To appear in *Algorithmica, Special Issue for ESA'96*

On Minimum-Area Hulls*

Esther M. Arkin† Yi-Jen Chiang‡ Martin Held§ Joseph S. B. Mitchell¶
Vera Sacristan|| Steven S. Skiena** Tae-Cheon Yang††

(March 1996; revised March 1997)

Abstract

We study some minimum-area hull problems that generalize the notion of convex hull to star-shaped and monotone hulls. Specifically, we consider the *minimum-area star-shaped hull* problem: Given an n-vertex simple polygon P, find a minimum-area, star-shaped polygon P^* containing P. This problem arises in lattice packings of translates of multiple, non-identical shapes in material layout problems (e.g., in clothing manufacture), and has been recently posed by Daniels and Milenkovic. We consider two versions of the problem: the *restricted* version, in which the vertices of P^* are constrained to be vertices of P, and the *unrestricted* version, in which the vertices of P^* can be anywhere in the plane. We prove that the restricted problem falls in the class of "3SUM-hard" (sometimes called "n^2-hard") problems, which are suspected to admit no solutions in $o(n^2)$ time. Further, we give an $O(n^2)$ time algorithm, improving the previous bound of $O(n^5)$. We also show that the unrestricted problem can be solved in $O(n^2 p(n))$ time, where $p(n)$ is the time needed to find the roots of two equations in two unknowns, each a polynomial of degree $O(n)$.

We also consider the case in which P^* is required to be monotone, with respect to an unspecified direction; we refer to this as the *minimum-area monotone hull* problem. We give a matching lower and upper bound of $\Theta(n \log n)$ time for computing P^* in the restricted version, and an upper bound of $O(nq(n))$ time in the unrestricted version, where $q(n)$ is the time needed to find the roots of two polynomial equations in two unknowns with degrees 2 and $O(n)$.

Keywords: minimum-area hulls, star-shaped polygons, monotone polygons, 3SUM-hardness, lower bounds, computational geometry, design and analysis of algorithms.

*An extended abstract of this paper was presented at the 4th European Symposium on Algorithms, Barcelona, Spain, September, 1996.

†Department of Applied Mathematics and Statistics, SUNY Stony Brook, NY 11794-3600; estie@ams.sunysb.edu. Supported in part by NSF Grant CCR-9504192.

‡Department of Applied Mathematics and Statistics, SUNY Stony Brook, NY 11794-3600; yjc@ams.sunysb.edu. Supported in part by NSF Grant DMS-9312098.

§Universität Salzburg, Institut für Computerwissenschaften, Jakob-Haringer Str. 2, A-5020 Salzburg, AUSTRIA; held@cosy.sbg.ac.at. Supported in part by NSF Grants DMS-9312098 and CCR-9504192.

¶Department of Applied Mathematics and Statistics, SUNY Stony Brook, NY 11794-3600; jsbm@ams.sunysb.edu. Supported in part by Hughes Research Laboratories and NSF Grant CCR-9504192.

||Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Pau Gargallo 5, 08028 Barcelona, Spain; vera@ma2.upc.es. Supported in part by Projecte UPC PR-9410.

**Department of Computer Science, SUNY Stony Brook, NY 11794-4400; skiena@cs.sunysb.edu. Supported in part by NSF Grant CCR-9109289 and ONR Award 400x116yip01.

††Dept. of Computer Science and Statistics, Kyungsung University, 110, DaeYeon-dong, Nam-gu, Pusan, 608-736, KOREA; tcyang@csd.kyungsung.ac.kr. Supported in part by Ministry of Education of KOREA while visiting SUNY at Stony Brook.

1 Introduction

The problem of approximating the shape of a geometric object by a simpler object arises in many real-world applications of geometric computing. A widely used approximation of an object is its convex hull. The convex hull of a set S is the "smallest" convex set containing S, where "smallest" can take on various meanings — minimal with respect to subsets (i.e., no convex set is properly contained within it), minimum area, minimum perimeter, etc — and still produce the same hull.

In this paper, we consider minimum-area hull problems in which we are to compute a set S^* that contains set S such that S^* has minimum possible area among all supersets of S that lie in some class $\mathcal{P}$. In the convex hull problem, $\mathcal{P}$ is the class of convex sets. In this paper, we consider the classes of *star-shaped* sets and of *monotone* sets.

Specifically, we address problems in which S is a simple polygon, P, and we consider two versions of the problem: the *restricted* version, in which the vertices of P^* are constrained to be vertices of P, and the *unrestricted* version, in which the vertices of P^* can be anywhere in the plane.

In particular, we answer the following problem posed by Daniels [7]:
Given a simple polygon P having n vertices, how efficiently can one compute a minimum-area star-shaped simple polygon P^* that contains P and has as its vertices a subset of those of P (the restricted problem, for star-shaped hulls)?

Previously, an $O(n^5)$ time dynamic programming algorithm was known [8]. In contrast, we provide an $O(n^2)$ time algorithm, and, further, we give evidence that this may be the best possible algorithm — we show that the problem is in the class of "3SUM-hard" (sometimes called "n^2-hard") problems, which are suspected to admit no solutions in $o(n^2)$ time [13].

Related Applications. Minimum-area hull problems arise in many geometric applications, including stock cutting and intersection detection, where we desire outer approximations to geometric shapes.

- In sheet stock cutting problems, we must cut a predefined set of shapes from a given roll of material (e.g., cloth), and we seek to use as little material as possible. Motivated by a project for the apparel industry, Daniels [7] considered the problem of computing a densest lattice packing of copies of a polygon P. (See Mount and Silverman [20] for the case of lattice packings of convex shapes.) Daniels worked within a three-dimensional search space associated with quadratic forms and known in lattice theory as the "Minkowski reduction domain". Each point in this space is associated with a parallelogram (and its rotations); each parallelogram is a building block for a lattice. A ray emanating from the origin in this space is associated with homothets of a parallelogram. Daniels showed that, for star-shaped P, this means that a "no" answer for any point in the space allows one to answer "no" for all points on a line segment from the origin to that point. This, combined with properties of surfaces corresponding to constant parallelogram area, allows one to perform binary search on the area of the parallelogram to find the densest lattice packing. This motivated Daniels to pose the problem of finding a minimum-area star-shaped enclosure for a polygon.

  One common method of shape simplification is to replace each shape with its (axis-aligned) bounding box. While this permits one to use a variety of simple bin packing heuristics for boxes, it may well be a very crude approximation to the original shapes, resulting in considerable material waste.

- Some approaches to layout problems find minimal enclosures for the polygonal shapes and then find a good placement for the enclosures. Ideally, one wants a "tight" enclosure for which crucial operations (such as finding Minkowski sums) can be performed quickly. Minkowski sums of convex polygons can be found in linear time, but the convex hull is often not a very tight enclosure. Li [15] shows that the Minkowski sum of two star-shaped polygons is star-shaped, and he shows how to compute it in time $O(mn \log mn)$ for polygons having m and n vertices. (In contrast, for arbitrary simple polygons, the Minkowski sum may have size $\Omega(m^2n^2)$.) This algorithm is at the core of the marker layout software system of Daniels, Li and Milenkovic [9, 16, 18, 19].

- As another example, outer approximations of shapes are useful to achieve fast intersection detection in virtual reality walk-throughs of geometric environments, by bounding complicated figures by simplified (superset) shapes. Only if the simplified shape is intersected need we perform a more expensive test against the original shape.

In all of these applications, we seek bounding shapes (or hulls) of polygonal shapes, where the shape is of minimum area. The convex hull of a given shape is the smallest area convex polygon that contains it. Of course, the area of the convex hull can be arbitrarily large relative to the original shape. Better approximations become possible with non-convex shapes that generalize the class of convex shapes, such as star-shaped and monotone polygons (monotone with respect to an unspecified direction). While star-shaped hulls and monotone hulls can also be arbitrarily bad outer approximations of a general shape, they are potentially much better than convex hulls in many practical situations, such as automatic marker layout.

Results. Our results include:

- We give an $O(n^2)$ time algorithm for computing a minimum-area star-shaped polygon, P^*, containing a given simple polygon P, such that P^* is restricted to have its vertices be a subset of the vertices of P. The previous best bound was given by an $O(n^5)$ time dynamic programming algorithm [8].

- We show that computing such a vertex-restricted star-shaped hull of a simple polygon P is a 3SUM-hard problem, meaning that the problem may have no solutions in $o(n^2)$ time. To our knowledge, this is the first instance of a problem defined on a simple polygon that has been shown to be 3SUM-hard. (Often, problems on simple polygons have $O(n)$ time algorithms; very few problems on simple polygons are known to have superlinear lower bounds.)

- We study the *unrestricted* star-shaped hull problem, providing an algorithm that runs in $O(n^2p(n))$ time, where $p(n)$ is the time needed to find the roots of two equations in two unknowns, each a polynomial of degree $O(n)$.

- We study also the monotone hull problem in which we must compute a minimum-area polygon P^* that is monotone with respect to some direction (the direction is to be chosen to minimize the area) and that contains a given simple polygon P. If P^* is restricted to use a subset of the vertices of P, then we obtain an $O(n \log n)$ time algorithm for computing P^*, and we show this bound to be tight by proving an $\Omega(n \log n)$ lower bound. We also consider the unrestricted monotone hull problem, which we solve in $O(nq(n))$ time, where $q(n)$ is the time needed to find the roots of two polynomial equations in two unknowns with degrees 2 and $O(n)$.

Related Work. Chang and Yap [3, 4] have studied optimization problems for inclusion and enclosure problems, and have obtained polynomial-time algorithms for finding a maximum-area convex subset of a polygon, a minimum-area enclosing convex k-gon, and a minimum enclosing k-gon of a fixed shape. Other convex optimal enclosure problems, particularly those defined by point sets, have been studied in [1, 10]. Fleischer et al. [12] have studied problems of simultaneously approximating a shape from outside and inside. Finally, Toussaint and ElGindy [22] have examined monotone hulls of simple polygons in the context of computing convex hulls.

2 Star-Shaped Hulls

In this section we consider the minimum-area star-shaped hull problem. A set S is *star-shaped with respect to point* $z \in S$ if for every point $w \in S$ the line segment (z, w) is contained in S. We say that S is *star-shaped* if it is star-shaped with respect to some point z, and we refer to such a point z as a *center* point of S. The set of all such center points is called the *kernel* of S. Let P be the given polygon, and P^* be a desired minimum-area star-shaped polygon containing P. Notice that in the restricted version all vertices of P^* must be vertices of P, and thus P^* is always contained in the convex hull $CH(P)$ of P. However, in the unrestricted version P^* is not necessarily contained in $CH(P)$, and the kernel of P^* need not be contained in $CH(P)$ either (see Fig. 1). We denote by $P'(z)$ the minimum-area star-shaped polygon containing P such that z lies in the kernel of $P'(z)$, for any fixed point z in the plane.

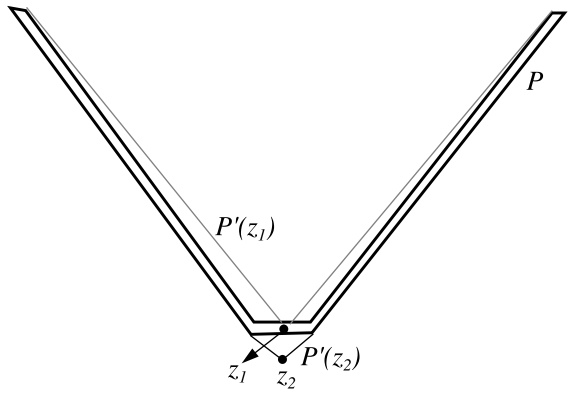

Figure 1: An example of the minimum-area unrestricted star-shaped hull, whose kernel is not contained in the convex hull $CH(P)$ of P. Here $P'(z_1)$, with point z_1 in its kernel, is the union of P and the two triangles each bounded by a dotted segment and the boundary of P, and $P'(z_2)$, with point z_2 in its kernel, is the union of P and the bottom triangle having z_2 as a vertex and two thin solid segments incident on z_2 as two sides. Note that by stretching the two 'arms' of P we can make the area of $P'(z_2)$ smaller than that of $P'(z_1)$, and hence the minimum-area P^* (and its kernel) is not contained in $CH(P)$.

2.1 3SUM-Hardness

First we show that the restricted version of the minimum-area star-shaped hull problem is 3SUM-hard. Gajentaan and Overmars [13] describe a large class of 3SUM-hard problems that are suspected to admit no solutions in $o(n^2)$ time. In particular, these problems are proved to be at least as hard as the following base problem 3SUM: *Given a set S of n integers, are there three elements of S*

that sum up to 0? The best known algorithm for 3SUM takes $\Theta(n^2)$ time. Erickson and Seidel [11] have obtained an $\Omega(n^2)$ lower bound for 3SUM but only under a very weak model of computation. Among those 3SUM-hard problems is the problem POINT-ON-3-LINES, described below. We show that our problem is at least as hard as (a slightly restricted version of) POINT-ON-3-LINES.

Theorem 1 *The restricted version of the minimum-area star-shaped hull problem is* 3SUM*-hard.*

Proof. We reduce the following to our problem: Given a set S of n lines in the plane, are there some three lines of S that pass through a common point? This problem, known as POINT-ON-3-LINES, is known to be 3SUM-hard [13].

We assume that the slopes and the y-intercepts of the lines in S are integers. We claim that this assumption does not affect the 3SUM-hardness of the problem. Indeed, the hardness proof in [13] proceeds by showing that 3-POINTS-ON-LINE is 3SUM-hard, and that POINT-ON-3-LINES is the exact dual of 3-POINTS-ON-LINE. Although not explicitly stated, what is actually shown for the former is that the *grid-point version* of 3-POINTS-ON-LINE, where the given points are grid points, is 3SUM-hard. Using a dualization that preserves integrality (e.g., mapping line $y = mx + b$ to point (m, b) and *vice versa*), the integrality assumption implies that our restricted version of POINT-ON-3-LINES becomes the dual of the grid-point version of 3-POINTS-ON-LINE, and thus is still 3SUM-hard.

Now we begin the reduction to our hull problem. Our plan is to construct a rectangle R with area C and cut off a little notch $\triangle_i$ of constant area c for each line $l_i \in S$ such that for the resulting simple polygon P the hull P^* has area at most $C - 3c$ if and only if at least three lines of S pass through the same point.

Let $\mathcal{A}$ be the arrangement of the lines of S. A *cell altitude* in $\mathcal{A}$ is a line segment, (v, u), joining a vertex v of $\mathcal{A}$ to a point u on an edge, e, of $\mathcal{A}$, such that (v, u) lies within some (convex) cell (2-face) of $\mathcal{A}$ and (v, u) is perpendicular to e. By the integrality assumption, it is easy to see that, in time $O(n)$, we can identify a small positive constant $\epsilon > 0$ such that the smallest cell altitude in $\mathcal{A}$ is larger than ϵ; in particular, we can take ϵ to be any positive constant less than $\frac{1}{\overline{m}\cdot\sqrt{1+\overline{m}^2}}$, where $\overline{m}$ is the maximum (integer, positive) slope among the slopes of the lines of S. (A cell altitude, h, determined by a vertex v and edge e can be written as $h = H/\sqrt{1+m_e^2} \geq H/\sqrt{1+\overline{m}^2}$, where H is the *vertical* distance from v to the line, of slope m_e, through e. Then we just note that $H = |K/(m_1 - m_2)| \geq 1/\overline{m}$, where $K \geq 1$ is an integer determined by the six parameters of the three lines determining the altitude, and m_1 and m_2 are the integer slopes of the two lines intersecting at the vertex v that determines h.) This means that if we fatten each line by thickness ϵ, no cell of $\mathcal{A}$ disappears. Also, we compute the convex hull, *CH*, of the vertices of $\mathcal{A}$ in $O(n \log n)$ time [2]. Without loss of generality, we assume that all lines of S are non-horizontal; this is always possible since we can rotate the coordinate system and the transformation takes $O(n)$ time. We then construct rectangle R large enough so that (i) R contains all vertices of *CH* (and thus all vertices of $\mathcal{A}$), and (ii) all lines of S intersect R at the top and bottom sides, and (iii) the topmost vertex of $\mathcal{A}$ is below the top side of R by distance y_0 for some constant $y_0 > 0$. Note that given the convex hull *CH*, such R can be constructed in $O(n)$ time. We denote by s the height of R, and B and T the bottom and top sides of R, respectively.

We specify, for each line l_i, the notch $\triangle_i$ to be cut off from R (see Fig. 2). Suppose l_i intersects B at u. We make two lines l_i' and l_i'' parallel to l_i such that their intersections u_1 and u_2 with B are to the left and right of u by distance $\epsilon/2$. Let l_i' and l_i'' intersect T at v_1 and v_2, respectively. If the slope of l_i is positive (see Fig. 2(a)), then we add segment (u_1, v_2), and make a horizontal line h below T by distance y', where y' is a positive constant chosen to satisfy $y' < y_0$ and $y' < s/2$.

Let p be the intersection between (u_1, v_2) and line h. We construct a line passing through u_2 and p, intersecting T at point t. Then $\triangle_i$ is taken as $\triangle tpv_2$. The construction for the case where the slope of l_i is negative is symmetric (see Fig. 2(b)): we add segment (u_2, v_1), intersect (u_2, v_1) with line h at p, obtain point t by intersecting T with the line passing through u_1 and p, and take $\triangle_i$ as $\triangle tpv_1$.

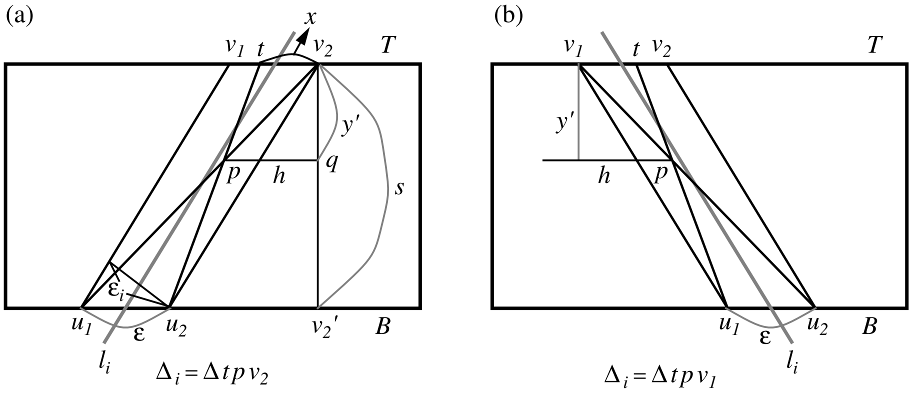

Figure 2: Construction in the proof of the 3SUM-hardness in Theorem 1: (a) the slope of line l_i is positive; (b) the slope of l_i is negative.

We claim that each notch $\triangle_i$ constructed above has the same area. To see this, consider the case where the slope of l_i is positive; the same argument applies to the other case. We refer to Fig. 2(a). Let (v_2, v_2') be the segment passing through v_2 and orthogonal to T; (v_2, v_2') intersects line h and side B at points q and v_2', respectively. Triangles $\triangle v_2pt$ and $\triangle u_1pu_2$ are similar, and segments (q, v_2) and (q, v_2') are a pair of corresponding altitudes. Thus we have $\frac{x}{\epsilon} = \frac{y'}{s-y'}$. Since y' is a constant, x is a constant, and thus the area of $\triangle_i$, given by $\frac{1}{2}xy'$, is also a constant c, for each i. This completes the proof of the claim.

Since all intersection points of lines of S are below T by distance at least $y_0 > y'$, they are all below point p. Thus no notch $\triangle_i$ interferes the intersection points. Also, since $y' < \frac{1}{2}s$, we have $x = \frac{y'}{s-y'}\epsilon < \epsilon$, and thus $\triangle_i$ is entirely inside the gap g_i delimited by (v_1, u_1) and (v_2, u_2). Notice that the distance ϵ_i between (v_1, u_1) and (v_2, u_2) is at most ϵ. If we replace each line l_i with gap g_i, then this is the same as fattening each l_i with thickness $\epsilon_i \leq \epsilon$, and thus no cell of $\mathcal{A}$ will disappear. This shows that above horizontal line h, all gaps g_i are well separated from each other, so that no notches $\triangle_i$ intersect each other. Recall that $P'(z)$ is the minimum-area star-shaped polygon containing P such that z lies in the kernel of $P'(z)$, for any fixed point z. It is easy to see that for the restricted version, $P'(z)$ does not contain any part of $\triangle_i$ if z lies in $\triangle pu_1u_2$, and $P'(z)$ contains the entire $\triangle_i$ otherwise. Call this triangle $\triangle pu_1u_2$ associated with line l_i the *wedge* w_i of l_i. Consider three lines l_i, l_j and l_k of S that are pairwise intersecting. If they pass through the same point, then clearly there exists a point $z \in w_i \cap w_j \cap w_k$, so that the area of $P'(z)$ is at most $C - 3c$. If the three lines do not pass through the same point, then since fattening the lines does not make any cell of $\mathcal{A}$ disappear, we have that $w_i \cap w_j \cap w_k = \emptyset$, so that there is no point z with $P'(z)$ excluding *all* three notches $\triangle_i, \triangle_j$, and $\triangle_k$. Therefore there are at least three lines of S passing through the same point if and only if the area of P^* is at most $C - 3c$. The theorem then follows by noting that the steps in our reduction take $O(n \log n)$ time in total. □

We comment that the above proof does not apply to the unrestricted version of the problem. In fact we do not know of any non-trivial hardness results for that version of the problem.

2.2 Efficient Algorithms

We give algorithms for computing P^* for both restricted and unrestricted versions. Recall that for any fixed point z in the plane, $P'(z)$ is the minimum-area star-shaped polygon containing P such that z lies in the kernel of $P'(z)$.

Our overall strategy for both restricted and unrestricted versions is as follows:

1. Partition the entire plane into *cells* such that for all points z lying in the same cell, the polygons $P'(z)$ are all *structurally equivalent* (to be defined later in Section 2.2.1). We call this partitioning the *equi-structure cell decomposition.*

2. For each cell C, compute the polygon (denoted by $P'(C)$) whose area is the minimum among all polygons $P'(z)$ with $z \in C$. Look at polygons $P'(C)$ for all cells C and report the one with the minimum area as a minimum-area hull P^*.

We elaborate the above steps in the following sections.

2.2.1 The Equi-structure Cell Decomposition

In the unrestricted version $P'(z)$ can be obtained as follows (see Fig. 3(a)): For each *convex* vertex v of P, we construct ray (z, v) emanating from z. Suppose ray (z, v) intersects the boundary of P at v' right before reaching v (v' does not exist if segment (z, v) lies entirely inside P, e.g., (z, v_1) in Fig. 3(a)). If ray (z, v), after passing through v, can go to infinity without being blocked by P, then we include v as a vertex of $P'(z)$; point v' is also included as a vertex of $P'(z)$ if segment (v', v) lies outside P. Then $P'(z)$ is obtained by traversing such vertices v and v' in order along the boundary of P. Additionally, z is itself a vertex of $P'(z)$ if z can see to infinity; see Fig. 1. The vertices, such as v' and possibly z, of $P'(z)$ that are not vertices of P are called the *augmented* vertices; the other vertices, v, of $P'(z)$ are called the *original* vertices. We adopt the convention that whenever we talk about an augmented vertex v', it is implied that v' is generated by ray (z, v) for some original vertex v in the above process.

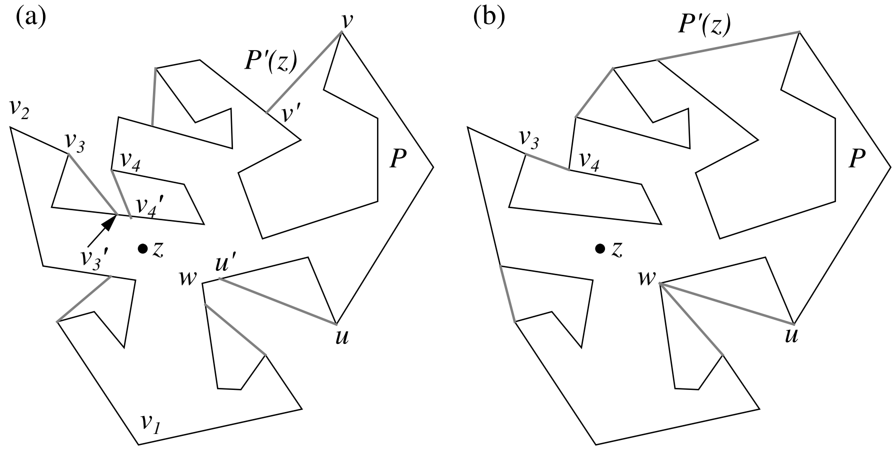

Figure 3: Minimum-area star-shaped polygon $P'(z)$ containing P with point z lying in the kernel of $P'(z)$: (a) unrestricted version; (b) restricted version in which the vertices of $P'(z)$ are constrained to be vertices of P.

For two points z_1 and z_2, we say that $P'(z_1)$ and $P'(z_2)$ are *structurally equivalent* (or simply *equivalent*) if they have the same sequence of original vertices, and their corresponding augmented vertices lie in the interior of the same edge of P. Clearly, as we move point z around, the structure of $P'(z)$ changes when and only when one of its augmented vertices, v', becomes some original vertex, u. Recall from our description of $P'(z)$ that z is on the line defined by u and v, and that ray (u, v) goes to infinity without being blocked by P. Then ray (u, v) is just one of the two boundaries of the *visibility cone* $Cone(u)$ with apex at u that defines the maximum angular range in which u can 'see' to infinity. Thus the only critical events for $P'(z)$ to change its structure are those when z moves across some line extension of a ray of $Cone(u)$, for some vertex u of P. Therefore, we can obtain the desired equi-structure cell decomposition of the plane by extending the two rays of $Cone(u)$ into two lines, for each vertex u, and build an arrangement of complexity $O(n^2)$ by intersecting the $O(n)$ lines. We are exploiting the fact that, since a vertex u gives rise to at most one cone of directions that see to infinity, and only the defining lines of these cones can give rise to critical events, we get an arrangement of only $O(n)$ lines, versus the naive bound of $O(n^2)$ lines that are determined by all pairs of vertices. (The $O(n^2)$ lines are necessary, though, to define the critical events in the case of computing hulls of sets of disconnected polygons; see the remarks in Section 4.)

Now our task is to compute the visibility cone $Cone(u)$ for each vertex u of P. In general, $Cone(u)$ does not exist if there is no way for u to see to infinity. Also, both rays of $Cone(u)$ are extensions of the edges of the convex hull $CH(P)$ incident on u when u is on $CH(P)$. To compute all cones $Cone(u)$, we do the following.

Algorithm *Vis-Cones*

1. Compute the convex hull $CH(P)$ of P.
2. For each vertex $u \in CH(P)$, extend the two edges of $CH(P)$ incident on u into two rays.
3. For each edge $e \in CH(P) \setminus P$, put a 'segment light' on e and light the boundary of P. For each vertex $u \in P \setminus CH(P)$ that is lit (i.e., $Cone(u)$ exists), compute $Cone(u)$. This entire step can be done in $O(n)$ time, using the algorithm of [14, 23].

By the above steps we obtain the $O(n)$ lines defining the critical events in $O(n)$ time, and by computing the arrangement of the lines we obtain the equi-structure cell decomposition in $O(n^2)$ time.

Lemma 2 *Given an n-vertex simple polygon P, for the unrestricted version we can partition the plane into $O(n^2)$ cells in $O(n^2)$ time such that for all points z in the same cell, the polygons $P'(z)$ are all equivalent.*

Now consider the restricted version in which the vertices of $P'(z)$ are constrained to be vertices of P. We can obtain the restricted $P'(z)$ from the unrestricted $P'(z)$ as follows (see Fig. 3(a)(b)): For each edge (u, u') in the unrestricted $P'(z)$ where u' is an augmented vertex in the interior of an edge e of P, we slide u' along e in the direction which keeps $P'(z)$ still containing the entire segment (z, u) (and thus enlarges the area of $P'(z)$), until the endpoint w of e is reached; we then replace chain (u, u', w) in the boundary of the unrestricted $P'(z)$ with edge (u, w). If before reaching endpoint w during sliding we encounter another augmented vertex v', then we replace the chain (u, u', v', v) with edge (u, v) (for example, chain (v_3, v_3', v_4', v_4) of $P'(z)$ in Fig. 3(a) is replaced by edge (v_3, v_4) in Fig. 3(b)). An equivalent description for obtaining restricted $P'(z)$ from unrestricted $P'(z)$ is that we traverse the vertex sequence of unrestricted $P'(z)$ and "short-cut" the tour by skipping all augmented vertices.

Recall that two equivalent polygons $P'(z_1)$ and $P'(z_2)$ differ only by the positions of their augmented vertices. But in the restricted version there are no augmented vertices, therefore $P'(z_1)$ and $P'(z_2)$ are equivalent if and only if they are identical. Clearly, the equi-structure cell decomposition computed in Lemma 2 for the unrestricted version is also an equi-structure cell decomposition for the restricted version. Moreover, all polygons $P'(z)$ with z in the same cell are identical.

Corollary 3 *Given an n-vertex simple polygon P, for the restricted version we can partition the plane into $O(n^2)$ cells in $O(n^2)$ time such that for all points z in the same cell, the polygons $P'(z)$ are all identical.*

2.2.2 Minimizing the Area

We discuss how to obtain a minimum-area hull P^*, after computing the equi-structure cell decomposition. First of all, for each cell C, we want to compute the *structure* of polygon $P'(C)$, i.e., the vertex sequence of $P'(z)$, the original vertices of $P'(z)$, and the edges of P in which the augmented vertices of $P'(z)$ lie, for any point $z \in C$. We carry out this task by computing $P'(z)$ with z at each of the $O(n^2)$ vertices of the cells. We start at any point z. For each convex vertex v of P, we perform a *ray shooting query* to see if ray (z, v) goes to infinity after passing through v (i.e., if the ray emanating from v with the same direction goes to infinity), and if so, we perform again a ray shooting query to obtain the edge of P first hit by the ray (v, z). Each ray shooting query can be performed in $O(\log n)$ time after preprocessing P [5, 6], and thus $P'(z)$ is obtained in $O(n \log n)$ time. Now as we walk z from one cell vertex to an adjacent vertex, the structure of $P'(z)$ only has $O(1)$ changes, and we can detect and perform this change also in constant time. We store the changes incrementally, as we walk through all the vertices of the cells, so the entire task is done in $O(n^2)$ time. For the restricted version, we carry out this task in the same way, except that we also have to skip the augmented vertices in the vertex sequence of each $P'(z)$. Clearly we can still perform this task in $O(n^2)$ time. In the following, we discuss the objective area function to be minimized.

Restricted Version
In the restricted version, since polygons $P'(z)$ are identical for all points z in the same cell, we only need to compute the area of $P'(z)$ with z at each of the $O(n^2)$ vertices of the cells, and then take the minimum among them. To compute the area of $P'(z)$, we decompose $P'(z)$ into at most n triangles by adding segments connecting z to all vertices of $P'(z)$. Suppose $\triangle_i = (z, v_i, v_{i+1})$ is one such triangle, and the coordinate of z is $(\bar{x}, \bar{y})$. The area of $\triangle_i$, denoted by $f_i(\bar{x}, \bar{y})$, is given by $\frac{1}{2} \cdot |\overline{v_i v_{i+1}}| \cdot h_i$, where h_i is the distance from z to the line l_i defined by v_i and v_{i+1}. Note that v_i and v_{i+1} are vertices of P, so $|\overline{v_i v_{i+1}}|$ is a constant, and line l_i is given by $y = m_i x + b_i$ for some constants m_i and b_i. It is easy to see that $f_i(\bar{x}, \bar{y}) = A_i \bar{x} + B_i \bar{y} + C_i$ for some constants A_i, B_i and C_i, so that the area of $P'(z)$, denoted by $F(\bar{x}, \bar{y})$, is given by $F(\bar{x}, \bar{y}) = \sum_{i=1}^{n} f_i(\bar{x}, \bar{y})$. After computing $F(x, y)$ at $z = (\bar{x}, \bar{y})$, the function value at another point $z' = (\bar{x} + \Delta x, \bar{y} + \Delta y)$ is obtained by $F(\bar{x} + \Delta x, \bar{y} + \Delta y) = \sum_{i=1}^{n}(A_i \bar{x} + B_i \bar{y} + C_i) + \sum_{i=1}^{n} A_i \Delta x + \sum_{i=1}^{n} B_i \Delta y = F(\bar{x}, \bar{y}) + (\sum_{i=1}^{n} A_i)\Delta x + (\sum_{i}^{n} B_i)\Delta y$. Note that $\sum_{i=1}^{n} A_i$ and $\sum_{i=1}^{n} B_i$ can be precomputed, so $F(\bar{x} + \Delta x, \bar{y} + \Delta y)$ is obtained in $O(1)$ time. Of course, the area function $F'(x, y)$ for vertex z' adjacent to vertex z in the cells is somewhat different from $F(x, y)$, but differs only by $O(1)$ terms since only $O(1)$ triangles $\triangle_i$ are different. Therefore, as we walk from one vertex of the cells to another, only $O(1)$ terms of A_i, B_i and C_i are different in the area function, and we can store the difference incrementally. In this way, we evaluate the area of $P'(z)$ for each subsequent vertex z in $O(1)$ time each, and thus obtain the minimum-area hull P^* in $O(n^2)$ time after walking through all $O(n^2)$ vertices of the cells.

Theorem 4 *Given an n-vertex simple polygon P, we can compute a minimum-area star-shaped polygon P^* containing P such that all vertices of P^* are original vertices of P in $O(n^2)$ time.*

Unrestricted Version

In the restricted version, we have the property that there is always a vertex z of some cell giving $P'(z) = P^*$. But in the unrestricted version this is not the case. In fact, there exist examples in which any optimal P^* must have its kernel point z lying in the *interior* of some cell (see Fig. 4). Therefore for each cell C we have to consider its interior points as well.

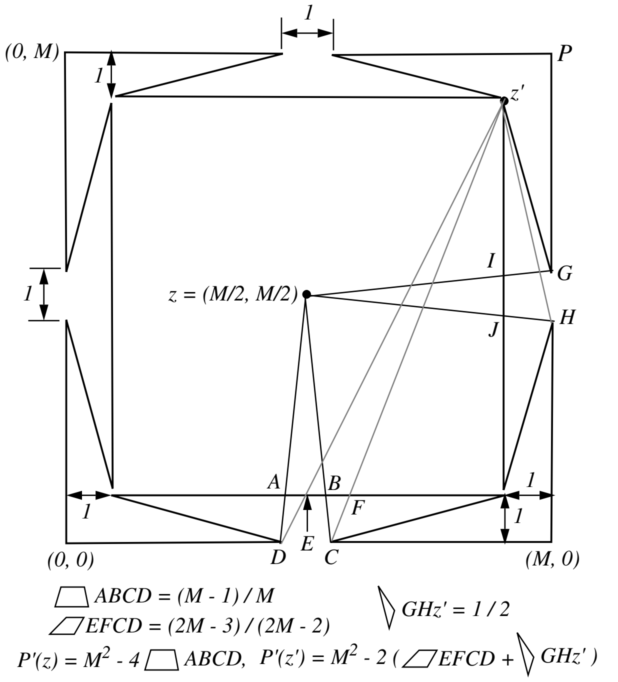

Figure 4: Example in which any optimal P^* must have its kernel point lying in the *interior* of some cell: z is an interior point of the cell which is the inner square, and M is a large number. For any vertex z' of this cell, $P'(z')$ has area larger than $P'(z)$. In particular, $area(ABCD) - area(EFCD) = (-M+2)/[2M(M-1)] \approx -1/(2M)$ but $area(GHJI) - area(GHz') = 1/2 - 1/M \approx 1/2$, and thus $P'(z)$ saves more area than $P'(z')$.

We use the locus approach, letting $z = (\bar{x}, \bar{y})$ be any point in cell C, and compute the area function of $P'(z)$. Again, we decompose $P'(z)$ into $O(n)$ triangles by adding segments connecting z to all vertices of $P'(z)$. Consider the area function $f_i(\bar{x}, \bar{y})$ of one such triangle $\triangle_i = (z, v_i, v_{i+1})$. If v_i and v_{i+1} are both original vertices, then as we have seen in the restricted version $f_i(\bar{x}, \bar{y}) = A_i\bar{x} + B_i\bar{y} + C_i$ for some constants A_i, B_i and C_i. But if one or both of v_i and v_{i+1} are augmented vertices, $f_i(\bar{x}, \bar{y})$ becomes more complicated. Consider the more general case in which both v_i and v_{i+1} are augmented vertices. Then recall from Section 2.2.1 that v_i and v_{i+1} lie on the same edge e of P and they are generated by rays (z, u_1) and (z, u_2) for some original vertices u_1 and u_2 of P (i.e., $v_i = u_1'$ and $v_{i+1} = u_2'$ in our notational convention); let $y = mx + b$ be the line equation of e, for some constants m and b. To see what $f_i(\bar{x}, \bar{y})$ looks like, assume that z is above e, $u_1 = (0, 0)$, and $u_2 = (a, 0)$, for some constant a. Then it can be shown that $f_i(\bar{x}, \bar{y}) = \frac{a\bar{y}(\bar{y} - m\bar{x} - b)^2}{2(\bar{y} - m\bar{x})(\bar{y} - m\bar{x} + am)}$, i.e., $f_i(\bar{x}, \bar{y}) = g_i(\bar{x}, \bar{y})/h_i(\bar{x}, \bar{y})$, where g_i and h_i are polynomials in two variables with degrees 3 and 2, respectively. The area function of $P'(z)$ is then given by $F(\bar{x}, \bar{y}) = \sum_i f_i(\bar{x}, \bar{y}) = G(\bar{x}, \bar{y})/H(\bar{x}, \bar{y})$, where G and H are polynomials in two variables with degree $O(n)$ each. To minimize F, we take two derivatives of F with respect to $\bar{x}$ and to $\bar{y}$, set them to 0, and find the roots. Here we have two equations in two unknowns, each a polynomial of degree $O(n)$. Let $p(n)$ be the time needed for the best known algorithm to find the $O(n^2)$ roots; $p(n)$ is some polynomial in n. After finding the roots, we then evaluate the value of the area function F at these $O(n^2)$ points, and find the minimum. Clearly the evaluation time is dominated by $p(n)$. We perform this process for each cell C and then find the global minimum; the total running time is $O(n^2 p(n))$.

Theorem 5 *Given an n-vertex simple polygon P, we can compute an unrestricted minimum-area star-shaped polygon P^* containing P in $O(n^2 p(n))$ time, where $p(n)$ is the time needed to find the roots of two equations in two unknowns, each a polynomial of degree $O(n)$.*

3 Monotone Hulls

In this section we consider the minimum-area monotone hull problem. Given a line l, a polygonal chain γ is *monotone with respect to l* if any line orthogonal to l intersects γ in a single point or in a single interval or not at all. A simple polygon Q is *monotone with respect to l* if its boundary can be partitioned into two chains that are both monotone with respect to l. Projecting all points of Q on l to obtain an interval on l, the two vertices of Q that yield the endpoints of the interval are called the *extreme vertices* of Q with respect to l. Given a simple polygon P and a line l, we denote by $P'(l)$ the minimum-area polygon that contains P and is monotone with respect to l. Again, in the restricted version the vertices of $P'(l)$ must be the original vertices of P, and in the unrestricted version this constraint is removed. In the minimum-area monotone hull problem, our goal is to minimize $P'(l)$ over all choices of l.

3.1 Lower Bound

First we show that the restricted version of the minimum-area monotone hull problem has a lower bound of $\Omega(n \log n)$.

Theorem 6 *Given an n-vertex simple polygon P, computing a minimum-area monotone polygon P^* containing P such that all vertices of P^* are the original vertices of P requires $\Omega(n \log n)$ time in the worst case.*

Proof. The following *ϵ-closeness problem* is known to require $\Omega(n \log n)$ time in the worst case [21]: Given $\epsilon > 0$ and n real numbers $x_1, x_2, \cdots, x_n$, determine whether any two x_i and x_j $(i \neq j)$ are at distance at most[1] ϵ. We reduce the ϵ-closeness problem to our problem as follows.

Without loss of generality, we assume that $x_1, x_2, \cdots, x_n$ are positive. We create a large rectangle R, and cut off n little triangular pieces $\triangle_i$ from the top side of R, each with the same constant area c (see Fig. 5(a)). We call the resulting polygon P. Triangle $\triangle_i$ is decided by x_i and ϵ, for each i. The top side of $\triangle_i$ lies on the top side of R. The right side of $\triangle_i$ is a segment with slope x_i, and the left side is a segment with slope $(x_i + \epsilon)$. The angle θ_i defined by the left and right sides of $\triangle_i$ is then given by $\theta_i = \tan^{-1}(x_i + \epsilon) - \tan^{-1} x_i$. Now we can completely specify $\triangle_i$ once the length t_i of its left side is decided. But the area of $\triangle_i$ is $c = \frac{1}{2}(a_i + b_i) t_i \sin\theta_i$, where $a_i = t_i \cos\theta_i$ and $(t_i \sin\theta_i)/b_i = \tan\phi_i = x_i$ (see Fig. 5(b)). Therefore $c = \frac{1}{2} t_i^2 \sin\theta_i (\cos\theta_i + \frac{\sin\theta_i}{x_i})$, which determines t_i. Finally, we make the triangles pairwise disjoint, say, by making the

[1]It is "less than ϵ" in the original statement. But changing it to "at most ϵ" does not affect the lower bound.

upper-right vertices of $\triangle_i$ and of $\triangle_{i+1}$ separated by distance $2r$, where r is the maximum among all right side lengths r_i, so that the triangles do not intersect each other (see Fig. 5(a)). Also, we make R large enough to contain all the triangles. Clearly, this construction takes $O(n)$ time.

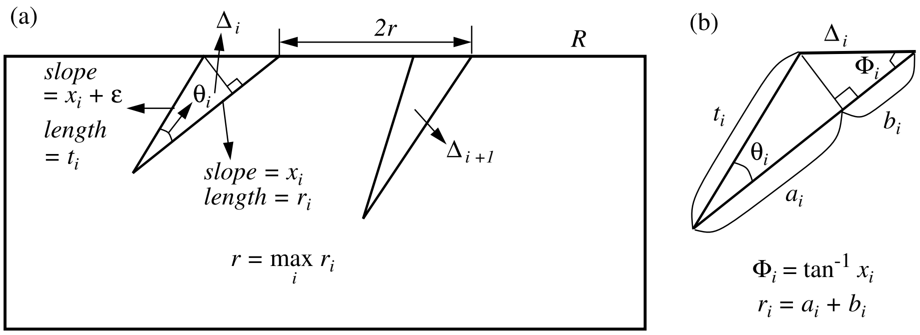

Figure 5: Construction in the proof of the $\Omega(n \log n)$ lower bound in Theorem 6: (a) constructing polygon P; (b) computing the area of $\triangle_i$.

Let l be a line and h be any line orthogonal to l. Given P constructed above, polygon $P'(l)$ (in the restricted version) does not contain any part of triangle $\triangle_i$ if the slope of h is between x_i and $x_i + \epsilon$, and contains the entire $\triangle_i$ otherwise. Let the area of R be some constant C. Then there are x_i and x_j ($i \neq j$) within distance ϵ if and only if the area of P^* is at most $C - 2c$. This shows that if we can solve our problem in time T, then we can solve the ϵ-closeness problem in time $O(n) + T$. The theorem then follows by the $\Omega(n \log n)$ lower bound of the ϵ-closeness problem. $\square$

We comment that the above proof does not apply to the unrestricted version of the problem. In fact we do not know of any non-trivial lower bounds for that version of the problem.

3.2 Efficient Algorithms

Now we discuss how to solve the problem efficiently. Recall that $P'(l)$ is the minimum-area polygon which contains P and is monotone with respect to l, for a given line l. In the unrestricted version, such $P'(l)$ is obtained as follows (see Fig. 6(a)). Consider a *convex* vertex v of P whose two incident edges form a nonmonotone chain with respect to l and for which v sees to infinity in exactly one of the two directions orthogonal to l. (Thus, v is not one of the two extreme vertices of P with respect to l.) We construct a ray emanating from v, orthogonal to l and directed toward the interior of P; we denote by v' the first intersection point between this ray and the boundary of P. Then $P'(l)$ is obtained by traversing the boundary of P and "short-cutting" the tour by edges (v, v'), starting from either extreme vertex with respect to l. Again we call the vertices v' of $P'(l)$ that are not vertices of P the *augmented* vertices, and the other vertices v of $P'(l)$ the *original* vertices. For the restricted version, we can obtain the restricted $P'(l)$ by short-cutting the boundary of the unrestricted $P'(l)$ using a sliding process similar to that in Section 2.2.1 (see Fig. 6(a)(b)). Again, this sliding process is equivalent to traversing the vertex sequence of unrestricted $P'(l)$ and skipping the augmented vertices.

Recall from Section 2.2.1 that the visibility cone $Cone(u)$ of a vertex $u \in P$ is the cone with apex at u and two rays that define the maximum angular range in which u can 'see' to infinity. Suppose we use Algorithm *Vis-Cones* presented in Section 2.2.1 to compute all rays in $Cone(u)$ for all vertices $u \in P$, extend the rays to lines, and sort the lines by their slopes. We denote by S the set of these lines in sorted order. Also, we use the same notion of structural equivalence

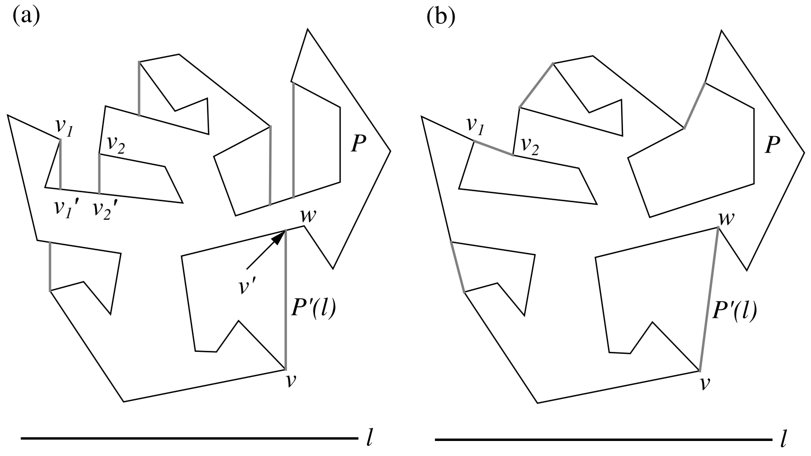

Figure 6: Minimum-area monotone polygon $P'(l)$ which contains P and is monotone with respect to line l: (a) unrestricted version; (b) restricted version in which the vertices of $P'(l)$ are constrained to be vertices of P.

defined in Section 2.2.1. From the above description of $P'(l)$, it is easy to see that as we rotate a given line l, the structure of $P'(l)$ stays unchanged when the slope of line h is between the slopes of two consecutive lines of S, where h is any line orthogonal to l. (Intuitively, the two directions of h give the directions along which we can 'see' into the 'pockets' of P.) As the size of S is $O(n)$, we have $O(n)$ critical events corresponding to structural changes of $P'(l)$. Our overall strategy for solving the monotone hull problem is similar to that for the star-shaped hull problem, with the set S serving the purpose of the equi-structure cell decomposition:

1. Compute set S described above, so that polygons $P'(l)$ with the slope of line h (orthogonal to l) between the slopes of two consecutive lines of S are all equivalent.

2. For each interval I defined by the slopes of two consecutive lines of S, compute the polygon (denoted by $P'(I)$) whose area is the minimum among all polygons $P'(l)$ with the slope of line h (orthogonal to l) in the range of I. Look at polygons $P'(I)$ for all I and report P^* as the one with the minimum area.

Restricted Version

As noted before, equivalent polygons are identical in the restricted version, so we only need to compute polygon $P'(l)$ with line l orthogonal to each line in S. We easily compute the first polygon $P'(l)$ in $O(n)$ time, simply by finding the visibility from above and below. Subsequent polygons $P'(l)$ involve $O(1)$ changes each, and can be computed incrementally in $O(1)$ time each by walking along the boundary of P. Therefore in Step 1 we perform Algorithm *Vis-Cones* in $O(n)$ time and perform a sorting in $O(n \log n)$ time to obtain set S, and in Step 2 we spend $O(n)$ time to compute the first $P'(l)$ and additional $O(n)$ time to compute all the other polygons. The overall time complexity is thus $O(n \log n)$, which is optimal by Theorem 6.

Theorem 7 *Given an n-vertex simple polygon P, we can compute a minimum-area monotone polygon P^* containing P such that all vertices of P^* are the original vertices of P, in optimal $\Theta(n \log n)$ time in the worst case.*

Unrestricted Version

The difference between restricted and unrestricted versions in the monotone hull problem is similar to the one in the star-shaped hull problem. In the restricted version, we have the property that some line h in S always gives $P'(l) = P^*$ where l is orthogonal to h. But in the unrestricted version this is no longer true. In Fig. 7 we give an example in which any optimal P^* must be some $P'(l)$ with line l *not* orthogonal to any line in S. This can be viewed as the slope of line h (orthogonal to l where $P^* = P'(l)$) being in the *interior* of an interval I defined by two consecutive slopes of the lines in S.

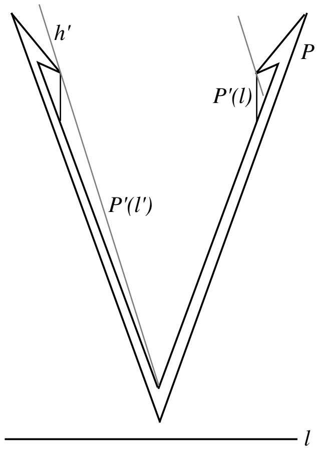

Figure 7: Example in which any optimal P^* must be some $P'(l)$ with line l *not* orthogonal to any line in S: h' is in S, and $P'(l')$ with l' orthogonal to h' is larger than $P'(l)$; in fact this occurs for each line in S.

To deal with a possible "interior" solution, we use a method similar to the locus approach. Let h be a line whose slope is anywhere inside an interval I defined by the slopes of two consecutive lines h_1 and h_2 of S, and consider $P'(l)$ where line l is orthogonal to h. Let θ be the angle such that the slope of h is $\tan\theta$; similarly define θ_1 for h_1. We want to find the value of θ that minimizes the area of $P'(l)$. By our construction, polygons $P'(l)$ and $P'(l_1)$ are equivalent, where line l_1 is orthogonal to h_1. Consider an original vertex v_i of P that generates augmented vertices v_i' and v_i'' respectively in $P'(l_1)$ and in $P'(l)$. Recall from our construction that v_i' and v_i'' lie on the same edge, say e, of P, and line (v_i, v_i') is parallel to h_1 (similarly line (v_i, v_i'') is parallel to h). The area difference between $P'(l)$ and $P'(l_1)$ is the sum of the signed area of $\triangle_i = (v_i, v_i', v_i'')$ for each such original vertex v_i. Note that $P'(l_1)$ is fixed and thus its area is a constant, say C. Then the function $F(\theta)$ of the area of $P'(l)$ is given by $F(\theta) = \sum_i f_i(\theta) + C$, where f_i is the function of the signed area of $\triangle_i$, $i \leq n$. Let H be the distance between vertex v_i and edge e, and θ_i be the angle such that a line orthogonal to e has slope $\tan\theta_i$. Then it can be shown that $f_i(\theta) = A$ or $-A$, where $A = \frac{1}{2}H^2[\tan(\theta - \theta_i) - \tan(\theta_1 - \theta_i)]$; see Fig. 8. Note that θ_i and θ_1 are constants. To find the minimum of $F(\theta)$, we take its derivative, set it to 0, and find the roots. To find the roots, we change the variables in the equation by $\sin\theta = x, \cos\theta = y$, and solve the equation with an additional equation $x^2 + y^2 = 1$. Therefore we have two polynomial equations in two unknowns with degrees 2 and $O(n)$. Let $q(n)$ be the time needed for the best known algorithm to find the $O(n)$ roots; $q(n)$ is some polynomial in n (smaller than $p(n)$). After finding the roots, we then evaluate the value of the area function F at these $O(n)$ points, and find the minimum. The evaluation time is dominated by $q(n)$. We perform this process for each interval I in S and then find the global minimum; the total running time is $O(nq(n))$.

Theorem 8 *Given an n-vertex simple polygon P, we can compute an unrestricted minimum-area*

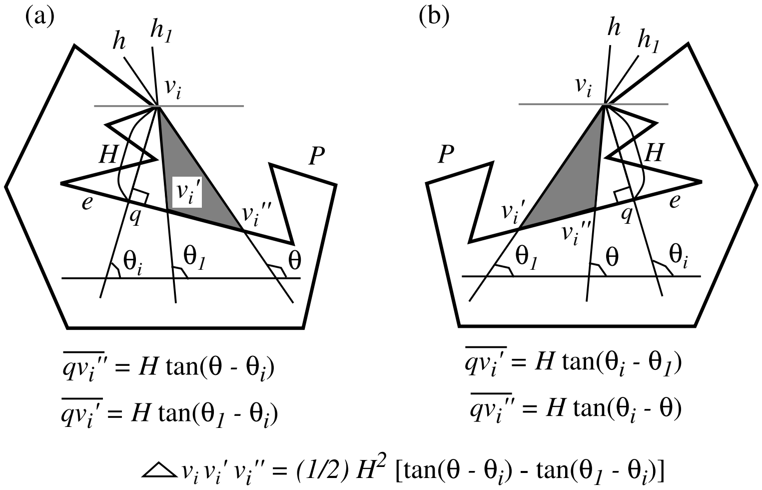

Figure 8: Computing the function $f_i(\theta)$ of the signed area of $\triangle_i = (v_i, v_i', v_i'')$: (a) both edges of P incident on v_i are on the left when walking along ray (v_i', v_i); (b) both edges of P incident on v_i are on the right when walking along ray (v_i', v_i). In (a), $\triangle_i$ belongs to $P'(l)$ but not to $P'(l_1)$, and thus $f_i(\theta) = A$, where $A = \frac{1}{2}H^2[\tan(\theta - \theta_i) - \tan(\theta_1 - \theta_i)]$; in (b), the opposite occurs, and thus $f_i(\theta) = -A$.

monotone polygon P^* containing P in $O(nq(n))$ time, where $q(n)$ is the time needed to find the roots of two polynomial equations in two unknowns with degrees 2 and $O(n)$.

4 Conclusion

We have studied some minimum-area hull problems that generalize the notion of convex hull to star-shaped and monotone hulls of simple polygons.

A natural generalization of the problem is to consider these same hull problems for disconnected sets — sets of points, or sets of (disjoint) polygons. For the unrestricted version, for a set of points, both minimum-area hulls are trivial, having zero area. The restricted version, however, can be solved using similar techniques to those discussed in this paper. For example, for a set of points, the star-shaped hull can be computed by considering how the minimum-area star-shaped polygon, $P'(z)$, varies as the center point z moves about in the plane. Changes in $P'(z)$ can only occur when z crosses a line determined by some pair of points; i.e., changes occur when z changes cells in the arrangement of size $O(n^4)$ induced by the $O(n^2)$ lines through pairs of points. (In the disconnected case, we cannot exploit the fact that each vertex has at most a single cone of directions in which it sees to infinity, as we used in the hull problem on simple polygons.) By tracking the constant-size changes to $P'(z)$ as z moves from cell to cell, we can obtain an overall complexity of $O(n^4)$ time. A similar approach can apply to the case of multiple polygons. We leave it to future work to explore this generalization further.

It would also be interesting to see if one can reduce the space complexity ($O(n^2)$) of our $O(n^2)$-time algorithm for star-shaped hulls of a simple polygon. Can topological sweep be applied to reduce the space complexity to linear?

A natural question also arises in higher dimensions: How efficiently can one compute minimum volume hulls for polyhedra or points in three (or more) dimensions?

Finally, we mention that some recent work of Majhi et al. [17] has examined a problem similar to our own, and applied a similar solution method, to compute an orientation of a simple polygon that minimizes the total "contact length" for "supports" that are constructed when the polygon is manufactured using layered manufacturing techniques, such as stereolithography.

Acknowledgements

We thank Karen Daniels for very useful discussions concerning the problems considered here, and their applications. E. Arkin is supported in part by NSF Grant CCR-9504192. Y.-J. Chiang is supported in part by NSF Grant DMS-9312098. M. Held is supported in part by NSF Grants DMS-9312098 and CCR-9504192. J. Mitchell is supported in part by Hughes Research Laboratories and NSF Grant CCR-9504192. V. Sacristan is supported in part by Projecte UPC PR-9410. S. Skiena is supported in part by NSF Grant CCR-9109289 and ONR Award 400x116yip01. T.-C. Yang was supported in part by Ministry of Education of KOREA while visiting SUNY at Stony Brook.

References

[1] E. M. Arkin, S. Khuller, and J. S. B. Mitchell. Geometric knapsack problems. *Algorithmica*, 10:399–427, 1993.

[2] M. J. Atallah. Computing the convex hull of line intersections. *J. Algorithms*, 7:285–288, 1986.

[3] J. S. Chang. Polygon optimization problems. Report 240, Comput. Sci. Div., New York Univ., New York, NY, 1986.

[4] J. S. Chang and C. K. Yap. A polynomial solution for the potato-peeling problem. *Discrete Comput. Geom.*, 1:155–182, 1986.

[5] B. Chazelle, H. Edelsbrunner, M. Grigni, L. Guibas, J. Hershberger, M. Sharir, and J. Snoeyink. Ray shooting in polygons using geodesic triangulations. In *Proc. 18th Internat. Colloq. Automata Lang. Program.*, volume 510 of *Lecture Notes in Computer Science*, pages 661–673. Springer-Verlag, 1991.

[6] B. Chazelle and L. J. Guibas. Visibility and intersection problems in plane geometry. *Discrete Comput. Geom.*, 4:551–581, 1989.

[7] K. Daniels. The restrict/evaluate/subdivide paradigm for translational containment. In *Fifth MSI Stony Brook Workshop on Computational Geometry*, 1995.

[8] K. Daniels and V. Milenkovic. Personal communication, 1995.

[9] K. Daniels and V.J. Milenkovic. Multiple translational containment. Part I: An approximation algorithm. *Algorithmica*, to appear.

[10] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger. Finding minimum area k-gons. *Discrete Comput. Geom.*, 7:45–58, 1992.

[11] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. In *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 528–536, 1993.

[12] R. Fleischer, K. Mehlhorn, G. Rote, E. Welzl, and C. K. Yap. Simultaneous inner and outer approximation of shapes. *Algorithmica*, 8:365–389, 1992.

[13] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom. Theory Appl.*, 5:165–185, 1995.

[14] L. J. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987.

[15] Z. Li. Compaction algorithms for non-convex polygons and their applications. Ph.d. Thesis, Harvard University, Division of Applied Sciences, 1994.

[16] Z. Li and V. Milenkovic. A compaction algorithm for non-convex polygons and its application. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 153–162, 1993.

[17] J. Majhi, R. Janardan, M. Smid, and P. Gupta. On some geometric optimization problems in layered manufacturing. Technical Report, Fakultät für Informatik, Otto-von-Guericke-Universität Magdeburg, Germany, January, 1997.

[18] V. Milenkovic, K. Daniels, and Z. Li. Automatic marker making. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 243–246, 1991.

[19] V. Milenkovic, K. Daniels, and Z. Li. Placement and compaction of nonconvex polygons for clothing manufacture. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 236–243, 1992.

[20] D. M. Mount and R. Silverman. Packing and covering the plane with translates of a convex polygon. *J. Algorithms*, 11:564–580, 1990.

[21] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[22] G. T. Toussaint and H. ElGindy, A counterexample to an algorithm for computing monotone hulls of simple polygons. *Pattern Recogn. Lett.*, 1:219–222, 1983.

[23] G. T. Toussaint. A linear-time algorithm for solving the strong hidden-line problem in a simple polygon. *Pattern Recogn. Lett.*, 4:449–451, 1986.