Computing Circular Separability

Joseph O'Rourke,[1] S. Rao Kosaraju,[1] and Nimrod Megiddo[2]

[1] Department of Electrical Engineering and Computer Science, Johns Hopkins University, Baltimore, MD 21218

[2] IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120-6099, and Tel-Aviv University, Tel-Aviv, Israel

Abstract. Two sets of planar points S_1 and S_2 are circularly separable if there is a circle that encloses S_1 but excludes S_2. We show that deciding whether two sets are circularly separable can be accomplished in $O(n)$ time using linear programming. We also show that a smallest separating circle can be found in $O(n)$ time, and largest separating circles can be found in $O(n \log n)$ time. Finally we establish that all these results are optimal.

1. Introduction

We consider the problem of determining whether two sets of planar points are circularly separable. More precisely, let S_1 and S_2 be two sets of points in R^2 with $|S_1| + |S_2| = n$. We seek to detect whether there is a circle C such that each point of S_1 is interior to or on the boundary of C, while each point of S_2 is exterior to or on the boundary of C. If the sets are circularly separable, we additionally want to find smallest and largest separating circles.

This problem has application to pattern recognition and image processing: Determining whether a collection of pixels is a digital image of a disk can be reduced to circular separability. Recently an $O(n^2)$ algorithm for digital disk recognition has been offered [11]; we improve this speed to $O(n)$.

Our results are as follows:

Detection of circular separability $O(n)$,

Finding a smallest separating circle $O(n)$,

Finding all largest separating circles $O(n \log n)$.

Moreover we show that all these results are optimal, with the consequence that it is easier to find a smallest circle than a largest circle.

Finally, the problem of spherical separability in arbitrary dimensions is discussed.

2. Detection of Circular Separability

Let $S_1 = \{(x_i, y_i): i \in I_1\}$ and $S_2 = \{(x_i, y_i): i \in I_2\}$. If either S_1 or S_2 is empty, then the sets are trivially separable. Assume then that neither is empty. We first transform these sets from R^2 to R^3 by the mapping $P(x, y) \rightarrow (x, y, x^2 + y^2)$. Call the transformed sets S_1' and S_2'. The equation of a plane in the transformed space is

$$ax + by + (x^2 + y^2) = c, \tag{1}$$

which may be rewritten as

$$\left(x + \tfrac{1}{2}a\right)^2 + \left(y + \tfrac{1}{2}b\right)^2 = c + \tfrac{1}{4}(a^2 + b^2). \tag{2}$$

Thus a plane in the transformed space corresponds to a circle in the original space centered at $(-\tfrac{1}{2}a, -\tfrac{1}{2}b)$ as long as

$$c + \tfrac{1}{4}(a^2 + b^2) \geq 0. \tag{3}$$

Similarly, every circle in the original space,

$$(x - A)^2 + (y - B)^2 = R^2, \tag{4}$$

may be expanded to

$$(-2A)x + (-2B)y + (x^2 + y^2) = R^2 - (A^2 + B^2). \tag{5}$$

The correspondence $A = -\tfrac{1}{2}a$, $B = -\tfrac{1}{2}b$, and $R^2 = c + \tfrac{1}{4}(a^2 + b^2)$ demonstrates that this is a plane in the transformed space, and the requirement that $R^2 \geq 0$ leads again to condition (3). Thus, there is a one-to-one correspondence between circles in the original space and planes satisfying Eq. (3) in the transformed space.

Geometrically, the mapping P raises each point (x, y) to the paraboloid $z = x^2 + y^2$. Any plane $ax + by + z = c$ satisfying (3) cuts the paraboloid in an ellipse, which projects to a circle on the xy-plane. See Fig. 1. Since S_1 is nonempty, for any $(x_1, y_1) \in S_1$, a separating circle must satisfy $(x_1 - A)^2 + (y_1 - B)^2 \leq R^2$. This assures that R^2 is nonnegative, which in turn implies that Eq. (3) is always satisfied. In the transformed space, this means that any plane separating S_1' from S_2' must cut the paraboloid. Thus, the problem of circular separability in the plane reduces to linear separability in three dimensions.

It is well known that linear separability in turn reduces to linear programming [5]. Thus, we can detect whether two sets are circularly separable, and
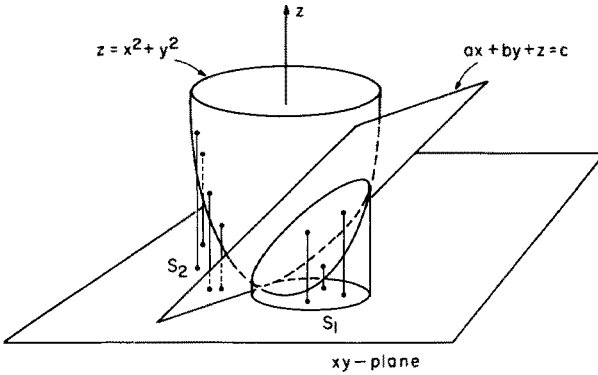
Fig. 1. Each plane that cuts the paraboloid corresponds to a circle in the plane, and vice versa.

indeed find a separating circle if they are, in $O(n)$ time using Megiddo's or Dyer's linear-time algorithm [2, 12].

We can now improve on Kim and Anderson's $O(n^2)$ algorithm for recognition of digital disks to $O(n)$, where n is the number of pixels on the boundary of the given convex region Q [11]. Let S_1 be the integral coordinate points representing the n boundary pixels. Let S_2 be the collection of points representing all pixels exterior to Q but 8-connected to a point in S_1; that is, adjacent horizontally, vertically, or diagonally. S_2 has at most $8n$ points, and can be easily constructed in linear time. Q is a digital disk iff there exists a circle that encloses S_1 and strictly excludes S_2. This does not quite match our definition of separability because of the strict exclusion requirement, but we will now show that the strict inequalities do not cause special difficulty.

Let I_1 and I_2 be index sets for S_1 and S_2, respectively. Then Q is a digital disk iff there exist a, b, and c such that

$$\begin{aligned}
ax_i + by_i + \left(x_i^2 + y_i^2\right) &\leq c \quad (i \in I_1), \\
ax_i + by_i + \left(x_i^2 + y_i^2\right) &> c \quad (i \in I_2).
\end{aligned} \tag{6}$$

This problem can be solved as a four-variable linear programming problem:

$$\underset{a,b,c,d}{\text{maximize}}\ d \tag{7}$$

such that

$$\begin{aligned}
ax_i + by_i + \left(x_i^2 + y_i^2\right) &\leq c \quad (i \in I_1), \\
ax_i + by_i + \left(x_i^2 + y_i^2\right) &\geq c + d \quad (i \in I_2).
\end{aligned}$$

There is a solution to (6) if and only if the optimal value for the system (7) is positive. However, there is no need to solve the four-variable problem. If there is a feasible solution with some positive d, then the system has a basic solution with

some positive d. The smallest possible positive value of d in a basic solution is bounded from below by

$$\frac{1}{24\max_i |x_i y_i|}.$$

This follows from an upper bound on the absolute value of any determinant of a 4×4 submatrix of the coefficient matrix, where the coefficients of a, b, c, and d are x_i, y_i, -1, and either 0 or 1, respectively. Thus, we may set d to such a lower bound and solve the resulting system of weak inequalities in three variables using the linear-time algorithm, establishing that the digital disk recognition problem can be solved in $O(n)$ time.

3. Finding a Smallest Separating Circle

The problem of finding a smallest separating circle may be formulated from Eqs. (1) and (2) as follows:

$$\underset{a, b, c}{\text{minimize}} \tfrac{1}{4}a^2 + \tfrac{1}{4}b^2 + c \tag{8}$$

such that

$$ax_i + by_i + \left(x_i^2 + y_i^2\right) \le c \qquad (i \in I_1),$$
$$ax_i + by_i + \left(x_i^2 + y_i^2\right) \ge c \qquad (i \in I_2).$$

This is a convex quadratic minimization problem in R^3 with linear constraints, and again can be solved by Megiddo's techniques in $O(n)$ time [12]. As in Section 2, we can argue that the minimum of the objective function is nonnegative.

4. Finding Largest Separating Circles

Megiddo's algorithm only applies to *convex* quadratic functions; negation of the objective function in Eq. (3) leads to a concave function for the largest separating circle problem. Thus, another approach is required. We first observe that if S_1 and S_2 are linearly separable, then the largest separating circle has infinite radius. In the transformed space, this corresponds to the existence of a vertical plane separating S_1' and S_2', cutting the paraboloid in a parabola, which projects to a straight line in the original space. This case can be dispensed with in linear time using the algorithms in [2, 12]. We base our algorithm for the finite radius case on the following lemma.

Lemma. *Any largest separating circle of finite radius for S_1 and S_2 must either pass through three points of S_2, or it must pass through two points of S_2 and one of S_1.*
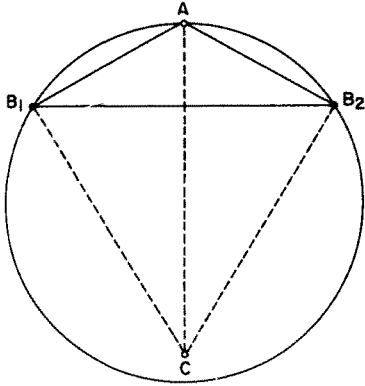
Fig. 2. An example where the largest separating circle is determined by one point A of S_1 and two points B_1 and B_2 of S_2.

Proof. Let C be a largest circle of finite radius. Since three points determine a circle, if C passes through less than three points of $S = S_1 \cup S_2$, then there is freedom to expand it while maintaining its contacts with the points of S, contradicting maximality. If C passes through just one point p of S_2, then it may be expanded by moving its center directly away from p while adjusting the radius to maintain contact with p. This again contradicts maximality. The only possibilities remaining are those claimed in the statement of the lemma. $\square$

An example of the second possibility is illustrated in Fig. 2. $S_1 = \{A, C\}$ and $S_2 = \{B_1, B_2\}$, where A is the apex and B_1 and B_2 the two base vertices of an isosceles triangle obtuse at A, and AC is perpendicular to $B_1 B_2$, with the length of AC just smaller than the diameter of the circle determined by $B_1 A B_2$, so that angle $B_1 C B_2$ is acute. A largest circle that excludes S_2 must have its center on the line determined by A and C, and this center must either be as far above C or as far below A as possible. The upper extreme circle is determined by $B_1 C B_2$, and the lower by $B_1 A B_2$. The latter circle is larger since the angle at A is obtuse while that at C is acute. Moreover, we can place C such that it lies strictly inside this circle. Thus the largest circle is determined by one point A of S_1, and two points B_1 and B_2 of S_2.

Observe now that any plane separating S_1' from S_2' in the transformed space must separate the convex hulls H_1 and H_2 of S_1' and S_2'. It follows from the lemma, then, that a separating plane π corresponding to a largest separating circle contacts the hulls in one of the following two manners:

(a) π contains ≥ 3 points of H_2,
(b) π contains ≥ 2 points of H_2 and ≥ 1 point of H_1.

In case (a), π is determined by a face of H_2, and in case (b), by an edge of H_2 and a vertex of H_1. This taxonomy permits the largest circles to be found by the following search algorithm.

(1) Compute the convex hulls H_1 and H_2 of S_1' and S_2'.
(2) $\{a\}$ **for** each face F of H_2 **do**
(3)   Determine if π containing F separates H_2 from H_1.
(4) $\{b\}$ **for** each edge e of H_2 **do**
(5)   Determine the vertex v of H_1 hit first by a plane rotating about e.
(6)   Determine if π containing e and v separates H_2 from H_1.

The convex hulls (step 1) can be computed in $O(n \log n)$ time [14]. The two **for** loops (steps 2 and 4) each iterate $O(n)$ times since a polyhedron of n vertices has $O(n)$ edges and faces. Thus the algorithm will have the claimed $O(n \log n)$ complexity if both loop bodies can be executed in $O(\log n)$ time. But each of these is what Edelsbrunner and Maurer call an "extremal query," and for which they provide an algorithm that requires $O(n \log n)$ preprocessing time and $O(n)$ space, and which can answer each such query in $O(\log n)$ time [6].

More precisely, they define an extremal query as follows. Let H be a convex polyhedron of n vertices in R^3, let π be a plane not intersecting H, and let l be a line (possibly at infinity) in π. An extremal query asks to determine a point of H hit first when π is rotated counterclockwise about l; if l is at infinity, the rotation degenerates to a translation. Their algorithm employs the $\log n$-height hierarchical data structure of nested polyhedral due to Kirkpatrick [4, 10].

Step 3 can be accomplished with a translation query between π and H_1: H_2 and H_1 are separable by π iff the point of H_1 that would first be hit by translating π is on the opposite side of π from H_2. Step 5 is a rotation query with l determined by e. Step 6 can then be settled in constant time by examining the relationship between π at e, and the two faces of H_2 adjacent to e.

Thus all largest circles can be found in $O(n \log n)$ time by computing the radii of every circle associated with a plane π explored by the search algorithm, and saving the maxima. Thus the largest circles can be found in $O(n \log n)$ time; the space requirements for the data structures are $O(n)$ [6].

Note that with a suitable modification of the lemma and the algorithm, the smallest circles could also be found with this technique. Finally we note that if S_1 is empty, our problem does *not* become the largest empty circle problem [16, 18] since that problem requires the center of the circle to be within the convex hull of S_2. If S_1 is empty, the largest separating circle has radius infinity.

5. Lower Bounds

It is clear that $\Omega(n)$ is a lower bound on the three problems considered since every point of both sets must be examined. This establishes the optimality of the detection and smallest circle algorithms.

We establish an $\Omega(n \log n)$ bound on the largest circle problem by reduction from the maximum gap problem [9]. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a given set of real numbers for which the maximum gap is to be computed, that is, the pair of points consecutive on the real line that determine the largest separation. In linear time find the minimum L and maximum R of X. Transform X to X' by mapping each point x to $x' = (x - L)/(R - L)$, again in $O(n)$ time. Let $S_1 = \{(\frac{1}{2}, \frac{1}{2})\}$, and let

$$S_2 = \{(x_i', 0): x_i' \in X'\} \cup \{(x_i', 1): x_i' \in X'\}.$$

We claim that a largest separating circle for these sets determines the maximum gap of X, as illustrated in Fig. 3.

By the lemma, a largest circle must either be determined by three points of S_2 or two of S_2 and one of S_1. If C passes through the point of S_1, then it must have
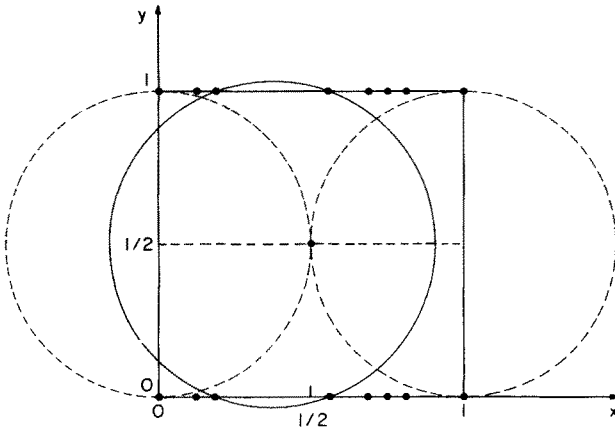
Fig. 3. Reduction of the maximum gap problem to finding the largest separating circle.

radius $\frac{1}{2}$ and be centered at $(0, \frac{1}{2})$ or $(1, \frac{1}{2})$ (the dashed circles in Fig. 3). But neither of these circles is maximal since they can be enlarged by moving their centers toward $(\frac{1}{2}, \frac{1}{2})$. Therefore, C must pass through three or more points of S_2. The symmetry of S_2 about $y = \frac{1}{2}$ implies that C must pass through precisely four distinct points, and be centered on the line $y = \frac{1}{2}$. It is clear, then, that a largest circle passes through points determining a maximum gap for X'; the unscaled version of these points determines a maximum gap for the original set X.

Since the maximum gap problem is known to require $\Omega(n \log n)$ operations in the decision tree model of computation [9], the largest separating circle problem algorithm in the previous section is asymptotically optimal. We should note, however, that the maximum gap problem can be solved in linear time if the floor function is permitted [17].

6. Discussion

Algorithms for the minimum spanning circle [17] and the largest empty circle [17, 18] employ the furthest-point and closest-point Voronoi diagrams, respectively. In some sense a mixture of both diagrams is needed to find separating circles. This has recently been made precise by Fisk [8], who described an $O(n^2)$ algorithm based on explicit use of both diagrams. The recent connections discovered between Voronoi diagrams and the paraboloid transformation [7] show that our search algorithm can be interpreted as using the Voronoi diagrams in another guise. The underside of H_2 (composed of the faces whose outward pointing normals have negative z components) projects to the Delaunay triangulation of S_2 on the xy-plane, the dual of the closest-point Voronoi diagram. The top side of H_1 projects to the dual of the furthest-point Voronoi diagram of S_1. Thus, a plane that supports H_2 on the underside and H_1 on the top side can be viewed as being determined by the two Voronoi diagrams.

Both the problems and the algorithms generalize to spherical separability in d-dimensions. Megiddo's linear algorithms still apply to the detection, digital disk recognition, and smallest sphere problems, albeit with large dimension-dependent constants [13]. These constants have recently been improved to $O(3^{d^2})$ by Clarkson [1] and Dyer [3]. The largest-sphere algorithm must spend $O(n^{\lceil d/2 \rceil})$ to compute the convex hulls [15], and can no longer depend on efficient extremal queries. Since the hulls may have as many as $O(n^{\lfloor d/2 \rfloor})$ facets, and the queries cost $O(n)$, the complexity is as listed below:

Detection of spherical separability	$O(3^{d^2}n)$,
Finding a smallest separating sphere	$O(3^{d^2}n)$,
Finding all largest separating spheres	$O(n^{\lfloor d/2 \rfloor + 1})$.

Acknowledgments

References

1. K. L. Clarkson, Linear Programming in $O(n3^{d^2})$ Time, Info. Proc. Letters, to appear, 1986.
2. M. E. Dyer, Linear Time Algorithms for Two- and Three-Variable Linear Programs, SIAM J. Computing, 13 (1984), 31–45.
3. M. E. Dyer, On a Multidimensional Search Technique and its Application to the Euclidean One-Center Problem, Department of Mathematics and Statistics, Teesside Polytechnic, manuscript, 1985.
4. D. P. Dobkin and D. G. Kirkpatrick, Fast Detection of Polyhedral Intersections, Theoretical Computer Science, 17 (1983), 241–253.
5. D. P. Dobkin and S. P. Reiss, The Complexity of Linear Programming, Theoretical Computer Science, 11 (1980), 1–18.
6. H. Edelsbrunner and H. Maurer, Finding Extreme Points in Three Dimensions and Solving the Post-Office Problem in the Plane, Info. Proc. Lett. 21 (1985), 39–47.
7. H. Edelsbrunner and R. Seidel, Voronoi Diagrams and Arrangements, Proc. ACM Symp. on Computational geometry, Baltimore, MD, June 1985, 251–262.
8. S. Fisk, Separating Points by Circles, and the Recognition of Digital Disks, IEEE Trans. on PAMI, to appear, 1986.
9. D. T. Lee and F. P. Preparata, Computational Geometry: A Survey, IEEE Trans. Computers, C-33, 12 (1984), 1072–1101.
10. D. G. Kirkpatrick, Optimal Search in Planar Subdivisions, SIAM J. on Computing 12 (1983), 28–35.
11. C. E. Kim and T. A. Anderson, Digital Disks and a Digital Compactness Measure, Proc. of 16th ACM Symposium on Theory of Computing, Washington, DC, 1984, 117–124.
12. N. Megiddo, Linear-Time Algorithms for Linear Programming in R^3 and Related Problems, SIAM J. on Computing 12 (1983), 759–776.

13. N. Megiddo, Linear Programming in Linear Time When the Dimension is Fixed, JACM, 31 (1984), 114–127.
14. F. P. Preparata and S. J. Hong, Convex Hulls of Finite Sets of Points in Two and Three Dimensions, CACM, 20 (1977), 87–93.
15. R. Seidel, A Convex Hull Algorithm Optimal for Point Sets in Even Dimensions, Tech. Rep. 81-14, University of British Columbia, Sep. 1981.
16. M. I. Shamos, Computational Geometry, Ph.D. thesis, Yale Univ., New Haven, CT, 1978.
17. M. I. Shamos and D. Hoey, Closest-Point Problems, Proc. 16th IEEE Symposium on Foundations of Computer Science, Los Angeles, CA, 1975, 151–162.
18. G. T. Toussaint, Computing the Largest Empty Circle with Location Constraints, Inter. J. Computer and Info. Sci., 12 (1983), 347–358.