ISSN 0105-8517

A simple algorithm for computing the smallest enclosing circle

Sven Skyum

DAIMI PB – 314
June 1990

COMPUTER SCIENCE DEPARTMENT
AARHUS UNIVERSITY
Ny Munkegade, Building 540
DK-8000 Aarhus C, Denmark

A simple algorithm for computing the smallest enclosing circle[1]

Sven Skyum

May 22, 1990

Abstract

We present a simple iterative algorithm for computing the smallest enclosing circle and the farthest-point Voronoi diagram of a pointset and the ordinary Voronoi diagram of a convex polygon. The algorithm(s) takes $O(n \log n)$ time for n points. This is not optimal for any of the problems, but the simplicity of the algorithm(s) makes it a better alternative for medium sized problems than earlier published methods.

1 Introduction

Suppose we are given n points $S = \{p_1, p_2, \ldots, p_n\}$ in the Euclidian plane R^2. The smallest enclosing circle of S, $SEC(S)$, is the circle with minimal radius enclosing all points in S. It is trivial and well-known that $SEC(S) = SEC(H)$, where $H \subseteq S$ are the extreme points of the convex hull of S.

In the next section we present the algorithm for computing $SEC(S)$. The algorithm is closely related to construction of the farthest-point Voronoi diagram and if S are points forming the vertices of a convex polygon to the construction of the ordinary Voronoi diagram too. The construction of Voronoi diagrams is presented in Section 3. The algorithms take $O(n \log n)$ time, are very easy to implement, and numerically sound. Migiddo ([3]) has given linear time algorithms for linear programming in R^3 which applies to the enclosing circle problem. Aggraval, Guibas, Saxe and Shor ([1]) recently gave linear algorithms for computing the Voronoi-diagrams of points when these form the vertices of a convex

[1]This research was (partially) supported by the ESPRIT II Basic Research Actions Program of the EC under contract No. 3075 (project ALCOM).

polygon. Both algorithms are recursive algorithms and the involved constants hidden in $O(n)$ are large.

2 The Algorithm

Assume we are given n points $S = \{p_1, p_2, \ldots, p_n\}$ in R^2, where S forms the vertices of a convex polygon. More specifically the points are stored in a double linked list such that next(p_i) (before(p_i)) is the clockwise (anticlockwise) neighbour of p_i on the polygon. In the sequel we will just say that S is a convex set of points.

radius(p, q, r) denotes the radius of the circle through the three points p, q and r if they are different. If two points are identical, then it denotes half the distance between one of those and the third one. angle(p, q, r) denotes the angle between the line segments from p to q and q to r. It will always be the case that $p \neq q$ and $q \neq r$, but not necessarily the case that $p \neq r$.

Algorithm 1

```
if |S| ≠ 1 then
  finish := false;
  repeat
  (1) find p in S maximizing
        (radius(before(p), p, next(p)), angle(before(p), p, next(p)))
      in the lexicographic order;
  (2) if angle(before(p), p, next(p)) ≤ π/2 then
        finish := true
      else
        remove p from S
      fi
  until finish
fi;
```

The algorithm will terminate since either the size of S is 1 to start with or the size of S will decrease at most until it has size 2 in which case the involved angle is 0. In fact, it will decrease to size 2, 3 or 4.

Upon termination, the last chosen p (possibly the only point in S to start with) will have the property that $SEC(\text{before}(p),\ p,\ \text{next}(p)) = SEC(S_0)$, where S_0 is the original pointset. This follows from the following Observations and Lemma.

The first two Observations are proven by standard geometrical arguments and not included here.

The line segment from a point p to q is denoted by $\overline{pq}$ and t is said to be to the right (left) of $\overline{pq}$ if the points p, q and t form a right (left) turn.

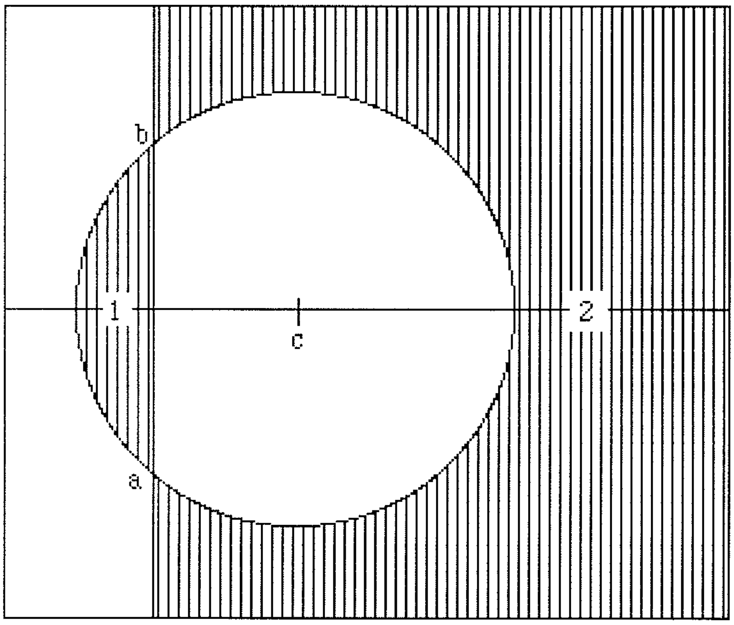

Figure 1: Observation 1

Observation 1

If a and b are points in R^2, C a circle through a and b, with radius r and centre c to the right of $\overline{ab}$ then $r < \text{radius}(a, b, p)$ for a point p inside C to the left of $\overline{ab}$ (area 1 on Figure 1) or outside C to the right of $\overline{ab}$ (area 2 on Figure 1).

Observation 2

If a, b and c are three points in R^2 and $\mathcal{C}$ a circle with radius less than radius(a, b, c) that encloses a and c, then $\mathcal{C}$ encloses b if and only if angle$(a, b, c) \geq \pi/2$.

Lemma 1

Let S be the vertices of a convex polygon in R^2. If (a, b, c) maximizes (radius(a, b, c), angle(a, b, c)) in the lexicographic order, then

i) a, b and c are consecutive vertices on the polygon.

ii) circle(a, b, c) encloses all points in S.

Proof

Case 1: angle$(a, b, c) \leq \pi/2$.
All angles in the triangle with vertices a, b and c are less than or equal to $\pi/2$, since angle(a, b, c) is the larger of the three. Since radius(a, b, c) is maximal, Observation 1 applied to $\{a, b\}$ implies that no point in S can be in areas numbered 3, 4 or 6 on Figure 2. Applied to $\{b, c\}$ and $\{a, c\}$ it follows that no point in S can be in areas numbered 2, 4, 5 or 1, 5, 6. Since S is a convex set of points, all points of S must be on the circle through a, b and c so circle(a, b, c) encloses S. That a, b and c are consecutive is then an implication of angle(a, b, c) being maximal among all occurring angles. Note that S can only contain one more point than a, b and c and that the points then form the vertices of a square.

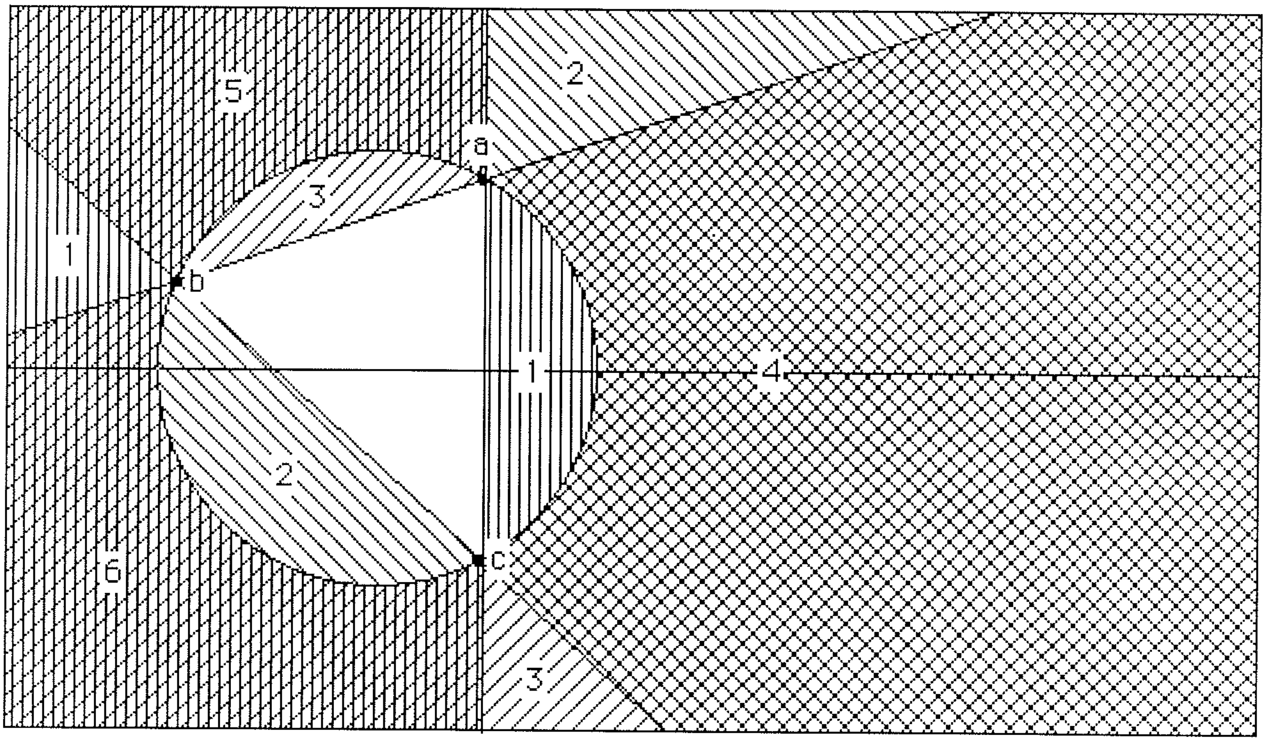

Figure 2: Lemma 1, case 1

Case 2: angle$(a, b, c) > \pi/2$.
Applying Observation 1 again to $\{a, b\}$, $\{b, c\}$, and $\{a, c\}$ ensures that no point in S can be in area 1 on Figure 3. A point p from S cannot be in area 2 because then b would be a convex combination of a, p and c violating S being a convex set of points. The maximality of angle(a, b, c) ensures once again that a, b and c are consecutive. If p is in $S - \{a, b, c\}$ then p must be situated in the unhatched area and statement (ii) of the Lemma follows. $\square$

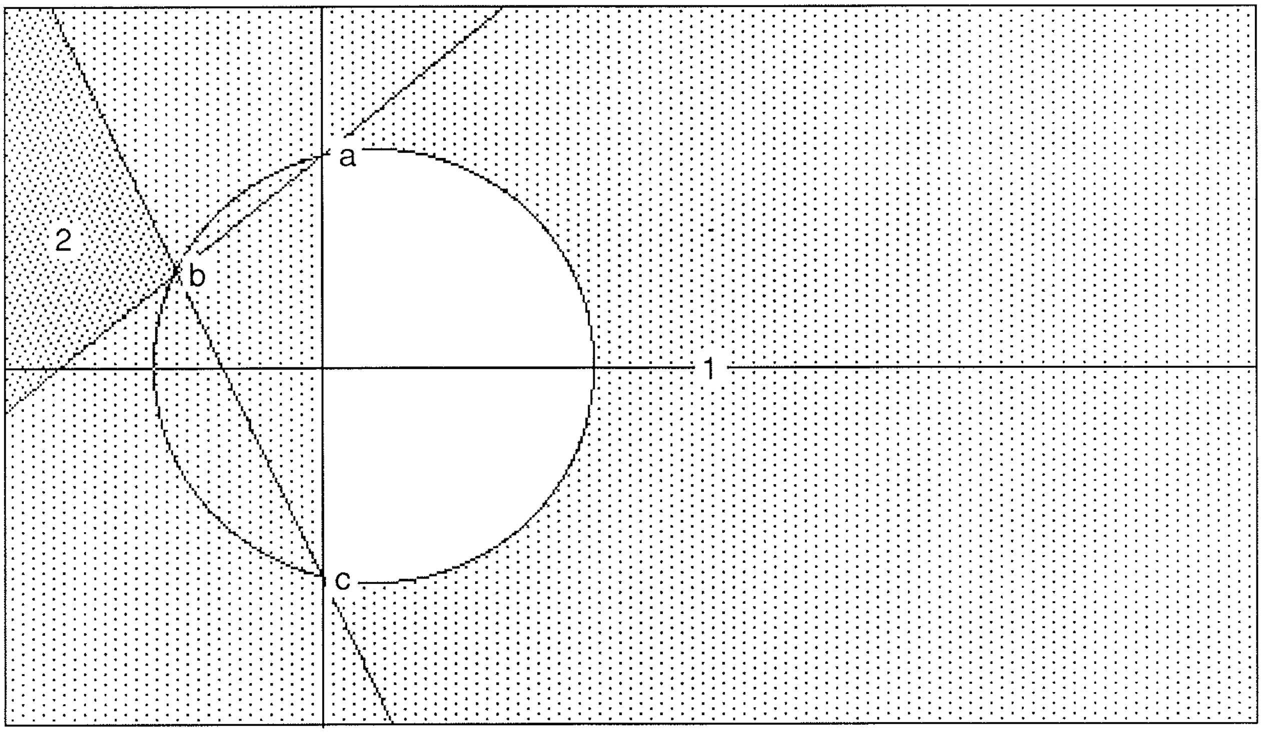

Figure 3: Lemma 1, case 2

The correctness of Algorithm 1 now follows. Observation 2 and Lemma 1 imply that if the "else" part of statement (2) is executed, then $SEC(S) = SEC(S - \{p\})$ and in the case of the "then" part being executed no circle with radius less than radius(before(p), p, next(p)) can contain before(p), p, and next(p), so p and its neighbours determine $SEC(S)$ which in turn is the smallest enclosing circle of the original given pointset.

Algorithm 1 can easily be implemented to run in time $O(n \log n)$. By removal of a point from S we only have to recompute the radii and angles for the old neighbours which can be done in constant time. Note that the new radii are not less than the old ones. Several datastructures support the actual deletions and insertions involved in statements (1) and (2) in overall time $O(n \log n)$.

Remarks

(1) If we a priori know that the radius of $SEC(S)$ is bounded above by R, we may successively remove points from S where radius(before(p), p, next(p))$> R$ without testing for maximality.

(2) If S does not form the vertices of a convex polygon to start with, Graham's scan (see [2] or [4]) can be incorporated naturally in Algorithm 1 by letting radius(a, b, c) be infinite if c is to the left of $\overline{ab}$.

(3) Remark (1) and (2) implies that by altering Algorithm 1 as indicated in (1) the existence (and a possible construction) of an enclosing circle with given radius R is tested (constructed) in linear time for a star shaped polygon.

3 Construction of Voronoi diagrams

In this section we demonstrate that with a simple extension, basically the same algorithm as the one presented in the previous section can be used to construct the farthest-point Voronoi diagram of a convex pointset S.

Let centre(a, b, c) for three non colinear points in R^3 denote the centre of the circle through a, b and c.

We will treat the farthest-point Voronoi diagram of S, denoted by $V_{-1}(S)$, as a graph (K, E) where the degree of the Voronoi-vertices K are either 1 or 3. If v has degree 1 it is a vertex "at infinity" on a bisector of two neighbour points in S (an "endpoint" of the half infinite line segments of the diagram). If v has degree 3, it is centre(a, b, c) of three points in S and no points in S are farther away from centre(a, b, c) than a, b and c.

If (v_1, v_2) is a Voronoi-edge from E, then for some points a and b in S, the line segment $\overline{v_1 v_2}$ is contained in the bisector of a and b and no points in S are farther away from points on $\overline{v_1 v_2}$ than a and b.

Note that if no four points in S are cocircular then $V_{-1}(S)$ is unique. Otherwise the distance between v_1 and v_2 for some edges (v_1, v_2) in E might be 0.

In Algorithm 2 to follow $v(p)$ will be a point on the bisector of p and next(p). On removal of p, $v(p)$ will be a vertex of $V_{-1}(S)$.

Initially $v(p)$ is a point on the bisector of p and next(p) "at infinity" to the right of $\overline{p\,\text{next}(p)}$.

Algorithm 2

```
for all p in S add v(p) to K;
if n > 2 then
   repeat
      find p maximizing
         (radius(before(p), p, next(p)), angle(before(p), p, next(p));
      q := before(p);
      c := centre(q, p, next(p));
      add c to K;
      add (c, v(p)) and (c, v(q)) to E;
      v(q) := c;
      next(q) := next(p);
      before(next(q)) := q;
      n := n - 1;
   until n = 2;
   add (v(q), v(next(q)) to E
else
   if n = 2 then {S = {p1, p2}}
      add (v(p1), v(p2)) to E
   fi
fi;
```

Lemma 1 from Section 2 ensures that when p is chosen the circle(before(p), p, next(p)) with centre $c =$ centre(before(p), p, next(p)) encloses all points of S. Thus c is a Voronoi-vertex and $(c, v(p))$ as well as $(c, v(\text{before}(p))$ are Voronoi-edges. That all Voronoi-vertices and edges are found follows by recognizing, that if $n > 1$, the number of vertices of degree 3 for Voronoi-diagrams is $n - 2$ and the number of edges is $2n - 3$ matching the number of vertices and edges created by Algorithm 2.

To construct the ordinary Voronoi-diagram $V(S)$, where vertices are points of minimal equal distance to three points in S instead of maximal distance and equivalently edges determined by minimal distance to pair of points, it suffices to alter Algorithm 2 by adding a minus before radius in line 5, that is to choose p such that the corresponding radius is minimal and among those the p maximizing the angle. In addition $v(p)$ must initially be a point on the bisector of p and next(p) "at infinity" to the left of $\overline{p\,\mathrm{next}(p)}$.

The correctness of the construction is a consequence of the following Lemma 2 which is an analog of Observation 2 and Lemma 1. The proof is similar and not included here.

Lemma 2

Let S be the vertices of a convex polygon in R^2. If (a, b, c) maximizes $(-\mathrm{radius}(a, b, c), \mathrm{angle}(a, b, c))$ in lexicographic order, then

i) a, b and c are consecutive vertices on the polygon.

ii) No point from S is inside $\mathrm{circle}(a, b, c)$.

iii) If b is inside $\mathrm{circle}(a', b', c')$ for three points a', b' and c' from S then either a or c is inside too.

References

[1] Aggraval A., Guibas L.J., Saxe J., Shor P.W.: A Linear-Time Algorithm for Computing the Voronoi Diagram of a convex Polygon. *Discrete Comput Geom 4; 491-604* (1989).

[2] Graham R.L.: An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Info. Proc. Lett. 1; 132-133* (1972).

[3] Megiddo N.: Linear-Time Algorithms for Linear Programming in R^3 and Related Problems. *Siam J. Comput. 12; 759-776* (1983).

[4] Preparata F.P., Shamos M.I.: *Computational Geometry – An Introduction.* Springer-Verlag 1985.