Annulus Placement Problems for Scaled and Offset Polygons (extended abstract)*

Gill Barequet† Prosenjit Bose‡ Matthew T. Dickerson§

Abstract

A polygon annulus is the region contained in some outer polygon but not contained in an inner polygon. Typically, the outer and inner polygons are concentric scaled or offset copies of the same polygon. Offset polygons and offset polygon annulus placement problems have recently been studied in [BDG] and [BBDG]. In this paper we study two related annulus placement problems: the first variant fixes the outer containing polygon and aims to maximize the inner polygon; the second variant fixes the inner polygon and aims to minimize the outer polygon.

1 Introduction

There has been considerable interest in various problems arising in geometric tolerancing, especially in the optimal placement of some annulus region. One set of problems studied recently in [BDG] and [BBDG] involves the optimal placement of some polygon annulus regions with respect to a set of points, where optimality is measured either by minimizing the size of the annulus region necessary to contain all the points, or maximizing the number of points contained in some polygon annulus of fixed size. The polygon annulus itself can either be defined as the difference region between two *scaled* copies of some polygon P or between two *offset* copies for the offseting function defined in [BBDG]. In addition to the tolerancing application, these problems arise also in the robot localization problem [GMR].

In the case where the annulus shape is a disk, it was recently shown in [BBBRW] that when either the inner disk or the outer disk is of fixed size, the placement problem can be solved more efficiently. Computing the optimal placement of an annulus is a fundamental aspect of quality control in the manufacturing sciences. In fact, the width of the thinnest circular annulus containing a set of points is the measure used by the American National Standards Institute (see Foster [F], pp. 40-42) and the International Standards Office for testing roundness. Fixing the size of the inner or outer shape of an annulus is an important aspect of quality control. For example, if a manufactured object is to fit inside a *sleeve*, the outer annulus must be fixed. On the other hand, if a part must fit over a peg, then the inner annulus is crucial.

In this paper we explore a set of polygon annulus placement problems where one of the two annuli (inner or outer) is fixed. In all cases, the goal for a polygon P and a point set S is to find an annulus region of minimum width (so-called the *tolerance* that contains all of the points in S. We solve problems for annulus

*Work on this paper by the first author has been supported in part by the U.S. Army Research Office under Grant DAAH04-96-1-0013. Work by the second author has been supported by the Natural Science and Engineering Research Council of Canada under grant OGP0183877. Work by the third author has been supported by the National Science Foundation under Grant CCR-93-1714.

†Center for Geometric Computing, Dept. of Computer Science, Johns Hopkins University, Baltimore, MD 21218. E-mail: `barequet@cs.jhu.edu`

‡School of Computer Science, Carleton University, Herzberg Room 5302, 1125 Colonel By Drive, Ottawa, Ontario, Canada, K1S 5B6. E-mail: `jit@uqtr.uquebec.ca`

§Dept. of Mathematics and Computer Science, Middlebury College, Middlebury, VT 05753. E-mail: `dickerso@middlebury.edu`

regions defined either by the normal convex polygon scaling distance function or by the offseting function. We give algorithms for the following problems:

Problem 1 *Given a polygon P and a set of points S, find a placement τ that maximizes $0 \leq \delta \leq 1$, such that all points in S lie in the annulus region between P and a scaled copy of P with scale factor δ.*

Problem 2 *Given a polygon P and a set of points S, find a placement τ that minimizes $\delta \geq 1$ such that all points in S lie in the annulus region between P and a scaled copy of P with scale factor δ.*

In other words, in both problems we seek to minimize the width of an annulus region between the inner and outer polygons. In the first problem, we fix the outer polygon and maximize the inner, while in the second problem we fix the inner polygon and minimize the outer. We define the the borders of the annulus to belong to it, so by saying that a point is contained by the outer (resp., inner) annulus polygon, we implicitly mean that it can (resp., cannot) lie on it.

Note that the *scaled* polygons of the two problems correspond to the convex distance functions (also called Minkowski functionals [KN, p. 15]), which are extensions of the notion of scaling circles (in the Euclidean case) to convex polygons. There have been several papers (e.g., [CD, KW, MKS]) which explore the Voronoi diagram based on these distance functions.

However, in many applications, including those dealing with manufacturing processes, defining distance in terms of an *offset* from a polygon is more natural than scaling. This is because the relative error of the production tool (a milling head, a laser beam, etc.) is independent of the location of the produced feature relative to some artificial reference point (the origin). Thus it is more likely to allow (and expect) local errors bounded by some tolerance, rather than scaled errors relative to some (arbitrary) center. We therefore study versions of the same problems for offset as well as for scaled polygons.

We adopt the terminology of [BBDG] and denote by $I_{P,\delta}$ (resp., $O_{P,\delta}$) the inner (resp., outer) offset version of P by δ.

Problem 3 *Given a polygon P and a set of points S, find a placement τ that minimizes δ such that all points in S lie in the annulus region between P and a $I_{P,\delta}$.*

Problem 4 *Given a polygon P and a set of points S, find a placement τ that minimizes δ such that all points in S lie in the annulus region between P and a $O_{P,\delta}$.*

We will provide algorithmic solutions to all of the above problems. In particular, we give general solutions that solve both the scaled and offset versions of the problem. Our solutions will make use of the Voronoi diagrams of the set S based on the scaled or offset distance function defined by P. The solutions differ only in the computation of the Voronoi diagrams.

1.1 Definitions, Background and Related Work

We first give a formal definition of the polygon offset function. (We follow the notation of [BDG] and [BBDG].) The *outer* ε-offset of P is obtained by translating each edge $e \in P$ by ε in a direction orthogonal to e and by extending it so as to meet the translations of the edges neighboring to e. In other words, the new edge e' lies on a line parallel to that containing e and at distance ε from it on the outside of P. The edge e' is trimmed by the lines parallel and at distance ε (outside of P) of the neighboring edges of e. The *inner* offset is defined in a similar way: For each edge $e \in P$ we construct a line parallel to it and at distance ε on the inside of P. We denote the inner and outer ε-offset polygons of P by $I_{P,\varepsilon}$ and $O_{P,\varepsilon}$, respectively.

The reference [BDG] investigates basic properties of polygon-offset distance functions, with particular attention paid to how it could be used in the definition of Voronoi diagrams. It also described an efficient algorithms for computing compact representations of the nearest- and furthest-neighbor diagrams.

In [BBDG] several annulus placement problems are solved based on offset polygons. Most of the problems involved fixing δ (the width of the annulus region) and maximizing the number of points contained. Algorithms are given for simple and convex polygons, and for translation only as well as for a general placement with translation and rotation. This reference also solves some related decision problems including an on-line variant. It is suggested that this approach to annulus placement may provide a robust solution to various problems that arise in robot localization, with the presence of "noisy data" that need not be contained in the best placement.

In the case where the annulus shape is a disk, it is shown in [BBBRW] that when the inner or the outer disk is of fixed size, the placement problem can be solved more efficiently than for the general placement problem. In this paper we extend this approach to polygonal annulus regions. We seek a placement of a polygonal annulus region that contains all points, while fixing one border of the annulus and minimizing the width (or tolerance) of the region.

1.1.1 Polygon Placement and Reflection Terminology

We now define more precisely what we mean by a *placement* of a polygon. We will assume that each polygon has a fixed reference point. For scaled polygons, it is natural to use the center of scaling (without loss of generality, the origin of the coordinate system), which is assumed to be contained by the polygon. For offset polygons, the natural reference is the offseting center which is the point (or in the degenerate case the segment) to which the inner polygon collapses when the polygon is offset inward. By translating a copy of the polygon P to some point q, we mean the unique translation of P so that its reference point is translated to q. Similarly, when we speak of the *reflection* of P, we mean the rotation of P by π around its reference point. The translation of a reflection of P to a point q translates the polygon so that the reference point of the reflected copy is mapped to q.

2 Preliminary Observations

Before presenting the algorithms, we begin with several observations and lemmas that are used later to prove the correctness of the methods. Some of the following observations are simple, or are analogous to well-known facts with respect to other distance functions. We include them here for completeness.

Our algorithms make use of Voronoi diagrams based on the scaled (Minkowski) and offset distance functions. In both cases the *bisector* of two points p, q is (in the non-degenerate case) the polyline of points equidistant from both p and q. That is, it is the set of points x such that $d(p,x) = d(q,x)$. Note that we could also define it symmetrically as the set of points x such that $d(x,p) = d(x,q)$, which identifies with the first definition when we reflect the underlying polygon. Since neither distance function is symmetric, these two definitions result in different bisectors. However all of the following observations and lemmas hold regardless of which definition is used.

Observation 1 (Scale and Offset) *The bisector between two points p and q has a distinct point x (the* median *of the bisector, analogous to the midpoint of pq in the Euclidean distance function), whose distance to (from) p and q is minimum of all points on the bisector. In both directions along the bisector from x, distances to p (q) are monotonically non-decreasing.*

This claim is obvious for the Euclidean metrics. It is easily verified that this is true also for both our new distance functions.

Observation 2 (Scale and Offset) *Given a point p, a line L, and a point q on L, there exists a direction along L from q such that $d(x,p) \geq d(q,p)$ for all points x in that direction.*

Note that this is true for $d(p,x) \geq d(p,q)$ as well as for $d(x,p) \geq d(q,p)$, but the directions might be different!

[[[ADD FIG]]]   [[[HERE]]]

Figure 1: A nonstarshaped Voronoi cell (for the offset distance function)

It is proven in [BDG] that the offset polygon distance function does not satisfy the triangle inequality, and in fact, for colinear points it satisfies a reverse inequality. We show here another interesting contrast between the offset polygon distance function and the scaled polygon distance function.

Lemma 3 (Offset) *There exist nearest-neighbor Voronoi regions for the offset polygon distance function that are non-starshaped.*

Proof. We show the claim by example. Figure 1(a) shows a rectangle P which defines a convex-offset distance function. Figure 1(b) shows the respective Voronoi diagram of three points p, q, r equally spaced along a line. The Voronoi cell that corresponds to the site q is not starshaped. □

In contrast, we remind the reader of the following.

Lemma 4 (Scale) *Voronoi cells for scaled polygon distance functions are starshaped polygons. Moreover, the point site p is in the kernel of its respective cell.* □

2.1 Feasible Regions of Placement

We now give more observations and lemmas that will be used in our algorithms. In particular, we aim to bound the region in which the fixed-size polygon can be placed. That is, for Problems 1 and 3 we ask where the possible placements of the fixed-size outer polygon that contain all the points are, and for Problems 2 and 4 where the possible placements of the fixed-size inner polygon that do not contain any of the points are. We denote these regions as the sets of "feasible placements." The following observations are well-known, and have been used in several algorithm for different polygon placement problems [BDG]. In the following P be a polygon, P_R is its reflection, and $q(P_R)$ is a translation of P_R to the point q.

Observation 5 *A translation τ of P contains q if and only if $q(P_R)$ contains the point to which τ translates P.*

Generalizing this for a set S of n points we have:

Observation 6 *A translation τ of a polygon P contains all the points in a set S if and only if τ translates P to a point in the intersection of the n reflected copies of P_R translated to the points of S.*

Based on this observation, we define a *feasible region* for placements of the annulus region in Problems 1 and Problems 3. The feasible region of placements is given by the intersection of n reflected copies of P translated to each of the points in S. This feasible region, according to Observation 6, gives the set of all possible placements where the fixed outer polygon contains all the points in S. The goal then becomes to find the largest inner polygon (scaled or offset) that can be placed inside this region without containing any point of S. If the feasible region for a particular polygon and a point set is empty, then there is no solution; i.e., there is no placement of the annulus region that contains all the points because there is no placement that makes all the points contained by the (fixed) outer polygon.

Hence we pose the following problem, a solution for which we will use as a substep of our algorithms for solving Problems 1 and 3:

Problem 5 *Given a polygon P and a set S of n of points, find the intersection of n translated copies of P placed at the points in S. Report if the intersection is empty.*

There is an analogous idea for Problems 2 and 4, where we are interested in finding a placement of the inner polygon such that it does not contain any point of S.

Observation 7 *A translation τ of a polygon P contains no points in a set S of n points if and only if τ translates P into the region determined by the complement of the union of n reflected copies of P translated to the points of S.*

Based on this observation, we define a different *feasible region* for placements of the annulus region in Problems 2 and 4. The feasible region of placements is given by the complement of the union of n reflected copies of P translated to each of the points in S, plus the boundary edges of the region. This feasible region gives the set of all possible placements where the fixed inner polygon does not contain any of the points in S. The goal then becomes to find the smallest outer polygon (scaled or offset) that can be placed inside this region while containing all points in S.

3 Basic Algorithms for Polygon Placement

In this section we describe three alternate approaches to solving Problem 5.

3.1 Intersecting Copies of a Polygon

As was mentioned in Observation 6, an important primitive in many polygon placement algorithms is to compute the intersection of several translated copies of a polygon, usually reflected copies of some input polygon. It was shown in [BDP] how the prune-and-search technique of Kirkpatrick and Snoeyink can be used for (efficiently) finding the intersection points of two translated copies of a polygon. We now describe several ways to compute all the vertices of the polygon that is the intersection of n translations of some input polygon with m vertices. These algorithms use well-known techniques and are included here only for completeness and have running times of $O(nm)$, $O(n \log h + hm)$ (where h is the number of points that belong to the convex hull of the point set), and $O(n(\log n + \log m) + m)$,

3.1.1 Brute Force

A brute force approach is to start with two copies of the polygon, and compute their intersection using any of several algorithms for intersecting convex polygons in time $O(m)$. Each of the remaining $n - 2$ polygons can then iteratively be intersected with the polygon resulting from the previous step. Since each resulting intersection is still a convex polygon with at most m edges, all parallel to the edges of the original polygon, each step requires $O(m)$ time for a total of $O(nm)$. This brute force approach is not only simple, but is linear in n, and so is a good approach when m is small.

3.1.2 Using the Convex Hull

Since the intersection of the n translated copies of a polygon is exactly the set of placements of the reflected polygon containing all the points in S, and furthermore the set of placements containing the convex hull of S is exactly the set of placements containing all the points, we might make use of a preprocessing step to compute the convex hull of the point set S. Let h be the size of the convex hull of S. We compute the convex hull in $O(n \log h)$ time. We then use the previous algorithm but only on the remaining h points, for a total running time of $O(n \log h + hm)$ which is faster if $h = o(n)$ and $\log h = o(m)$.

3.1.3 A Furthest-Neighbor Voronoi Diagram Approach

A final approach makes use of the furthest-neighbor Voronoi diagram to compute the intersection of the n polygons. First we compute the furthest-neighbor Voronoi diagram of the n points. For convex distance this is done by [MKS] in time $O(n(\log n + \log m) + m)$. For convex-offset distance this is done by [BDG] in the same amount of time. Now, we follow the first step (out of three steps) of Lemma 2.1 of [DGR, p. 124], which constructs the intersection of n congruent circles in $O(n)$. Specifically, in [DGR] the authors find the portion of the intersection in each cell of the furthest-neighbor Voronoi diagram. They do that in a simple walking (on the diagram) method. All we need to observe is that they amortize the number of jumps between cells of the diagram, and obtain (for the circles case) an $O(n)$ time bound due to the complexity of the diagram. For both our distance functions we do $O(m \log n)$ such operations. From the compact representation of the diagram we will need to explicitly compute only the portions that belong to the intersection. This happens $O(n)$ times, and we spend $O(\log m)$ time for each time. The size of the accumulated output is $O(m \log n)$. In total we have an $O(n(\log n + \log m) + m)$-time algorithm. This approach is faster than the first approach if $\log n = o(m)$ and faster than the second approach if the above condition is met and in addition $h = \Theta(n)$.

4 Minimizing the Annuls for a Fixed Outer Polygon

We now address the problem of minimizing an annulus region by fixing the outer polygon and maximizing the inner polygon (Problems 1 and 3). We solve this problem for both scaled and offset polygons. In the following theorem, the Voronoi diagram is for the appropriate distance function, either scaled (Minkowski) or offset-polygon. Our algorithm is based on the following theorem and corollary.

Theorem 8 *Given a convex feasible region of possible translations of a polygon P, the largest (scaled or offset) empty polygon (containing no points in S) must be on one of the following points:*

1. *On a vertex of the nearest neighbor Voronoi diagram of S;*
2. *On an intersection of an edge of this Voronoi diagram with the feasible region;*
3. *At a vertex on the boundary of the feasible region; specifically a vertex which represents the intersection of two reflected copies of P (as opposed to an arbitrary vertex of one reflected copy of P).*

Corollary 9 *The optimal placement of the annulus region has at least 3 contact points between the set S and the inner or outer boundary of the annulus region, at least one of which is in contact with the inner boundary (the maximized inner polygon).*

Proof. Suppose for contradiction that the minimum polygon annulus region is placed in the feasible region (ensuring that all the points are contained in the outer polygon) but not in one of the three possibilities listed in the stated theorem. We show that the inner polygon could then be enlarged, thus shrinking the size of the annulus region and contradicting the assumption. By our assumption, the annulus region has been located either inside a Voronoi region or on a Voronoi edge not on the boundary of the feasible region. Suppose the polygon is placed at a point x inside a Voronoi region for a point q and not on the boundary of the feasible region. This implies that q is the nearest neighbor of x, and the definition of our distance function further implies that the maximum inner polygon defining the annulus region has q and no other point in S on its boundary. It is thus possible to move x in some direction farther away from q without x becoming closer to another point in S than it is to q. In doing so, we increase the size of the maximum polygon placed at x containing no points in S, hence we have a contradiction. Suppose next that the placement x is inside a Voronoi region but on the boundary of the feasible region. If it is on a vertex of the polygon defining the feasible region, we are in case 3. If it is on an edge of the feasible region, then by Observation 2 we can again move x in some direction farther from its nearest neighbor q, and as in the previous case we have a contradiction. Suppose finally that the best placement x is on an edge of the Voronoi diagram. Recall that an edge of the Voronoi diagram between cells VOR(p) and VOR(q) is part of the bisector between p and q defined by the appropriate distance function. By Observation 1 there is some direction along this bisector that will move x farther from its two nearest neighbors in S, again increasing the size of the largest possible polygon not containing any points. The only thing that would prevent the movement of x along this bisector is if x also sits on the boundary of the feasible region, in which case we are in case 2 and the theorem holds. □

4.1 An Algorithm

The algorithm based on Theorem 8 is simple. First we construct the feasible region. This is done by computing the intersection of n convex polygons of size m, which can be done in $O(nm)$ time. Next we construct the nearest-neighbor Voronoi diagram of S based on polygon P and the appropriate (scaled or offset) distance function. Compact representations for both diagrams (nearest- and furthest-neighbor) can be computed in $O(n(\log n + \log m) + m)$ time. Finally, we check a discrete set of at most n Voronoi vertices, n intersections between Voronoi edges and the convex feasibility region, and m vertices of the feasible region to find which allows the maximal polygon. For Voronoi vertices we can test containment in the feasibility region in $O(\log m)$ time. For Voronoi edges we can find intersections with the feasibility region in $O(\log m)$ time. To find the maximal inner polygon, we just need to know the distance to the nearest neighbor in S. For Voronoi vertices and edges, this is known. For vertices on the feasibility region we can do point location in the compact Voronoi diagram in $O(\log n)$ time. [[[BUT WON'T WE NEED $O(\log n \log m)$ FOR REALLY COMPUTING THE DISTANCE ???]]] So the total running time for checking the $O(n + m)$ possible locations is $O(n \log m + m \log n)$ time. The total running time of the algorithm including the construction of the Voronoi diagram is $O(n(\log n + m))$ time.

GILL PLEASE CHECK THIS, AND ADD/CORRECT FOR THE COMPACT VERSION OF THE VD
GILL: I AM NOT SURE I BELIEVE ALL PARTS OF THE ANALYSIS, BUT AS STATED, THE $O(nm)$ KILLS EVERYTHING, SO THE MISTAKE CANNOT BE TOO BIG. NEED TO THINK ABOUT IT.

5 Minimizing the Annuls for a Fixed Inner Polygon

In this section we address the problem of minimizing an annulus region by fixing the inner polygon and minimizing the outer polygon (Problems 2 and 4). As in the previous section, we solve this problem both for the scaled and offset polygons. In what follows, the furthest-neighbor Voronoi diagram is for the appropriate distance function, either scaled (Minkowski) or offset-polygon. Here, S is the input point set, and P is the fixed inner polygon.

Lemma 10 *The feasible region is the complement of the interior of the union of the n reflected copies of P placed at points of S.*

Proof. Follows from Observation 7. □

Note that this (polygonal) feasible region may have two different types of vertices. One type (denoted a *P-vertex*) is simply a vertex of a reflected copy of the polygon. The second vertex type (denoted an *I-vertex*) is an intersection of two copies of the reflected polygon. Note that P-vertices are all convex with respect to the feasible region (moving past a P-vertex counterclockwise around the feasible region is a left turn) whereas I-vertices are not. This yields two observations:

Observation 11 *An I-vertex is equidistant from two points in S according to the polygon distance function.*

Observation 12 *If e is an edge adjacent to a P-vertex of a feasible region, and L is the line defined by edge e, then it is possible to move some distance ϵ in both directions along L from the P-vertex without leaving the feasible region.*

Let U be the union of n reflected copies of P placed at the points of S. By Observation 11, placing P at an I-vertex of the boundary of U results in P having at least two points of S on its boundary. This means that an I-vertex of U is on an edge or vertex of the nearest-neighbor Voronoi diagram of S. Furthermore, since each edge of this diagram corresponds to two points in S, and the two copies of the reflected polygon associated with those points intersect in at most two points, each Voronoi edge can be associated with at most two I-vertices. Since the Voronoi diagram (in its compact representation) has $O(n)$ edges (each of which may be a polyline of complexity m) and $O(n)$ vertices, U can have at most $O(n)$ I-vertices. (We note for contrast that n polygons can certainly intersect in $\Theta(n^2)$ points, but only $O(n)$ of these points may be I-vertices of the boundary of U. The rest are interior to U.) There may also be at most $O(nm)$ P-vertices. Therefore, the complexity of the boundary of U is $O(nm)$. The polygon U can be computed in $O(nm \log(nm))$ time by using a divide-and-conquer approach. We summarize with the following:

Lemma 13 *The union U of n reflected copies of P has $O(n)$ I-vertices, $O(nm)$ P-vertices and the complexity of its boundary is $O(nm)$. It can be computed in $O(nm \log(nm))$ time.*

Given an edge e of the furthest-neighbor Voronoi diagram of S, we refer to the two points of S equidistant from e as the *generators* of e. We refer to a point in the feasible region as a *feasible point.*

Theorem 14 *The center of the smallest enclosing polygon is in the feasible region on one of the following:*

1. *A vertex of the furthest-neighbor Voronoi diagram;*
2. *A point on an edge of the furthest-neighbor Voronoi diagram provided it is the median of the bisector of its generators (see Observation 1);*
3. *the intersection point of an edge e of the furthest-neighbor Voronoi diagram and the boundary of the feasible region that is closest to the median of the bisector of the generators of e; or*
4. *an I-vertex of the feasible region (see Observation 2).*

Proof. Assume that the center of the minimum polygon annulus lies in the feasible region. We will show that if the center is not in one of the four places listed in the theorem, then the outer polygon can always be shrunk and thus contradict our assumption.

The center c lies in the feasible region and must lie either in the interior of a Voronoi cell, on an edge of the Voronoi diagram, or on an a vertex of the Voronoi diagram. If c lies on a vertex of the Voronoi diagram, we are in case 1. If c lies on an I-vertex of the feasible region, we are in case 4.

Suppose c is on an edge e of the Voronoi diagram and in the interior of the feasible region. If c is on the median of the bisector of the generators of e, then we are in case 2. Otherwise, moving c towards the centroid reduces the size of the outer polygon, which is a contradiction.

Suppose c is on an edge e of the Voronoi diagram and on an edge of the feasible region. If c is on the median of the bisector of the generators of e, then we are in case 2. If we can move c towards the median while remaining in the feasible region, then we reduce the size of the outer polygon, which is a contradiction. If we cannot move c towards the median while remaining in the feasible region, then we are in case 3.

Suppose that the center c lies in the interior of the furthest-neighbor Voronoi cell of a point $q \in S$ and in the interior of the feasible region. This implies that q is on the boundary of the outer polygon. Furthermore, no other point of S lies on the boundary of the outer polygon; therefore, it is possible to reduce the size of the outer polygon by moving c towards q, hence a contradiction.

Suppose that c lies in the interior of the furthest-neighbor Voronoi cell of a point $q \in S$ and is on the boundary of the feasible region. If c is on an I-vertex, we are in case 4. If c is on an edge, then by Observation 2 there is a direction along which we can move c while reducing the size of the outer polygon, contradicting our assumption. Similarly, if c is on a P-vertex, then by Observation 12 there is also a direction along which we can move c while reducing the size of the outer polygon. □

Corollary 15 *The optimal placement with fixed inner polygon and a minimum outer polygon has either:*

1. *two points on the outer polygon on a diameter;*
2. *three points on the outer polygon;*
3. *two points on the outer polygon and one on the inner polygon; or*
4. *one point on the outer polygon and two on the inner polygon.*

Proof. Follows from Theorem 14. □

5.1 Algorithm

Theorem 14 implies a natural approach to computing the minimum polygon annulus. First, compute all the possible locations for the center as listed in the theorem. Second, for each location, compute the size of the annulus. Output the smallest of these annuli. Since the true center must lie in one of these locations, we can find the minimum polygon annulus.

First, we need a way of deciding if a point x is in the feasible region. To do this we compute the nearest-neighbor Voronoi diagram of S based on reflected polygon P and the appropriate (scaled or offset) distance function. We preprocess the diagram for planar point location. Once we know the closest point of S to x, we can determine whether it is in the feasible region or not. We thus need $O(nm\log(nm))$ time for preprocessing and $O(\log(nm))$ for each query.

There are $O(n)$ vertices in the furthest-neighbor Voronoi diagram. Containment in the feasible region can be verified in $O(n\log(nm))$ time. There are $O(n)$ medians in the furthest-neighbor Voronoi diagram and they can be verified in $O(n\log(nm))$ time.

There are $O(n)$ vertices on the boundary of the feasible region (see Lemma 13). All of these vertices can be verified in $O(n\log(nm))$ time.

To compute the intersection point of an edge e of the furthest-neighbor Voronoi diagram and the boundary of the feasible region that is closest to the median of the bisector of the generators of e, we note that each edge of the diagram is a polygonal chain of at most m segments. If the median is on e and is feasible, then no other candidate on e is smaller (distance-wise). Therefore, we need only consider the edges where the median is not feasible. In this case, we direct the segments of the edge towards the median. For each segment we need only the first intersection with the feasible region. This can be viewed as a ray shooting query. For each directed segment $\vec{st}$ directed from s to t, we want the intersection of $\vec{st}$ with U closest to s. Preprocessing U for ray shooting queries is too costly. Instead, we perform two plane sweeps to compute the intersections between U and the directed segments, one for the segments directed to the left and one for the segments directed to the right. After the first intersection for a given segment is found, we remove it from the event queue. Therefore, each segment is processed at most twice, once when it is placed in the queue and once for its first intersection. Since there are $O(nm)$ segments, and the boundary of U has $O(nm)$ segments, each of the two sweeps takes $O(nm\log(nm))$ time.

All of the candidates are generated and verified in $O(nm\log(nm))$ time. We conclude with the following:

Theorem 16 *The minimum polygon annulus with fixed inner polygon can be computed in* $O(nm\log(nm))$ *time.*

6 An Alternate Approach

[[[GILL, THE FOLLOWING IS UP TO YOU TO POLISH, CLARIFY, REVISE. SOME OF THE TEXT APPEARED BEFORE.]]]

We describe a general deterministic $O(n \log n \log m + m)$-time algorithm for solving all the optimization problems 1–4 at once: Given a set S of n points and a convex polygon P with m vertices, find the minimum-width annulus of P that covers S, either when the inner border or the outer border of the annulus is fixed, and for both scaling and offseting. For this purpose we use, as explained above, the distance-function $\mathcal{D}_P$ that corresponds to P and compute the nearest- and furthest-neighbor Voronoi diagrams of S with respect to $\mathcal{D}_P$. For both distance functions (scaling and offseting) compact representations of both diagrams (nearest- and furthest-neighbor) can be computed in $O(n(\log n + \log m) + m)$ time. Next we use the method of [DGR] (where the authors minimize the width of a *circular* annulus) and consider the overlay of the two diagrams. (Note that in Section 3.1.3 we use only a portion of this method.) As is well-known, the center of the minimum-width annulus that contains S is either a vertex of one of the two diagrams (possibly a vertex at infinity in the furthest-neighbor diagram) or a point of intersection between the two diagrams. Given a specific value of δ, we place δ-annuli (when either the inner or outer border is fixed) centered at all the points of S and observe (like in [DGR]) the overlay for determining whether the intersection of all annuli is nonempty. (The intersection contains the loci of all feasible placements of the annulus so that it covers S.) This step takes $O(n \log m)$ time in all cases. Finally, a parametric-searching algorithm (in its simplest form) is applied for optimizing (minimizing) the value of δ for which the intersection of all the annuli is nonempty. Over all, the whole procedure requires $O(n \log n \log m + m)$ time.

7 Summary and Open Problems

[[[FILL IN]]] [[[To be filled in by Matthew later this week? But feel free to add any important thoughts you might have, especially open questions.]]]

References

[BBDG] G. Barequet, A. Briggs, M. Dickerson, and M.T. Goodrich, Offset-polygon annulus placement problems, *Proc. 5th Workshop on Algorithms and Data Structures*, Halifax, Nova Scotia, Canada, 1997, to appear.

[BDG] G. Barequet, M. Dickerson, and M.T. Goodrich, Voronoi Diagrams for Polygon-Offset Distance Functions, *Proc. 5th Workshop on Algorithms and Data Structures*, Halifax, Nova Scotia, Canada, 1997, to appear.

[BDP] G. Barequet, M. Dickerson, and P. Pau, Translating a convex polygon to contain a maximum number of points, *Proc. 7th Canadian Conf. on Computational Geometry*, Québec City, Québec, Canada, 1995, 61–66; full version to appear in: *Computational Geometry: Theory and Applications.*

[BBBRW] M. de Berg, P. Bose, D. Bremner, S. Ramaswami, and G. Wilfong, Computing Constrained Minimum-Width Annuli of Point Sets, to appear in C.A.D. (special issue on manufacturing), also *Proc. 5th Workshop on Algorithms and Data Structures*, Halifax, Nova Scotia, Canada, 1997, to appear.

[CD] L.P. Chew and R.L. Drysdale, Voronoi diagrams based on convex distance functions, Technical Report PCS-TR86-132, Dept. of Computer Science, Dartmouth College, Hanover, NH 03755, 1986; Preliminary version appeared in: *Proc. 1st Ann. ACM Symp. on Computational Geometry*, Baltimore, MD, 1985, 324–244.

[DGR] C.A. Duncan, M.T. Goodrich, and E.A. Ramos, Efficient approximation and optimization algorithms for computational metrology, *Proc. 8th Ann. ACM-SIAM Symp. on Discrete Algorithms*, New Orleans, LA, 1997, to appear.

[F] L. W. Foster, GEO-METRICS II: The application of geometric tolerancing techniques. Addison-Wesley, 1982.

[GMR] L. Guibas, R. Motwani, and P. Raghavan, The robot localization problem, in: *Algorithmic Foundations of Robotics*, A.K. Peters, Ltd., 1995, 269–282.

[KN] J.L. Kelley and I. Namioka, Linear Topological Spaces, Springer Verlag, 1976.

[KS] D. Kirkpatrick and J. Snoeyink, Tentative prune-and-search for computing fixed-points with applications to geometric computation, *Fundamental Informaticæ*, 22 (1995), 353–370.

[Kl] R. Klein, Concrete and abstract Voronoi diagrams, *LNCS 400*, Springer Verlag, 1989.

[KMM] R. Klein, K. Mehlhorn, and S. Meiser, Randomized incremental construction of abstract Voronoi diagrams, *Computational Geometry: Theory and Applications*, 3 (1993), 157–184.

[KW] R. Klein and D. Wood, Voronoi diagrams based on general metrics in the plane, *Proc. 5th Symp. on Theoretical Computer Science*, 1988, 281–291, *LNCS 294*, Springer Verlag.

[MKS] M. McAllister, D. Kirkpatrick, and J. Snoeyink, A compact piecewise-linear Voronoi diagram for convex sites in the plane, *Discrete Computational Geometry*, 15 (1996), 73–105.

[MMO] K. Mehlhorn, S. Meiser, and Ó'Dúnlaing, On the construction of abstract Voronoi diagrams, *Discrete Computational Geometry*, 6 (1991), 211-224.

[MMR] K. Mehlhorn, S. Meiser, and R. Rasch, Furthest site abstract Voronoi diagrams, Technical Report MPI-I-92-135, Max-Planck-Institut für Informatik, Saarbrücken, Germany.