Polygonal Approximation by Boundary Reduction *

Laurence Boxer † Chun-Shi Chang ‡ Russ Miller § Andrew Rau-Chaplin ¶

Abstract

We give a sequential algorithm for approximating a given polygon P by another polygon P' such that P' is a "good approximation" of P, and has fewer edges. We formalize the notion of a "good approximation" in terms of the Hausdorff metric and show through experimentation that the application of this metric leads to visually satisfying approximations. Our algorithm modifies that of [Leu and Chen] to produce output that better approximates the input.

Key words and phrases: *polygon, Hausdorff metric, analysis of algorithms*

1 Introduction

In [Leu and Chen, 1988], a sequential algorithm is given for replacing a polygon P by another polygon P' such that P' has fewer edges than P. The algorithm can be implemented in optimal $\Theta(n)$ time, where n is the number of edges of the input polygon P. The basic idea of this algorithm is as follows. Repeatedly, a run of 2 or 3 consecutive edges that differs from its chord (*i.e.*, the line segment that connects the extreme endpoints of the run) by less than a pre-specified tolerance value (if such a run exists) is replaced by its chord. The process terminates when either the resulting polygon has a small number of remaining edges or there are no more runs that differ from their chords by more than the tolerance value. The replacement of runs that deviate only slightly from linearity by their chords can be regarded as a technique of "smoothing" or "noise reduction." The algorithm produces good results for a large class of input polygons.

* *Pattern Recognition Letters* 14 (1993), 111-119

† Department of Computer and Information Sciences, Niagara University, Niagara University, NY 14109, USA. Research partially supported by a grant from the Niagara University Research Council. This paper was written during the author's sabbatical visit to the State University of New York - Buffalo.

‡ Department of Computer Science, State University of New York - Buffalo, Buffalo, New York 14260, USA. Research partially supported by NSF grant IRI-9108288.

§ Department of Computer Science, State University of New York - Buffalo, Buffalo, New York 14260, USA. Research partially supported by NSF grant IRI-9108288.

¶ School of Computer Science, Carleton University, Ottawa, Ontario K1S 5B6, Canada. Research partially supported by the Natural Sciences and Engineering Research Council of Canada.

Unfortunately, if the repetition of the smoothing step is not properly controlled, the result of this smoothing process may not be a good approximation to the original polygon P. In this paper we formalize the notion of a "good approximation" in terms of the Hausdorff metric d_H [Nadler, 1978], *i.e.*, we require that neither P nor P' have a point whose distance to the nearest point of the other polygon is larger than the tolerance value τ. We show that the algorithm of [Leu and Chen] may result in large values of $d_H(P, P')$, and we show how to modify this algorithm so that $d_H(P, P')$ remains small. Our algorithm pays a price for its "accuracy" in terms of its asymptotic efficiency. That is, the running time of our algorithm is $O(n + r^2)$, where $r = |E(P)| - |E(P')|$ is the number of edges removed from P to obtain P'. In practice, however, we often find that our algorithm is actually faster than that of [Leu and Chen], as our modification more carefully controls the number of "global" smoothing steps.

The paper is organized as follows. In Section 2, we define terminology. In Section 3, we present the algorithm of [Leu and Chen], an example that demonstrates how this algorithm can produce unsatisfactory results, and our algorithm, which improves upon the output of that of [Leu and Chen]. In Section 4, we give some experimental results, comparing the outputs of the two algorithms on the same inputs.

2 Preliminaries

2.1 Arc Terminology

Much of the terminology we use is taken from [Leu and Chen]. Input to the problem includes a circularly-ordered set of vertices $V = \{v_0, v_1, \ldots, v_{n-1}\}$ of a polygon, and a tolerance value τ. Output is a circularly-ordered subset of V. Vertices are represented by their Cartesian coordinates. All vertices and line segments are assumed to be in the Euclidean plane E^2.

The maximum deviation of a polygonal arc from its chord occurs at one of the vertices of the arc. Let $\{P, Q, R, S\} \subset V$. For an arc of the form $\overline{PQ} \cup \overline{QR}$, the maximum arc-to-chord deviation is just the distance from Q to $\overline{PR}$. Figure 1 shows an arc of three line segments, $\overline{PQ}$, $\overline{QR}$, and $\overline{RS}$, and the arc's chord, $\overline{PS}$. Let d_1 and d_2 be, respectively, the distances from Q to $\overline{PS}$ and from R to $\overline{PS}$. Without loss of generality, assume all of $\overline{PQ}$, $\overline{RS}$, and $\overline{PS}$ are non-vertical, with slopes m_1,

Figure 1: Computing arc-to-chord deviation

m_2, and m_0, respectively. Then d_1 and d_2 can be computed via

$$d_1 = \begin{cases} |\overline{PQ}| \sin(|\arctan m_0 \ - \ \arctan m_1|) & \text{if the perpendicular from } Q \text{ to } \overleftrightarrow{PS} \text{ meets } \overline{PS}; \\ \min\{|PQ|,\ |QS|\} & \text{otherwise;} \end{cases}$$

$$d_2 = \begin{cases} |\overline{RS}| \sin(|\arctan m_0 \ - \ \arctan m_2| & \text{if the perpendicular from } R \text{ to } \overleftrightarrow{PS} \text{ meets } \overline{PS}; \\ \min\{|PR|,\ |RS|\} & \text{otherwise.} \end{cases}$$

The *maximum arc-to-chord deviation* between arc $\overline{PQ} \cup \overline{QR} \cup \overline{RS}$ and its chord $\overline{PS}$ is $\max\{d_1, d_2\}$.

A *local minimum deviation arc (LMDA)* is an arc A of 2 or 3 segments whose maximum arc-to-chord deviation is less than or equal to those of its *neighboring arcs*, where a neighboring arc of A is an arc $B \neq A$ of 2 or 3 segments such that $A \cap B$ is a segment or a union of 2 segments. In Example 3.1, we have neighboring arcs that are tied according to the criteria above, so we use the following tie-breakers. Suppose neighboring arcs A and B have equal maximum arc-to-chord deviations that are less than or equal to those of all their neighbors.

- If A has 2 segments and B has 3 segments, let A be an LMDA.
- If A and B have the same number of segments and A has a vertex that precedes all the vertices of B in the circular order of the input, then A is an LMDA.
- If A is an LMDA, then B is not an LMDA.

It follows from the second tie-breaker that (contrary to the assertion of [Leu and Chen]) the output is sensitive to which vertex is first in the circular order of input. (If all neighboring arcs of 2

or 3 segments have *distinct* maximum arc-to-chord deviations, then the assertion of [Leu and Chen] that the output is independent of the starting point, is correct.)

The first two tie-breaker rules are arbitrary, and we have constructed Example 3.1 with them in mind. Were we to use other tie-breaking rules, Example 3.1 could be modified accordingly to illustrate the undesirable behavior of the algorithm of [Leu and Chen].

2.2 Hausdorff Metric

The *Hausdorff metric* measures how well two geometric objects A and B approximate each other with respect to their positions in a metric space. Roughly, A and B are close in the Hausdorff metric if and only if neither has a point that is far away from a nearest point of the other.

In particular, let d be the Euclidean metric for E^2. Let $\varepsilon > 0$ and let A be a nonempty subset of E^2. The "ε neighborhood of A in E^2," denoted $N_d(\varepsilon, A)$, is defined to be

$$N_d(\varepsilon, A) = \{x \in E^2 \mid d(x, a) < \varepsilon \text{ for some } a \in A\}.$$

If $z \in E^2$, let

$$d(z, A) = \inf\{d(z, a) \mid a \in A\}.$$

Definition 2.1 [Nadler, 1978] Let A and B be nonempty, bounded, and closed subsets of E^2. The *Hausdorff distance* $d_H(A, B)$ is

$$d_H(A, B) = \inf\{\varepsilon > 0 \mid A \subset N_d(\varepsilon, B) \text{ and } B \subset N_d(\varepsilon, A)\}. \blacksquare$$

Alternately, we have the following.

Theorem 2.2 [Nadler] $d_H(A, B) = \max\{\sup_{a \in A} d(a, B), \sup_{b \in B} d(b, A)\}$. $\blacksquare$

The function d_H is a metric for nonempty, bounded, closed subsets of E^2 [Nadler].

3 Algorithms

3.1 The Algorithm of Leu and Chen

The algorithm of [Leu and Chen] is given below, somewhat restated. We assume the input includes the vertices $v_0, v_1, \ldots, v_{n-1}$ in circular order, and the tolerance value τ. If, during a given itera-

tion of the outer loop, the remaining vertices are, in circular order, $v_{j_0}, \ldots, v_{j_{N-1}}$, then arithmetic expressions in subscripts are understood to be *mod* N.

Let $N = n$;
IF $N > 6$ {N is the number of edges in the boundary} THEN
  REPEAT {assume the remaining vertices are, in circular order, $v_{j_0}, \ldots, v_{j_{N-1}}$}
    FOR $i := 0$ TO $N - 1$ DO
      Compute the maximum arc-to-chord deviations of the arcs
        $A_i = \overline{v_{j_i} v_{j_{i+1}}} \cup \overline{v_{j_{i+1}} v_{j_{i+2}}}$ and $B_i = A_i \cup \overline{v_{j_{i+2}} v_{j_{i+3}}}$;
    ENDFOR;
    FOR $i := 0$ TO $N - 1$ DO
      IFA_i or B_i is an LMDA whose maximum arc-to-chord deviation is less than τ,
        THEN mark the LMDA for removal at the end of the iteration
    ENDFOR;
    replace the LMDAs marked by their respective chords and
      decrement the value of N accordingly
  UNTIL $N \leq 6$ OR no LMDAs are removed
ENDIF {$N > 6$}
ELSE {$N \leq 6$} consider all arcs of 2 or 3 edges, removing one
  (if such exists) with smallest maximum arc-to-chord deviation $< \tau$

At first glance, the algorithm appears to require $O(n^2)$ time. However, we observe that only values of i corresponding to neighbors of arcs removed on a given iteration need be considered on the next iteration. The first iteration of the REPEAT loop requires $\Theta(n)$ time. Since $O(n)$ arcs are removed by a performance of the entire algorithm, it follows that all iterations of the REPEAT loop beyond the first require $O(n)$ time. Thus, the algorithm may be implemented in $\Theta(n)$ time.

3.2 An Example of Poor Performance

If P' is the output that results from applying a polygonal approximation algorithm to a polygon P using $\tau > 0$ as a bound for the tolerance of deviation in replacement of an arc by its chord, it is

desirable that

$$d_H(P, P') < \tau.$$

The following example shows that this goal is not achieved by the algorithm presented above. We say the remaining polygon P' is *determined* by a circularly ordered subset $\{v_0, v_1, \ldots, v_{m-1}\}$ of its vertices if all the vertices of P' are contained in $\bigcup_{i=0}^{m-1} \overline{v_i\ v_{(i+1)\ mod\ m}}$.

Example 3.1 *Suppose s is a positive integer, M is a positive constant, $0 < \tau < M/12$ (where τ is the error tolerance), and the input polygon P is determined by the following vertices.*

$$\begin{array}{lcl} L_k & = & (2\tau + (-1)^{(k+1)}2\tau, \frac{k\tau}{4}),\ k = 0, 1, \ldots, 2s; \\ R_k & = & (M + (-1)^k 2\tau, \frac{k\tau}{4}),\ k = 0, 1, \ldots, 2s; \\ T_L & = & (2\tau, s\tau); \\ M_L & = & (\frac{M+2\tau}{3}, \frac{3s\tau}{4}); \\ M_R & = & (\frac{2(M+2\tau)}{3}, \frac{3s\tau}{4}); \\ T_R & = & (M, s\tau), \end{array}$$

where a circular order of these vertices is

$$R_0, R_1, \ldots, R_{2s}, T_R, M_R, M_L, T_L, L_{2s}, L_{2s-1}, \ldots, L_0.$$

Suppose other vertices of P are as follows. Each of the line segments $\overline{L_{2i}L_{2i+1}}$ and $\overline{L_{2i+1}L_{2i+2}}$ has $2^{(2i+1)} - 1$ distinct vertices of P with $x-$coordinates satisfying $\tau < x < 3\tau$, $i = 0, \ldots, s-1$. Each of the line segments $\overline{R_{2i}R_{2i+1}}$ and $\overline{R_{2i+1}R_{2i+2}}$ has $2^{(2i)} - 1$ distinct vertices of P with $x-$coordinates satisfying $M - \tau < x < M + \tau$, $i = 0, \ldots, s-1$. Then the algorithm outputs a polygon P' with 5 or 6 vertices in the circular order

$$R_{2s}, T_R, M_L, T_L, L_{2s},$$

(respectively,

$$R_{2s}, T_R, M_R, M_L, T_L, L_{2s}),$$

and $d_H(P, P') = \Theta(\tau \log n)$, where n is the number of vertices of P. See Figure 2.

Proof (Sketch): It may happen that on the first iteration, the vertex M_R is removed. This depends on the relation among M, s, and τ. In Figure 2, we do not show M_R removed. In the following, we assume without loss of generality that M_R is not removed.

The behavior of the algorithm on this example is as follows. On odd-numbered iterations, the bottom of the remaining polygon P' is shifted upward on the right side. On even-numbered iterations, the bottom of P' is shifted upward on the left side.

We proceed by induction on even values of j, the number of iterations of the REPEAT loop. For $j = 0$, the following hold. The remaining polygon is determined by the subset of its vertices, in circular order,

$$\{R_j, R_{j+1}, \ldots, R_{2s}, T_R, M_R, M_L, T_L, L_{2s}, L_{2s-1}, \ldots, L_j\}.$$

Each of the line segments $\overline{L_{2i}\ L_{2i+1}}$ and $\overline{L_{2i+1}\ L_{2i+2}}$ has $2^{(2i+1-j)} - 1$ distinct vertices of P with $x-$coordinates satisfying $\tau < x < 3\tau$, $i = j/2, \ldots, s-1$. Each of the line segments $\overline{R_{2i}\ R_{2i+1}}$ and $\overline{R_{2i+1}\ R_{2i+2}}$ has $2^{(2i-j)} - 1$ distinct vertices of P with $x-$coordinates satisfying $M - \tau < x < M + \tau$, $i = j/2, \ldots, s-1$. In particular, the line segments $\overline{R_j\ R_{j+1}}$ and $\overline{R_{j+1}\ R_{j+2}}$ have no remaining vertices of P except for their endpoints.

Suppose the above hold after j iterations, for some even j such that $0 \leq j < 2s - 1$. On the $(j+1)^{th}$ iteration, the following occur. The segments $\overline{L_{2i}\ L_{2i+1}}$ and $\overline{L_{2i+1}\ L_{2i+2}}$ each mark $2^{(2i-j)}$ LMDAs of 2 segments each (each such LMDA has maximum arc-to-chord deviation of 0), leaving each of $\overline{L_{2i}\ L_{2i+1}}$ and $\overline{L_{2i+1}\ L_{2i+2}}$ with $2^{(2i-j)} - 1$ vertices of P that have $x-$coordinates satisfying $\tau < x < 3\tau$, $i = j/2 \ldots, s-1$. Similarly, the segments $\overline{R_{2i}\ R_{2i+1}}$ and $\overline{R_{2i+1}\ R_{2i+2}}$ each mark $2^{(2i-j-1)}$ LMDAs of 2 segments each (each such LMDA has maximum arc-to-chord deviation of 0), leaving each of $\overline{R_{2i}\ R_{2i+1}}$ and $\overline{R_{2i+1}\ R_{2i+2}}$ with $2^{(2i-j-1)} - 1$ vertices of P that have $x-$coordinates satisfying $M - \tau < x < M + \tau$, $i = \frac{j}{2} + 1, \ldots, s-1$. Also, the arc

$$\overline{L_j\ R_j} \cup \overline{R_j\ R_{j+1}} \cup \overline{R_{j+1}\ R_{j+2}}$$

is marked as an LMDA and replaced by its chord. Thus, at the end of this iteration, the remaining polygon is determined by the circularly ordered subset of its vertices

$$\{R_{j+2}, \ldots, R_{2s}, T_R, M_R, M_L, T_L, L_{2s}, L_{2s-1}, \ldots, L_j\}.$$

On the $(j+2)^{th}$ iteration, the following occur. The segments $\overline{L_{2i}\ L_{2i+1}}$ and $\overline{L_{2i+1}\ L_{2i+2}}$ each mark $2^{(2i-j-1)}$ LMDAs of 2 segments each (each such LMDA has maximum arc-to-chord deviation of 0), leaving each of $\overline{L_{2i}\ L_{2i+1}}$ and $\overline{L_{2i+1}\ L_{2i+2}}$ with $2^{(2i-j-1)} - 1$ vertices of P that have $x-$coordinates

satisfying $\tau < x < 3\tau$, $i = \frac{j}{2} + 1 \ldots, s-1$. Similarly, the segments $\overline{R_{2i}\ R_{2i+1}}$ and $\overline{R_{2i+1}\ R_{2i+2}}$ each mark $2^{(2i-j-2)}$ LMDAs of 2 segments each (each such LMDA has maximum arc-to-chord deviation of 0), leaving each of $\overline{R_{2i}R_{2i+1}}$ and $\overline{R_{2i+1}R_{2i+2}}$ with $2^{(2i-j-2)}-1$ vertices of P that have $x-$coordinates satisfying $M - \tau < x < M + \tau$, $i = \frac{j}{2} + 1, \ldots, s-1$. Also, the arc

$$\overline{R_{j+2}\ L_j} \cup \overline{L_j\ L_{j+1}} \cup \overline{L_{j+1}\ L_{j+2}}$$

is marked as an LMDA and replaced by its chord. Thus, at the end of this iteration, the remaining polygon is determined by the circularly ordered subset of its vertices

$$\{R_{j+2}, \ldots, R_{2s}, T_R, M_R, M_L, T_L, L_{2s}, L_{2s-1}, \ldots, L_{j+2}\}.$$

This concludes the induction.

After $2s$ iterations, the remaining vertices of P', in circular order, are $R_{2s}, T_R, M_R, M_L, T_L, L_{2s}$, and

$$d_H(P, P') \ = \ d_H(\overline{L_0\ R_0}, \overline{L_{2s}\ R_{2s}}) \ = \ \frac{s\tau}{2}.$$

It is easily seen that $s = \Theta(\log n)$, so $d_H(P, P') \ = \ \Theta(\tau \log n)$. ■

A cruder measure of the closeness of P and P' is a comparison of their diameters. It is desirable that P and P' have approximately equal diameters. However, in the previous example, given any $\varepsilon > 0$, we can make $\frac{diam(P')}{diam(P)} < \frac{1}{2} \ + \ \varepsilon$ by choosing s so that $s\tau$ is sufficiently large compared to M. Similar observations can be made about a comparison of the areas of P and P'.

In the example above, the use of a polygon determined by a proper subset of its vertices was a notational convenience. We could have replaced the vertices interior to the arcs $\overline{R_i\ R_{i+1}}$ and $\overline{L_i\ L_{i+1}}$ by vertices slightly off these segments such that no 3 consecutive vertices were collinear, and still have obtained an example in which $d_H(P, P') = \Theta(\tau \log n)$.

3.3 Our Algorithm

The example discussed above shows that the algorithm of [Leu and Chen] can produce an output polygon P' such that $d_H(P, P')$ is unacceptably large. In the following, we describe a modification that avoids this problem. In the algorithm of [Leu and Chen], each LMDA is checked only for the

distance(s) of its internal vertex (vertices) to its chord. Suppose the polygonal arc $\overline{PQ} \cup \overline{QR}$ is an LMDA (a similar discussion applies to arcs of the form $\overline{PQ} \cup \overline{QR} \cup \overline{RS}$), where

$$P = v_i,\ Q = v_j,\ R = v_k,\ |j - i| > 1,\ |k - j| > 1,$$

and previous iterations of the REPEAT loop have already removed the vertices

$$V' = \{v_{i+1}, \ldots, v_{j-1}, v_{j+1}, \ldots, v_{k-1}\}.$$

By considering the distances of the members of V', as well as that of Q, from the chord $\overline{PR}$ in determining LMDAs, we make sure that the output polygon P' satisfies $d_H(P, P') < \tau$.

We feel there is another way in which the decision of which LMDAs should be replaced by their chords could be improved. The union of an LMDA and its chord bounds a region R that is a triangle, or the union of 2 triangles, or a quadrilateral. If R is too large (in area) relative to the entire image I containing P, the LMDA should not be replaced by its chord. There are several reasonable ways to determine what is "too large." We have chosen to use the following. We assume there is another input value α, a positive number representing a relative area tolerance. Let $A(X)$ denote the area of X. Let B_R be a minimal-area rectangle containing R. We say R is $\alpha - large$ if

$$\frac{A(B_R)}{A(I)} \geq \alpha.$$

Notice that whether or not R is $\alpha-$large can be determined, for a given LMDA, in $\Theta(1)$ time.

With these considerations, we give the following algorithm. If, during a given iteration of the outer loop, the remaining vertices are, in circular order, $v_{j_0}, \ldots, v_{j_{N-1}}$, then arithmetic expressions in sub-subscripts (*e.g.*, $i + 1$ in $v_{j_{i+1}}$) are understood to be *mod* N, while arithmetic expressions in first order subscripts (*e.g.*, $j + 1$ in v_{j+1}) are understood to be *mod* n.

Let $N = n$;
IF $N > 6$ {N is the number of edges in the boundary} THEN
  REPEAT {assume the remaining vertices are, in circular order, $v_{j_0}, \ldots, v_{j_{N-1}}$}
    IF this is the first iteration THEN FOR $i := 0$ TO $N - 1$
    ELSE FOR all i such that there is a vertex v_q removed during the previous
      iteration such that v_q is between (in the circular order of P)

v_{j_i} and $v_{j_{i+1}}$, or $v_{j_{i+1}}$ and $v_{j_{i+2}}$, or $v_{j_{i+2}}$ and $v_{j_{i+3}}$ {1}

DO

Compute the maximum arc-to-chord deviations of the arcs

$A_i = \overline{v_{j_i} v_{j_{i+1}}} \cup \overline{v_{j_{i+1}} v_{j_{i+2}}}$ and $B_i = A_i \cup \overline{v_{j_{i+2}} v_{j_{i+3}}}$;

ENDFOR;

IF this is the first iteration THEN FOR $i := 0$ TO $N - 1$

ELSE FOR all i such that there is a vertex v_q removed during the previous

iteration such that v_q is between (in the circular order of P)

v_{j_i} and $v_{j_{i+1}}$, or $v_{j_{i+1}}$ and $v_{j_{i+2}}$, or $v_{j_{i+2}}$ and $v_{j_{i+3}}$ {2}

DO

IF A_i or B_i is an LMDA of maximum arc-to-chord deviation less than τ

AND whose chord $\overline{v_{j_i} v_{j_{i+2}}}$ (respectively, $\overline{v_{j_i} v_{j_{i+3}}}$) is less than τ

from each member of $\{v_{j_i}, v_{j_i+1}, \ldots, v_{j_{i+2}}\}$ (respectively, $\{v_{j_i}, v_{j_i+1}, \ldots, v_{j_{i+3}}\}$)

AND such that the region bounded by the LMDA and its chord is not $\alpha - large$

THEN mark the LMDA for removal at the end of the iteration

ENDFOR;

replace the LMDAs marked by their respective chords;

make a list of the indices of vertices removed on this iteration; {3}

decrease N by the number of vertices removed on this iteration

UNTIL $N \leq 6$ OR no LMDAs are removed

ENDIF $\{N > 6\}$

ELSE $\{N \leq 6\}$ consider all arcs of 2 or 3 edges, removing one

(if such exists) with smallest maximum arc-to-chord deviation $< \tau$

such that the region bounded by the LMDA and its chord is not $\alpha-$large

We note that the (remaining) vertices are considered in circular order, so that the list constructed at line {3} is ordered without need for a sort step. It follows that identifying the next value of i such that there is a v_q satisfying the condition of {1}, {2} may be done in $\Theta(1)$ time. Hence it is easily seen that this algorithm runs in $O(n + r^2)$ time, where r is the number of vertices removed from P to obtain the output. Since $r = O(n)$, this gives a running time of $O(n^2)$. However, by

comparing an LMDA A with vertices v_q removed in previous iterations whose order in P is between vertices of A, we often find that fewer iterations are necessary than for the $\Theta(n)$ time algorithm of [Leu and Chen], so that on a large class of examples, our algorithm is the faster one.

4 Experimental Results

We have implemented both of the algorithms discussed in this paper on a Sun Workstation, using the C language. We give several figures, starting with Figure 2, that compare the results of the two algorithms on the same inputs. We feel that for several of the examples, our algorithm produces significantly better output. The figures show output edges as darker line segments than those input edges not coinciding with output edges. Subfigures labeled "Original" show the output of the algorithm of [Leu and Chen]; those labeled "Improved" show the output of our algorithm. Running times are measure in CPU seconds. "Improved" figures in which we show $\alpha = 0$ are examples in which we did not use a relative area tolerance.

Figure 2 compares the two algorithms on an instance of Example 3.1. We observe that on *all* instances of Example 3.1 in which $s \geq 2$, our algorithm leaves P' with a lowest edge of $\overline{L_4\ R_4}$, while the algorithm of [Leu and Chen] leaves P' with a lowest edge of $\overline{L_{2s}\ R_{2s}}$.

Figure 3 compares the two algorithms on an almost-circular polygonal input. We see that our algorithm gives output that better approximates the input with respect to d_H, the more so when we use a relative area tolerance.

On the irregular input of Figure 4, the two algorithms yield the same output when a relative area tolerance is not used in our algorithm. When we use a relative area tolerance, our algorithm produces output that is slightly better in approximating the input with respect to d_H.

5 Further Remarks

We have given an algorithm for smoothing a polygon by removing interior vertices of arcs that differ from their chords by less than a small pre-specified tolerance. Our algorithm improves on that of [Leu and Chen] in the sense of keeping the Hausdorff distance between input and output small and in the sense of not making large changes in area. The algorithm of [Leu and Chen] can

be implemented in optimal $\Theta(n)$ time, while ours requires $O(n + r^2)$ time, where r is the number of input vertices removed by the algorithm. However, in practice, our algorithm often runs faster than that of [Leu and Chen].

References

[Leu and Chen] Leu, J-G., and L. Chen (1988). Polygonal approximation of 2-D shapes through boundary merging, *Pattern Recognition Letters* 7, 231-238.

[Nadler] Nadler, S.B., Jr. (1978). *Hyperspaces of Sets*, Marcel Dekker, Inc., New York.

Figure 2: The example of Section 3.2, using $s = 5,\ M = 250,\ n = 2052$

Figure 3: Comparison on almost-circular input

Figure 4: Comparison on irregular input