Efficient Proximity Probing Algorithms for Metrology

Aviv Adler[1], Fatemeh Panahi[2], A. Frank van der Stappen[3] and Ken Goldberg[4]

***Abstract*—Metrology, the theoretical and practical study of measurement, has applications in automated manufacturing, inspection, robotics, surveying, and healthcare. The *geometric probing* problem considers how to optimally use a probe to measure geometric properties. In this paper, we consider a *proximity probe* which, given a point, returns the distance to the boundary of the nearest object. When there is an unknown convex polygon P in the plane, the goal is to minimize the number of probe measurement needed to exactly determine the shape and location of P. We present an algorithm with upper bound of $3.5n + k + 2$ probes, where n is the number of vertices and $k \leq 3$ is the number of acute angles of P. The algorithm requires constant time per probe, and hence $O(n)$ time to determine P. We also address the related problem where the unknown polygon is a member of a known finite set Γ and the goal is to efficiently determine which polygon is present. When m is the size of Γ and n' is the maximum number of vertices of any member of Γ, we present an $O(n'm)$ algorithm with an upper bound of $2n+2$ probes.**

Note to Practitioners: ***Abstract*—This paper was inspired by the problem of using a sensor with low-dimensional output, such as a range sensor, to determine the shape of an object, which is typically too complex for a single measurement to characterize. Existing approaches to shape measurement generally use sensors with high-dimensional output, such as cameras, or use low-dimensional sensors to measure fixed points, such as with Scanning Probe Microscopy (SPM). It is shown here that by employing an algorithm that uses the results of previous measurements to determine how the next measurement is taken, a non-directional range sensor can be used to efficiently and exactly determine the shape of a convex polygon. This suggests an alternative approach to obtaining information on the shape of an object in cases where low-dimensional sensors are more accurate, faster, or cheaper than their counterparts.**

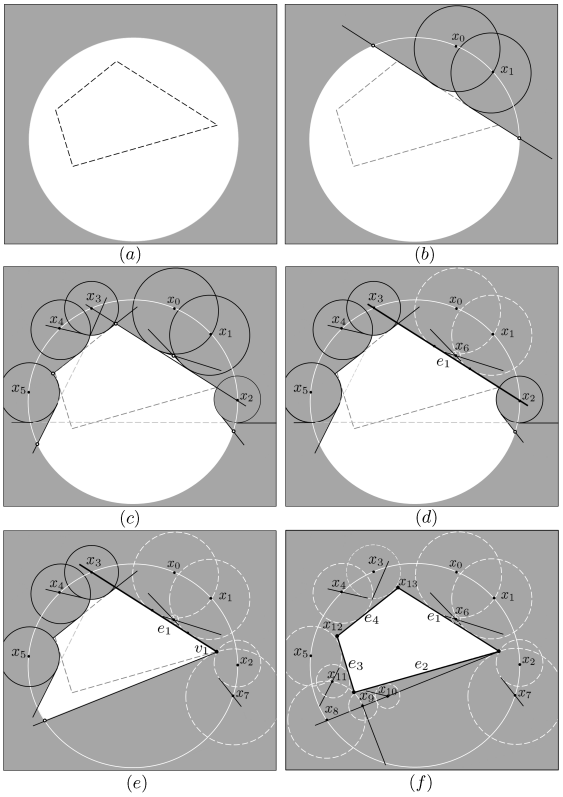

Fig. 1. An illustration of Algorithm 1 probing to determine the shape and location of a polygon with $n = 4$ vertices and $k = 1$ acute angle. This instance is solved with 14 ($\leq 3.5n + k + 2$) probes.

I. Introduction

METROLOGY has applications in manufacturing, inspection, robotics, surveying, and healthcare ([5], [6]). An important aspect of metrology is the problem of how to most efficiently use a given measurement device, or *probe*, to determine properties of the environment. When the measurement device and object of interest are geometric, the problem of obtaining information about the object through repeated use of the device is known as *geometric probing*. One version of this problem is to deduce the shape of an unknown object using as few probes as possible.

A number of researchers have developed efficient algorithms for probing convex polytopes. A pioneering paper, by Cole and Yap [10], studied probes which travel along a straight line chosen by the algorithm and stop when they collide with the polygon (later referred to as *finger probes* [1], [11], [12], [13]). Different probe types and algorithms were presented by Dobkin et al. [12], and generalized to higher-dimensional cases. These include the finger probes previously studied by Cole and Yap; *hyperplane probes*, which consist of a hyperplane (whose angle is chosen by the algorithm) which sweeps over the whole space and stops when it collides with the polygon; and *silhouette probes* (also called *projection probes* [17]), which provide the projection of the polygon onto

[1]Aviv Adler is with the Department of Mathematics, Princeton University, Princeton NJ, USA

[2]Fatemeh Panahi is with the Department of Information and Computing Sciences, Utrecht University, Utrecht, Netherlands. She is supported by the Netherlands Organization for Scientific Research (NWO)

[3]A. Frank van der Stappen is with the Department of Information and Computing Sciences, Utrecht University, Utrecht, Netherlands

[4]Ken Goldberg is with the Departments of IEOR and EECS, UC Berkeley, Berkeley CA, USA

This work has been supported in part by by the U.S. National Science Foundation under Award IIS-1227536: Multilateral Manipulation by Human-Robot Collaborative Systems, and by grants from Google, Cisco, and Flextronics.

a chosen subspace. Other probes which have been studied for convex polygons include *x-ray probes* ([1], [11], [16]), which measure the length of intersection between a chosen line and the unknown polygon, and *half-plane probes* [18], which measure the area of intersection between a chosen half-plane and the unknown polygon. A related problem, identifying a convex polygon from a known set, was considered by Goldberg and Rao in [15] using diameter probes.

Further work on the theme of determining complex geometric information with multiple relatively simple sensing devices can also be found in recent work on geometric pursuit-evasion games [9].

In this paper, we consider *proximity probes* which, given a point, return the distance to the boundary of the object in question, which we assume to be a convex polygon. This is an expanded and updated version of our conference paper [20], where we used proximity probes to determine the shape of an unknown convex polygon. We refer to this as Problem 1 and present Algorithm 1 to solve it, as illustrated in Fig. 1. This journal paper also includes an algorithm and analysis or *Problem 2*: identifying a convex polygon from a known set.

Metrology is used in many automation applications, such as semiconductor manufacturing and MEMS to inspect the shape of etched silicon structures ([2], [19]) using scanning probe microscopy (SPM) ([2], [3], [4]), and virtual metrology (VM) ([7], [8]), which uses measurements of parameters during the production of wafers to statistically predict production properties. Proximity probes could also be relevant for sonar sensors in 2D and 3D (underwater) applications.

The remainder of the paper is organized as follows. In Section II, we introduce Problem 1 and the definitions necessary for the algorithms. In Section III, we present our algorithm for Problem 1 and analyze its complexity per probe; we also present a complete example of Algorithm 1 for a simple polygon P. In Section IV we show an upper bound on the number of probes needed by Algorithm 1. Section V deals with Problem 2, presenting and analyzing an algorithm meant to minimize the number of probes necessary to identify P from the set Γ. Finally, in Section VI, we summarize our results and discuss future work.

II. Problem 1: Formulation and Preliminaries

We assume that all points and objects lie in the plane and that all positioning and measurements are exact.

For any two points or closed sets of points a, b, $\mathrm{dist}(a, b)$ denotes the Euclidean distance between a and b; for a lclosed subset S of the plane, $\partial(S)$ denotes its boundary, $\mathrm{Int}(S)$ denotes its interior, $\bar{S}$ denotes the closure of its complement (so both S and $\bar{S}$ contain $\partial(S)$), and $\mathrm{Conv}(S)$ denotes its convex hull. We also define *zero-disk* to mean a disk containing only its center.

In addition, for any disk of positive radius C and point z on its boundary, we define $L(C, z)$ to be the line tangent to C at z. We also define $H(C, z)$ to be the half-plane bordered by $L(C, z)$ which contains C, and $\bar{H}(C, z)$ to be the half-plane bordered by $L(C, z)$ which does not contain C.

A. Problem Formulation

Let P be an unknown convex polygon with n vertices and edges contained in a known disk D, and let the *probing function* f_P be defined over the the plane as

$$f_P(x) = \begin{cases} \mathrm{dist}(x, P) & : x \notin \mathrm{Int}(P) \\ -1 & : x \in \mathrm{Int}(P) \end{cases}$$

The probing algorithm is not explicitly given this function, but is allowed to call it as many times as necessary to find P exactly; the goal of this paper is to find an algorithm which minimizes the upper bound of probes necessary (and allows the next probe to be efficiently computed at each step). The points x for which it calls the function f_P are the *probes*, and the disks of radius $f_P(x)$ centered at these points are the *probe disks*, abbreviated as *p-disks* (by convention, if $f_P(x) = -1$ then no disk is produced). Every p-disk is by definition incident to P at exactly one point.

B. Condensed Probe Disks

Suppose we have two (distinct) p-disks C_a, C_b such that $C_a \subset C_b$. Since they both must be incident to P at exactly one point, they must be incident to P at the same point (otherwise it is impossible for one to contain the other); this point will by definition be the only point in $\partial(C_a) \cap \partial(C_b)$, which we call $p_{a,b}$. Furthermore, P must be interior disjoint with the half-plane $H(C_b, p_{a,b})$ since P is convex, $p_{a,b} \in C_b, P$, and C_b is not a zero-disk (because $\emptyset \neq C_a \subset C_b$). We thus define the *condense* operation on C_a, C_b which outputs $p_{a,b}$ as a zero-disk and associates with it the half-plane $H(C_b, p_{a,b})$; the products of this operation are called *condensed probe disks* (abbreviated as *cp-disks*). Furthermore, any p-disks which neither contain nor are contained by other p-disks (and so cannot be used by the condense operation) are also considered to be cp-disks. Note that cp-disks, like p-disks, must have exactly one intersection point with P (since cp-disks are either p-disks or zero-disks produced by the condense operation). If C^* is a cp-disk produced by the condense operation (and hence C^* is a point), we let $H(C^*)$ be its associated half-plane and $L(C^*)$ be the line bordering $H(C^*)$.

A small note: it is possible for a p-disk C_a to be contained in several other p-disks, none of which are contained in each other; however, this can only happen when C_a is a zero-disk and also at a vertex of P. In these cases, C_a will be condensed with every disk containing it to produce multiple condensed cp-disks.

C. Clockwise Ordering of cp-Disks

For any cp-disk C^*, let $p(C^*)$ be its intersection point with P. We note that by imposing a clockwise direction on the boundary of P, we can impose a clockwise order on the set of $p(C^*)$ for all cp-disks C^* (it is possible for two cp-disks to have the same contact point on P; but this can only happen on vertices of P). This then imposes a clockwise (cyclic) ordering on the set of cp-disks, where if multiple cp-disks happen to have the same contact point with P, they can be ordered by the lines tangent to them at the common contact point (a zero-disk C^*_{zero} produced by the condense operation is considered to have the line $L(C^*_{zero})$ as its tangent; a zero-radius cp-disk not produced by the condense operation cannot share a contact point with another p-disk or cp-disk since it would be contained by the other disk and hence not be a cp-disk by definition).

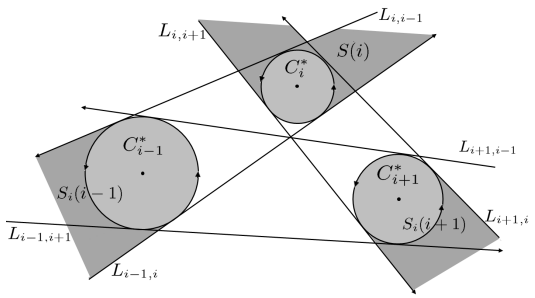

Fig. 2. An example of three consecutive cp-disks, which shows the shadow sets of C_{i-1}^*, C_{i+1}^* with respect to C_i^* and the neighbor-shadow set of C_i^*.

From now on we will attach indices to the cp-disks indicating their order. Specifically, we will let X be the ordered set of cp-disks, and implicitly label the disks in X as $C_1^*, C_2^*, ..., C_\alpha^*$. Since the ordering of cp-disks is cyclic, we assume that additions and subtractions on indices are performed modulo the number of disks.

Remark: It should be noted that in general, given an arbitrarily probed set of cp-disks, prior knowledge of P is necessary to deduce their exact ordering by the above criteria, and thus the ordering cannot be used by the algorithm. However, we will show that our algorithm chooses probes in such a way that this labeling can always be determined exactly without any prior knowledge of P.

D. Shadow Sets

Suppose we have two cp-disks C_i^*, C_j^*; for both we define a counterclockwise direction on their boundary. We define the lines $L_{i,j}$ and $L'_{i,j}$ to be the lines tangent to both C_i^* and C_j^* such that

- for both lines, C_i^* and C_j^* lie on the same side
- $L_{i,j}$ is given a direction coinciding with the counterclockwise direction imposed on the two cp-disks, while $L'_{i,j}$ is given a direction opposing the counterclockwise direction
- Both lines, in their given directions, intersect C_i^* before C_j^*.

Note that $L_{i,j}$ is the same line as $L'_{j,i}$ but with the opposite direction imposed on it.

We now define the rays $l_{i,j}, l'_{i,j}$ to be the rays respectively lying on $L_{i,j}, L'_{i,j}$ with their sources at the respective points of tangency with C_j^*. For a ray l, we define $H_{right}(l)$ to be the quarter plane lying directly to the right of the ray, and $H_{left}(l)$ is analogously defined.

We then define the *shadow set* cast by C_j^* on C_i^* as

$$S_i(j) = C_j^* \cup (H_{left}(l_{i,j}) \cap H_{right}(l'_{i,j}))$$

This set cannot contain any point of $\mathrm{Int}(P)$, since P cannot have any point in $\mathrm{Int}(C_j^*)$, must be incident to C_i^*, and is convex; similarly, P cannot contain any point of $\mathrm{Int}(S_i(j))$. Fig. 2 illustrates shadow sets for three consecutive cp-disks.

Note that the boundary of C_j^* is partly on the boundary of $S_i(j)$ and partly in its interior; since P cannot contain any point of $\mathrm{Int}(S_i(j))$, its point of intersection with C_j^* must be on the part of $\partial(C_j^*)$ which is also on $\partial(S_i(j))$. We call this the *feasible arc* C_i^* imposes on C_j^* and denote it $\zeta_i(j)$.

E. The Neighbor-Infeasible Region

We first define the set $S_{i-1}(i) \cup S_{i+1}(i)$ to be the *neighbor-shadow set* of C_i^* (abbreviated as *ns-set*), denoted as $S(i)$ for convenience (Fig. 2). Similarly, we define the set $\zeta_{i-1}(i) \cap \zeta_{i+1}(i)$ to be the *neighbor-feasible arc* of C_i^*, abbreviated as *nf-arc*; we denote it as $\zeta(i)$ for convenience.

For cp-disks produced by the condense operation, we instead use $S(i)$ to refer to the half-plane $H(C_i^*)$. Note that since no cp-disk can be contained in $\mathrm{Int}(H(C_i^*))$, $H(C_i^*)$ is a superset of $S_{i-1}(i) \cup S_{i+1}(i)$ for these cp-disks.

The *neighbor-infeasible region* R can now be defined as

$$R = \bigcup_{i=1}^{m} S(i) \cup \bar{D}$$

Intuitively, for each C_i^*, we simply take the ns-set of C_i^*, the half-planes associated with all cp-disks generated by the condense function, and the complement of D (the disk which we were initially given as containing P). Since R is composed of these pieces, P must be entirely contained in (the closure of) the complement of R.

We will show later that our algorithm behaves in such a way that the complement of R (the *neighbor-feasible region*) is a single connected piece; we therefore will assume it to be the case now. The boundary of R will then be naturally split into the following two basic types of pieces, which we call *sections*:

1) arcs of the boundary of D
2) connected subsets of the boundaries of the sets $S(i)$; we denote $\partial(S(i)) \cap \partial(R)$ as $\partial_R(S(i))$

Note that the second type of section has two possibilities:

a) if C_i^* was not produced through the condense operation, $\partial_R(S(i))$ is naturally split into at most three pieces, namely

- the nf-arc $\zeta(i)$
- a segment of the ray $l_{i-1}(i)$ (which we will denote $l(i)$ for convenience)
- a segment of the ray $l'_{i+1}(i)$ (which we will denote $l'(i)$ for convenience)

The other two pieces of the boundary of $S(i)$, namely $l'_{i-1}(i)$ and $l_{i+1}(i)$ cannot lie on $\partial(R) = \partial(\bar{R})$ because P in that case would impose the wrong ordering of the cp-disks.

b) if C_i^* was produced through the condense operation, $\partial_R(S(i))$ is just $L(C_i^*)$

Remark: Although the neighbor-infeasible set R is interior disjoint with P by definition, it is not necessarily the case that it is the full set of all infeasible points, i.e. the points which, given the p-disks, can't be contained in P.

F. Confirmation of Vertices and Edges, and the Query Set

We say a point v is *confirmed* if by considering X it can be shown that v is a vertex of P, and we say a line L is confirmed if by considering X it can be shown that l contains an edge of P; an edge e of P is also referred to as confirmed if the line extending it is confirmed. Any vertices or edges of P which are not confirmed are called *unconfirmed*. The list of confirmed vertices is denoted V_c and the list of confirmed edges is denoted E_c.

Now we consider $\partial(R)$, as described above as a collection of pieces of the boundaries of the $S(i)$ and D. Since $\partial(R)$ is continuous, there will be points which lie on more than one of the specified sections. Some of these points will lie on confirmed vertices or edges of P. The ones which do not will be called the *query set* Q, from which we will always probe (except for the very first probe). Furthermore, we define the *preferred query set* Q^* to be the subset of Q which does not contain any intersection points between two p-disks.

To confirm a vertex or line, we need to count how many p-disks are incident to it; an easy way to compute this from the set of cp-disks is to count the number of cp-disks tangent to L, double-counting those produced by the condense operation (since they correspond to two p-disks). Note that this means the number of cp-disks involved is at most the number of p-disks involved.

Furthermore, note that the set of all cp-disks passing through a point or tangent to a line must be consecutive.

We can confirm a point v as a vertex of P in these cases:

- if 3 p-disks pass through v
- if v is probed and $f_P(v) = 0$ (this implies that $v \in \partial(P)$; the fact that v was in Q, which is a necessary condition for being probed by the algorithm, means that v sits in a corner of R and thus cannot be in the middle of an edge of P, meaning it must be a vertex of P)
- if a segment of (confirmed or unconfirmed) line L on $\partial(R)$ and two p-disks touch v
- if segments of (confirmed or unconfirmed) lines L, L' on $\partial(R)$ and one p-disk touch v

If we confirm a vertex on a previously unconfirmed line, we can automatically confirm the line as well.

Additionally, we can confirm a line L as containing an edge of P if L is tangent to three p-disks. The cp-disks representing these three p-disks will necessarily be consecutive in X because they all have contact points with P on the same edge (and no other cp-disks will have contact points in the interior of this edge, since in that case L would have been confirmed earlier), and so given a cp-disk C_i^* we just need to check the three consecutive triples containing it.

In addition, if line L is tangent to two p-disks and passes through the intersection point v of the boundaries of two other p-disks, then both L and v can be confirmed. Also, if L is tangent to a p-disk and goes through the intersections of the boundaries of two different pairs of p-disks (call these points v_1, v_2), we can confirm v_1, v_2 and L.

Whenever a vertex v is confirmed, it automatically implies that probing v would return $f_P(v) = 0$; this means we can place a p-disk there *without* explicitly probing it, and perform the condense operation with any existing p-disks which happen to contain v. Since they all have the same contact point v with P, they will be consecutive in X, and later on we will show that there cannot be more than 3 such disks for any v, so this process takes constant time.

Similarly, whenever a line L is confirmed, we always have at least one, and often more than one, cp-disks tangent to L; at each tangent point x we know that $f_P(x) = 0$ so we may place a p-disk there without actually executing the probe function, and perform the condense operation with the original tangent p-disk to create a new cp-disk. Since an edge is always confirmed if it is incident to 3 cp-disks, the number of condense operations we need to perform is at most 3 for each confirmed line; thus this process takes constant time.

Remark: Thanks to the fact that we use the condense operation when we confirm vertices and edges (without requiring new probes), the lines corresponding to these condense operations are automatically incorporated into $\partial(R)$.

III. Algorithm 1

We now present an efficient algorithm for solving the probing problem described in Section II. The algorithm maintains the circular ordered list X of cp-disks, sorted in clockwise order of their intersection point with P around $\partial(P)$, an algebraic representation of the neighbor-infeasible region R, lists of the confirmed vertices (V_c) and edges (E_c) of P, and representations of the query set Q and preferred query set Q^*. We present it in two parts: the first dealing with how to generate the next probe given X, R, V_c, E_c, Q, and Q^*, and the second dealing with how to update these objects given a new probe result. The algorithm terminates once (a) at least one vertex and edge have been confirmed and (b) every confirmed vertex is on two confirmed lines and every confirmed line contains two confirmed vertices.

In addition, a some extra information and pointers will be stored in these lists in order to allow the algorithm to execute all the steps in constant time, most notably pointers in Q for each element which point to its neighbors (in both X and Q); however, we omit the exact details.

A. Algorithm for Generating New Probes

The algorithm for generating new probes is divided into two distinct phases (preceded by a one-probe initialization): in Phase 1, we probe arbitrarily from the preferred query set Q^* when possible; when it is not, we choose instead from Q (both Q^* and Q are by definition a subset of the boundary of R) until some edge is confirmed; in Phase 2 (once an edge is confirmed), we probe points designed to confirm the vertices and edges of P in (roughly) clockwise order.

We also add the following definitions for reference in the algorithm:

- the first edge of P to be confirmed is denoted e_1 (i.e. the edge contained by the first line confirmed)
- the edges and vertices of P in clockwise order are $e_1, v_1, e_2, v_2, ..., e_n, v_n$
- for any edge e_i, we let L_i^* be the line containing e_i; note that it is the lines, not the edges themselves, which are directly confirmed by the algorithm
- at any given step of the algorithm, we let t be the largest index such that $e_1, v_1, e_2, v_2, ..., e_{t-1}$ are all confirmed (we can determine t from E_c and V_c without any extra direct knowledge of P)
- l is a ray originating on some point on e_{t-1} which we know is in P (for all $t > 2$, we use v_{t-2}; otherwise we use the contact point of some p-disk with the confirmed line containing e_{t-1}) and extending e_{t-1} in the direction coinciding with the clockwise direction around the boundary of P (this direction is also determinable from E_c and X without any extra knowledge of P)

- for any set S and ray γ, let $\rho(\gamma, S)$ be the furthest point along γ which is also in S

At the start, X, V_c, E_c, Q, Q^* are empty and $R = \bar{D}$, so we simply probe from an arbitrary point on the boundary of D. Because $P \subset \mathrm{Int}(D)$, this disk will have positive radius; because it is the first p-disk, it cannot be condensed and is thus also a cp-disk. In addition, it will not have any neighbors in X since it is the only disk in X, so its shadow set is by convention defined to be itself. Thus, R is simply the union of this disk and the complement of D, and the boundary of R will consist of an arc of this disk plus an arc of D. Hence, by definition, Q consists of the two points of intersection between the boundaries of D and the first cp-disk.

Algorithm Steps:

1) While no line has been confirmed, at each step we check if Q^* has at least one element. If it does, we choose an arbitrary point $x \in Q^*$ and probe it; if not, we choose an arbitrary point $x \in Q$ and probe it.
2) Once a line has been confirmed, we let the edges and vertices of P, the index t, and the ray l be defined as above. We repeat the following step until both e_t and v_{t-1} are confirmed (at which point, by definition, the index t increases, and we start Phase 2 again; we terminate once v_t is confirmed on e_1).
   Let $x = \rho(l, \bar{R})$; an intuitive idea of x is that it is the furthest clockwise point on the confirmed line containing e_{t-1} which is not in the neighbor-infeasible region R. We note then that since x is the furthest point on $l \subset L^*_{t-1}$, it must also be on some other object on the boundary of R; hence, either $x \in V_c$ (if x happens to be v_{t-1} and is already confirmed) or $x \in Q$.
   If $x \in Q$ then it must be both on L^*_{t-1} and some other piece of the boundary of R. In particular, it can be on the following
   - an nf-arc $\zeta(i)$ of some
   - another confirmed line
   - an unconfirmed line, either corresponding to the output of a condense function or incident to two (consecutive) cp-disks
   - the boundary of D

   We then do the following:
   a) if $x \in V_c$, call *Next Edge*
   b) if $x \in Q$ and $x \notin \zeta(i)$ for all i, probe x
   c) if $x \in Q$ and $x \in \zeta(i)$ for some i, then it is one endpoint of the arc $\zeta(i) \cap \partial(R)$; let x' be the other endpoint. This point by definition will either be x's neighbor in Q or will be an endpoint of $\zeta(i)$, and hence is retrievable in constant time

Remark: Although in Phase 1 we are allowed to probe any $x \in Q^*$ (or, if Q^* is empty, any $z \in Q$) at each step, if we wish to minimize the time complexity of choosing the next probe at each step, we need a retrieval method which produces a member of Q^* or Q in constant time; having either a stack or a queue as an additional data structure for Q^* and Q are the most natural ways of achieving this.

The Next Edge Procedure

This procedure is called when e_{t-1} and v_{t-1} are both confirmed but e_t is not confirmed. Let us consider the set of cp-disks incident to v_{t-1}; they will be consecutive in X, and will have been produced by the condense function (at the moment that v_{t-1} was confirmed). Let C^*_i be the last cp-disk among them; let $\mathrm{N}_Q(C^*_i)$ be C^*_i's next neighbor (in the clockwise direction) in Q. We then probe $\mathrm{N}_Q(C^*_i)$ (updating the maintained information as we go so i and $\mathrm{N}_Q(C^*_i)$ can change after each probe) until the next edge is confirmed, at which point t can be updated and we return to the main loop of Phase 2. We note that $\mathrm{N}_Q(C^*_i)$ is actually the point on $L(C^*_i)$ furthest from v_{t-1}.

The Pseudocode

For the pseudocode, we introduce some extra notation and functions (and show, where necessary, that these functions can be computed efficiently). We define the sets E^*_c, V^*_c to be respectively the subset of E_c consisting of those lines which do not contain two points from V_c, and the subset of V_c consisting of those points which are not contained by two lines from E_c. Intuitively, E^*_c and V^*_c consist of the confirmed lines and vertices whose adjacent vertices and lines, respectively, have not been confirmed yet. These sets are easy to maintain with flags attached to both E_c and V_c.

For the case (c) of Phase 2, if $x \in \zeta(i)$, then we denote the other endpoint of the arc $\zeta(i) \cap \partial(R)$ as $q(x)$.

For any $x \in Q$, we note that since we can retrieve its neighbors in X in constant time, we can determine whether it is on some nf-arc in constant time; we will treat this as a binary valued funtion $\mathrm{nf}(x)$ which is $true$ when x is on some nf-arc, and $false$ otherwise.

The *RandomElement* function refers to random or arbitrary choice of some element from a set; the *Probe* function refers to the full update algorithm (described in Section IIIB), which uses and modifies all the objects in the program. Most object updates occur within the Probe function.

Note that by the time Phase 2 starts, by definition, we will have at least one member of E_c; note also that maintaining Q^* is only necessary for Phase 1.

B. Algorithm for Handling a New Probe

The algorithm for updating the maintained information (X, R, E_c, V_c, Q, Q^*) is relatively simple since we usually probe from the set Q (since $Q^* \subset Q$). To update X in this case, we merely note that each point $x \in Q$ is specifically linked to two consecutive 'neighbors' in X.

If the new p-disk contains or is contained by one or both of the 'neighbor' cp-disks of its center, we perform the condense operation; this check trivially takes constant time since it has only two neighbors. It cannot contain or be contained by any non-neighboring cp-disks, and therefore checking whether the condense operation has to be used has constant time complexity per step.

The only case where we do not probe from Q is in Phase 2, when line L^*_{t-1} containing edge e_{t-1} is meets $\zeta(i)$ (by definition at an endpoint of $\zeta(i) \cap \partial(R)$) and, in addition, the other endpoint of $\zeta(i) \cap \partial(R)$ is not in Q. Even if we cannot determine it from our observations alone, our original definition of the ordering (depending on P) is still valid;

Algorithm 1 Identifying P using proximity probes

procedure DETERMINEP(D)
  $V_c, E_c \leftarrow null$ ▷ Initialization
  $\partial(R) \leftarrow \partial(D)$
  $x \leftarrow$ RandomElement($\partial(D)$)
  run Probe(x)
  while $E_c = null$ **do** ▷ Phase 1
    if $Q^* \neq null$ **then**
      $x \leftarrow$ RandomElement(Q^*)
    else
      $x \leftarrow$ RandomElement(Q)
    run Probe(x)
  while $E_c^* \neq null$ and $V_c^* \neq null$ ▷ Phase 2
    $x \leftarrow \rho(L, R)$
    if $x \in V_c$ **then** ▷ Case a:
      run NextEdge(x) ▷ $x = v_{t-1}$
    else if nf(x) $= false$ **then** ▷ Case b:
      run Probe(x) ▷ x is not on an nf-arc
    else ▷ Case c:
      $x' \leftarrow q(x)$ ▷ x is on an nf-arc
      run Probe(x')
  return V_c ▷ Return P as a set of vertices
end procedure

procedure NEXTEDGE(x)
  while $\neg\exists e \in (E_c \backslash e_{t-1}) | x \in e$
    $x' \leftarrow \mathrm{N}_Q(C_i^*)$
    run Probe(x')
end procedure

because the new disk has its center on the neighbor-feasible arc of C_i^*, it must be a neighbor of C_i^*. Furthermore, since it is the other (further clockwise around the boundary of $\bar{R}$) endpoint of $\zeta(i) \cap \partial(R)$, the remaining set of points at which C_i^* can be incident to P, which is a subset of $\zeta(i) \cap \partial(R)$, is counterclockwise from all points of the new disk (around the boundary of $\bar{R}$). Hence, the new disk cannot be between C_{i-1}^*, C_i^* and can be inserted between C_i^*, C_{i+1}^*.

The remainder of the updates involve updating V_c and E_c, and in turn updating Q to not include confirmed vertices or edges; as any vertex or line is automatically confirmed when three p-disks are tangent to it, and thus these checks remain in constant time. Updating the relevant stored information is constant for each element of R, Q, Q^*, V_c, E_c and X we update, and for each set only a bounded number of elements (the neighbors of the probed point) are updated, so the total updating time has complexity $O(1)$ per probe.

C. Example

These concepts are shown in Fig 1. The vertices of the polygon are labeled in clockwise order, with v_1 being the acute angle vertex; the edges are labeled in clockwise order as specified previously. The main features are represented as follows: cp-disks are shown by black circles, and p-disks which have been condensed are shown by dashed white circles; the infeasible region R is shown in gray; and Q is shown by large dots. In Phase 1 of the algorithm, Q^* is denoted by empty dots; in Phase 2, the next probe chosen is denoted by an empty dot.

The panels show: *(a)* P and D before any probes; *(b)* the initial probe x_0 on the boundary of D and x_1 is one of the intersection points of the disk resulting from probing x_0 and $\partial(D)$; *(c)* illustration of all but one of the probes chosen during Phase 1 of the algorithm; *(d)* after 7 probes the edge e_1 of P is *confirmed* by the algorithm, and the disks incident to it are condensed; *(e)* in Phase 2, case (c) of the algorithm occurs, resulting in a probe at x_7. This confirms v_1, and therefore we condense the disks which are incident to it (centered at x_2 and x_7); *(f)* after 14 probes, all the edges and vertices are confirmed, and so P has been determined.

IV. COMPLEXITY BOUNDS ON ALGORITHM 1

We first establish the following notation. Let v be a vertex of P; we then write $\angle_P(v)$ to refer to the angle of P at v. If v is confirmed, we note that this means the algorithm would have condensed the disks incident to v, so that $\bar{R}$ would have an angle at v; we write $\angle_R(v)$ to refer to this angle.

Note that $\angle_P(v)$ is always contained in $\angle_R(v)$ and that $\angle_R(v)$ never increases as the algorithm goes on.

A. Preliminary Lemmas

Lemma IV.1. *Assume that v is the intersection point on $\partial(R)$ of the boundaries of two p-disks C_i and C_j, neither of which contains the other. If we probe from $x \in \bar{R}$ such that $x \neq v$, the resulting p-disk C cannot pass through v unless $\angle_P(v)$ is acute. If $\angle_P(v)$ is acute and C passes through v, then $\angle_R(v)$ becomes acute.*

Proof. We first note that if either C_i or C_j is a zero-disk, the other would contain it, violating the givens of the lemma.

Assume that the new p-disk C resulting from probing $x \in \bar{R}$ ($x \neq v$) passes through v; since C_i, C_j, C all pass through $v \in \partial(R)$, it must be a vertex of P. Let l be the ray with source at v, tangent to C_i and pointing into C_j; let l' similarly be the ray with source at v tangent to C_i and pointing into C_j. The convex hull of these two rays is then a cone, and is entirely contained within the interior of the infeasible region R (with the exception of v itself). Since $x \in \bar{(}R)$, $x \notin \mathrm{Int}(R)$, and $x \neq v$, we know that x is not in this cone.

We note that since the cone can be seen as the intersection of two half-planes (whose boundaries are the line extensions of the two rays), its complement can be seen as the union of the two complements of these half-planes (which are themselves half-planes). Therefore, since x is in the complement, it must be in at least one of the half-planes; suppose without loss of generality it is in the half-plane whose border is the extension (denoted L) of l. However, let us now consider the line $L(C, v)$; this line will form an acute angle with L, and will be the line bounding the condensation of C with the zero-disk we place at v when it is confirmed. Thus, P must be between L and $L(C, v)$, which means that since the angle (which will be the new $\angle_R(v)$) of these two lines at v is acute, $\angle_P(v)$ must be acute as well. □

Lemma IV.2. *Let v be a confirmed vertex of P, and let $x \in \bar{R}$ be the next probed point which produces a disk C (Fig. 3). Note that v is already confirmed, so $x \neq v$ since we don't*

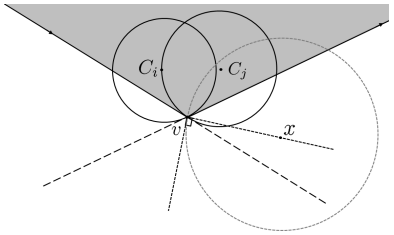

Fig. 3. If v is a confirmed vertex and the disk resulting from $x \in R$ is tangent to v, then $\angle_P(v)$ is acute while $\angle_R(v)$ is not acute.

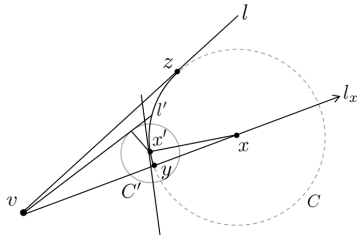

Fig. 4. The ray l' with source at v and tangent to C' cannot be tangent to C' inside of C, where C' is a disk centered at an arbitrary point lying on $\partial(C)$ and between y and z.

probe confirmed vertices. Then C can be incident to v only if $\angle_P(v)$ is acute, $\angle_R(v)$ is not acute; furthermore, afterwards, $\angle_R(v)$ will be acute (so no new p-disk can be incident to v).

Proof. Suppose C is incident to v. The proof that $\angle_P(v)$ is essentially the same as the proof for IV.1; therefore we must show that this can only happen when $\angle_R(v)$ is not acute.

Suppose to the contrary that $\angle_R(v)$ is acute. Then $x \in \bar{R}$, which is contained in $\angle_R(v)$, C is centered within the cone corresponding to $\angle_R(v)$, which is acute. But that means that the complement of $H(C, v)$, which must contain P, has v as its only intersection with the cone corresponding to $\angle_R(v)$. Thus, $P \subset \bar{R} \cap \bar{H}(C, v) = v$, which is obviously a contradiction. □

Corollary IV.3. *Let v be a vertex of P such that when v is confirmed, it is* not *by being probed directly. Then,*

- *if $\angle_P(v)$ is not acute, when the algorithm finishes the number of p-disks incident to it is at most 2*
- *if $\angle_P(v)$ is acute, when the algorithm finishes the number of p-disks incident to it as at most 3*

This corollary follows directly from the preceding lemma.

Lemma IV.4. *Let e be a confirmed edge and v be one of its endpoints. Let $x \in \bar{R}$ such that x doesn't lie on the line extending e. If we probe from x, the resulting disk cannot be incident to v unless v is an acute angle vertex of P.*

Proof. Since e is a confirmed edge, one of the half-planes bordered by the line extending e must contain R; therefore, since $x \in \bar{R}$, x cannot lie in the other half-plane bordered by e. Assume the p-disk created by probing x, denoted by C, is incident to v. The new feasible region created by v is a region lies the angle between e and $L(C, v)$, which is an acute angle. □

We note that as long as we only probe from points in $\partial(R)$ which are not confirmed vertices or in the interior of any line segment on $\partial(R)$ contained by a confirmed line, we will never create a p-disk which will be incident to the interior of any previously confirmed edge.

B. Undesirable Confirmations

The bounds derived in the previous section are only violated (by 1) if v is confirmed while incident to three p-disks, one of which is the zero-disk centered at v itself (this applies regardless of whether $\angle_P(v)$ is acute). However, we note that if one of the two non-zero p-disks is also tangent to one of the edges of P adjacent to v, we may associate it with that edge instead (so that the bound is not considered violated), and hence need only worry about the possibility that neither of the non-zero p-disks are tangent to an adjacent edge. We call such cases *undesirable confirmations*, as each occurrence increases our bound on the number of probes needed to confirm P.

We now present two geometric lemmas which are necessary for our bound on the number of undesirable confirmations; we omit the proofs here, but interested readers can find the full arguments in our technical report [21].

Lemma IV.5. *Let C be a disk centered at x and v be a point. Let l_x be the ray with source at v and passing through x. Let y be the first intersection point of L and $\partial(C)$. In addition, let l be the ray with source at v and tangent to C lying to the left of l_x, and let z be the point where l is tangent to C. Finally, let x' be any point on the arc of $\partial(C)$, and C' be a disk centered at x'. Then the ray l' with source at v and tangent to C' (such that C' is to the right of l', as in Fig. 4) is tangent to C' outside C.*

Lemma IV.6. *In Phase 2 of Algorithm 1, if at any step our probe was from case (b) (where the ray l intersects some other straight-line piece L of the boundary of R), where the angle between L and l is is not acute, and the probe does not confirm a vertex, our next probe will also be of case (b), and also with a non-acute angle.*

We note that an easy corollary of this lemma is that if case (b) occurs at a non-acute angle, it will continue until a new vertex is confirmed; since undesirable confirmations by definition cannot happen in case (b), the next confirmed vertex cannot be undesirable. We may now use this to present a bound on the number of undesirable confirmations.

Lemma IV.7. *At most one undesirable confirmation occurs during Phase 1.*

Proof. First, since we only probe from Q during Phase 1, any probe which returns a zero-disk must confirm a vertex, as no point in Q can correspond to the interior of an edge of P. In this case, by definition, an undesirable confirmation occurs if and only if we probe from a point x which is in the intersection of two p-disks and receive $f_P(x) = 0$; therefore, by definition, if we probe from Q^* we will not get an undesirable confirmation.

Suppose that we have one undesirable confirmation so far,

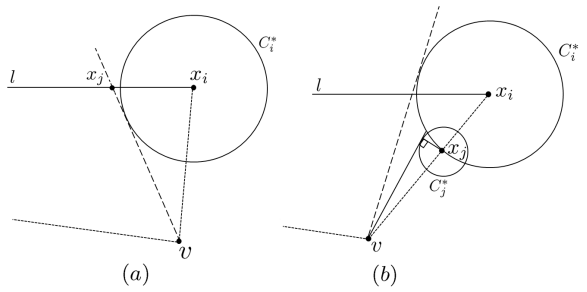

Fig. 5. v is the vertex identified by an undesirable confirmation and C_i^* is the p-disk generated by the subsequent probe. (a) Any probe which falls on l before the next undesirable confirmation of a vertex must be the intersection of l and a line on $\partial(R)$ where the angle at the intersection is not acute, and must be the result of case (b) of Algorithm 1. (b) Any probe after x_i which is the last probe lying on l falls on the intersection of ζ_i and a tangent line.

for vertex v of P, and suppose we are still in Phase 1; therefore, E_c is empty. In particular, this means that segments of the lines produced by the condense operation on disks incident to v are on the boundary of R. Given one of these segments, we see that one endpoint will be v and the other endpoint cannot be a confirmed vertex (since otherwise we could confirm the line and go to Phase 2); furthermore, the other endpoint cannot be incident to two p-disks either since we would still be able to confirm the line in question. Therefore, the other endpoint will be in Q^*, so Q^* will not be empty. Therefore, once one undesirable confirmation occurs in Phase 1, there cannot be another until Phase 2 begins. □

Lemma IV.8. *Let m be the number of undesirable confirmations which occur over the course of the algorithm. Then $m \leq n/2 + 1$.*

Proof. We first wish to show that in Phase 2 only every other confirmed vertex can be an undesirable case. More specifically, we assume v is confirmed by an undesirable confirmation, and show that the next vertex to be confirmed cannot be an undesirable confirmation. We let l and e_t be defined as in the algorithm. We also assume to the contrary that the next vertex confirmation after v is undesirable.

Since v is an undesirable confirmation, it must have been confirmed in case (c), where the endpoint of l on the boundary of R falls on the intersection of the boundaries of two cp-disks. When v is confirmed, these two cp-disks are condensed, and the next probe must be of case (b), at the intersection between l and one of the lines L produced through v by the condense operation. We let this intersection be x_i and the resulting cp-disk be C_i^* (Fig. 5).

If any probe afterwards (before the confirmation of the next vertex, which we have assumed to be undesirable) falls on l, it cannot be of case (c) and cannot be the assumed undesirable next confirmation; therefore, it was generated by case (b), as the intersection of l and a line on $\partial(R)$ tangent to C_i^*. However, since the center of C_i^* is on l, the angle at this intersection cannot be acute (Fig. 5(a)), and therefore by Lemma IV.6, the next confirmed vertex cannot be an undesirable case, thus producing a contradiction.

Therefore, we may assume that x_i is the last probe to fall on l before the next vertex is confirmed; thus, every probe between x_i and the next confirmation is of case (c) (since an occurrence of case (a) would require the next confirmation to happen first, and case (b) by definition always falls on l). It is clear by a simple inductive argument that every probe between x_i and the new undesirably-confirmed vertex falls on the boundary of C_i^*: in case (c), the next probed point will be on the other endpoint of the nf-arc $\zeta(i)$. If the resulting p-disk intersects l outside of C_i^*, then we have a contradiction as C_i^* can immediately be shown to be disjoint with P, which by definition is not possible, so C_i^* must remain the cp-disk l intersects; but then, since we have assumed that every probe until the next vertex confirmation is of case (c), it must remain on the boundary of C_i^* until the next vertex is confirmed.

We now note that each new probe which does not confirm a new vertex decreases $\zeta(i)$. We note as well that because x_i is located where one of the lines associated with v (when it is confirmed) intersects l, $\zeta(i)$ by definition is initially the arc between l and the line joining v and x_i (Fig. 5(b)); we let this arc of C_i^* be called ζ^*.

We now consider the probe immediately preceding the probe which results in the undesirable confirmation. By the above, this must occur on ζ^*; but then by Lemma IV.5, after this probe neither of $\zeta(i)$'s endpoints are intersections of the boundaries of two cp-disks. But this immediately implies that the next probe, which is at an endpoint of $\zeta(i)$, cannot be an undesirable confirmation. Hence, since none of these probes can be an undesirable confirmation, the next vertex confirmation will not be undesirable, contradicting our assumption that it is.

Thus, we have proved that in Phase 2, whenever an undesirable confirmation occurs, the next vertex to be confirmed cannot be an undesirable confirmation. Since we have already proved that in Phase 1 there can be at most one undesirable confirmation, there are in total at most $n/2 + 1$. □

C. Bounding the Number of Probes Used

We now wish to find an upper bound for the number of probes used by Algorithm 1; this is achieved by analyzing the number of p-disks that can be incident to any edge or vertex of P when it is confirmed. We now assume that no undesirable confirmation occurs; we will then note that since each undesirable confirmation adds at most one probe, and by Lemma IV.8 there are at most $n/2+1$, we can add this to the bound we derived to obtain the true bound.

At any given step in the algorithm, let $\phi(e)$ and $\phi(v)$ denote the number of p-disks incident to unconfirmed edge e and unconfirmed vertex v respectively; and let $\omega(e)$ and $\omega(v)$ denote the number of p-disks which are incident to confirmed edge e and confirmed vertex v, respectively.

We first consider the number of p-disks any object can have adjacent to it at the moment it is first confirmed; by convention, if a p-disk is incident to both some confirmed vertex and some confirmed line, we associate it with the vertex only. We perform this analysis on the two basic phases of the algorithm.

For Phase 1 (i.e. confirming the first edge), there are two possible cases for the number of probes which will suffice to confirm the first edge e_1 with clockwise endpoint v_1:

- If $\phi(v_1) \leq 1$ three disks are sufficient to confirm e_1.

- If v_1 is confirmed or $\phi(v_1) = 2$, then two disks are sufficient to confirm e_1.

We will conduct the same analysis for Phase 2 by computing the possible values of $\omega(v_{i-1})$ and $\omega(e_i)$ when they are first confirmed (which depends on whether v_{i-1} is acute or not) for $1 < i \leq n$. We note that no vertex can be confirmed on $\partial(D)$ because $P \in \text{Int}(D)$.

Case 1: v_{i-1} is not confirmed and $\phi(v_i) \leq 1$. Since v_{i-1} is not confirmed but e_{i-1} is confirmed, $\phi(v_{i-1}) \leq 1$. We consider the two possible sub-cases: either v_{i-1} is not an acute angle vertex of P, or it is.

- Suppose v_{i-1} is not an acute angle vertex. It could either have been confirmed by case (b) or case (c) from Phase 2 of the algorithm.
  - Suppose it was confirmed by case (b); let x be the point probed. For case (b) of the algorithm to confirm a vertex, the result of the probe must be 0 (i.e. $f_P(x) = 0$), and this new zero-disk is the only disk incident to v_{i-1}; thus $\omega(v_{i-1}) = 1$. In this case, x (which is actually v_{i-1}) cannot lie on the boundary of D (as in this case $x \in P \subset \text{Int}(D)$), so x is on a segment of an (confirmed or unconfirmed) line L on $\partial(R)$; this line will then be confirmed as e_i with $\omega(e_i) = 2$.
  - Suppose it was confirmed by case (c). By Lemma IV.4, the new p-disk cannot pass through v_{i-1}, so $\omega(v_{i-1}) = 1$. We observe that to confirm v_{i-1}, the new p-disk must reduce the feasible arc of the previous p-disk containing v_{i-1} to a single point; to do this, it must confirm e_i. Hence, since $\omega(v_{i-1}) = 1$ and $\phi(v_i) \leq 1$, we get $\omega(e_i) = 2$.

  Therefore, in all cases, $\omega(v_{i-1}) = 1$ and $\omega(e_i) = 2$.
- If v_{i-1} is an acute angle vertex. This is similar to the above case, except that as Lemma IV.4 doesn't hold for acute angles, we include the possibility that in case (c) the resulting p-disk will pass through v_{i-1}. If so, v_{i-1} is confirmed, and the next iteration of the algorithm will be case (a). As $\phi(v_i) \leq 1$, $\omega(e_i) = 2$, and when v_{i-1} is confirmed in the next iteration $\omega(v_{i-1}) \leq 2$.

Case 2: v_{i-1} is not confirmed and either $\phi(v_i) = 2$ or v_i is confirmed. This case is similar to case 1, except that because $\phi(v_i) = 2$ (or v_i is confirmed), e_i is incident to at most one disk, and v_{i-1} will be confirmed immediately after e_i is confirmed. So, $\omega(e_i) = 1$ and $\omega(v_{i-1}) \leq 2$, if v_{i-1} is acute and $\omega(v_{i-1}) = 1$ if it is not.

Case 3: v_{i-1} is confirmed and $0 \leq \phi(v_i) \leq 1$. We consider the two possible sub-cases: either v_{i-1} is not an acute angle vertex of P, or it is.

- v_{i-1} is not an acute angle vertex. Since v_{i-1} is confirmed before e_i and v_{i-1} is not an acute angle, by Lemma IV.4, $\omega(v_{i-1}) = 2$, and case (a) will immediately follow in the algorithm. The next edge e_i will be confirmed by two incident disks since $\phi(v_i) \leq 1$, so $\omega(e_i) = 2$.
- v_{i-1} is an acute angle vertex. According to Lemma IV.4, it is possible that v_{i-1} has been confirmed with three disks as v_{i-1} is an acute angle. Therefore, $\omega(v_{i-1}) \leq 3$. As in the previous case, $\omega(e_i) = 2$.

Case 4: v_{i-1} is confirmed and either $\phi(v_i) = 2$ or v_i is confirmed. We again consider the same two possible sub-cases as in the above cases.

- v_{i-1} is not an acute angle vertex. As in case 3, $\omega(v_{i-1}) = 2$, but the next edge will be confirmed with one incident disks since v_i is incident to more than one disk (or already confirmed), so $\omega(e_i) = 1$
- v_{i-1} is an acute angle vertex. As in case 3, $\omega(v_{i-1}) \leq 3$, and $\omega(e_i) = 1$ since v_i is incident to multiple disks (or already confirmed).

Finally, it is clear that v_n will be confirmed with one disk. Table I summarizes the result for the above four cases.

Theorem IV.9. *Our algorithm uses at most $3n+m+k+1 \leq 3.5n+k+2$ probes to find P, where $k \leq 3$ is the number of acute angles of P; each probe is computed in $O(1)$ time, thus leading to an overall time complexity of $O(n)$.*

TABLE I
THE NUMBER OF INCIDENT P-DISKS TO v_{i-1}, e_i FOR $(1 < i \leq n)$

Case	v_{i-1}	v_i	v_{i-1}: Not acute $\omega(v_{i-1}), \omega(e_i)$	v_{i-1}: acute $\omega(v_{i-1}), \omega(e_i)$
1	NC	NC, $\phi(v_i) \leq 1$	$1, 2$	≤ 2, 2
2	NC	C or $\phi(v_i) = 2$	$1, 1$	≤ 2, 1
3	C	NC, $\phi(v_i) \leq 1$	$2, 2$	≤ 3, 2
4	C	C or $\phi(v_i) = 2$	$2, 1$	≤ 3, 1

Proof. We note that no p-disk generated at any point by the algorithm can be incident to a previously-confirmed edge or to a previously-confirmed non-acute angle vertex once both edges adjacent to it have been confirmed. Note also that since the algorithm never probes from the interior of $\bar{R}$, the algorithm never uses a probe which returns -1. Therefore, the number of probes needed is equal to the sum of the number of p-disks incident to each edge and vertex of P when they are confirmed, with the possible additional k for the acute angles already taken care of by assuming the worst case at time of confirmation. Let n_j be the number of times case j occurs, and k_j be the number of times case j occurs with an acute vertex; then $\sum_{j=1}^{4} n_j = n - 1$ and $\sum_{j=1}^{4} k_j \leq k$ since the cases begin once e_1 is confirmed.

We now consider the number of p-disks incident to each edge and vertex of P when they are confirmed, assuming no undesirable confirmations:

- e_1 is incident to at most 3 p-disks when it is confirmed
- For $j = 1, 4$, by Table I we note that $\omega(v_{i-1}) + \omega(e_i) \leq 4$ if v_{i-1} is acute, and $\omega(v_{i-1}) + \omega(e_i) = 3$; hence at most $3n_j + k_j$ probes were used.
- For $j = 2$, by Table I we note that $\omega(v_{i-1}) + \omega(e_i) \leq 3$ if v_{i-1} is acute, and $\omega(v_{i-1}) + \omega(e_i) = 2$; hence at most $2n_2 + k_2$ probes were used
- For $j = 3$, by Table I we note that $\omega(v_{i-1}) + \omega(e_i) \leq 5$ if v_{i-1} is acute, and $\omega(v_{i-1}) + \omega(e_i) = 4$; hence at most $4n_3 + k_3$ probes were used

Consider what happens in case 3 (with vertex v_{i-1} and edge e_i); it occurs when v_{i-1} is incident to two disks (or is confirmed) before e_i is confirmed. If $i = 2$, then e_1 must have

been adjacent to 2 p-disks. If $i > 2$, then case 3 was preceded by either case 2 or case 4; if it was case 4, then since v_{i-1} was already confirmed, e_{i-1} must have been confirmed with one fewer p-disk than our above bounds.

Thus, every instance of case 3 (which requires one more probe per vertex-edge pair than cases 1 or 4), there is a corresponding instance either of case 2 (which requires one fewer probe per vertex-edge pair than cases 1 or 4) or of case 4 (or the base case) in which at least one fewer probe was used than the bound above. So, since case 3 is the only case in which more probes are required than cases 1 and 4, and since we showed that every instance of case 3 is 'offset', we can bound the total number of probes needed by the number needed if only cases 1 and 4 occurred.

Thus, the pairs $(v_1, e_2), ..., (v_{n-1}, e_n)$ plus e_1 require at most $3n+k$ probes to confirm; the final vertex v_n requires one more, giving an upper bound of $3n + k + 1$ probes with the assumption that no undesirable confirmations occurred. Each undesirable case increases the upper bound by at most 1, and the number of such cases (by Lemma IV.8) is $m \leq n/2 + 1$. Hence, we compute our true upper bound as $3n+m+k+1 \leq 3.5n + k + 2$ probes. Finally, we note that in Section III(B) we showed that each probe requires $O(1)$ computations, and therefore the total time required by the algorithm is $O(n)$. □

V. Problem 2: Identifying P from a Finite Set

We now consider the problem of identifying a convex polygon P with n vertices from a known finite set. Let Γ be a (known) finite set of convex polygons and D be a (known) disk. We are now asked to identify P using as few probes as possible, knowing that $P \subset \text{Int}(D)$ and that $P \in \Gamma$ (where a polygon is considered a member of Γ if it can be rotated and translated to match an element of Γ). Let $m = |\Gamma|$ and n' be the maximal number of vertices on any polygon in Γ. We show that $2n + 2$ probes are sufficient to find P, with time complexity $O(n'm)$.

Remark: Since our algorithm will proceed by confirming the edges of P in counterclockwise order, P is *not* considered to be in Γ if a reflection is required to produce a match to an element. However, if we wish to include reflections, we can run our algorithm on an augmented set consisting of Γ plus the reflections of all elements of Γ; this does not increase the number of probes needed to find P by the result above, but it doubles the computation time needed for the algorithm.

Let e_{min} and e_{max} be the minimum and maximum, respectively, over lengths of all edges in Γ, and ψ_{max} be the maximum over all angles, of the polygons in Γ.

Lemma V.1. *Let $x_0 \notin D$ be a point and C_0 be the p-disk (of radius $r_0 > 0$) generated by probing x_0. Let x_1 be another point such that $\text{dist}(x_1, x_0) < d$ where $d = \min\{e_{min}, r_0 \sin(\frac{\pi-\psi_{max}}{2})\}$. Let C_1 be the p-disk generated by probing x_1. Then:*

- *If C_0 is incident to an edge, C_1 will be incident to the same edge or one of its endpoints.*
- *If C_0 is incident to a vertex, C_1 will be incident to the same vertex or one of its adjacent edges.*

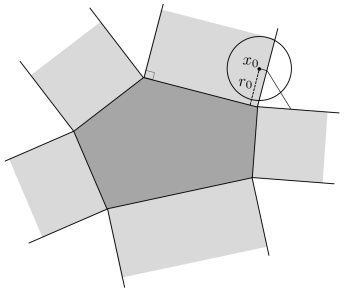

Fig. 6. The partitioning of the exterior of a polygon. Any probes in the same part are incident to a common vertex or edge.

Proof. Consider all the perpendicular half-lines of the edges of P rooted at its vertices and contained in the exterior of P. These lines partition the exterior of P to $2n$ regions. Note that if we probe from any point inside one of these regions, the resulting p-disk hits the edge or vertex bordering the region (Fig. 6). After probing from x_0, if we choose x_1 inside the disk centered at x_0 with radius d, it is easy to see that x_1 lies in the same region that x_0 lies in, or in an adjacent region. □

Lemma V.2. *Let C_0 and C_1 be two intersecting disks centered at x_0 and x_1, respectively. Let $p_{0,1}$ be an intersection point of their boundaries . If there are two intersection points, let $p_{0,1}$ be the one closer to $L_{0,1}$. For $i = 0, 1$, let p_i be the point where $L_{0,1}$ is tangent to C_i. Then, $\text{dist}(p_0, p_{0,1}) < \text{dist}(x_0, x_1)$.*

Proof. Let $\angle p_0 x_0 p_{0,1} = \alpha$ and let $\angle p_1 x_1 p_{0,1} = \beta$ where p_1 is the tangent point of $L_{0,1}$ on C_1^*. It is clear that $\alpha + \beta \leq \pi$. (Note that $\alpha + \beta = \pi$ when there is only one intersection point.) $\angle p_0 p_{0,1} p_1 = \pi - (\alpha + \beta)/2 \geq \pi/2$. Thus, the triangle $p_{0,1} p_0 p_1$ is obtuse and $\text{dist}(p_0, p_{0,1}) < \text{dist}(p_0, p_1)$. Since $\text{dist}(p_0, p_1) \leq \text{dist}(x_0, x_1)$ (since p_0, p_1 are projections of x_0, x_1 on $L_{0,1}$), $\text{dist}(p_0, p_{0,1}) < \text{dist}(x_0, x_1)$. □

We now present Algorithm 2 as a series of lemmas. Algorithm 2 maintains a list of vertices and edges confirmed, as well as a point on e_1 (after it is confirmed). At each step (after initialization, which confirms the first edge), we then take the last vertex or edge confirmed (in clockwise order) and show that it is possible to determine the next edge or vertex with a single probe. This allows us to uniquely determine P using $2n+2$ probes (one probe per vertex or edge, plus initialization cost of 2 probes). Lemma V.3 shows that either the first edge can be confirmed with 3 probes, or the first edge and first vertex can be confirmed with 4 probes; Lemma V.4 shows that afterwards, the next edge or vertex can be determined using only one probe. Theorem V.5 then puts them together to obtain our upper bound on the number of probes needed.

Lemma V.3. *There is an approach that can find either the first edge of P with three disks or find the first edge and one of its endpoints using 4 probes.*

Proof. For any point x_i, let C_i be the p-disk (with radius r_i) generated by probing it. First we probe from a point $x_0 \in \partial(D)$. Choose $x_1 \in \partial(D)$ such that $\text{dist}(x_1, x_0) < d$ where $d = \min\{e_{min}, r_0 \sin(\frac{\pi-\psi_{max}}{2})\}$. By Lemma V.1, if we probe from x_1, C_1 and C_0 are incident with the same edge, the same

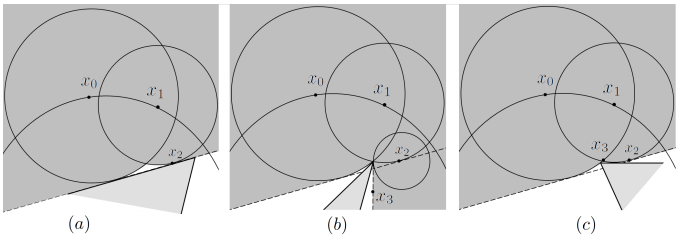

Fig. 7. The third probe from x_2 results in the disk C_2. There are three possible cases: (a) If C_2 is a zero disk, then three probes are sufficient to confirm the first edge. (b) If $\partial(C_2)$ intersects the intersection of $\partial(C_0), \partial(C_1)$, the intersection point is confirmed as a vertex and probing from x_3 will be tangent to the adjacent edge. (c) Otherwise, probing from x_3 (the intersection point of $\partial(C_0), \partial(C_1)$) will confirm an edge and one of its endpoints.

vertex or an edge and its endpoint. Since $d \leq r_0$, the resulting disks are not disjoint. Let $p_{0,1}$ be the intersection of ζ_0 and ζ_1. Without loss of generality, assume that $r_0 \leq r_1$. We note that it is not possible for C_1 to contain C_0 completely (if $C_0 \subset C_1$, then x_0, x_1 and $p_{0,1}$ become collinear while $p_{0,1}$ has to be a point of P; however, since x_0 and x_1 lie on $\partial(D)$, $p_{0,1}$ lies outside of D which is not possible and thus produces a contradiction). Thus, we can choose x_2 to be the intersection of $L_{0,1}$ and C_1. Note that the intersection of $\partial(C_1)$ and $\partial(C_2)$ has to lie on ζ_1.

- If $r_2 = 0$, then $L_{0,1}$ is confirmed as an edge (Fig. $7(a)$).
- If $r_2 = \text{dist}(x_2, p_{0,1})$ then $p_{0,1}$ is confirmed as a vertex. Call this confirmed vertex v. Choose $x_3 \in \partial(R)$ on $L_2(v)$ such that $\text{dist}(x_3, v) < e_{min}$. Note that it is not possible for C_3 to intersect v. The tangent line of C_3 from v is confirmed as the first edge (Fig $7(b)$).
- If $0 < r_2 < \text{dist}(x_2, p_{0,1})$, then we choose x_3 on $p_{0,1}$. Note that Lemma V.2 implies $\text{dist}(x_2, x_3) < d$ (Fig $7(c)$).
  - If $r_3 = 0$, $p_{0,1}$ is confirmed as a vertex. By Lemma V.1, the tangent line of C_2 from v is confirmed as its incident edge.
  - If $r_3 > 0$ then C_3 must be tangent to $L_{1,2}$. Then, $L_{1,2}$ is confirmed as a line and its intersection with C_0 is confirmed as its endpoint.

□

Lemma V.4. *Let $e_1, ..., e_n$ be the edges of P in counterclockwise order. Suppose we have a line passing through e_1 and a point p lying on the interior of e_1. Let v_n and v_1 be the unknown endpoints of e_1. We can find v_1 using a single probe.*

Proof. Without loss of generality assume that the given line is horizontal and v_n is the left endpoint of e_1. First, suppose that v_n is given. Based on the edge lengths in Γ, there are a finite number of candidate vertices for v_1. Each of the candidate vertices is associated with a region (determined by the polygon which the candidate is based on) from which a p-disk will pass through it. If we probe from a point which in the intersection of all these regions, the disk will hit v_1. This intersection is not empty and is formed by the vertical line which passes through the rightmost candidate vertex and the perpendicular line to its next incident edge which has the lowest intersection point with the rightmost vertical line (Fig. 8).

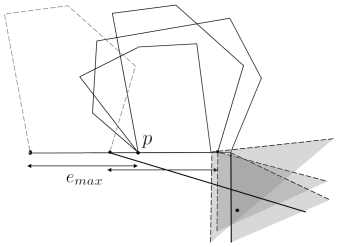

Fig. 8. The intersection of the corresponding regions to all candidate vertices.

Consider the case in which v_n is not given; we bound its location on the given line. Note that $|e_1| \leq e_{max}$, so v_n lies between p and p' where $p' \in D$ is the leftmost point such that $\text{dist}(p, p') \leq e_{max}$. Again, we aim to find a region which is the intersection of all regions for the candidate vertices when v_1 can lie between p and p'. This region is encompassed between two lines. One is the vertical line that passes through the rightmost candidate vertex, assuming that v_1 lies on p. The other is the line perpendicular to the next incident edge of e_1 which has the lowest intersection point with this vertical line assuming that v_n lies on p' (Fig. 8). □

Theorem V.5. $2n + 2$ *probes are sufficient to determine a convex polygon P from a set of models Γ.*

Proof. By Lemma V.3, to determine the first edge and a point on it, either 3 probes are sufficient or 4 probes are used and one of its endpoints is confirmed as well. By Lemma V.4, the endpoint of the edge can be determined by a single probe. To find the next edge incident to the endpoint, a single probe can be used from a point which lies on the extension of the current edge from its determined endpoint with the distance less than e_{min}. The p-disk generated by this probe will be guaranteed to hit the next edge. Using this strategy, we need $3 + n + (n - 1)$ probes to identify P.

Computing e_{max}, e_{min} and ψ_{max} requires $O(n'm)$ time. Finding the region to probe from requires computing the lowest intersection point of $O(n'm)$ lines with a vertical line. The incident edge of a vertex can be determined in $O(1)$. Thus, the time complexity of the algorithm is $O(n'm)$. □

VI. Conclusion and Future Work

This addresses proximity probe and presents two new algorithms. Algorithm 1 determines the shape an unknown convex polygon P (with n vertices, $k \leq 3$ of which are acute angle vertices) requiring at most $3.5n + k + 2$ probes, with the position of each probe requiring $O(1)$ time to compute. Algorithm 2 addresses the problem introduced in [15] to identify P from among a finite set Γ. Algorithm 2 solves this using at most $2n + 2$ probes, with total computation time $O(n'm)$, where m is the size of Γ and n' is the maximal size of the polygons in Γ.

There are many interesting opportunities for future work. One is to improve on these upper bounds and study cases that provide lower bounds. Another very important question is how to model and develop algorithms for the case where probe measurements are not precise and instead lie within

some known bounds of the true value. Instead of precisely identifying the unknown polygon, our goal here is to give an approximation of it.

Another question is the case where proximity probes operate from inside the polygon or polyhedron. Also of interest is using multiple probes in parallel, where data from the set of probes is used to compute the next set of probes. We are also interested in extending these results to 3D and higher dimensions and to curved and non-convex objects, perhaps using the approach that Boissonnat and Yvinec [14] developed to extend finger probes to non-convex polyhedra.

REFERENCES

[1] Skiena, S. S., *Problems in geometric probing*, Algorithmica, 4(4):599-605,1989.

[2] Kalinin, S. V. and Gruverman, A., *Scanning Probe Microscopy of Functional Materials*, 2011.

[3] Mokaberi, B. and Requicha, A.A.G., *Drift compensation for automatic nanomanipulation with scanning probe microscopes*, IEEE Transactions on Automation Science and Engineering, 3(3):199-207, 2006.

[4] Eichhorn, V., Bartenwerfer, M. and Fatikow, S., *Nanorobotic assembly and focused ion beam processing of nanotube-enhanced AFM probes*, IEEE Transactions on Automation Science and Engineering, 9(4): 679-686, 2012.

[5] Dotson, C. L., *Fundamentals of Dimensional Metrology*, 2006.

[6] Czichos, H., Saito, T., and Smith, L. E., *Springer Handbook of Metrology and Testing*, 2011.

[7] Susto, G. A., Schirru, A., Pampuri, S., De Nicolao, G. and Beghi, A., *An Information-Theory and Virtual Metrology-based approach to Run-to-Run Semiconductor Manufacturing Control*, Proceedings of the IEEE Conference on Automation Science and Engineering, 358-363, 2012.

[8] Pampuri, S., Schirru, A., Susto, G.A., De Luca, C., Beghi, A., and De Nicolao, G. *Multistep Virtual Metrology Approaches for Semiconductor Manufacturing Processes*, Proceedings of the IEEE Conference on Automation Science and Engineering, 91-96, 2012.

[9] Klein, K. and Suri, S, *Capture bounds for visibility-based pursuit evasion*, Proceedings of the 29th ACM Annual Symposuim on Computational Geometry, 329-338, 2013.

[10] Cole, R. and Yap, C. K.. *Shape from probing*, Journal of Algorithms, 8(1):19-38, 1987.

[11] Skiena S. S., *Interactive Reconstruction via Geometric Probing*, Proceedings of the IEEE 80, 1364-1383, 1992.

[12] Dobkin, D., *Edelsbrunner, H., and Yap, C. K., Probing convex polytopes*, Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing, 424-432, Berkeley, California, 1986.

[13] Guha, S. and Kiêu Trong, K., *Recognizing convex polygons with few finger probes*, Pattern Analysis and Applications, 12(2):193-199, 2009.

[14] Boissonnat, J. D. and Yvinec, M., *Probing a scene of nonconvex polyhedra*, Algorithmica, 8:321-342, 1992.

[15] Rao, A. S. and Goldberg, K. Y., Shape from diameter: *Recognizing polygonal parts with a parallel-jaw gripper*, Intl. J. of Robotics Research, 13(1):16-37, 1994.

[16] Meijer, Henk and Skiena, Steven S., *Reconstructing Polygons from X-Rays*, Geometriae Dedicata, 61(2), pp 191-20, 1996.

[17] Li, S.-Y. R., *Reconstruction of polygons from projections*, Information Processing Letters, 28:235-240, 1988.

[18] Skiena S. S., *Probing Convex Polygons with Half-Planes*, Journal of Algorithms 12, 359-374, 1991.

[19] Niemann, J., *Electrical Measurements on Nanoscale Materials*, Keithley Instruments tutorial paper, 2004.

[20] Panahi, F., Adler, A., van der Stappen, A.F., and Goldberg, K., *An Efficient Proximity Probing Algorithm for Metrology*, Proceedings of the IEEE Conference on Automation Science and Engineering, 342-349, 2013.

[21] F. Panahi, A. Adler, A.F. van der Stappen, K. Goldberg, *An Efficient Proximity Probing Algorithm for Metrology*, Technical Report UU-CS-2013-010, Dept. of Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands, 2013.

Aviv Adler is a senior at Princeton University, where he is pursuing his B.A. in Mathematics. His research interests include computational geometry, motion planning, graph theory, and algorithmic automation.

Fatemeh Panahi received her B.S. degree from Shahid Beheshti University, Tehran, Iran in 2007 and her M.S. degree from Amirkabir University of Technology, Tehran, Iran in 2009, both in Computer Science. In 2011 she joined the University of Utrecht, Utrecht, The Netherlands where she currently pursuing her Ph.D in Computer Science. Her research interests include algorithmic automation, automatic manipulation and computational geometry.

A. Frank van der Stappen (M03) received the M.Sc. degree from Eindhoven University of Technology, Eindhoven, The Netherlands, in 1988 and the Ph.D. degree from Utrecht University, Utrecht, The Netherlands, in 1994. He is currently an Associate Professor with the Department of Information and Computing Sciences, Utrecht University. His research interests include algorithmic automation, path and motion planning, grasping, simulation, and geometric algorithms.

Dr. van der Stappen is an Associate Editor for the IEEE Transactions on Automation Science and Engineering.

Ken Goldberg is Professor of Industrial Engineering and Operations Research at UC Berkeley, with appointments in Electrical Engineering and Computer Science, the School of Information, and the UC San Francisco Dept of Radiation Oncology. He served (2006-2009) as Vice-President of Technical Activities for the IEEE Robotics and Automation Society and is Editor-in-Chief of the IEEE Transactions on Automation Science and Engineering. Goldberg is Founding Co-Chair of the IEEE Technical Committee on Networked Robots and Founding Chair of the IEEE Transactions on Automation Science and Engineering (T-ASE) Advisory Board. Goldberg has published over 170 refereed papers and eight US patents and was awarded the NSF Presidential Faculty Fellowship in 1995, the Joseph Engelberger Award for Robotics Education in 2000, the IEEE Major Educational Innovation Award in 2001 and was named IEEE Fellow in 2005.