INFORMS Journal on Computing
Vol. 12, No. 4, Fall 2000

0899-1499/100/1204-0317 $05.00

Approximation of Point Sets by 1-Corner Polygonal Chains

JOSÉ-MIGUEL DÍAZ-BÁÑEZ / *Departamento de Matemática Aplicada II, Universidad de Sevilla, Spain, Email: dbanez@cica.es*

FRANCISCO GÓMEZ / *Departamento de Matemática Aplicada, Universidad Politécnica de Madrid, Spain, Email: fmartin@eui.upm.es*

FERRAN HURTADO / *Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Spain, Email: hurtado@ma2.upc.es*

(Received: July 1999; revised: May 2000; accepted: June 2000)

In this paper we consider some problems that belong to the interplay between the field of Facility Location and the area of Computational Geometry. Specifically, given a set *S* of points in the plane, we discuss several variations of the problem of finding monotone 1-corner polygonal chains that minimize the maximum vertical distance to *S*.

Several problems on approximating geometric objects by polygonal chains have been studied by the computational geometry community due to its relevance to fields such as geographic information systems, pattern recognition and CAD/CAM. Some examples are the papers by Imai and Iri (1986, 1988), Toussaint (1985), Melkman and O'Rourke (1988), Chan and Chin (1996), and O'Rourke (1981).

A lot of attention has been paid to the *simplification problem*: to approximate a given polygonal chain C_1 by another one C_2 whose vertices are some of the vertices in C_1 (or at least use their x-coordinates). When a subchain $C_1' \subset C_1$ is approximated by a line segment $s \in C_2$, some error is made. The error of the approximating chain C_2 is defined to be the maximum error over all its line segments. Two main problems are considered:

- the *min-number* problem: to minimize the number of vertices of an approximating polygonal chain with the error within a given bound;
- the *min-*ϵ problem: to minimize the error of an approximating polygonal chain consisting of a given number of vertices.

Certainly the maximum Euclidean distance between s and C_1' is a natural measure for the error, but many other criteria have been considered: the parallel-strip criterion (Toussaint 1985), the parallel-rectangle criterion (Ballard 1981), the minimum-width rectangle criterion (Imai and Iri 1988), the L_1-approximation (Gentle et al. 1988), the L_2^2-approximation (Chin et al. 1992), and the L_∞-approximation (Kurozumi and Davis 1982).

The approximation of discrete point sets by polygonal chains has arisen in the context of Statistics and operations research, as a natural extension of the classical best-fit-line problem. Here the vertical distance, also called the *Tchebyshev error* of the fit, is commonly used to define the error instead of the Euclidean distance (Rice 1964). That is the framework of our work, and throughout this paper, *distance* will mean *vertical distance*.

Hakimi and Schmeichel (1991) posed the following problem. Let $k > 0$ be an integer; then for every given set of points S in the plane we want to construct a monotone polygonal chain C with k or fewer corners (vertices) such that $d(S, C)$ (vertical distance) is as small as possible. They solve two variants of the problem: when C is required to have its corners on points in S, the *discrete problem*, and when its corners can be on any point in the plane, the *free problem*. Those problems can be seen as min-ϵ problems for discrete point sets. For both of them, Hakimi and Schmeichel give $O(n^2 \log n)$ algorithms that do not work in the presence of degeneracies (they do not admit points with the same x-coordinate).

Results in this paper correspond to variations on the problems posed by Hakimi and Schmeichel, for the case $k = 1$, adding the restriction that the monotone chains must start and end at specified anchor points a and b. This assumption is quite natural in many real applications (see Drezner and Wesolowsky 1989); on the other hand, we show that our algorithms can also be extended to deal with non anchored chains. Many more variations on these problems have been considered by Díaz-Báñez (1998) and in Díaz-Báñez et al. (1999).

We present here an $O(n \log n)$ algorithm for the 1-corner discrete problem, and an $O(n \log n)$ algorithm for the 1-corner free problem, none of them making degeneracy assumptions. For the 1-corner discrete case we also solve within the same time bound the problem in which a and b are not fixed but must each satisfy a feasible set of linear constraints. We also provide as observations $O(n \log^2 n)$ versions of our algorithms that use a more elementary approach and would be easy to implement. All these results are described in Section 2 after some preliminary lemmas in Section 1. We conclude in Section 3 with some remarks and open problems.

1. Notation, Definitions and Preliminary Results

Hereafter we will call $S = \{a = p_0, p_1, \ldots, p_n, p_{n+1} = b\}$ the given set of points, will denote the x-coordinate of a point q by $x(q)$, and will assume that the points in S are already

Subject classifications: Analysis of algorithms: computational complexity, statistics, regression.
Other key words: Analysis of algorithms, data structures, transportation: location.

given in lexicographical order. The points a and b are the *anchor points*, and are assumed to satisfy $x(a) < x(p_1)$ and $x(p_n) < x(b)$. These anchor points have been added in the notation to the point set to be fitted, which is really $S \backslash \{a, b\}$, for the sake of making easier later descriptions.

Some subsets of S will appear often and have a specific notation:

$$L_i = \{p_0, p_1, \ldots, p_i\},\ R_i = \{p_i, p_{i+1}, \ldots, p_{n+1}\},$$

$$S_{ij} = \{p_i, p_{i+1}, \ldots, p_j\}.$$

The convex hull of a point set T will be denoted by $CH(T)$. Let l and r be the leftmost point and the rightmost point in T, respectively (ties are lexicographically broken). The part of the boundary of $CH(T)$ above the line lr is called the *upper hull*; the *lower hull* is the part below the line.

A polygonal chain is said to be *x-monotone*, when the intersection with every vertical line is either empty or a connected set. A polygonal chain is said to be *strictly monotone*, when it does not contain vertical segments; equivalently, the intersection with every vertical line is either empty or a point. The polygonal chains for which we are looking in this paper will be always strictly monotone, unless specifically stated.

Observe that if we are thinking of approximating *functions*, then strictly monotone chains are the suitable ones, while if we think more in terms of locating a route minimizing the maximum distance to customers, non-strictly monotone chains might be considered. In this paper, that possibility arises only in Section 3.

A handful of results will be crucial in the rest of the paper. The next two are straightforward to prove.

Lemma 1. *Let T be a point set, and let l and r be the leftmost point and the rightmost point in T, respectively. The following properties hold:*

(1) The maximum distance between T and the line lr is attained at some vertex of the polygon $CH(T)$;

(2) furthermore, the distance function to the line lr is unimodal along the boundary of the upper hull (see Toussaint 1984), and analogously along the lower hull.

Lemma 2. *Let T be a point set, and let l and r be the leftmost point and the rightmost point in T, respectively. The rays with origin l going to the right of $x = x(l)$ are described by the polar angle from the upper vertical ray from l (refer to Figure 1); let $\phi_{0,1}$ and $\phi_{0,2}$ correspond to the rays supporting $CH(T)$ ($\phi_{0,1} \leqslant \phi_{0,2}$). The following properties hold:*

(1) The maximum distance between the rays through l and the part of the upper hull of T above the rays is an increasing function in the interval $[\phi_{0,1}, \pi]$. When a ray through l rotates clockwise, the corresponding parallel line supporting the upper hull advances and rotates clockwise along the boundary.

(2) The maximum distance between the rays through l and the part of the lower hull of T below the rays is a decreasing function in the interval $[0, \phi_{0,2}]$. When a ray through l rotates clockwise, the corresponding parallel line supporting the lower hull advances and rotates clockwise along the boundary.

(3) Let r^* the ray from l which minimizes the maximum distance to the set T: this maximum distance must be attained by one vertex from the upper hull and another one from the lower hull.

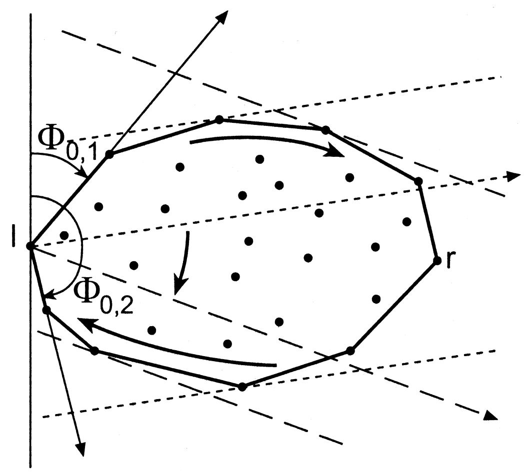

Figure 1. Polar angle and maximum distance.

In order to obtain the ray r^* in the lemma above, in time linear in $|T|$, we can easily adapt the approach proposed by Meggido (1983, 1984), who solves the problem of finding the line that minimizes the maximum distance to a point set using linear programming. Nevertheless, if $CH(T)$ is already available, and given in a suitable data structure, then r^* can be computed much faster, as we show next. We give first an easy solution, then a more efficient one that uses more complicated techniques. In both cases we use the notation of Lemma 2.

Observation 1. *If $CH(T)$ is given in a data structure allowing binary search in the upper and lower hulls, the ray r^* can be computed in $O(\log^2 n)$ time using nested binary search.*

Proof: Assume without loss of generality that the upper hull contains at least as many edges as the lower hull, extend its edge in median position to a line e_m, and compute (in $O(\log n)$ time) the line t_m parallel to e_m supporting the lower hull. If r_m is the ray through l in the same direction, by comparing its distance to e_m and to t_m we know in which sense the ray should rotate towards r^*, and half of the upper chain can be discarded. The process can be iterated, requiring a total of $O(\log n)$ steps. ∎

A technique allowing one in some cases to avoid nested binary search is the *tentative prune-and-search* described by Kirkpatrick and Snoeyink (1995). In their paper they consider problems that involve searching for a special k-tuple with one element drawn from each of k lists, and provide general techniques in the framework of efficiently computing fixed-points of monotone continuous functions. For one function binary search is enough, for three functions they develop the tentative prune-and-search technique, and for two they prove that standard prune-and-search solves the problem. After some definitions from Kirkpatrick and Sno-

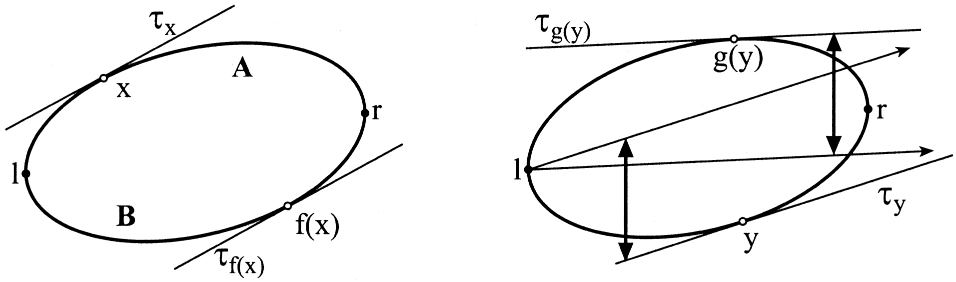

Figure 2. The functions f and g for Lemma 3.

eyink (1995), we reproduce the later result, repeatedly used in this paper, and show next how to use it for computing r^*.

We consider hereafter monotone continuous functions that are defined on sets having the topology of the real interval [0, 1]. A set $\mathcal{F}$ forms a set of *basic functions* if the functions in $\mathcal{F}$ can be evaluated in constant time and a fixed-point of the composition of k functions in $\mathcal{F}$ can be computed in time that depends only on k.

We say that a function $f: A \rightarrow B$ is *piecewise-basic* with complexity $\|f\| = n$ if $A = [a_0, a_n]$ is partitioned into n *basic intervals* by $a_0 \leqslant a_1 \leqslant \cdots \leqslant a_n$ and f is defined piecewise by functions $f_i: [a_{i-1}, a_i] \rightarrow B$ from $\mathcal{F}$. The ordered list $a_0 \leqslant a_1 \leqslant \cdots \leqslant a_n$ is assumed to be given in an array or balanced binary search tree as well as the individual functions f_i.

Theorem 1 (Kirkpatrick and Snoeyink (1995)). *Suppose we are given two piecewise-basic functions: f monotone decreasing over the domain A and g monotone increasing over the domain B. If $B \subseteq f(A)$ and $A \subseteq g(B)$, then we can determine a fixed point of $g \circ f$ in $O(\log \|f\| + \log \|g\|)$ steps.*

The proof of the theorem proceeds by showing that one could prune half of one of the lists for the functions at each step in constant time. For applying this theorem to our problem, we need to have well-defined pairs point/tangent, which is not obvious because the tangent in the vertex of a polygon is not well-defined and all the points interior to an edge have the same tangent. The kinetic framework of Guibas et al. (1983) allows a well-behaved parameterization of pairs point/tangent, which we assume hereafter. More details and alternatives can be found in Kirkpatrick and Snoeyink (1995). The tangent at point q will be denoted τ_q.

Lemma 3. *If $CH(T)$ is given in a data structure allowing binary search in the upper and lower hulls, the ray r can be computed in $O(\log n)$ time.*

Proof: Let A and B be the upper and the lower hull, respectively. Parameterize A and B by x-coordinate. We define functions $f: A \rightarrow B$ and $g: B \rightarrow A$ as follows (Figure 2). $f(x)$ is the point in B such that τ_x and $\tau_{f(x)}$ are parallel. $g(y)$ is the point in A such that the distance between τ_y and its parallel ray through l is the same than the distance between $\tau_{g(y)}$ and its parallel ray through l. Obviously, if x is a fixed point of $g \circ f$ then r^* is parallel to τ_x. As f is decreasing and g is increasing, we can apply Theorem 1 for finding such a fixed point in $O(\log n)$ time, and therefore the claim is proved. ■

We next turn our attention from rays to full lines. For every angle $\theta \in [-\pi, \pi]$ there are two lines with slope tan (θ) that support $CH(T)$, one touching above, $a(\theta)$, another touching below, $b(\theta)$. Let us denote by $l(\theta)$ the line that having slope tan (θ) minimizes the vertical distance to T; obviously $l(\theta)$ is parallel and equidistant to $a(\theta)$ and $b(\theta)$.

The line l^* that minimizes the vertical distance to T will have some slope θ^* (i.e., $l^* = l(\theta^*)$) and the contact points of $a(\theta^*)$ and $b(\theta^*)$ will lie on the same vertical.

The preceding results of this section are easily adapted for dealing with full lines instead of rays, and the following lemma holds:

Lemma 4. *The vertical distance between the lines $a(\theta)$ and $b(\theta)$ is a decreasing function in the interval $[-\pi, \theta^*]$ and an increasing function in the interval $[\theta^*, \pi]$. If $CH(T)$ is given in a data structure allowing binary search in the upper and lower hulls, the line l^* can be computed in $O(\log n)$ time.*

We also consider worth mentioning the following weaker result:

Observation 2. *If $CH(T)$ is given in a data structure allowing binary search in the upper and lower hulls, the line l^* can be computed in $O(\log^2 n)$ time using nested binary search.*

2. 1-Corner Polygonal Chains

The first natural extension of the best-fit-line problem is to look for two rays with the same origin (or segments sharing an endpoint) giving the best approximation for a point set. This is the 1-corner problem, which we consider in this paper.

2.1 Discrete-Corner Polygonal Chains with Fixed Endpoints

The problem of finding the best 1-corner discrete polygonal chain starting at a and ending at b admits a straightforward solution by brute force: take in turn each point $p_1, \cdots, p_n$ as a candidate for a corner point; for every candidate, $n - 1$ distances from the corresponding polygonal to the rest of the points can now be computed. This leads to an algorithm of quadratic complexity. However, by combining some observations we are next showing a more efficient algorithm.

Theorem 2. *Given a set of n points, a discrete-corner polygonal chain with fixed endpoints that minimizes the maximum distance can be found in $O(n \log n)$ time and $O(n)$ space.*

Proof: The algorithm we propose works as follows. We progress incrementally from left to right and for every point p_i, we consider the set $L_i = \{a, p_1, \ldots, p_i\}$. The convex hull $CH(L_i)$ can be obtained from the previous one, and the upper and lower hull can be maintained in a data structure allowing binary search, the overall process requiring $O(n)$ time (O'Rourke 1998). An array for every hull, starting at a and ending at p_i is enough, as deletions and insertions are always performed at the tail. The maximum vertical distance between L_i and the line ap_i can be computed and stored in $O(\log n)$ time, because property 1 from Lemma 1 ensures that searching on the convex hull suffices and property 2 that we can do it in $O(\log n)$). A second sweep, now from right to left, will allow the computation and storage of the maximum vertical distance between $R_i = \{p_i, p_{i+1}, \ldots, p_n, b\}$ and the line p_ib. Finally, we can consider every point $p_i \neq a$, b, $i = 1, \cdots, n$ as candidate bend for a 1-corner chain, compute the maximum distance from the chain to the point set in constant time (using the two stored values), and maintain the minimum value found in the process. The overall complexity is $O(n \log n)$ as claimed. ■

2.2 Discrete-Corner Polygonal Chains Between Convex Polygons

It is natural to consider also the case in which the starting point of the chain a and the endpoint b are not fixed but must belong to some prescribed region. In particular, we present here efficient algorithms for solving the problem of finding a 1-corner discrete polygonal chain that starts and ends at two given convex polygons P_1 and P_2 (i.e., we must have $a \in P_1$ and $b \in P_2$), assuming that the point set P we want to approximate is within the strip defined by the rightmost point of P_1 and the leftmost point of P_2. Notice that each convex polygon can be considered to be the feasible region of a set of linear constraints that a and b must satisfy.

We first characterize the solution of the problem of finding the ray with origin p_i that minimizes the maximum distance to $L_i = \{p_0, p_1, \ldots, p_i\}$ and intersects the convex polygon P_1. Of course a similar characterization applies to the ray from p_i that minimizes the maximum distance to $R_i = \{p_i, p_{i+1}, \ldots, p_{n+1}\}$ and intersects P_2.

Lemma 5. *Let l_i^* be the half-line with origin p_i that minimizes the maximum distance to L_i. Then the ray with origin p_i that minimizes the maximum distance to $L_i = \{p_0, p_1, \ldots, p_i\}$ and intersects the convex polygon P_1 is*

(1) *l_i^* if $l_i^* \cap P_1 \neq \emptyset$, and elsewhere*
(2) *the half-line that has origin in p_i, supports P_1, and gives smallest angle with respect to l_i^*.*

Proof: Obviously if l_i^* intersects P_1 we cannot improve on this solution. If $l_i^* \cap P_1 = \emptyset$ we have to rotate the ray towards P_1 as little as possible according to Lemma 2, therefore the solution we are looking for is the support line of P_1 from p_i that is angularly closest to l_i^* as claimed. ■

Now we have all the ingredients for our next result.

Theorem 3. *The minimax problem for a discrete-corner polygonal chain between convex polygons can be solved within $O(n \log n)$ time and $O(n)$ space.*

Proof: As in the proof of Theorem 2, we progress from left to right and for every point p_i construct $CH(L_i)$, maintaining the upper and lower hull in data structures allowing binary search, the overall process requiring $O(n)$ time (O'Rourke 1998). We can compute then l_i^* in time $O(\log n)$ using Lemma 3. To check whether l_i^* intersects P_1 is trivially done in $O(\log n)$ time. In the negative, the rays from p_i that support P_1 can also be computed in $O(\log n)$ time. A second sweep, now from right to left, will allow the computation and storage for every p_i of the best ray from p_i with respect to R_i that intersects P_2. After these two steps, we have the best two rays from every p_i and we can pick the best pair, obtaining the claimed complexities. ■

By replacing in the previous proof Lemma 3 by Observation 1, a weaker result is obtained:

Observation 3. *The minimax problem for a discrete-corner polygonal chain between convex polygons can be solved in $O(n \log^2 n)$ time and $O(n)$ space with nested binary search.*

Notice that in the results above, P_1 and P_2 could have been the halfplane to the left of $x = x(p_1)$ and the halfplane to the right of $x = x(p_n)$, i.e. we have obtained too the solution of the problem without anchors.

2.3 Free-Corner Polygonal Chains

We consider now the problem of finding 1-corner polygonal chains that fit a set of points in the plane and whose corners are not necessarily at a point from S, but whose beginning point and endpoint are fixed.

Given two consecutive points p_j and p_{j+1} with different x-coordinates, we consider the vertical strip $B_i = \{q | x(p_i) \leq x(q) \leq x(p_{i+1})\}$. We first solve the problem of finding the best 1-corner polygonal chain whose corner is in B_i. This problem can be solved in linear time following the main geometric ideas of the forthcoming lemmas, which would lead to a quadratic solution for the more general problem above, if this single-shot method is used for $B_0, B_1, \ldots, B_n$. Using more structure and a suitable incremental updating, more efficient solutions are possible, as we are showing in this subsection.

Observation 4. *If $CH(L_i)$ and $CH(R_{i+1})$ are given in data structures allowing binary search in the upper and lower hulls, the best 1-corner polygonal chain whose corner is in B_i can be computed in $O(\log^2 n)$ time using nested binary search.*

Proof: Let s_a be the line through a that minimizes the maximum distance to L_i and let s_b be the line through b that minimizes the maximum distance to R_{i+1}. If the two lines intersect within B_i we already have the best 1-corner polygonal. If they intersect at a point q (possibly at infinity) outside B_i, many cases arise. For the sake of clarity we only describe one of the cases, as the others are analogous.

Let t and d (refer to Figure 3) be the points in which s_a and s_b intersect the vertical line $x = x(p_i)$, respectively. Let e^- be the maximum distance from s_a to the points in L_i and let e^+

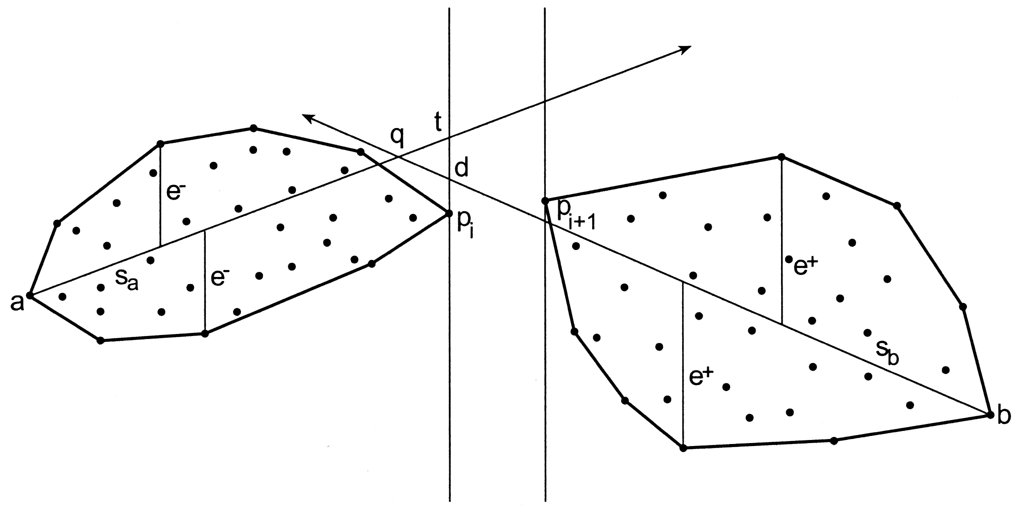

Figure 3. Assumed position for Observation 4.

be the maximum distance from s_b to the points in R_{i+1}. Let us assume that $x(a) \leqslant x(q) \leqslant x(p_i)$, that q is above the line ab, and that t is above d and that $e^- \leqslant e^+$.

If the maximum distance e_d^- from the line ad to L_i is less than or equal than e^+ then the polygonal a-d-b is the solution. Otherwise $e_d^- > e^+$, and by Lemma 2 there is a point m in the segment td such that the maximum distance from the line am to L_i is equal to the maximum distance from the line mb to R_{i+1}. Observe that the polygonal a-m-b is the solution we were looking for, because if $x \in B_i$, $x \neq m$ is above bm, the line xb fits R_{i+1} worse than mb does, and if x is on or below bm then the line ax fits L_i worse than am does. Now we have to show how to obtain the point m.

Let A' be the set of edges on the upper hull of L_i such that the corresponding parallel lines through a intersect the segment td. Similarly, let B' be the set of edges on the lower hull of R_{i+1} such that the corresponding parallel lines through b intersect the segment td.

Take the median segment in say A' (assuming B' not to have more segments than A'), and construct the parallel line through a, which will intersect td in a point z. Compute the maximum distance e_z^+ from the line zb to R_{i+1} and the maximum distance e_z^- from the line az to L_i. According to whether $e_z^- < e_z^+$ or $e_z^+ < e_z^-$ we know which half part of A' can be discarded, and the process is repeated.

We already know from Observation 1 that s_a and s_b can be obtained in time $O(\log^2 n)$. e_d^-, A', and B' can be computed in time $O(\log n)$. Finally, the computation of m above can be done within $O(\log^2 n)$ time, because we have $O(\log n)$ steps, each one requiring $O(\log n)$ time for the computation of e_z^+ and e_z^-. ∎

It is again worthwhile to consider whether the nested binary search leading to $O(\log^2 n)$ time can be avoided. Theorem 1 again comes to the rescue.

Lemma 6. *If $CH(L_i)$ and $CH(R_{i+1})$ are given in data structures allowing binary search in the upper and lower hulls, then the best 1-corner polygonal chain whose corner is in B_i can be computed in $O(\log n)$ time.*

Proof: We borrow all notation and assumptions from the proof of the preceding Lemma. We know from Lemma 3 that s_a and s_b can be computed in $O(\log n)$ time; what remains is to prove that the point m can also be computed within the same time bound.

Let A and B be the upper hull of L_i and the lower hull of R_{i+1}, respectively. Parameterize A and B by the x-coordinates. If $u \in A$, let τ_u be the tangent at u and let a_u be the line through a parallel to τ_u. If $v \in B$, let τ_v be the tangent at v and let b_v be the line through b parallel to τ_v. We define functions $f: A \to B$ and $g: B \to A$ as follows (Figure 4). $f(x)$ is the point in B such that $d(\tau_x, a_x) = d(\tau_{f(x)}, b_{f(x)})$. $g(y)$ is the point in A such that the intersection of $a_{g(y)}$ and b_y is a point on the line $x = x(p_i)$.

Obviously, if x is a fixed point of $g \circ f$ then $a_x \cap (x = x(p_i)) = m$. As f is decreasing and g is increasing, we can apply Theorem 1 for finding such a fixed point in $O(\log n)$ time, and the claim is proved. ∎

Now we are ready to solve the problem described at the beginning of this subsection.

Theorem 4. *The minimax problem for a free-corner polygonal chain with fixed endpoints can be solved in $O(n \log n)$ time and $O(n)$ space.*

Proof: We incrementally progress from right to left, and for every point p_i construct $CH(R_i)$ from $CH(R_{i+1})$, keeping pointers to the points deleted from the hulls in every step, in such a way that the process can be reversed. Then we progress from left to right, and for every point p_i construct $CH(L_i)$ and $CH(R_{i+1})$ from $CH(L_{i-1})$ and $CH(R_i)$, obtained in the previous step, and maintain the upper and lower hulls in data structures allowing binary search. The overall process requires $O(n)$ time. This way we are at every step in the situation of Lemma 6, and the result follows. ∎

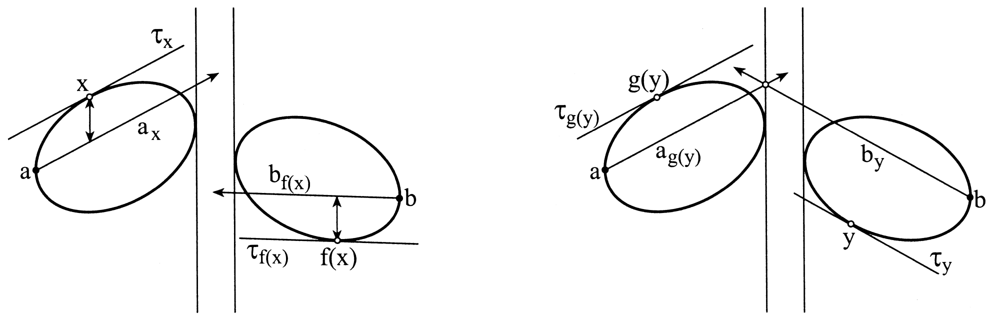

Figure 4. The functions f and g for Lemma 6.

By using Observation 4 instead of Lemma 6, a weaker result is obtained:

Observation 5. *The minimax problem for a free-corner polygonal chain with fixed endpoints can be solved in $O(n \log^2 n)$ time and $O(n)$ space with nested binary search.*

We finally consider the problem of finding a free-corner monotone polygonal chain defined from $x = -\infty$ to $x = +\infty$ (i.e., two rays with common origin such that every vertical intersects their union in exactly one point), minimizing the vertical distance to the given point set; in other words, the non-anchored version of the problem. It is clear that the lines best fitting L_i and R_{i+1} now play the role of rays through a and b in the preceding results of this subsection; with Lemma 4 and Observation 2 in hand instead of Lemma 2, Observation 1 and Lemma 3, it is easy to see that slight modifications allow us to obtain results similar to Theorem 4 and Observation 5 for the non-anchored situation:

Theorem 5. *The minimax problem for a free-corner monotone polygonal chain defined from $x = -\infty$ to $x = +\infty$ can be solved in $O(n \log n)$ time and $O(n)$ space.*

Observation 6. *The minimax problem for a free-corner monotone polygonal chain defined from $x = -\infty$ to $x = +\infty$ can be solved in $O(n \log^2 n)$ time and $O(n)$ space with nested binary search.*

3. Concluding Remarks and Open Problems

We have discussed in this paper several variations on the problem of finding 1-corner polygonal chains that minimize the maximum vertical distance to a given point set with constraints on the bend position or on the endpoints of the chain.

Many more variations on these problems have been considered by Díaz-Báñez (1998) and in Díaz-Báñez et al. (1999). In particular, it is proved there that for constant k it is possible to use a dynamic-programming approach for the k-corners discrete problem allowing degeneracies. Also, results are given in which total length is used as a constraint instead of the number of bends.

As for open problems, let us mention that if the approximation by polygonal chains is thought as a route-planning problem, it could correspond to an obnoxious service, in which case the max-min version should be considered instead of the min-max variations studied here.

Acknowledgments

Most of the problems in this paper were posed and partially solved during the Workshop on Computational Geometry organized by Godfried Toussaint at the Bellairs Research Institute of McGill University in the winter of 1997. We thank the anonymous referees for useful comments and Jack Snoeyink for clarifying to us important details of the paper by Kirkpatrick and Snoeyink (1995). The first author is partially supported by Grupo de Investigación en Localización (Junta de Andalucía) FQM241. The second author is partially supported by Proyectos MEC-DGES-SEUID PB96-0005-C02-01 and PB98-0933. The third author is partially supported by Proyectos MEC-DGES-SEUID PB96-0005-C02-02 and PB98-0933, and CUR Gen. Cat. SGR1999-00356.

References

Ballard, D.H. 1981. A hierarchical representation for curves. *Communications of the ACM* **24** 310–321.

Chan, W.S., F. Chin. 1996. Approximation of polygonal curves with minimum number of line segments or minimum error. *International Journal of Computational Geometry & Applications* **6** 59–77.

Chin, F., A. Choi, Y. Luo. 1992. Optimal generating kernels for image pyramids by piecewise fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14** 1190–1198.

Díaz-Báñez, J.M. 1998. Location of Linear and Piecewise Linear Structures. Ph.D. thesis (in Spanish), Univ. de Sevilla, Sevilla, Spain.

Díaz-Báñez, J.M., F. Gómez, F. Hurtado. 1999. Some Problems on Approximation of Point Sets by Polygonal Chains. Technical Report 11-1999, Dept. de Mat. Aplic., EUI, Univ. Pol. de Madrid, Madrid, Spain.

Drezner, Z., G.O. Wesolowsky. 1989. Location of an obnoxious route. *J. Operational Research Society* **40** 1011–1018.

Gentle, J.E., V.N. Sposito, S.C. Narula. 1988. Algorithms for unconstrained L_1 simple linear regression. *Computational Statistics and Data Analysis* **6** 335–339.

Guibas, L., L. Ramshaw, J. Stolfi. 1983. A kinetic framework for computational geometry. *Proc. 24th FOCS* 100–111.

Hakimi, S.L., E.F. Schmeichel. 1991. Fitting polygonal functions to a set of points in the plane. *Graphical Models and Image Processing* **53** 132–136.

Imai, H., M. Iri. 1986. An optimal algorithm for approximating a piecewise linear function. *Journal of Information Processing* **9** 159–162.

Imai, H., M. Iri. 1988. Polygonal approximations of curve-formulations and algorithms. G.T. Toussaint, ed. *Computational Morphology*, North Holland, Amsterdam.

Kirkpatrick, D., J. Snoeyink. 1995. Tentative prune-and-search for computing fixed points with applications to geometric computation. *Fundamenta Informaticae* **22** 353–370.

Kurozumi, Y., W.A. Davis. 1982. Polygonal approximation by the minimax method. *Computer Graphics and Image Processing* **19** 248–264.

Megiddo, N. 1983. Linear time algorithms for linear programming in IR^3 and related problems. *SIAM J. Comput.* **12** 759–776.

Megiddo, N. 1984. Linear programming in linear time when the dimension is fixed. *Journal of the Association for Computing Machinery* **31** 114–127.

Melkman, A., J. O'Rourke. 1988. On polygonal chain approximation. G.T. Toussaint, ed. *Computational Morphology*. North Holland, Amsterdam.

O'Rourke, J. 1981. An on-line algorithm for fitting straight lines between data ranges. *Comm. ACM* **24** 574–578.

O'Rourke, J. 1998. *Computational Geometry in C*, 2nd ed. Cambridge University Press, New York.

Rice, J. 1964. *The Approximation of Functions, Vol. 1: The Linear Theory*. Addison-Wesley.

Toussaint, G.T. 1984. Complexity, convexity and unimodality. *International Journal of Computer and Information Sciences* **13** 197–217.

Toussaint, G.T. 1985. On the complexity of approximating polygonal curves in the plane. *Proc. IASTED, International Symposium on Robotics and Automation*, Lugano, Switzerland.