Computing Largest Circles Separating Two Sets of Segments

Jean-Daniel Boissonnat* Jurek Czyzowicz‡ Olivier Devillers* Jorge Urrutia§
Mariette Yvinec¶

Abstract: A circle C separates two planar sets if it encloses one of the sets and its open interior disk does not meet the other set. A separating circle is a largest one if it cannot be locally increased while still separating the two given sets. An $\Theta(n \log n)$ optimal algorithm is proposed to find all largest circles separating two given sets of line segments when line segments are allowed to meet only at their endpoints. This settles an open problem from a previous paper.[3] In the general case, when line segments may intersect $\Omega(n^2)$ times, our algorithm can be adapted to work in $O(n\alpha(n) \log n)$ time and $O(n\alpha(n))$ space, where $\alpha(n)$ represents the extremely slowly growing inverse of Ackermann function.
Keywords: Computational geometry, circles, separability.

1 Introduction

Let $\mathcal{C}$ denote a family of Jordan curves in the plane. Two sets P and Q in the plane are $\mathcal{C}$-separable if there exists $\xi \in \mathcal{C}$, such that every point of one of these sets lies in the closed region inside ξ, and every point of the other set lies in the closed region outside ξ. In this paper we restrict our consideration to elements of $\mathcal{C}$ being circles. A circle $C(X, r)$, with center X and radius r, separating P from Q is said to be a largest separating circle if there is a neighbourhood B of X such that there is no separating circle with radius strictly greater than r centered in B. We propose an optimal algorithm to find all largest circles separating two given sets of line segments P and Q.

Some previous research on this subject concerned polygonal separability[10, 1, 19] or its extension to higher dimensions, where the construction of a polyhedron with small number of faces, separating two given polyhedra was considered.[6, 18, 4] Line or hyperplane separability of two given sets of points may be solved using linear programming.[17]

The problem of circular separability was first asked in the context of applications in pattern recognition and image processing - recognition of digital disks.[12, 11] Kim and Anderson[12] gave a quadratic algorithm to determine the circular separability of two finite sets of points. Bhattacharya[2] computed in $O(n \log n)$ time the planar region at which may be centered all circles separating two given point sets. O'Rourke, Kosaraju and Megiddo[20] presented optimal algorithms for the circular separability of point sets. The first one determine the circular separability of two given point sets and find the smallest circle separating circle in linear time and the second one find all the largest separating circles in $O(n \log n)$ time. These authors have used the paraboloid transformation which convert the smallest separating circle problem for two points sets into a convex quadratic minimization problem in three dimensions. This method generalizes to spherical separability in higher dimensions, however it does not apply to the problem of circular separability of two simple polygons. In this case, the method leads to the computation of the convex hull of the segments of parabola in 3D which is an unsolved problem. This problem has been already considered[3] and a linear algorithm to find the smallest separating circle of two polygons has been proposed.

In the present paper we consider the problem of finding all largest circles separating two given sets of line segments. We suppose that different line segments may meet only at their endpoints. An $O(n \log n)$ algorithm is given to solve this problem. This settles an open question from previous paper,[3] asking for the largest circle separating two simple polygons. As our algorithm works in the case when segments degenerate to single points, it may be considered as a generalization of a result from O'Rourke, Kosaraju, and Megiddo result.[20]

Our algorithm can be adapted to work in the general case when line segments may intersect. In this case, it works in $O(n\alpha(n) \log^2 n)$ deterministic time or in $O(n\alpha(n) \log n)$ randomized time and $O(n\alpha(n))$ space, where $\alpha(n)$ is the extremely slowly growing inverse of Ackermann function.

2 Preliminaries

We will refer to the *hierarchical representation* of convex polygons introduced by Dobkin and Kirkpatrick.[14, 8] Originally, such a representation was introduced in the context of the point location problem, in which the construction of a hierarchy of planar maps permitted point location queries in optimal $O(n \log\ n)$ time.

*INRIA, B.P.93, 06902 Sophia-Antipolis cedex (France). First-name.Name@ sophia.inria.fr
†Dept. Informatique, Univ. du Québec à Hull (Canada). czyzowicz@ uqah.uquebec.ca
‡Research supported by NSERC.
§Dept. of Computer Science, University of Ottawa(Canada). jorge@ csi.uottawa.ca
¶I3S, URA CNRS 1376, Sophia Antipolis (France).

Hereafter, we use outer hierarchical representations. An outer hierarchical representation of a convex polyhedron D is a nested sequence $D_0 \supset D_1 \supset \ldots \supset D_k$ of convex polyhedra, such that

1. D_0 is a tetrahedron,

2. D_k is the polyhedron D,

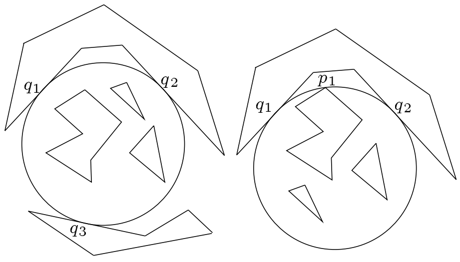
3. the set F_i of faces of D_i is obtained from F_{i+1} by removing a subset I_{i+1} of pairwise non adjacent faces of D_{i+1}. Extending the remaining faces $F_{i+1} \setminus I_{i+1}$ will then form polyhedron D_i.

It may be proved that in any convex polyhedron D_{i+1} it is possible to find a constant fraction of its faces of bounded degree which are pairwise non adjacent. As a consequence, the hierarchy of a convex polyhedron D with n vertices has a depth $k = O(\log n)$. The whole hierarchical representation takes $O(n)$ space and it may be computed in $O(n \log\ n)$ time. After computing the hierarchical representation of a convex polyhedron, line intersection queries may be performed in $O(\log\ n)$ time.

The paper will use a well-known transformation Φ, mapping circles in the xy-plane to points in the three-dimensional space which we will call the *space of circles*. According to this transformation, the image of a circle of radius r, centered at (x_0, y_0), is the point (x_0, y_0, r). Observe that images of the circles passing through a point (x_1, y_1) lie on the surface of a cone originating at $(x_1, y_1, 0)$ with vertical axis and 45 degrees apex angle. Such a cone will be called a *lifting cone* and denoted by $LC(x_1, y_1)$. Observe as well, that the image of a circle tangent to a given line l lie on a halfplane containing l and having 45 degree angle with xy-plane. There are two such *lifting halfplanes*, $LH^-(l)$ and $LH^+(l)$, denoting the images of the circles tangent to l and centered, respectively, on the left- or the right-hand side of the oriented line l. The space of circles in fact is a halfspace, as only points with non-negative z-coordinate belong to it.

Let S denote the set of line segments (sites) $s_1, s_2, \ldots, s_m$ in the plane. The closest site Voronoi diagram of S, noted $CSVor(S)$, is the partition of the plane into m regions, such that any point belonging to the i-th region is closer to s_i than to any other segment of S. Suppose that we want to decide whether a query disk contains any point of a given set S. Such a query may be answered quickly if the closest site Voronoi diagram of the set S has been precomputed. The center of the query disk is localised in a Voronoi cell determining the closest segment s_i of S. The radius of the disk is then compared to the distance from its center to s_i.

Similarly, in order to decide whether a query disk contains entirely a given set of line segments S, we will precompute the furthest site Voronoi diagram of S, noted $FSVor(S)$, which is just the furthest site Voronoi diagram of the vertices of the convex hull, $CH(S)$, of S.



Case 1 Case 2

Figure 1: For Lemma 1

For the purpose of the paper it is useful to introduce the following three-dimensional structure, which contains all the information included in the furthest site Voronoi diagram $FSVor(S)$. For each vertex v of $CH(S)$, take a cone originating at v having vertical axis and 45 degrees apex angle. Let $UE(S)$ denote the upper envelope of all such cones. A point of $UE(S)$ corresponds, in terms of transformation Φ, to a tangent enclosing circle for S that is a circle which encloses S but touches S in some points. The portions of cones belonging to $UE(S)$ are limited by segments of hyperbola, where two neighbouring cones meet. Observe that the projection of these segments of hyperbola onto the xy-plane forms the furthest site Voronoi diagram $FSVor(S)$. All of its cells are convex, unbounded polygonal regions.

3 Largest Separating Circles.

In the sequel, a segment is said to lie inside (resp. outside) a given circle C if it is included in the closed region which is inside (resp. outside) C; such a segment and the circle C are allowed to be tangent which mean that they meet in a single point. Two sets of segments P and Q are said to be in general position if they do not admit parallel segments and if there is no circle tangent to four segments of $P \cup Q$.

A largest circle C separating two given sets of segments P and Q must be constrained in at least three points by the elements of P and Q. Suppose that P lies inside and Q lies outside circle C. When the two sets of segments are in general position, it is easy to see that C either is tangent in three points to the elements of Q, or is tangent in two points to the elements of Q and meets a vertex of the convex hull $CH(P)$. We can conclude that a largest separating circle must verify one of the two conditions given by the following lemma. The proof of this lemma is straightforward.

Lemma 1 *If C is a largest circle separating two given sets of segments P and Q in general position, P lying inside and Q lying outside C, then one of the following two conditions must be verified (see Figure 1) :*

1. *C is tangent to three segments of Q in points q_1, q_2 and q_3 such that each of the three arcs determined*

on C by these points is smaller that a semicircle (see Fig. 1, case 1).

2. *C is tangent to two segments of Q in points q_1 and q_2, and meet the convex hull $CH(P)$ in a vertex p_1, , such that the arc of C extending from q_1 through p_1 to q_2 is smaller that a semicircle, (see Fig 1, case 2).*

If Q admits parallel segments, a largest separating circle may be tangent to two parallel segments of Q without meeting P or touching a third segment in Q (see Fig. 2, case $1'$ or $2'$). In such a case there is an infinite number of circles tangents to those two segments of Q which are largest separating circles according to the above definition. However all those circles are deduced by translation from two extremes circles which,in addition to the two contact points with parallel segments of Q, have a third contact point with either set P or Q. Our algorithm restrict to report only the largest separating circles that have at least three contact points. When point sets P and Q are not in general situation, those circles may be in one of the degenarate case listed in the lemma below.

Lemma 1 (bis) *Let C be a largest circle separating two sets of segments P and Q, such that P lies inside, Q lies outside and C has at least three contact points with sets P and Q. Then, in addition to cases 1 and 2 of lemma 1 above, C may be in one of the following degenerate cases (see Fig. 2) :*

$1'$. *C is tangent to two parallel segments of Q (in two antipodal points) and to a third segment in Q.*

$1''$. *C touches Q in two pairs of antipodal points.*

$2'$. *C is tangent to two parallel segments of Q and meet a vertex of $CH(P)$.*

$2''$. *C touches Q in two antipodal points q_1 and q_2 and P in two vertices p_1 and p_2 such that points P_1, q_1, p_2, and q_2 appear in that cyclic order on circle C.*

4 Intersecting Upper Envelope of Cones.

In this section we will apply the idea of hierarchical representation of convex polyhedra to some other types of surfaces.

For any set $S = s_1, s_2, ..., s_n$ of points (sites) in the xy-plane let $LC(S)$ denote the family of lifting cones $LC(s_1), LC(s_2), ..., LC(s_n)$. For each cone only the part with the positive values of z-coordinate is taken into consideration. We will be looking for the points of intersection of $UE(S)$, the upper envelope of the family of cones $LC(S)$, with the query curves belonging to a family F including some lines, parabolae and hyperbolae.

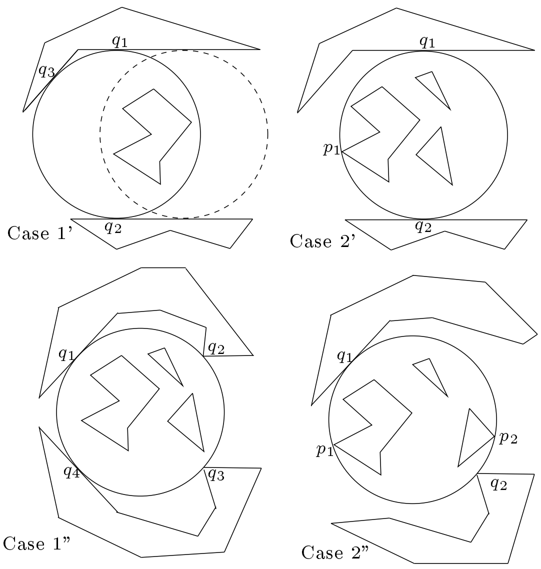


Figure 2: Degenerate cases of Lemma 1

Theorem 2 *Let $LC(S)$ be a family of lifting cones and F be a family of lines in the three-dimensional space. It is possible to preprocess the set of cones $LC(s_i)$, $i = 1, 2, ..., n$, in $O(n \log\ n)$ time and $O(n)$ space, so that the intersections of $UE(S)$ with a query line l may be found in $O(\log\ n)$ time.*

Proof (sketch): According to an earlier observation, the projection on the xy-plane of the segments of hyperbola, along which the portions of cones contributing to $UE(S)$ meet, is a straight-line planar subdivision whose cells are unbounded. It is possible to find, in time proportional to the graph size, a fraction of its faces, which are pairwise non adjacent and such that each face has a bounded number of edges. After eliminating from S the origins of the cones corresponding to these faces we are left with a subset S' of S. Continuing this process we construct a hierarchical representation of sets $S_1 \subset S_2 \subset ... \subset S_k = S$, where S_1 is a single point. We obtain as well a hierarchy $G(S_1)$, $G(S_2)$, ... , $G(S_k)$ of $k = O(\log\ n)$ straight-line planar subdivision, such that any face $G(S_i)$ intersects a bounded number of faces of $G(S_{i-1})$ and vice versa. Using Kirkpatrick's techniques,[14] this hierarchical representation may be found in $O(n \log\ n)$ time and $O(n)$ space.

Observe that because of the convexity of the intersection of the cones in $LC(S)$, l intersects $UE(S)$ in at most two points. In fact, if l intersects $UE(S)$ in two points it intersects each $UE(S_i)$, $i = 1, 2, ..., k$, in two points and if l intersects $UE(S)$ in a single point it intersects each $UE(S_i)$ in a single point as well. Suppose that we know the intersection x of a query line l with $UE(S_{i-1})$, and the localisation x' of the projection of x in one of the faces of $G(S_{i-1})$. In constant time, we can compute the intersection of l with $UE(S_i)$, as well as the face of $G(S_i)$ containing its projection. In $k = O(log\ n)$ steps, each one taking a constant time, we can compute the intersection of l with $UE(S)$. □

Note that, as $UE(S_i)$ is contained in $UE(S_{i-1})$, it may happen that l intersects $UE(S_{i-1})$ but not $UE(S_i)$, so after some steps of the search the process may detect no intersection points.

Suppose that, instead of a line, we can have as a query any curve $\zeta \in F$ belonging to one of the three following categories:

1. $\zeta = LC(x_1, y_1) \cap LC(x_2, y_2)$, intersection of two lifting cones, for any two points (x_1, y_1) and (x_2, y_2) of the xy-plane,
2. $\zeta = LC(x_1, y_1) \cap LH^+(l)$, intersection of a lifting cone with a lifting halfplane, for any point (x_1, y_1) and any oriented line l of the xy-plane,
3. $\zeta = LH^+(l_1) \cap LH^+(l_2)$, intersection of any two lifting halfplanes.

Family F contains now branches of hyperbola, parabola and lines. It is possible to prove that the above theorem generalises then to

Theorem 3 *Let F be a family of curves in three dimensions such that any curve $\zeta \in F$ is an intersection of two surfaces, each of these surfaces being either a lifting cone or a lifting halfplane.*

1. *There are at most two intersections of $\zeta \in F$ with $UE(S')$, for any $S' \subseteq S$.*
2. *It is possible to preprocess the cones $LC(S)$ in $O(n \log n)$ time and $O(n)$ space, so that the intersections of $UE(S)$ with a query curve $\zeta \in F$ may be found in $O(\log n)$ time.*

Proof: omitted

5 The Algorithm.

Before turning our attention to the algorithm we make a few simple observations stated in the following lemmas. They concern the images of some families of circles in the space of circles.

Lemma 4 *The image with respect to the transformation Φ of a family of circles passing through two given points s_1 and s_2 is the branch of hyperbola being the intersection of $LC(s_1)$ and $LC(s_2)$*

Lemma 5 *The image with respect to the transformation Φ of a family of circles tangent from the right-hand side to a given oriented line l_1, and passing through a given point s_1 lying on the right-hand side of l_1 is the parabola being the intersection of the cone $LC(s_1)$ with the halfplane $LH^+(l_1)$*

Lemma 6 *The image with respect to the transformation Φ of a family of circles tangent from the right-hand side to two given oriented lines l_1 and l_2 is the line of intersection of two halfplanes $LH^+(l_1)$ and $LH^+(l_2)$*

To find the largest circles separating two sets of line segments P and Q we will run twice the algorithm given below. First the algorithm looks for the largest circles C enclosing P and leaving Q outside. Both P and Q may meet a largest circle C in a finite number of points, but they must be disjoint with the open regions lying, respectively, inside or outside C. In the second run of the algorithm the roles of P and Q are reversed. The algorithm will report all largest separating circles with at least three contact points.

The idea of the algorithm is to search for all the circles which may possibly verify one of the conditions of Lemma 1 or 1(bis). Consider first conditions 1, $1'$ or $1''$. Any circle C tangent to Q in three points and not containing any point of Q inside C is centered at a vertex of $CSVor(Q)$, the closest site Voronoi diagram of the set of line segments Q. Each such center of a candidate circle is point-located in some face of $FSVor(P)$, the furthest site Voronoi diagram of the vertices of the convex hull $CH(P)$ of the set of line segments P. This way we can compute the distance from the center of C to the furthest point of P. If this distance appears to be smaller that the radius of C, C separates P and Q. In such a case, if C verifies conditions 1, $1'$ or $1''$, it is reported as a largest separating circle.

When the separating circle C verifies the conditions 2 of Lemma 1 or one of the degenerated condition $2'$ and $2''$, it must be internally tangent to some vertex p_1 of $CH(P)$ and externally tangent to some two points q_1 and q_2 of Q. The first condition means that $\Phi(C)$ lies on $UE(P)$, the upper envelope of the lifting cones of the vertices of $CH(P)$, within the face corresponding to vertex p_1. At the same time, the center of C lies then on a Voronoi edge of $CSVor(Q)$ equidistant from q_1 and q_2. Suppose that q_1 and q_2 are internal points of two edges of Q, then in the space of circle $\Phi(C)$ lies then on a segment belonging to a line determined according to Lemma 6. Similarly, if q_1 or q_2 are endpoints of segments of Q, the corresponding edge of $CSVor(Q)$ is mapped in the space of circles to a segment of parabola or to a segment of hyperbola as stated in Lemmas 4 and 5. Thus, to find the largest separating circles which fulfill conditions 2, $2'$ or $2''$, it is sufficient to scan, one by one, all the $O(n)$ edges of $CSVor(Q)$. For each edge of $CSVor(Q)$, the segment of line, parabola or hyperbola which is the transformed of the largest circles centered on this edge is intersected with the enveloppe $UE(P)$. The hierarchical representation of $UE(P)$ is used for this purpose. Each point of intersection corresponding to a circle which verify one of the conditions 2, $2'$ or $2''$ is reported as a largest separating circles.

Algorithm All Largest Separating Circles

Input: Two sets of line segments P and Q with a total of n segments meeting only at their endpoints.

Output: All largest separating circles C, with P inside, Q outside and at least three contact points.

1. Compute $FSVor(P)$ the furthest site Voronoi diagram of the vertices of the convex hull $CH(P)$ of set P; compute $UE(P)$ the representation of $FSVor(P)$ in the space of circles.
2. Compute the hierarchical representation of $UE(P)$.
3. Compute $CSVor(Q)$, the closest site Voronoi diagram of the set Q.
4. **for** each vertex v of $CSVor(Q)$
   - 4.1. Compute the distance $d(v, Q)$ from v to Q.
   - 4.2. Locate v in a face of $FSVor(P)$ and compute $d(v, FSVor(P))$, the distance from v to the furthest vertex of P.
   - 4.3. **if** $d(v, Q) >= d(v, FSVor(P))$ **and** one of the conditions 1, $1'$ or $1''$ holds for circle C centered in v with radius $d(v, Q)$ **then Output(C)**.
5. **for** each edge e of $CSVor(Q)$
   - 5.1. Compute a curve segment z in the space of circles corresponding to largest circles centered on e and externally tangent to Q and the curve ζ (line, parabola or hyperbola) supporting z.
   - 5.2. Compute x_1 and x_2, the two intersections of ζ with $UE(CH(P))$ if they exist.
   - 5.3. **for** $i = 1, 2$ **if** $x_i \in z$ **and** x_i is the image of a circle C_i such that conditions 2, $2'$ or $2''$ hold **then Output(C_i)**.

End of the Algorithm

Correctness of the algorithm is obvious considering Lemmas 1, 1(bis) and the previous discussion.

6 Complexity of the Algorithm.

The computation of the furthest site Voronoi diagram in step 1 takes $O(n \log n)$ time and $O(n)$ space by well known algorithms.[13] To obtain in linear time the upper envelope $UE(P)$, each of $O(n)$ faces of $FSVor(P)$ is transformed into a portion of a cone, and each of the $O(n)$ edges of FSVor(S(P)) is transformed into a segment of hyperbola.

The hierarchical representation of $UE(P)$ from step 2 is computed in $O(n \log\ n)$ time and $O(n)$ space by Theorem 3.

The Voronoi diagram of the set of line segments from step 3 is computed within $O(n \log\ n)$ time and $O(n)$ space using one of the well known algorithms.[15]

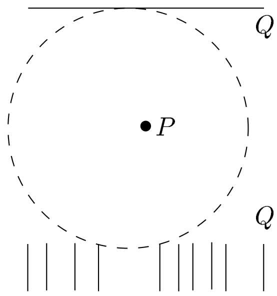


Figure 3: The lower bound example

The for loop from step 4 is run $O(n)$ times. Step 4.1 takes a constant time. The point location in the planar map $FSVor(CH(P))$ from step 4.2 takes $O(\log n)$ time, for example using Kirkpatrick technique.[14] Observe that a hierarchical representation of $FSVor(P)$, needed for this approach is implied by the structure computed in step 2. In step 4.3, in time proportional to the degree of vertex v we check whether one of the conditions 1, $1'$ or $1''$ is met. Over all iterations of the foor loop the complexity of step 4.3 is $O(n)$. Thus the total complexity of the for loop from step 4 is $O(n \log n)$.

Similarly, the for loop from step 5 is executed $O(n)$ times. Depending on the case, the curve segment z needed in step 5.1 is computed using one of the lemmas 4, 5 or 6. By Theorem 3, there are at most two intersections of z with $UE(CH(P))$ and they may be computed in $O(\log n)$ time. In constant time we then check in step 5.3 if the candidate circle meets one of the conditions 2, $2'$ or $2''$ in order to report it as a largest separating circle. We conclude that the loop 5 takes $O(n \log\ n)$ time.

We have thus proved

Theorem 7 *For two sets of line segments P and Q with a total of n segments meeting only at their endpoints, it is possible to compute in $O(n\ log\ n)$ time and $O(n)$ space all largest circles separating P and Q.*

Once all largest separating circles have been reported, the largest one can be easily found comparing their radii.

In order to show the optimality of our result we sketch the proof of a $\Omega(n\ log\ n)$ lower bound of our problem.[20] It is obtained by reduction of the maximum gap problem for which the $\Omega(n\ log\ n)$ lower bound was established in the linear decision-tree model of computation.[16] Let $X = x_1, x_2, ..., x_n$ be a set of points on the real line between x_{min} and x_{max} for which the maximum gap must be computed, i.e. the pair of consecutive points of X maximizing the distance between them. Let set Q contain n line segments, each one extending between the points $(x_i, -1)$ and $(x_i, 0)$, $i = 1, 2, ..., n$ and the $(n+1)$-th segment s extending between the points $(x_{min}, x_{max} - x_{min})$ and $(x_{max}, x_{max} - x_{min})$. Let P be a single point of coordinates $\left(\frac{x_{min}+x_{max}}{2}, \frac{x_{max}-x_{min}}{2}\right)$. Clearly, the largest circle separating P and Q is tangent to s and passes through segments at x_i and x_j defining the maximum gap in X (see Figure 3).

It follows from the algorithm that there is at most $O(n)$ largest separating circles with at least three contact points. Indeed, for each of $O(n)$ vertices of $CSVor(Q)$ there is at most one such candidate separating circle, and for each of $O(n)$ edges of $CSVor(Q)$ there are at most two candidate circles. The above example where values of X are equally spaced shows that there are sets P and Q actually admitting $O(n)$ largest separating circles.

7 Conclusions.

The paper gives an efficient algorithm for the problem of finding all largest circles separating two given sets of line segments. The solution is proven optimal in the linear decision-tree model of computation. However, for the general algebraic tree model, the lower bound for the maximum gap problem is not known so far. Thus in this model of computation the optimality of our solution remains open.

Note that the previous discussion does not imply an $\Omega(n\ log\ n)$ lower bound for the problem of the largest circle separating two given polygons. Indeed, it is not possible to build in linear time a polygon using the set of line segments from Figure 3.

It was supposed in this paper that the line segments meet only at their endpoints. For two arbitrary sets of segments we can obtain $\Omega(n^2)$ points of intersection. However, the following corollary states that we can tackle the problem of largest circles separating two arbitrary sets of line segments in less than quadratic time.

Corollary 8 *For two sets of line segments P and Q containing a total of n segments, it is possible to compute in $O(n\alpha(n)\log^2 n)$ deterministic time or in $O(n\alpha(n)\log n)$ randomized time and $O(n\alpha(n))$ space all locally largest circles separating P and Q.*

To prove this, observe that if there exists a circle C separating two sets of line segments P and Q, P lying inside and Q lying outside C, it is sufficient to take into consideration separation of the external cell of the arrangement of line segments of set P, and, containing it, a single cell of the arrangement of line segments of Q. The complexity of such cell[9] is at most $O(n\alpha(n))$ and it may be computed in $O(n\alpha(n)\log^2 n)$ deterministic time[5] or in $O(n\alpha(n)\log n)$ randomized time.[7] Once both cells are computed, we can apply our algorithm to $O(n\alpha(n))$ portions of line segments which do not meet outside their endpoints.

An interesting open problem is to extend the algorithm onto some other classes of objects like, for example, circles or figures bounded by line segments and circular arcs.

The original approach of the hierarchical representation[14] was applied to the problem of intersection of convex polyhedra (envelopes of planes in three dimensions) by query lines. The approach presented here applied this idea to the upper envelopes of some families of cones. Moreover, not only lines are used as the intersection queries, but also some other curves as parabola and hyperbola. It is of independent interest to extend this idea to some other families of surfaces.

References

[1] A. Aggarwal, H. Booth, J. O'Rourke, S. Suri, and C. K. Yap. Finding minimal convex nested polygons. *Inform. Comput.*, 83(1):98–110, October 1989.

[2] B. K. Bhattacharya. Circular separability of planar point sets. In G. T. Toussaint, editor, *Computational Morphology*, pages 25–39. North-Holland, Amsterdam, Netherlands, 1988.

[3] J.-D. Boissonnat, J. Czyzowicz, O. Devillers, and M. Yvinec. Circular separability of polygon. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms (SODA)*. 1995.

[4] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite vc-dimension. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 293–302, 1994.

[5] B. Chazelle, H. Edelsbrunner, L. Guibas, M. Sharir, and J. Snoeyink. Computing a face in an arrangement of line segments and related problems. *SIAM J. Comput.*, 22:1286–1302, 1993.

[6] G. Das and D. Joseph. The complexity of minimum convex nested polyhedra. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 296–301, 1990.

[7] M. de Berg, K. Dobrindt, and O. Schwarzkopf. On lazy randomized incremental construction. In *Proc. 26th Annu. ACM Sympos. Theory Comput.*, pages 105–114, 1994.

[8] D. P. Dobkin and D. G. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra. *J. Algorithms*, 6:381–392, 1985.

[9] H. Edelsbrunner, L. Guibas, and M. Sharir. The complexity of many cells in arrangements of planes and related problems. *Discrete Comput. Geom.*, 5:197–216, 1990.

[10] H. Edelsbrunner and F. P. Preparata. Minimum polygonal separation. *Inform. Comput.*, 77:218–232, 1988.

[11] S. Fisk. Separating points by circles and the recognition of digital discs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(4):554–556, 1986.

[12] C. E. Kim and T. Anderson. Digital disks and a digital compactness measure. In *Proc. 16th Annu. ACM Sympos. Theory Comput.*, pages 117–124, 1984.

[13] D. G. Kirkpatrick. Efficient computation of continuous skeletons. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 18–27, 1979.

[14] D. G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12:28–35, 1983.

[15] D. T. Lee and R. L. Drysdale, III. Generalization of Voronoi diagrams in the plane. *SIAM J. Comput.*, 10:73–87, 1981.

[16] U. Manber and M. Tompa. The complexity of problems on probabilistic, nondeterministic, and alternating decision trees. *J. ACM*, 32(3):720–732, 1985.

[17] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. ACM*, 31:114–127, 1984.

[18] J. S. B. Mitchell and S. Suri. Separation and approximation of polyhedral surfaces. In *Proc. 3rd ACM-SIAM Sympos. Discrete Algorithms*, pages 296–306, 1992.

[19] D. M. Mount. Intersection detection and separators for simple polygons. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 303–311, 1992.

[20] J. O'Rourke, S. R. Kosaraju, and N. Megiddo. Computing circular separability. *Discrete Comput. Geom.*, 1:105–113, 1986.