = \max_{v,\sigma,i} f_{v,i}^{\sigma}(x)$, where the maximum is taken over all visible vertices $v$, $\sigma \in \{l, r\}$, and indices $i$ corresponding to type* (ii) *or* (iii) *children of $v$.*

*Proof.* Let $x \in C$. Clearly, each $f_{v,i}^{\sigma}(x)$, when it is defined, is a geodesic distance from $x$ to some vertex $w$ of $P$, thus $F(x) \geq \max_{v,\sigma,i} f_{v,i}^{\sigma}(x)$. On the other hand, for each vertex $w$ of, say $P_1$, $d(x, w)$ is obtained by going along a straight segment from $x$ to some vertex $v$ (visible from $x$), and then following the shortest path $g(v, w)$ from $v$ to $w$. If $w = v$ then clearly $d(x, w) \leq f_{w,i}^{\sigma}(x)$ for some $\sigma$ and $i$. Otherwise, let $z$ be the next vertex after $v$ along $g(v, w)$. If $z$ is invisible from $C$ then so is $w$ and we have

$$d(x, w) = |xv| + d(v, w) \leq |xv| + c(v) \leq f_{v,i}^{\sigma}(x)$$

for some $\sigma$ and $i$. If $z$ is a type (ii) child of $v$, say $z = w_j$ in the above notation, then for some $i < j$ the function $f_{v,i}^{l}(x)$ is different from $-\infty$, and

$$d(x, w) = |xv| + |vz| + d(z, w) \leq |xv| + |vw_j| + d_l(w_j)$$

as is easily checked. By definition, then, $d(x, w) \leq f_{v,i}^{l}(x)$. Since a symmetric inequality can be obtained in case $z$ is a type (iii) child, the assertion of the lemma follows easily. □

Thus, to calculate $x^* = \text{RELCEN}(C)$, we need to calculate

$$\min_{x \in C} \max_{v,\sigma,i} f_{v,i}^{\sigma}(x).$$

It follows from Lemma 1 that the pointwise maximum of these functions, namely $F(x)$, is convex on $C$. Thus, considering $C$ as an interval on the $x$-axis, given any point $x_0 \in C$, we can determine whether $x^* < x_0$, $x^* > x_0$, or $x^* = x_0$, by calculating (in time proportional to the number of functions $g_j$) $F(x_0) = \max_j g_j(x_0)$ and its one-sided derivatives at $x_0$. If the left derivative $(F')^-(x_0)$ is positive (resp. the right derivative $(F')^+(x_0)$ is negative, resp. $(F')^-(x_0) \leq 0 \leq (F')^+(x_0)$) then $x^* < x_0$ (resp. $x^* > x_0$, $x^* = x_0$). These derivatives are defined in the following manner. If a single function $g_j$ attains $F$ at $x_0$ then $(F')^{\pm}(x_0) = g_j'(x_0)$. Otherwise we have $(F')^-(x_0) = \min_j g_j'(x_0)$, $(F')^+(x_0) = \max_j g_j(x_0)$, where these extrema are calculated over all $g_j$ attaining $F$ at $x_0$; if any of the functions $g_j$ is defined only on one side of $x_0$, it appears only in the expression defining the corresponding one-sided derivative of $F$ and not in that of the other one.

These observations allow us to apply Megiddo's technique for two-variable linear programming [Me1] to find $x^*$ in linear time. Since the functions $g_i$ have

a somewhat irregular shape, some fine-tuning of the technique is required. We perform $O(\log n)$ iterations; after the $j$th iteration we will have obtained a subinterval $I_j$ of $C$ where $x^*$ is known to lie, and a subset $G_j$ of the functions $g_i$ such that

$$|G_j| \leq m \cdot (\tfrac{3}{4})^j$$

and

$$\max_{g \in G_j} g(x) = F(x)$$

for all $x \in I_j$. As in Megiddo [Me1], each iteration will run in time linear in the number of remaining functions (i.e., the functions in $G_j$), and this will imply overall $O(n)$ performance of the algorithm. Initially, $I_0 = C$ and $G_0 = \{g_1, \ldots, g_m\}$. Consider the $j$th step. Combine the functions in $G_{j-1}$ into disjoint pairs. Let $g$, $h$ be such a pair. We can assume without loss of generality that neither $g$ nor $h$ is $-\infty$ throughout $I_{j-1}$, or else we could simply discard that function.

We define an *extended intersection* of $g$, $h$ to be a point $x_0 \in I_{j-1}$ such that $g(x) > h(x)$ for $x$ lying on one side of $x_0$ sufficiently near it, while $g(x) < h(x)$ on the other side of $x_0$ sufficiently near it. Since each of $g(x)$, $h(x)$ has the form $|xv| + c$ for $x$ in some interval $J$, and $-\infty$ otherwise, it is easily checked that $g$ and $h$ can have at most four extended intersections over $I_{j-1}$, of which at most two are real intersections, and the others occur at endpoints of the domains of $g$ or of $h$. (A special case arises when the domains of $g$, $h$ are disjoint; in this case we choose an arbitrary point $x_0$ lying between these two domains, and call it the (unique) extended intersection of $g$ and $h$.) See Fig. 6 for an illustration of extended intersections. It follows easily from definition that if $g$, $h$ have no extended intersections in $I_{j-1}$, then one of them, say $g$, is larger than $h$ throughout $I_{j-1}$, and consequently we simply discard $h$ in this case.

Let $S$ denote the collection of all $O(n)$ extended intersections of the pairs $g$, $h$. Find the median $x_0$ of these intersections. Calculate $F(x_0) = \max_{g \in G_{j-1}} g(x_0)$ and its one-sided derivatives at $x_0$ (in the manner explained above, which takes $O(|G_{j-1}|)$ time) to determine which side of $x_0$ contains $x^*$. Next take the extended intersections lying on that side of $x_0$, find their median $x_1$, and determine which side of $x_1$ contains $x^*$. Repeating this procedure one more time, we obtain a subinterval $I_j$ of $I_{j-1}$ known to contain $x^*$, so that at most one-eighth of the extended intersections lie in $I_j$. Since each pair of functions contributes at most four points to the set $S$, it follows that for at least half of the pairs $g$, $h$, all their extended intersections lie outside $I_j$, and consequently we can determine which of $g$, $h$ is dominated by the other function throughout $I_j$, and discard it from further consideration (the idea of using multiple medians at a single iteration of the algorithm is adapted from Zemel [Ze]). The remaining functions constitute the next subcollection $G_j$, and the process is repeated again. Note that we will have

$$F(x) = \max_{g \in G_j} g(x)$$

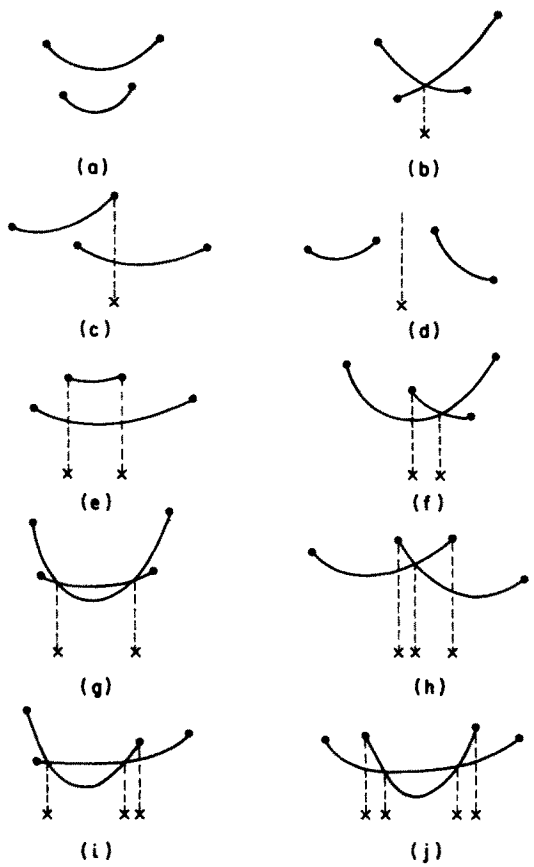


Fig. 6

for all $x \in I_j$, and that $|G_j| \leq \frac{3}{4}|G_{j-1}|$. This procedure stops either when at some median $x_0$ we have $(F')^-(x_0) \leq 0 \leq (F')^+(x_0)$, in which case $x^* = x_0$, or when $|G_j|$ becomes smaller than some prespecified constant. In this case $\min_{x \in I_j} \max_{g \in G_j} g(x)$ can be calculated in constant time by a brute-force method.

Let $c = x^*$ be the relative center of $P$ on $C$. Again we construct in $O(n)$ time the tree of shortest paths $Q(c)$ from $c$ to the vertices of $P$. Lemmas 2 and 3 allow us to determine either that $c$ is the geodesic center of $P$, or else on which side of $C$ that geodesic center must lie.

This completes the description of the RELCEN procedure, and leads to the following result.

**Theorem 1.** *The relative geodesic center of a simple polygon $P$ along a cut $C$, and the portion of $P$ cut by $C$ which contains the true geodesic center of $P$, can be calculated in linear time, given a triangulation of $P$.*

**Remark.** This algorithm is used as a subroutine in the complete algorithm GEOCEN given below, with the convention that if the output produced by RELCEN is the geodesic center of $P$ then the entire algorithm halts and outputs this center.

## 4. The Algorithm GEOCEN

We are now in position to describe the complete procedure for the calculation of geocen($P$). It consists of the following steps.

I. Triangulate $P$ (in time $O(n \log n)$ using the algorithm in Garey *et al.* [GJPT], or more efficiently using the algorithm of Tarjan and Van Wyk [TV]). Place the internal edges of the triangulation in a balanced binary tree $T$ [Ch] such that:

(i) For each subtree $T'$ of $T$, the edges stored at the nodes of $T'$ bound a collection of triangles that cover a connected simple subpolygon of $P$.
(ii) The number of edges in each of the two subtrees rooted at the left and the right children of a node $e$ of $T$ are both at least one-quarter of the number of edges in the subtree rooted at $e$.

Such a balanced decomposition tree of $P$ can be constructed in linear time [Gea], or, by an alternative technique, in $O(n \log n)$ time [Ch].

We next search the tree $T$ to locate the triangle $\triangle abc$ of the triangulation of $P$ which contains the geodesic center of $P$. At each node of $T$ along the search path, we apply RELCEN to the corresponding diagonal $e$ of $P$ to determine which side of $e$ contains geocen($P$), and then continue the search at the appropriate child of that node. Hence in $O(\log n)$ applications of RELCEN, each of which takes $O(n)$ time, we obtain the desired triangle. Thus the total time required by this step is $O(n \log n)$.

II. Next we want to locate a region within the triangle $\triangle abc$ in which, for each vertex $v$ of $P$, the structure of the path $g(x, v)$ and the analytic form of the corresponding function $d(x, v)$ are both uniquely determined by the vertex $v$.

To this end we construct the trees $Q(a)$, $Q(b)$, $Q(c)$ of shortest paths from $a$, $b$, and $c$, respectively, and record the $k = O(n)$ "tangents", i.e., extensions of shortest path edges toward the triangle $\triangle abc$ that actually enter that triangle (see Fig. 7). (As a matter of fact, we only need to calculate the portion of the tree $Q(a)$ consisting of shortest paths that do not cross the opposite side $bc$ of our triangle, and similarly for $Q(b)$, $Q(c)$.) Within any region determined by these tangents the structure of $g(x, v)$ and the analytic form of $d(x, v)$ are both uniquely determined for each vertex $v$. Our task is thus to determine the side of each one of these tangents which contains geocen($P$). To this end we use and adapt a technique used by Megiddo [Me1] in his linear-time algorithm for linear programming in $\mathbf{R}^3$. Specifically, we find the median slope of these tangents and let $T_1, \ldots, T_{\lfloor k/2 \rfloor}$ have positive slope and $T_{\lfloor k/2 \rfloor + 1}, \ldots, T_k$ have negative slope with

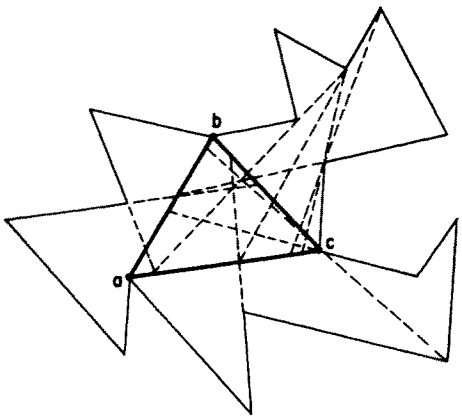


Fig. 7

respect to a new orientation of the axes where the $x$-axis is in the direction of the median slope. Form the $\lfloor k/2 \rfloor$ intersections of the lines $T_i$ and $T_{\lfloor k/2 \rfloor + i}$ for $i = 1, \ldots, \lfloor k/2 \rfloor$. Next find the median $x_1$ of the $x$-coordinates of these $\lfloor k/2 \rfloor$ intersection points. Take the cut $C_x$ determined by the line $x = x_1$ in the triangle $\triangle abc$ and apply RELCEN to $C_x$ to determine which side of $C_x$ contains the geodesic center. For the roughly $k/4$ intersection points lying on the other side of $C_x$ find their median $y$-coordinate $y_1$. Take the cut $C_y$ determined by the line $y = y_1$ in this triangle and apply RELCEN to $C_y$ to determine which side of $C_y$ contains the geodesic center. Now for each of the roughly $k/8$ pairs of tangents whose intersection lies in the quadrant determined by $C_x$ and $C_y$ and opposite to the quadrant containing the center, we can discard one of the two tangents (since the side of it containing geocen($P$) is now known). In addition, we add the two half-planes bounded by the cuts $C_x$ and $C_y$ and containing geocen($P$) to a collection of half-planes that the algorithm constructs. Thus in $O(\log n)$ applications of RELCEN we can find the side of each tangent containing geocen($P$), and at the same time obtain $O(\log n)$ half-planes whose intersection is known to contain geocen($P$). We then find an arbitrary point $w$ lying in the intersection of these half-planes, a task that can be accomplished in $O(\log n)$ time, using Megiddo's technique for two-dimensional linear programming [Me1], and then calculate the shortest-path tree $Q(w)$ in $O(n)$ time. By the preceding discussion, the combinatorial structure of $Q(w)$ and of $Q(\text{geocen}(P))$ are the same, so that the analytic form of each function $d(x, v)$ throughout the intersection of those half-planes can be readily determined. Clearly, this substep also runs in $O(n \log n)$ time.

**Remark.** In the preceding arguments we have assumed that geocen($P$) is an interior point of $P$. It is easy to check, using Lemma 2, that if geocen($P$) lies in the relative interior of an edge of $P$, the procedure sketched above will locate a region containing the center, and will properly obtain the fixed analytic form of

the functions $d(x, v)$ for $x$ in that region. Finally, if geocen($P$) is a vertex of $P$ (which is to say, geocen($P$) is $a$, $b$, or $c$) then it must be a reflex vertex, and so be incident to a side of $\triangle abc$ which is also a cut of $P$. But then a preceding application of RELCEN to that cut would have found geocen($P$) to lie on that cut, and consequently would have terminated our algorithm. Hence in step II, and in the succeeding step III, we can assume that geocen($P$) is not a vertex of $P$.

III. The problem is now reduced to finding the minmax of a family of functions $d_i(x) = d(x, v_i) = |xv_{k(i)}| + c_i$, where $c_i = d(v_{k(i)}, v_i)$, for $i = 1, \ldots, n$. Note that the preceding remark and discussion allow us to ignore $P$ completely and regard each of the functions $d_i(x)$ as being defined over the entire plane, because geocen($P$) must be the unique global minimum of $F(x) \equiv \max_i d_i(x)$. As noted in the Introduction, this problem can be restated as follows: given $n$ circles in the plane, find the smallest circle which contains all of them. The simpler one-center problem (in which instead of $n$ circles we are given $n$ points) has for some time now linear-time solutions [Me1] (see also Dyer [Dy2] for an adaptation of this technique to the weighted one-center problem). A very recent result of Megiddo [Me4], obtained after the original preparation of this paper, shows how to solve this last problem in time $O(n)$. Our original solution has time complexity $O(n \log n \log \log n)$, and is accomplished by yet another technique due to Megiddo [Me2] which uses parallel algorithms in the design of efficient sequential optimization algorithms. It can be found in a preceding version [PS] of this paper, but we omit the details here since this technique is fairly involved, and is superseded anyway by [Me4].

Megiddo's solution is an adaptation of his linear-time algorithm for linear programming in $\mathbf{R}^d$ [Me3] and proceeds roughly as follows. The problem is restated as: find a point $x^* = (x, z) \in \mathbf{R}^3$ which minimizes $z$, subject to the constraints $z - c_i \geq |xv_{k(i)}|$ for $i = 1, \ldots, n$. Now for any $i \neq j$ let $H_{ij}$ be the plane

$$(z - c_i)^2 - |xv_{k(i)}|^2 = (z - c_j)^2 - |xv_{k(j)}|^2.$$

If we can determine which side of $H_{ij}$ contains the point $x^*$ where the desired minimum of $z$ is attained, then we can eliminate one of the constraints $z - c_i \geq |xv_{k(i)}|$, $z - c_j \geq |xv_{k(j)}|$, from further consideration. This observation allows Megiddo to eliminate some fixed fraction of the constraints in linear time, using essentially the same ideas as in [Me3], thereby obtaining an overall linear-time performance.

We thus obtain our main result.

**Theorem 2.** *The geodesic center of a simple polygon having n sides can be calculated in $O(n \log n)$ time.*

## 5. Discussion

We have presented a rather involved algorithm for computing the geodesic center of an $n$-sided simple polygon in time $O(n \log n)$. While this constitutes a

significant improvement over the previous algorithm [AT], we have no matching nonlinear lower bound on the complexity of the problem. An obvious open problem is whether the geodesic center can be calculated in linear time (say when a triangulation of the polygon is given). In attempting to enhance the efficiency of our technique to achieve this goal, a major subproblem that arises is that each application of RELCEN to a cut of the polygon $P$ has to process the entire collection of the vertices of $P$, even when the center of $P$ is already known to lie in a smaller subpolygon of $P$. We do not see how to exploit the information provided by previous applications of RELCEN to reduce the complexity of subsequent applications of this procedure.

## Acknowledgments

The authors wish to thank Godfried Toussaint, for introducing the problem and for some helpful discussions on it, Arie Tamir, for discussions on the problem and for some helpful comments, observations, and references that helped us a lot in the preparation of the paper, Nimrod Megiddo, for discussion of the problem (which has eventually led to his improved linear-time solution of the last substep of the problem [Me4]), and János Pach, for discussions concerning the notion of $P$-convexity. Some ideas concerning this problem were discussed at the Workshop on Movable Separability of Sets at the Bellairs Research Institute of McGill University, and at the research seminar on computational geometry at the Courant Institute; we wish to thank the participants of these two groups for stimulating discussions of these ideas.

## References

[AT] Asano, T., and Toussaint, G. T. (1985). Computing the geodesic center of a simple polygon, Technical Report SOCS-85.32, McGill University.

[Ch] Chazelle, B. (1982). A theorem on polygon cutting with applications, *Proc. 23rd IEEE Symp. on Foundations of Computer Science*, pp. 339–349.

[Dy1] Dyer, M. E. (1984). Linear-time algorithms for two- and three-variable linear programs, *SIAM J. Comput.* **13**, pp. 31–45.

[Dy2] Dyer, M. E. (1986). On a multidimensional search technique and its applications to the Euclidean one-center problem, *SIAM J. Computing* **15**, pp. 725–738.

[GJPT] Garey, M. R., Johnson, D. S., Preparata, F. P., and Tarjan, R. E. (1978). Triangulating a simple polygon, *Inform. Process. Lett.* **7**, pp. 175–180.

[Gea] Guibas, L., Hershberger, J., Leven, D., Sharir, M., and Tarjan, R. E. (1987). Linear-time algorithms for visibility and shortest path problems inside a triangulated simple polygon, *Algorithmica* **2**, pp. 209–233.

[LP] Lee, D. T., and Preparata, F. P. (1984). Euclidean shortest paths in the presence of rectilinear barriers, *Networks* **14**(3), pp. 393–410.

[Lea] Lenhart, W., Pollack, R., Sack, J., Seidel, R., Sharir, M., Suri, S., Toussaint, G. T., Yap, C., and Whitesides, S. (1987). Computing the link center of a simple polygon, *Proc. 3rd ACM Symp. on Computational Geometry*, pp. 1–10.

[Me1] Megiddo, N. (1983). Linear-time algorithms for linear programming in $\mathbf{R}^3$ and related problems, *SIAM J. Comput.* **12**, pp. 759–776.

[Me2] Megiddo, N. (1983). Applying parallel computation algorithms in the design of serial algorithms, *J. Assoc. Comput. Mach.* **30**, pp. 852–866.

[Me3] Megiddo, N. (1984). Linear programming in linear time when the dimension is fixed, *J. Assoc. Comput. Mach.* **31**, pp. 114–127.

[Me4] Megiddo, N. (1989). On the ball spanned by balls, *Discrete Comput. Geom.*, this issue, pp. 605–610.

[PS] Pollack, R., and Sharir, M. (1986). Computing the geodesic center of a simple polygon, Technical Report 231, Computer Science Department, Courant Institute, New York University, September 1986.

[Su1] Suri, S. (1986). Computing the geodesic diameter of a simple polygon, Technical Report JHU/EECS-86/08, Department of Electrical Engineering and Computer Science, Johns Hopkins University.

[Su2] Suri, S. (1986). Polygon partitioning techniques for link distance problems, Technical Report, Department of Electrical Engineering and Computer Science, Johns Hopkins University.

[TV] Tarjan, R. E., and Van Wyk, C. J. (1988). An $O(n \log \log n)$-time algorithm for triangulating simple polygons, *SIAM J. Comput.* **17**, pp. 143–178.

[Ze] Zemel, E. (1984). An $O(n)$ algorithm for the linear multiple choice knapsack problem and related problems, *Inform. Process. Lett.* **18**, pp. 123–128.

*Received September 17, 1986, and in revised form July 30, 1987.*