COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING 36, 31–41 (1986)

Computational-Geometric Methods for Polygonal Approximations of a Curve

HIROSHI IMAI

Department of Computer Science and Communication Engineering, Kyushu University, Hakozaki, Fukuoka 812, Japan

AND

MASAO IRI

Department of Mathematical Engineering and Instrumentation Physics, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan

Received May 1985; revised May 13, 1986

In cartography, computer graphics, pattern recognition, etc., we often encounter the problem of approximating a given finer piecewise linear curve by another coarser piecewise linear curve consisting of fewer line segments. In connection with this problem, a number of papers have been published, but it seems that the problem itself has not been well modelled from the standpoint of specific applications, nor has a nice algorithm, nice from the computational-geometric viewpoint, been proposed. In the present paper, we first consider (i) the problem of approximating a piecewise linear curve by another whose vertices are a subset of the vertices of the former, and show that an optimum solution of this problem can be found in a polynomial time. We also mention recent results on related problems by several researchers including the authors themselves. We then pose (ii) a problem of covering a sequence of n points by a minimum number of rectangles with a given width, and present an $O(n \log n)$-time algorithm by making use of some fundamental established techniques in computational geometry. Furthermore, an $O(mn(\log n)^2)$-time algorithm is presented for finding the minimum width w such that a sequence of n points can be covered by at most m rectangles with width w. Finally, (iii) several related problems are discussed. © 1986 Academic Press, Inc.

1. INTRODUCTION

Piecewise linear or polygonal curves are often used to approximate complex boundaries of figures in cartography and other geographical data processing, computer graphics, pattern recognition, etc. In particular, when we want to reduce the size of a map, or compress the amount of cartographic data, the problem of approximating a given finer piecewise linear curve by another coarser piecewise linear curve such that the maximum distance of the former from the latter is bounded by a constant is of fundamental importance. The general problem for finding an approximate piecewise linear curve with the minimum number of segments seems quite difficult. We here formulate two types of approximation problems with additional constraints and present efficient algorithms for them, which will be practically useful enough.

A number of papers have been published for various kinds of piecewise linear approximation problems. Ichida and Kiyono [4] considered the problem of approximating a piecewise linear curve by a sequence of segments which may not be continuous. Their approach is simple, but the algorithm given in [4] is rather complicated. Kurozumi and Davis [9] generalized the result of [4] and gave an algorithm that finds an approximate piecewise linear curve whose maximum distance from the given is bounded by a constant. However, their algorithm does not necessarily find a curve with the minimum number of segments, nor is it shown in what sense the solutions of their algorithm are optimum. In both papers, the complexity of the algorithm is not well analyzed. Williams [17] and Sklansky and Gonzalez [14] considered the problem of approximating a piecewise linear curve with n vertices by another whose vertices are a subset of the former. Their algorithms are fast but do not necessarily give an optimal solution in the sense that the number of segments of the latter is minimum. Furthermore, even the maximum distance of a computed curve from the given may be greater than a given constant unless the given curve satisfies some monotonicity conditions.

In this paper, we first consider the problem studied in [14, 17], and show that, for that problem, an *optimum* solution can be found in $O(n^3)$ time. After submitting the original version of this paper (see [6]), further developments on this problem have been done by several researchers, including the authors themselves. We also mention the results of those developments briefly.

We then pose the problem of covering a sequence of n points by the minimum number of rectangles with a fixed width and an infinite length. This formulation of the problem will be more appropriate for the application to the cartographic problem of reducing the size of a map, and admits an algorithm more effective in terms of time complexity $O(n \log n)$. Our approach is somewhat similar to that of Ichida and Kiyono [4], but our algorithm presented here is much faster. The algorithm utilizes fundamental established algorithms in computational geometry, such as a linear-time algorithm for finding a convex hull of two convex polygons and the so-called "rotating-caliper" method of Shamos [13] and Toussaint [15].

We further consider the problem of finding the minimum width w such that a sequence of n points can be covered by at most m rectangles with width w. For this problem, an $O(mn(\log n)^2)$-time algorithm is presented. We also consider the problem of covering the points (vertices) of a polygon by rectangles, and how to obtain an approximate piecewise linear curve from the covering rectangles. Ballard [2] considered another problem of covering a sequence of points by rectangles. We refer to it algorithmically in Section 2 and discuss the relation between our problem and Ballard's at the end of Section 3. Finally, several related problems are discussed.

2. AN APPROXIMATE CURVE CONNECTING A SUBSEQUENCE OF POINTS

2.1. The Minimum Subsequence Approximation Problem

The minimum subsequence approximation problem (with respect to a given error bound w') is to find an approximate piecewise linear curve with the minimum number of segments such that its points $P_{i(1)}, P_{i(2)}, \ldots, P_{i(m)}$ are a subset of points $P_1, \ldots, P_n$ of the given curve (where $1 \equiv i(1) < i(2) < \cdots < i(m) \equiv n$) and that the "error" of each segment $P_{i(l)}P_{i(l+1)}$ $(l = 1, \ldots, m-1)$ is at most w', where the error of segment P_iP_j $(i < j)$ is defined appropriately according to respective applications. Often the error of segment P_iP_j is defined to be the maximum of the distance between the segment P_iP_j and each point P_k $(i \le k \le j)$. This error criterion will be referred to as (e1). For the minimum subsequence problem under the error criterion (e1), the so-called cone-intersection method is proposed in [14, 17], but it does not necessarily give the optimum curve, and what is worse it may produce a solution that does not meet the error criterion.

This problem is easily solved as follows. Consider a directed graph G of which each vertex v_i represents a point P_i in the given sequence and whose arcs (v_i, v_j) are placed from vertex v_i to vertex v_j $(i < j)$ if and only if the error of segment P_iP_j is at most w'. Note that G is by definition acyclic. To each arc (v_i, v_j), assign the weight $j - i - 1$, which is equal to the number of points which would be skipped if we connected P_i and P_j directly. Then, the approximate curve with the minimum number of segments is represented by the longest path from v_1 to v_n in the graph G where the length is measured by weights thus defined. Since G is acyclic, a longest path from v_1 to v_n can be found in time proportional to the number of arcs of the graph, i.e., $O(n^2)$. (Or, simply find a shortest path from v_1 to v_n by breadth-first search in G where the length of a path is defined to be the number of arcs in the path.)

A naive way of constructing G is to check, for each i, j $(1 \le i < j \le n)$, whether the distance of the segment P_iP_j and each point P_k with $i \le k \le j$ is at most w' or not. This will take $O(n^3)$ time. Thus, we see, from the above, that the problem can be solved in $O(n^3)$ time.

In the above algorithm, the most time-consuming part is the construction of G. It might be possible to construct the graph G in subcubic time with respect to n. In fact, quite recently, O'Rourke and Melkman [10, 11] have shown that G can be constructed in $O(n^2 \log n)$ time by essentially extending the cone-intersection method and hence the minimum subsequence problem under (e1) can be solved in $O(n^2 \log n)$ time.

2.2. *Extensions*

Besides the error criterion (e1), other error criteria have been used in different situations. Toussaint [16] considers an error criterion (termed the "*parallel-strip*" criterion) such that the error of segment P_iP_j is defined to be the maximum distance between the *line* connecting P_i and P_j and each point P_k $(i \le k \le j)$, which will be referred to as (e2). In [16], it is shown that, by using the on-line convex hull algorithms [1, 12], the graph G under this error criterion (e2) can be constructed in $O(n^2 \log n)$ time, and hence the minimum subsequence problem under (e2) can be solved in $O(n^2 \log n)$ time.

Ballard [2] considers an error criteria, in connection with his data structure named *strip trees*, such that the error of segment P_iP_j is defined to be the minimum width (divided by 2) of the rectangle which contains all the points $P_i, P_{i+1}, \ldots, P_j$ and whose sides are parallel or vertical to segment P_iP_j and pass P_i and P_j; the error of P_iP_j is $+\infty$ if such a rectangle does not exist. (The width of the rectangle is the distance of the two sides parallel to P_iP_j.) This error criterion will be referred to as (e3). It is easy to show that the minimum subsequence problem under this criterion (e3) can be solved in $O(n^2 \log n)$ time by means of the on-line convex hull algorithms, as in [16]. Thus, the minimum subsequence approximation problem under each of the error criteria (e1), (e2), (e3) can be solved in $O(n^2 \log n)$ time.

Imai and Iri [8] consider the problem of finding a subsequence of at most m points, where m is a given constant, that minimizes the error under respective criteria (e1), (e2), (e3) (the error of the subsequence is the maximum among the errors of segments connecting neighboring pairs of points in the subsequence). It is shown that the problem under (e1) can be solved in $O(n^2 (\log n)^2)$ time and the

TABLE 1
Summary of the Time Complexities of the Subsequence Approximation Problems

Error criteria	To minimize the number of points for a given error bound	To minimize the error for a given number of points
(e1)	$O(n^2 \log n)^a$	$O(n^2 (\log n)^2)^c$
(e2)	$O(n^2 \log n)^b$	$O(n^2 \log n)^c$
(e3)	$O(n^2 \log n)^c$	$O(n^2 \log n)^c$

[a] O'Rourke, Melkman [9, 10].
[b] Toussaint [16].
[c] Imai and Iri [8].
Note. n is the number of points.

problem under (e2) and (e3) in $O(n^2 \log n)$ time. We show these results on the subsequence approximation problems in Table 1.

3. COVERING A SEQUENCE OF POINTS BY RECTANGLES

3.1. *Preliminaries*

Given a sequence of n points $P_1, P_2, \ldots, P_n$, we say that rectangles $R_1, R_2, \ldots, R_r$ (in sequence) with width w cover the sequence of points if there are indices $i(0), i(1), \ldots, i(r)$ $(1 \equiv i(0) < i(1) < \cdots < i(r-1) < i(r) \equiv n)$ such that, for each $q = 1, \ldots, r$, rectangle R_q contains points $P_{i(q-1)}, \ldots, P_{i(q)}$. By a rectangle with width w, we mean a rectangle with width w and an infinite (i.e., sufficiently large) length. When we consider a covering rectangle with width w, we suppose that its length is made minimal with its width kept unchanged. Then, the problem is to find rectangles (in sequence) of width w which cover the given sequence of points and the number of which is minimum (Fig. 1). We shall give an $O(n \log n)$-time algorithm for this problem.

First, consider the problem of determining whether or not a set of points can be covered by a rectangle with width w. (Note that there is a rectangle with width w containing a set of points if and only if there is a line from which the maximum distance to those points is at most $w' = w/2$.) For this problem, the following lemma is almost obvious (e.g., [4]).

LEMMA 3.1. *There is a rectangle with width w which can cover a set of points if and only if there is a rectangle with width w covering the set of points such that one of the*

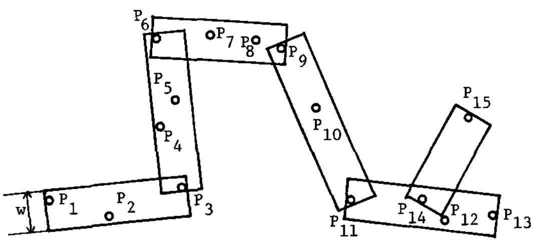

FIG. 1. Covering a sequence of points by rectangles.

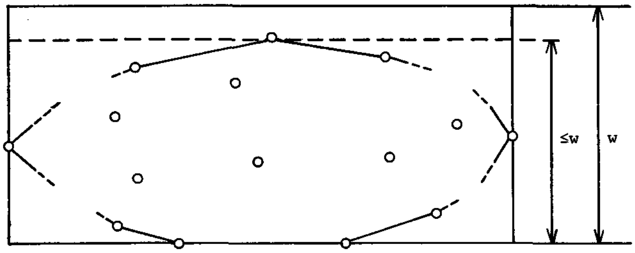

FIG. 2. The convex hull of points and a covering rectangle.

two parallel sides of distance w of the rectangle contains some edge of the convex hull of the set of points (Fig. 2).

Thus, the problem is directly connected with the convex hull of points. For points $P_1, \dots, P_n$, we denote by $\mathrm{CH}(i, j)$ the convex hull of points $P_i, \dots, P_j$ $(1 \le i \le j \le n)$. We also denote by $W(i, j)$ the minimum width of a rectangle that can cover points $P_i, \dots, P_j$. For an edge e of a convex hull, we call the vertex of the convex hull that is farthest from the line passing through e the "*antipodal point*" of e, and denote by $d(e)$ the distance of the antipodal point of e from the straight line through e. Then, from Lemma 3.1, we have

$$W(i, j) = \min\{d(e) | e \text{ is on } \mathrm{CH}(i, j)\}.$$

Given a convex hull with N edges, we can find the antipodal points of all the edges in $O(N)$ time by the so-called rotating-caliper method (Shamos [13], Toussaint [15]) well known in computational geometry. For completeness, we shall describe an $O(N)$-time algorithm for a convex hull whose vertices are $Q_0, \dots, Q_{N-1}$ in the clockwise order, where indices are counted modulo N.

ALGORITHM C (Caliper method).

```
(C1) Find the antipodal point $Q_k$ of edge $e_0 = Q_{N-1}Q_0$, and compute $d(e_0)$;
     $i := 1$;
(C2) calculate the distance $d_k$ and $d_{k+1}$ of the line passing through edge $e_i = Q_{i-1}Q_i$
     from points $Q_k$ and $Q_{k+1}$, respectively;
(C3) if $d_k \le d_{k+1}$ then $k := k + 1$ and return to (C2); otherwise ($Q_k$ is an
     antipodal point of $e_i$), $d(e_i) := d_k$; $i := i + 1$; if $i < N - 1$ then return to
     (C2) else halt.
```

The validity of Algorithm C is easy to demonstrate. Clearly, Algorithm C takes $O(N)$ time.

The convex hull of n points can be found in $O(n \log n)$ time (e.g., Graham [3]), so that we have the following lemma (a similar lemma for the Chebyshev approximation problem is shown in Shamos [13]).

LEMMA 3.2. *$W(1, n)$ can be computed in $O(n \log n)$ time. Furthermore, when the convex hull* CH $(1, n)$ *is given, $W(1, n)$ can be computed in $O(n)$ time.*

3.2. Finding the Minimum Number of Rectangles Covering a Given Sequence of Points

We now consider the problem of finding the minimum number of rectangles with width w that cover a sequence of points $P_1, \ldots, P_n$. For this problem, the following simple greedy algorithm yields the optimum solution, as can easily be shown.

```
ALGORITHM G (Greedy algorithm).
(G1) $i := 1$;
(G2) find $i^* = \max\{j \mid i < j \le n,\ W(i, j) \le w\}$;
(G3) cover points $P_i, \ldots, P_{i^*}$ by a rectangle with width $w$;
     if $i^* = n$, then halt; otherwise $i := i^*$ and return to (G2).
```

The main step of Algorithm G is step (G2). We shall show that step (G2) for $i = 1$ can be executed in $O(i^* \log i^*)$ time. Since $W(1, j)$ is nondecreasing in j, we can employ the doubling binary search technique to find i^*. First, we find m and l such that $W(1, m) \le w$, $W(1, l) > w$, and $m < l \le \min\{2m, n\}$ (or we find $W(1, n) \le w$) by the doubling search; i.e., starting with $m = 1$ and $l = 2$, we repeat doubling both m and l until the condition is satisfied. Then, i^* is known to lie in the interval between m and $l - 1$. The exact value of i^* can be determined by means of binary search as follows.

```
ALGORITHM B (Binary search).
(B1) $k := m$; $l := l - 1$; find and retain CH(1, $k$);
(B2) $h := \lceil (k + l)/2 \rceil$;
      find CH($k + 1, h$);
      find CH(1, $h$) by computing the convex hull of CH(1, $k$) and CH($k + 1, h$);
      compute $w_h := W(1, h)$ by Algorithm C;
(B3) if $w_h \le w$ then $k := h$ else $l := h - 1$;
      if $k = l$ then $i^* := k$ and halt; otherwise, retain CH(1, $k$) and return to (B2).
```

The validity of Algorithm B is obvious. The complexity is evaluated as follows: The execution of the doubling search is mainly to compute $W(1, 2^g)$ for $g = 1, 2, \ldots, M + 1$, where $M = \log_2 m$. Since $W(1, j)$ can be found in $O(j \log j)$ time (Lemma 3.2), we can execute the doubling search in $O(\sum_{g=1}^{M+1} 2^g \log 2^g) = O(i^* \log i^*)$ time (note that $m = O(i^*)$).

In Algorithm B, all the steps except the part for finding CH($k + 1, h$) in (B2) can be executed in $O(i^* \log i^*)$ time, because, in (B2), CH(1, h) can be found in $O(h)$ time by computing the convex hull of two convex polygons CH(1, k) and CH($k + 1, h$) (Shamos [13]), and then $W(1, h)$ can be computed in $O(h)$ time by Algorithm C. In the gth execution of (B2), CH($k + 1, h$) is the convex hull of at most $m2^{-g} = 2^{M-g}$ points, which can be found in $O(2^{M-g} \log 2^{M-g})$ time. Since (B2) is executed at most M times, computation of CH($k + 1, h$) in the whole execution of Algorithm B takes $O(\sum_{g=1}^{M} 2^{M-g} \log 2^{M-g}) = O(i^* \log i^*)$ time. (In the binary search, if one naively computed $W(1, h)$ in $O(h \log h)$ time, the whole algorithm would take $O(i^* (\log i^*)^2)$ time. So, in Algorithm B given here, $W(1, h)$ is computed as in step (B2).)

In Algorithm G, step (G2) for general i can be executed in $O((i^* - i) \log(i^* - i))$ time by means of the above algorithm, and hence Algorithm G can be completed in $O(n \log n)$ time. (Note that the n in "$\log n$" in the complexity function may be replaced by something like the average number of points covered by a rectangle in the optimum solution, so that, if that number is bounded by, say, a constant, then the algorithm runs in linear time.)

3.3. Covering a Sequence of Points by at Most m Rectangles with Minimum Width

We now consider the problem of minimizing width w such that a sequence of n points can be covered by at most m rectangles with width w, and present an $O(mn(\log n)^2)$-time algorithm. The algorithm is simple, and is described as follows, where we set $W(i, n+1) = \infty$ $(1 \le i \le n)$.

ALGORITHM A (Compute the minimum width w^*).

```
(A1) $m' := m$; $i := 1$; $w := 0$;
(A2) find $j$, by the doubling binary search, such that
       (i) $i < j \le n$,
       (ii) a sequence of points $P_i, \dots, P_n$ cannot be covered by at most $m'$ rectangles with width less than $W(i, j)$, and
       (iii) a sequence of points $P_i, \dots, P_n$ can be covered by at most $m'$ rectangles with width less than $W(i, j+1)$;
(A3) $w := \max\{w, W(i, j)\}$; $i := j$; $m' := m' - 1$;
     if $i = n$ then halt; otherwise, return to (A2).
```

At the end of algorithm A, w is the minimum width w^*.

LEMMA 3.3. *Algorithm* A *correctly computes the minimum width* w^*.

Proof. We have only to show the following claim: In a sequence of rectangles with width w^* covering the sequence of points, which is found by Algorithm G, the first rectangle covers points $P_1, \dots, P_{j_1}$, where j_1 is defined to be that j which is found in the first execution of step (A2) (if this is true, then the lemma can be easily proven by induction). Since a sequence of points $P_1, \dots, P_n$ can be covered by at most m rectangles with width less than $W(1, j_1+1)$, we have $w^* < W(1, j_1+1)$. Also, since the sequence cannot be covered by at most m rectangles with width less than $W(1, j_1)$, we have $W(1, j_1) \le w^*$. Hence, $W(1, j_1) \le w^* < W(1, j_1+1)$. This shows that the claim is true. □

Concerning the complexity of Algorithm A, step (A2) can be executed in $O(n(\log n)^2)$ time by means of the binary search and Algorithm G in Section 3.2. Since steps (A2) and (A3) are executed at most m times, the total time complexity of Algorithm A is $O(mn(\log n)^2)$ time. (A more detailed analysis shows that $(\log n)^2$ in the complexity function can be replaced by $\log n \cdot \log(n/m)$.)

3.4. Covering the Points of a Polygon by Rectangles

Consider a polygon whose vertices (or, points) are given in clockwise order; P_1, $P_2, \dots, P_n$. We say that rectangles (in sequence) cover the points of the polygon if they cover a sequence of points $P_i, P_{i+1}, \dots, P_n, P_1, \dots, P_i$ for some i with $1 < i \le n$.

We now consider, as in Section 3.2, the problem of finding the minimum number of rectangles covering the points of the polygon. The minimum number m^* can be found by the following algorithm, the validity of which is evident.

ALGORITHM.

```
(1) Find the minimum number of rectangles (in sequence) covering the sequence of
    points $P_1, P_2, \ldots, P_n, P_1$ by Algorithm G;
(2) find a covering rectangle $R$ among those rectangles except the last one in the
    sequence such that the number of points covered by $R$ is minimum;
(3) for each point $P_i$ covered by $R$, find the minimum number $m_i$ of rectangles
    covering the sequence of points $P_i, \ldots, P_n, P_1, \ldots, P_i$;
(4) let $m^* = \min\{m_i | P_i$ is covered by $R\}$.
```

(It should be noted that the number of those rectangles which are found in (1) is at most $m^* + 1$.)

This algorithm can be implemented by means of Algorithm G, where Algorithm G is executed $O(n/m^*)$ times, because the rectangle R which is found in step (2) covers $O(n/m^*)$ points. Hence, this algorithm runs in $O((n^2 \log n)/m^*)$ time.

We next consider, as in Section 3.3, the problem of minimizing width w such that the points of the polygon can be covered by at most m rectangles with width w. This minimum width w^* can be found by the following algorithm.

ALGORITHM.

```
(1) For each $i$ with $1 \le i \le n$, do the following (1.1) and (1.2) for the sequence of
    points $P_i, \ldots, P_n, P_1, \ldots, P_i \equiv Q_0, \ldots, Q_n$ ($W(0, j)$ is the minimum width of a
    rectangle that can cover points $Q_0, \ldots, Q_j$);
    (1.1) find $j$, by the binary search, such that
        (i) $0 < j \le n$,
        (ii) the sequence of points $Q_0, \ldots, Q_n$ cannot be covered by at most $m$
             rectangles with width less than $W(0, j)$,
        (iii) the sequence of points $Q_0, \ldots, Q_n$ can be covered by at most $m$ rectangles
              with width less than $W(0, j + 1)$ (we set $W(0, n + 1) = \infty$),
    (1.2) if the sequence of points $Q_0, \ldots, Q_n$ can be covered by at most $m$
          rectangles with width $W(0, j)$, then $w_i := W(0, j)$ else $w_i := \infty$;
(2) let $w^* = \min\{w_i | 1 \le i \le n\}$.
```

The validity of this algorithm can be shown by arguments similar to the proof of Lemma 3.3. Since step (1.1) for each i can be executed in $O(n(\log n)^2)$ time by means of the binary search and Algorithm G, the complexity of this algorithm is $O((n \log n)^2)$.

We summarize the results obtained in Sections 3.1–3.4 in Table 2.

3.5. *Finding an Approximate Piecewise Linear Curve*

Covering a sequence of points by rectangles with width w does not necessarily produce directly an approximate piecewise linear curve whose maximum distance from those points is at most $w' = w/2$, so that, in this subsection, we consider the connections between the problem of covering by rectangles and that of finding an approximate curve.

First, it should be noted that, in applications to the problem of reducing the size of a map and the like, covering by rectangles is quite satisfactory, because, in such cases, two points whose distance is within a tolerance, say w, can be considered hardly distinguishable from each other.

TABLE 2
Summary of the Time Complexities of the Rectangle Covering Problem for a Sequence of n Points

	To minimize the number of rectangles for a given width	To minimize the width for a given number m of rectangles
Sequence of points forming a path	$O(n \log n)$	$O(mn(\log n)^2)$
Sequence of points forming a polygon	$O(n^2 \log n)$	$O(n^2(\log n)^2)$

Second, a straightforward algorithm for determining an approximate curve by means of the algorithm for covering by rectangles will be to find the covering rectangles with width $\leq w/2$, and then to draw, as is easily done, an approximate piecewise linear curve whose maximum error is at most $w' = w/2$. The number of segments of the curve, however, may be greater than the minimum number of rectangles with width w that cover the sequence of points, and than the minimum possible number of segments satisfying the same error condition.

Last, we discuss an algorithm which does not stick to covering rectangles but utilizes the algorithmic techniques developed in the previous subsections. This algorithm proceeds as Algorithm G: In the step corresponding to (G2), we search for the maximum i^* of i' such that there is a line from which the distance to each point P_j $(i \leq j \leq i')$ is at most w' and which intersects a given line segment L (the latter condition is omitted in the case of $i = 1$); in the step corresponding to (G3), we take, as an approximate line, a line such that the maximum distance to points P_j $(i \leq j \leq i^*)$ is minimum and that it intersects L. (Here, it should be noted that this line does not necessarily intersect L at one of the two endpoints of L.) We then update the segment L to be the intersection of the line just taken and the disk with center P_{i^*} and radius w'. Using all the lines taken in this algorithm, we can easily obtain an approximate curve whose maximum error is bounded by w'.

This modified algorithm can be executed in $O(n \log n)$ time in total by algorithms similar to Algorithms B and C, where the following should be noted. Given a convex polygon and a segment L, we can find a line intersecting L, such that the maximum distance of the line from the polygon is minimum, in time proportional to the number of the vertices of the polygon by the rotating-caliper method. In applying the rotating-caliper method, a pair of calipers is not only a pair of an edge and its antipodal point as in Algorithm C but also may be a pair of "antipodal points."

This modified algorithm would find an approximate curve of fewer segments, but it is not known under what criterion the solution curve is optimal.

3.6. Relations with Ballard's Strip Tree

Ballard [2] considered a similar but different problem of covering by rectangles in connection with his data structure, called the strip tree. The strip tree is a data structure for representing a polygonal curve hierarchically in terms of the degree of approximation of the curve by rectangles.

Algorithmically, Ballard's covering by rectangles can be treated as the subsequence problem as discussed in Section 2. Ballard's paper [2] does not give an algorithm for finding the minimum number of rectangles covering a given sequence of n points in his sense, but only gives a heuristic algorithm. As noted in Section 2.2, the minimum covering by rectangles in Ballard's sense can be found in $O(n^2 \log n)$ time (see Sect. 2.2), whereas the minimum covering by rectangles proposed in this paper can be found in $O(n \log n)$ time.

Thus, algorithmically, our covering by rectangles is easier to handle than that in Ballard's sense. Furthermore, we can develop another "strip tree" by using our covering by rectangles, which has the advantage from the algorithmic viewpoint, and hence would deserve further investigation.

4. DISCUSSIONS

In this paper, we considered two types of problems in connection with the general piecewise linear curve approximation problem: (i) to find an approximate curve connecting a subsequence of points of the minimum cardinality, and (ii) to cover a sequence of points by the minimum number of rectangles. For these two, we gave the algorithms which correctly find the optimum solutions. Comparing the two from the viewpoint of size reduction of a map, the latter problem will be more useful in reality because it is computationally less complex than the former and, at the same time, ordinarily gives a better approximation in the sense that fewer segments are needed for the same tolerance.

However, for the general approximation problem, there still remain problems left open. For example, a challenging problem is to find a simple approximate curve when the given piecewise linear curve is simple (a piecewise linear curve is *simple* if no pair of nonadjacent segments intersects). Also, from the computational geometric point of view, it would be interesting to apply computational geometric shortest-path algorithms to the polygonal approximation problem, since, concerning the problem of approximating a piecewise linear function, Imai and Iri [7] gave an optimal algorithm by means of such algorithms (see also Imai [5]).

ACKNOWLEDGMENTS

The authors would like to thank Dr. Takao Asano of Sophia University, Tokyo for his useful comments on the algorithm in Section 3.2. They are also indebted to a referee for his valuable suggestions on the original version of this paper. Part of the work by the first author was done while he was at the Department of Mathematical Engineering, University of Tokyo.

REFERENCES

1. D. Avis, H. ElGindy, and R. Seidel, Simple on-line algorithms for convex polygons, in *Computational Geometry* (G. T. Toussaint, Ed.), pp. 23–42, North-Holland, Amsterdam, 1985.
2. D. H. Ballard, Strip trees: A hierarchical representation for curves, *Commun. ACM* **24**, No. 5, 1981, 310–321.
3. R. L. Graham, An efficient algorithm for determining the convex hull of a finite planar set, *Inf. Process. Lett.* **1**, 1972, 132–133.
4. K. Ichida and T. Kiyono, Segmentation of plane curves, *Trans. Electron. Commun. Eng. Japan* **58-D**, 1975, 689–696. [Japanese]
5. H. Imai, *Studies on Algorithms in Computational Geometry*, Doctoral dissertation, University of Tokyo, 1986. [Japanese]
6. H. Imai and M. Iri, *Computational-Geometric Methods for Polygonal Approximations of a Curve*, Research Memorandum RMI 85-01, Department of Mathematical Engineering, University of Tokyo, 1985.

7. H. Imai and M. Iri, An optimal algorithm for approximating a piecewise linear function, *J. Inf. Process. submitted.*
8. H. Imai and M. Iri, *Polygonal Approximations of a Curve—Formulations and Solution Algorithms.* Technical Report CSCE-86-C07, Department of Computer Science and Communication Engineering, Kyushu University, 1986.
9. Y. Kurozumi and W. A. Davis, Polygonal approximation by the minimax method, Comput. Vision Graphics Image Process. **19**, 1982, 248–264.
10. J. O'Rourke, Polygonal chain approximation: An improvement to an algorithm of Imai and Iri, Department of Electrical Engineering and Computer Science, Johns Hopkins University, 1985.
11. J. O'Rourke, private communication, 1985.
12. F. P. Preparata, An optimal real time algorithm for planar convex hulls, *Commun. ACM* **22**, 1979, 443–450.
13. M. I. Shamos, *Computational Geometry*, Ph.D. thesis, Department of Computer Science, Yale University, 1978.
14. J. Sklansky and V. Gonzalez, Fast polygonal approximation of digitized curves, *Pattern Recognit.* **12**, 1980, 327–331.
15. G. T. Toussaint, Solving geometric problems with the "rotating calipers," in *Proceedings, IEEE MELECON*, Athens, Greece, 1983.
16. G. T. Toussaint, On the complexity of approximating polygonal curves in the plane, in *Proceedings, IASTED Int. Symp. Robotics and Automation*, Lugano, Switzerland, 1985.
17. C. M. Williams, An efficient algorithm for the piecewise linear approximation of planar curves, *Comput. Vision Graphics Image Process.* **8**, 1978, 286–293.