Covering a Simple Polygon by Monotone Directions*

Hee-Kap Ahn[†] Peter Brass[‡] Christian Knauer[§] Hyeon-Suk Na[¶]
Chan-Su Shin[‖]

January 11, 2009

Abstract

In this paper we study the problem of finding a set of k directions for a given simple polygon P, such that for each point $p \in P$ there is at least one direction in which the line through p intersects the polygon only once. For $k = 1$, this is the classical problem of finding directions in which the polygon is monotone, and all such directions can be found in linear time for a simple n-gon. For $k > 1$, this problem becomes much harder; we give an $O(n^5 \log^2 n)$-time algorithm for $k = 2$, and $O(n^{3k+2})$-time algorithm for $k \geq 3$.

1 Introduction

Background A polygon P is said to be *monotone* in a direction α if every line in direction α intersects P in at most one connected component. Determining if a given polygon is monotone is a well-studied problem. Preparata and Supowit [7] presented a linear time algorithm to find all directions in which a given polygon is monotone. Bose and Kreveld [1] studied rotational plane sweep on a simple polygon with the restriction that the sweep line intersects the polygon in at most one connected component.

In this paper we consider a generalization of the monotonicity problem: given a simple polygon P, find a set $\mathcal{D}$ of k directions such that each point $p \in P$ is *covered* by at least one direction of $\mathcal{D}$, in the sense that the line through p in this direction intersects P in one connected component. We call a simple polygon having such a set $\mathcal{D}$ of k directions k*-monotone* for $\mathcal{D}$. Figure 1 illustrates some examples of polygons; the polygon in (a) is 1-monotone for the vertical direction, the polygon in (b) is not 1-monotone but 2-monotone for the horizontal and the vertical directions, and the polygon in (c) is not k-monotone for *any* $k \geq 1$ since any line through a point in the gray region intersects the polygon in at least two connected components.

To our best knowledge, no related work has been done so far, except those of [7] and [1] introduced above. Since every 1-monotone polygon is decomposable into two monotone chains, one may think that decomposing the boundary of a simple polygon into k monotone chains is

[*] Work by Ahn was supported by the Postech BSRI Research Fund–2008. Work by Na was supported by Korean Research Foundation Grant(KRF-2007-531-D00018). Work by Shin was supported by Korean Research Foundation Grant(KRF-2006-311-D00764).

[†] Department of Computer Science and Engineering, POSTECH, Korea. heekap@postech.ac.kr

[‡] Department of Computer Science, City College, New York, USA. peter@cs.ccny.cuny.edu

[§] Institute of Computer Science, Free University Berlin, Germany. knauer@inf.fu-berlin.de

[¶] Corresponding author. School of Computing, Soongsil University, Seoul, Korea. hsnaa@ssu.ac.kr

[‖] School of Electrical and Information Engineering, Hankuk University of Foreign Studies, Korea. cssin@hufs.ac.kr

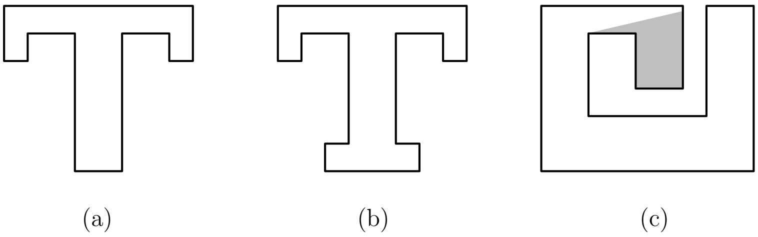


Figure 1: The polygon is (a) 1-monotone for the vertical direction, (b) 2-monotone for the horizontal and the vertical directions, (c) not k-monotone for any $k \geq 1$.

relevant to k-monotonicity. Indeed, decomposing the boundary of a simple polygon into some specific polygonal chains has been studied to determine the moldability and the castability of a polygon; Rappaport and Rosenbloom [8] gave a linear time algorithm to determine whether the boundary of a simple polygon can be decomposed into two monotone chains, where the two chains are not necessarily monotone in the same direction. Bose et al. [2] studied different type of decomposition of the boundary of a polygon; given a center r, one chain can be rotated in clockwise direction around r and the other can be rotated in counterclockwise direction around r, without either chain penetrating the interior of the polygon. However, these decompositions are barely related to k-monotonicity. For instance, some 2-monotone polygons like Figure 1(b) are not decomposable into two monotone chains, and some polygons decomposable into two monotone chains are not 2-monotone.

Monotonicity of three dimensional polyhedron was introduced by Toussaint [9], and studied by Bose and Kreveld [1], and Ha et al. [4].

Summary of results. We obtain the following results.

Theorem 1 *Given a direction α and a simple polygon P with n vertices, we can test in $O(n^2 \log^2 n)$ time if there is a direction β such that P is 2-monotone for $\{\alpha, \beta\}$. We return all directions β such that P is 2-monotone for $\{\alpha, \beta\}$ in the same time.*

Theorem 2 *Given a simple polygon P with n vertices in general position, we can find in $O(n^5 \log^2 n)$ time all pairs of two directions for which P is 2-monotone. If no such pair is found, then P is not 2-monotone.*

Theorem 3 *Given a fixed number $k \geq 3$ and a simple polygon P with n vertices, we can find in $O(n^{3k+2})$ time all sets of k directions for which P is k-monotone. If no such set is found, then P is not k-monotone.*

The remainder of this paper is organized as follows: We first consider the case $k = 2$ in Section 3. In Sections 3.1 and 3.2, we give algorithms proving Theorems 1 and 2, respectively. We then consider the general case $k \geq 3$ and give an algorithm for Theorem 3 in Section 4.

2 Preliminaries

Throughout this paper, $P = (v_1, \ldots, v_n)$ is a simple polygon with n vertices $v_1, \ldots, v_n$ ordered counterclockwise. We denote by ∂P the boundary of the polygon P. We regard directions as the angles in the range $[0, \pi)$ measured from the positive x-axis.

There are three different notations of a line: we use $\ell(p, q)$ to denote the line through two points p and q, $\ell(e)$ to denote the line containing an edge or a segment e, and $\ell_\gamma(p)$ to denote the line through p in direction γ. We use $\overline{pq}$ to denote the line segment connecting two points p and q. Note that a line defines a direction (or an angle) in the angle range $[0, \pi)$. Similarly, an edge on ∂P or a segment defines a direction in the angle range. We say that a direction γ *covers* a point p in P if $\ell_\gamma(p)$ intersects P in one connected component.

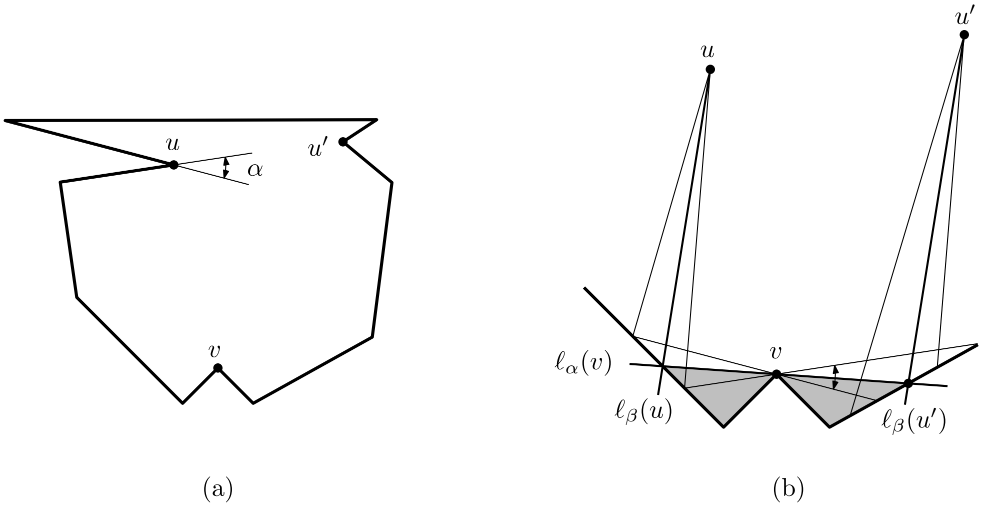


Figure 2: There is only one pair of directions for which the polygon is 2-monotone. These directions are not defined by edges or pairs of vertices.

Before proving the results, we show that the candidate directions for which a polygon is 2-monotone are not necessarily directions defined by edges or pairs of vertices of P. Consider a polygon with three reflex vertices u, u' and v as in Figure 2(a). To cover the points in the neighborhood of u, we must set one of two directions, say α, to lie in the exterior angular range between two edges incident to u. Indeed, any α in the range covers the whole polygon, except the points in the gray region lying below $\ell_\alpha(v)$, as shown in Figure 2(b). If there is a second direction that covers this gray region, then the polygon is 2-monotone. For some fixed α in that range, let β be the direction defined by u and the intersection of $\ell_\alpha(v)$ with the edge immediately to the left of v on $\ell_\alpha(v)$. Now place the third reflex vertex u' so that u' lies on the line in direction β passing through the intersection of $\ell_\alpha(v)$ with the edge immediately to the right of v on $\ell_\alpha(v)$. Then it is not difficult to see that the polygon is 2-monotone for $\{\alpha, \beta\}$. Clearly, these directions are not defined by any edges or pairs of vertices. Moreover, it is the only pair of directions for which the polygon is 2-monotone: for any other α within the range (induced by u), its correspondingly defined β cannot cover the region of P lying below $\ell_\alpha(v)$, with the infinite strip between $\ell_\beta(u)$ and $\ell_\beta(u')$, as shown in Figure 2(b), and thus no β can cover the region of P lying below $\ell_\alpha(v)$.

3 Monotonicity for two directions

Given a direction α, let P_α be the set of points $p \in P$ such that $\ell_\alpha(p) \cap P$ is one connected component. Then P_α is a collection of parallel strips to α, called *α-good strips*, and $P \setminus P_\alpha$ is separated by the α-good strips into *α-bad strips* such that every line of direction α within this strip intersects P in more than one component. As in Figure 3, α-good strips and α-bad strips appear alternately along the perpendicular direction of α and the number of such strips is at most n. An α-bad strip may contain $O(n)$ components of $P \setminus P_\alpha$, called *α-bad components*, but in total the number of α-bad components of P is at most $O(n)$ since the vertices of α-bad components are either the endpoints of α-good strips or the vertices of P. If we assume that the vertices of P are in general position (no three vertices of P are collinear), then an α-bad strip can contain only $O(1)$ components. Finally, note that α-bad components are not necessarily closed sets (containing the boundary in the sets) and that closures of two α-bad components in an α-bad strip can intersect only once at a reflex vertex of P_α in the top or bottom side of the strip. See Figure 3. For the horizontal direction α, this polygon has two α-bad strips and four α-bad components $S_1 \ldots, S_4$. The dashed line segments on the boundary of α-bad components are not contained in their α-bad components. The closures of S_1 and S_2 are completely disjoint and the closures of S_3 and S_4 intersect at a reflex vertex $u'' \in P_\alpha$ in the top.

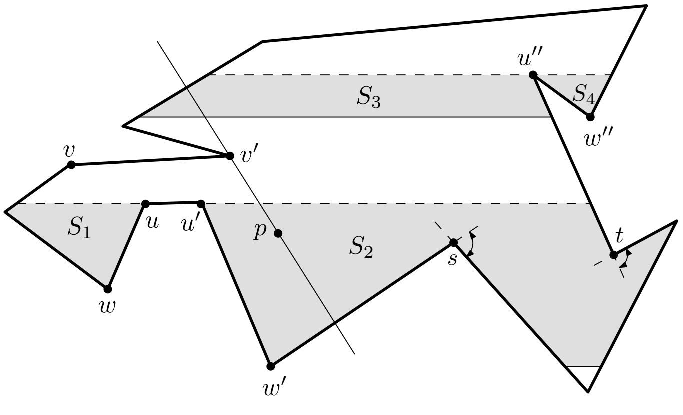


Figure 3: α-bad components $S_1, \ldots, S_4$.

If P is 2-monotone for α and some other direction β, every point p of α-bad components must be covered by β, that is, $\ell_\beta(p)$ must intersect P in one connected component. Thus we first consider the problem of finding a second direction β for given α such that all points of α-bad components are covered by β. We then consider the more general problem of finding two directions $\{\alpha, \beta\}$ for which P is 2-monotone.

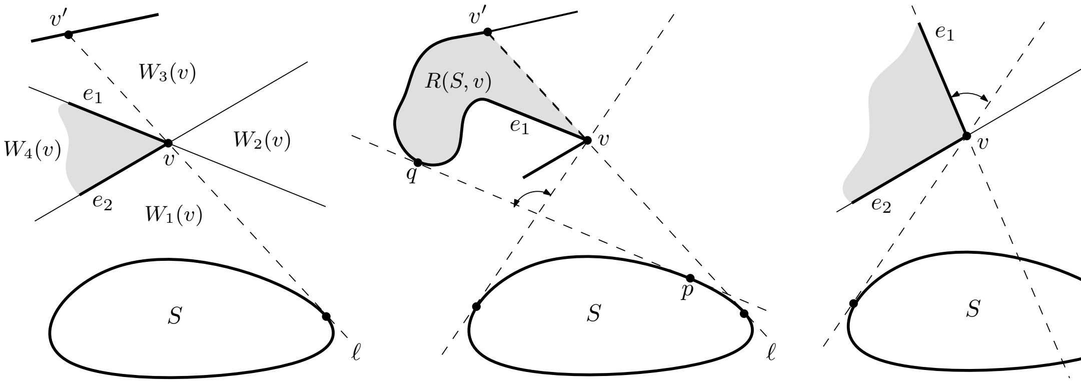
3.1 Finding a direction β for a fixed direction α

In this section, we describe an algorithm to solve the following problem: given a direction α, decide whether there is a direction β such that P is 2-monotone for $\{\alpha, \beta\}$. In fact, we will solve the more general problem of finding all directions β such that P is 2-monotone for $\{\alpha, \beta\}$.

We first compute all α-bad components by a plane sweep method in $O(n \log n)$ time. The second direction must cover all points in the α-bad components. We say a direction β is *forbidden* if there is a point p in an α-bad component such that $\ell_\beta(p)$ intersects P in more than one connected component. If we compute the union of all forbidden directions, then we can solve the problem by checking if the union is the whole angle space. If the union is the whole angle space, there is no valid second direction.

In the sequel, we will define an angle interval $f_\alpha(S, v)$ of forbidden directions for each pair (S, v) of an α-bad component S and a reflex vertex v of P. In Lemma 4 we will show that the union of $f_\alpha(S, v)$ for all pairs (S, v) is equal to the union of all forbidden directions. Without loss of generality, we assume that α is the horizontal direction.

Forbidden intervals. Given a pair (S, v) of an α-bad component S and a reflex vertex v of P, we define an angle interval $f_\alpha(S, v)$ of forbidden directions. Let e_1 and e_2 be the edges incident to v, where e_1 appears right before e_2 in counterclockwise orientation of ∂P. Two extensions $\ell(e_1)$ and $\ell(e_2)$ partition the region around v into four wedges as shown in Figure 4(a); the one intersecting the exterior of P locally around v is denoted by $W_4(v)$, and by $W_2(v)$ its diagonal wedge. The remaining two wedges are denoted by $W_1(v)$ and $W_3(v)$, where $W_1(v), W_2(v), W_3(v), W_4(v)$ are ordered in counterclockwise around v. The union of two wedges $W_1(v)$ and $W_3(v)$ forms a *double wedge* $W_{13}(v)$. Depending on the geometric relation between the wedges and S, we define the following three types of forbidden intervals.



(a) Definition of $R(S, v)$ (b) Type B (c) Type C

Figure 4: Forbidden intervals of Type B and Type C.

Type A: This is the case that $v \in S$. Let $f_\alpha(S, v)$ be the (open) angle interval of $W_{13}(v)$. Then any direction $\beta \in f_\alpha(S, v)$ is forbidden because we can translate the line $\ell_\beta(v)$ slightly so that it intersects both edges e_1 and e_2 incident to v and still passes through some point of P; thus this line in direction β intersects P in two or more components. For instance, in Figure 3 the vertices s and t define forbidden intervals of Type A for the same α-bad component S_2.

For the other two types, assume that $v \notin S$.

Type B: This is the case that $S \cap W_2(v) = \emptyset$ as in Figure 4(a) and 4(b). Let ℓ be the line through v and tangent to S such that e_1 and S lie in the same side of ℓ. Let v' be the first intersection of the ray from v along ℓ with ∂P (see Figure 4(a)). Then the segment $\overline{vv'}$ cuts P into two pieces, and we denote the one not intersecting S by $R(S, v)$. No point of $R(S, v)$ is visible from any point of S, that is, for any points $p \in S$ and $q \in R(S, v)$, $\ell(p, q)$ intersects P in more than one connected component and the direction is forbidden. So we take the double wedge containing $R(S, v)$ and S, whose boundaries are two inner tangent lines between them (see Figure 4(b)). We set $f_\alpha(S, v)$ to be the (open) angle interval of this double wedge, including the endpoint (direction) corresponding to the inner tangent $\overline{pq}$ if the tangent point p is contained in S.

Type C: This is the case that $S \cap W_2(v) \neq \emptyset$ (see Figure 4(c)). Define $f_\alpha(S, v)$ as the (open) angle interval of directions β such that $\ell_\beta(v)$ lies in the double wedge $W_{13}(v)$ and intersects S. Note that if $\ell(e_1)$ and $\ell(e_2)$ both intersect S, then $f_\alpha(S, v)$ consists of two sub-intervals.

Also, if S is contained in $W_2(v)$, then $f_\alpha(S, v) = \emptyset$. Using the translation argument used in the definition of Type A, we can prove that every direction in $f_\alpha(S, v)$ is forbidden.

Let $\mathcal{F}_\alpha$ be the union of $f_\alpha(S, v)$ for all pairs of α-bad components S and reflex vertices v of P.

Lemma 4 *The set $\mathcal{F}_\alpha$ is the union of all forbidden second directions.*

Proof. We have seen in the type definition that any direction in $\mathcal{F}_\alpha$ is forbidden, so what remains is to show that any forbidden direction for points in α-bad components is contained in $\mathcal{F}_\alpha$. Let β be a forbidden second direction. Then there is a point p in an α-bad component S such that the line $\ell_\beta(p)$ intersects P in more than one component. Among the intersection points between $\ell_\beta(p)$ and ∂P, let q be the closest point from p that are not visible from p, and let p' be the closest point of ∂P from p lying on $\overline{pq}$ (see Figure 5(a)). Note that p' and p may be identical.

We now consider the geodesic shortest path from p to q in P. Since q is not visible from p, it consists of more than one component and all its vertices, except p and q, are reflex. Let v be a reflex vertex in the path such that $\ell_\beta(v)$ is tangent to the path. Then β is contained in the interior of $W_{13}(v)$ since both edges incident to v lie in the same side of $\ell_\beta(v)$. We have two cases depending on whether q is in S or not.

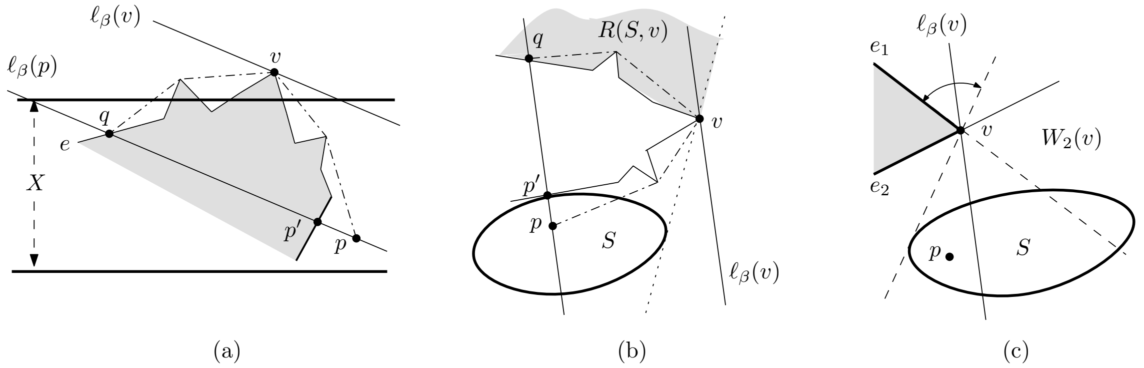


Figure 5: Illustration of the proof of Lemma 4.

Case 1: q is in S. In this case, we claim that the geodesic path from p to q must be contained in S. Then v is in S and the pair (S, v) defines the forbidden interval $f_\alpha(S, v)$ of Type A. Since β lies in the interior of $W_{13}(v)$, β is contained in this $f_\alpha(S, v)$.

To prove the claim, we denote by X the closure of the α-bad strip containing S and show that the geodesic path from p to q lies in X. Since α is horizontal, all we need is to show that the extreme points in y-coordinate along this path are contained in X. Let v' be an extreme point in y-coordinate along the geodesic path. Then v' must be contained in X; since otherwise v' must lie strictly above (or strictly below) X and any horizontal line lying between X and v' intersects P in more than one connected component, contradicting the fact that the strip right above (or right below) X is an α-good strip. Thus the geodesic path from p to q is contained in X.

Now remind that within the strip X, closures of any two bad components are either completely disjoint or can intersect only once at a reflex vertex of P_α. So, if the geodesic path lying in X and starting and ending at points of S were not contained in S, the path must go out to a neighboring component S' through a reflex vertex and then come back through a reflex vertex from some component S''. Since the path is geodesic, the outgoing and incoming vertices must be distinct and thus S' and S'' are different components both adjacent to S, which implies that closures of α-bad components S' and S'' are completely disjoint and cannot be connected by a path in X. Therefore the geodesic path starting and ending at $p, q \in S$ cannot leave S.

Case 2: q is not in S. In this case, if $S \cap W_2(v) = \emptyset$, then $R(S, v)$ is defined (see Figure 5(b)). Since q is the closest point invisible from p among the points in $\ell_\beta(p) \cap \partial P$, one portion of ∂P between p' and q is completely enclosed by the segment $\overline{pq}$ and the geodesic path from p to q. Note that v lies between p' and q along that portion. Thus q lies on the boundary of $R(S, v)$, and the direction β of $\overline{pq}$ for $p \in S$ and $q \in R(S, v)$ is contained in $f_\alpha(S, v)$ of Type B.

Now if $S \cap W_2(v) \neq \emptyset$, then the interior of S must intersect the line $\ell_\beta(v)$ since $\ell_\beta(v)$ separates $p \in S$ and $W_2(v)$ (see Figure 5(c)). So β is contained in $f_\alpha(S, v)$ of Type C.

This completes the proof of Lemma 4. □

Computing $\mathcal{F}_\alpha$. We now describe how to compute $\mathcal{F}_\alpha$. To answer the visibility queries quickly, we preprocess P in $O(n \log n)$ time so that ray shooting queries can be answered in $O(\log n)$ time [5]. In addition, we construct a data structure on the vertices $v_1, \ldots, v_n$ of P in $O(n \log n)$ time such that for each pair (i, j) with $i < j$, the convex hull of the vertices $\{v_i, v_{i+1} \ldots, v_j\}$ can be (implicitly) constructed in $O(\log^2 n)$ time [6].

We first compute all the α-bad components S and their convex hulls $\mathcal{CH}(S)$ by a plane sweep algorithm. This takes $O(n \log n)$ time since the total complexity of the components is linear. Next, we compute $f_\alpha(S, v)$ for each pair of an α-bad component S and a reflex vertex v of P as follows: The type of $f_\alpha(S, v)$ can be determined in $O(\log n)$ time, by checking if v is in S and then if the extension of the incident edges of v intersects S. Depending on the type of $f_\alpha(S, v)$, we use different methods as follows:

Type A: $f_\alpha(S, v)$ is the (open) angle interval of $W_{13}(v)$ and thus can be computed in $O(1)$ time.

Type B: We need to compute $R(S, v)$ and two inner tangent lines between $R(S, v)$ and S, as in Figure 4(b). To do this, we first compute two tangent lines from v to $\mathcal{CH}(S)$ in $O(\log n)$ time. To determine the cutting segment vv' that defines $R(S, v)$, we perform a ray shooting query from v along one of the two tangent lines in $O(\log n)$ time. If v' is a point on the edge (v_i, v_{i+1}) and v is equal to v_j for $i < j$, we determine the convex hull $\mathcal{CH}(R(S, v))$ of $\{v_{i+1}, \ldots, v_j\} \cup \{v'\}$ in $O(\log^2 n)$ time, by inserting v' into the precomputed convex hull of $\{v_{i+1}, \ldots, v_j\}$. We then compute the two tangents between $\mathcal{CH}(R(S, v))$ and $\mathcal{CH}(S)$ in $O(\log n)$ time by binary searching.

Type C: We only need to compute the tangents from v to $\mathcal{CH}(S)$ in $O(\log n)$ time by binary searching.

For each pair (S, v), we compute $f_\alpha(S, v)$ in $O(\log^2 n)$ time, so computing all such intervals takes $O(n^2 \log^2 n)$ time. Finally we compute the union of all forbidden intervals in $O(n^2 \log n)$ time, by using a simple greedy algorithm after sorting the interval endpoints, and check if the union is the whole angle space in $O(1)$ time. This completes the description of the algorithm and the proof of Theorem 1.

3.2 Finding two directions α and β

This section is devoted to the proof of Theorem 2: for given simple polygon P with n vertices in general position, compute in $O(n^5 \log^2 n)$ time all pairs $\{\alpha, \beta\}$ of two directions for which P is 2-monotone.

We initially set α to be the horizontal direction, and compute the interval system $\mathcal{F}_\alpha$ by the procedure described in Section 3.1. We then perform a sweep over the angle space of α in counterclockwise direction, and maintain the system $\mathcal{F}_\alpha$ during the sweep. Of course, since the exact values of the interval endpoints change continuously during the angular sweep, what we maintain is the combinatorial description of the intervals of $\mathcal{F}_\alpha$, that is, the ordered sequence of the interval endpoints.

Lids and owners. Let S be an α-bad component of P. The boundary between S and its neighboring α-good strip is a line segment in direction α, which we call a *lid* of S. Then S has two (top and bottom) lids, possibly of zero length. On the supporting line of each lid, there must be some vertex v of P, called the *owner* of the lid, such that $\ell_\alpha(v)$ intersects only the exterior or only the interior of P in the neighborhood of v. (If more than one vertex satisfy this condition, the leftmost vertex for the top lid and the rightmost vertex for the bottom lid must be chosen for the owner since we sweep the angular space in counterclockwise direction.) In Figure 3, the top and bottom owners of S_1, S_2, and S_3 are u and w, u and w', and u'' and w'', respectively.

Let $T(\alpha)$ be the set of endpoints of the lids of all α-bad components in P, and V the set of vertices of P. Then we can prove the following:

Lemma 5 *Let S be an α-bad component and v be a reflex vertex of P. The endpoints of the forbidden interval $f_\alpha(S, v)$ are defined by a pair of points from $V \cup T(\alpha)$.*

Proof. The endpoints of $f_\alpha(S, v)$ correspond to one of the followings: (i) lines extending the incident edges of v, or (ii) tangent lines between $R(S, v)$ and S, or (iii) tangent lines between S and v. Endpoints in (i) are defined by points from V, so consider the other two cases. For every tangent line between S and $R(S, v)$ or between S and v, its tangential points are either vertices of P (thus in V) or endpoints of lids of S, thus its direction is defined by a pair of points from $V \cup T(\alpha)$. □

Data structures. As mentioned before, we maintain $\mathcal{F}_\alpha$ during the angular sweep of α. Lemma 5 implies that what we need to maintain are: (i) the owners, lids, and convex hull of every α-bad component S, (ii) the convex hull of $R(S, v)$ for every pair (S, v) of an α-bad component S and a reflex vertex v that defines $R(S, v)$, and finally (iii) all *generating lines* defined by pairs of points from $V \cup T(\alpha)$. Dynamic data structures that can be used to maintain them are:

- $\mathcal{F}_\alpha$: During the sweep, we maintain all forbidden intervals $f_\alpha(S, v)$ and their union $\mathcal{F}_\alpha$, using the data structure $\mathcal{I}$, due to Cheng and Janardan [3]. This structure maintains the union of a set of m intervals, under insertions and deletions of intervals in $O(\log m)$ time. The union of the intervals can be listed in $O(k)$ time, if there are k components. In particular, it can be tested in $O(1)$ time whether the union is trivial.
- S, $R(S, v)$, $\mathcal{CH}(S)$ and $\mathcal{CH}(R(S, v))$: Both S and $R(S, v)$ can be represented as a simple polygon with a set of vertices or lid-endpoints on the boundary in counterclockwise order.

The data structure $\mathcal{C}$, due to Overmars and van Leeuwen [6], maintains the set of all bad components S and $R(S,v)$ where computing their convex hulls or answering various queries (such as computing tangents) can be done in $O(\log^2 n)$ time.

- Next event α: The standard priority queue $\mathcal{Q}$ is used to store the values of α at which the interval system changes combinatorially. This queue is maintained dynamically in $O(\log n)$ time per insertion and deletion.

Preprocessing. As a preprocessing step, we build a static data structure in $O(n \log n)$ time that supports visibility queries in $O(\log n)$ time. This can be used when we need to determine the first point at which a ray from a vertex of P hits ∂P. We build another data structure of Overmars and van Leeuwen [6] in $O(n \log n)$ time that enables us to (implicitly) determine the convex hull of $\{v_i, \ldots, v_j\}$ for any pair of vertices v_i and v_j in $O(\log^2 n)$ time. Finally, for each pair (v_i, v_j) of vertices of P, we compute the direction α_{ij} defined by them, and combine it with the information about how the lines $\ell_\alpha(v_i)$ and $\ell_\alpha(v_j)$ change their α-goodness or α-badness while moving from $\alpha_{ij} - \varepsilon$ to $\alpha_{ij} + \varepsilon$. This information is used to determine the type of component events that will be defined in the sequel. Using the first data structure for visibility queries, we can complete this in $O(n^2 \log n)$ time.

Events. We define two kinds of events at which the combinatorial structure of $\mathcal{I}$ or $\mathcal{C}$ changes. The first kind of events, called *component events*, happens when a bad component is created, deleted, split into two, or two bad components are merged into one, or the owners or lid-endpoints of bad components are changed. Any of these events result in creation/deletion of S and $R(S,v)$ in $\mathcal{C}$ and creation/deletion of intervals in $\mathcal{I}$. The second kind of events, called *interval events*, happens when the combinatorial description of intervals in $\mathcal{I}$ changes.

(a) Component events: These events arise whenever the sweeping line passes through a pair of vertices of P, i.e., $\alpha = \alpha_{ij}$ and some combinatorial change in the dynamic structure $\mathcal{C}$ are needed; such as creation, deletion, merging and splitting, or change of the owners or lid-endpoints of bad components in $\mathcal{C}$. For the vertices u and v on the sweeping line, denote the left one by $\mathrm{Left}(u,v)$ and the right one by $\mathrm{Right}(u,v)$.

- Owner/Lid-endpoint Change: At $\alpha = \alpha_{ij}$, an α-bad component S can change the top (resp. bottom) owner from v_i to v_j if both v_i and v_j stay in S and $v_j = \mathrm{Left}(v_i, v_j)$ (resp. $v_j = \mathrm{Right}(v_i, v_j)$). For instance, in Figure 3, the bottom owner of S_2 is changed from w' to t at the moment that α reaches the direction of $\ell(w', t)$. For this event, we just change the owner of S in $\mathcal{C}$.

  An endpoint of a lid of S with owner v_i moves from one edge to another incident to v_j if the status of $\ell_\alpha(v_j)$ stays at α-badness. For this event, we insert this vertex as a new vertex into S of $\mathcal{C}$ and update all information relevant to $\mathcal{CH}(S)$.

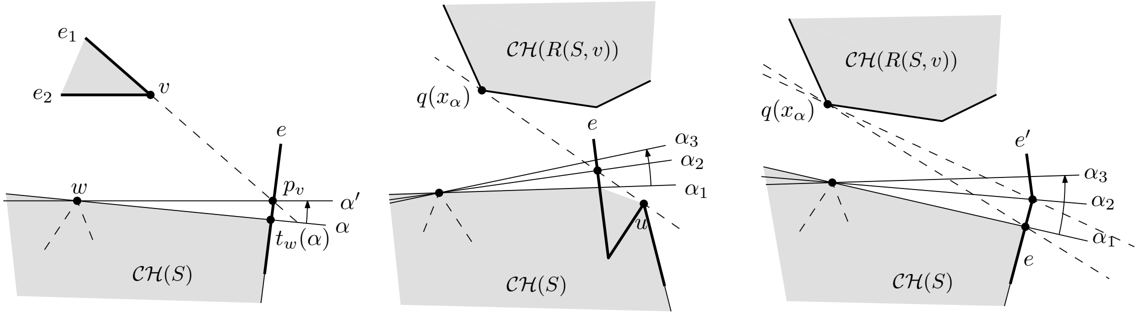
- Creation/Deletion: At $\alpha = \alpha_{ij}$, a new α-bad component S is created if (i) v_i and v_j are the endpoints of an edge in an α-good strip, and (ii) the status of one of $\ell_\alpha(v_i)$ and $\ell_\alpha(v_j)$ is changed from α-goodness to α-badness. The pair (v, v') in Figure 3 illustrates this case; The status of $\ell_\alpha(v)$ is changed from α-goodness to α-badness, while the status of $\ell_\alpha(v_j)$ stays the same at α-goodness. Then a new bad component S with the top owner $\mathrm{Left}(v_i, v_j)$ and the bottom owner $\mathrm{Right}(v_i, v_j)$ is created. Deletion proceeds exactly inverse to creation: when the top and bottom owners of S become collinear.

- Split/Merge: At $\alpha = \alpha_{ij}$, an α-bad component S can be split into three bad components S_1, S_2, and S_3 if (i) v_i is an owner of another component T and the status of $\ell_\alpha(v_i)$ stays at α-badness, and (ii) v_j was contained in S but not on the lids of S, and the status of $\ell_\alpha(v_j)$ is changed from α-badness to α-goodness. For instance, in Figure 3, $\ell(w, s)$ splits S_2 into three components. Then the top and bottom owners of S_1 are the S's top owner and Left(v_i, v_j), respectively, and the top and bottom owners of S_2 and S_3 are identically Right(v_i, v_j) and the S's bottom owner, respectively. At this event, we remove old component S and insert new components S_1, S_2 and S_3 into $\mathcal{C}$.

  Merging components proceeds exactly inverse to splitting: when the S_1's bottom owner Left(v_i, v_j) and the top owner Right(v_i, v_j) of S_2 and S_3 become collinear, three components S_1, S_2, and S_3 are merged into S.

Now we count the number of operations needed in $\mathcal{C}$, $\mathcal{I}$ and $\mathcal{Q}$ by the component events. The number of component events is $O(n^2)$ in total. Since the vertices of P are in general position, each component event calls $O(1)$ creation or deletion of bad components in $\mathcal{C}$. Thus we execute $O(n^2)$ creation or deletion of bad components in $\mathcal{C}$ in total. Creation or deletion of a bad component causes $O(n)$ insertion or deletion of $R(S, v)$ in $\mathcal{C}$ and thus $O(n)$ interval operations in $\mathcal{I}$ and $O(n)$ operations in $\mathcal{Q}$. Therefore, the total number of operations in $\mathcal{C}$, $\mathcal{I}$ and $\mathcal{Q}$ that are caused by component events during the sweep of α is $O(n^3)$.

(b) Interval events: These events arise when the combinatorial structure of the interval system in $\mathcal{I}$ changes. Every interval $f_\alpha(S, v) = (x_\alpha, y_\alpha)$ in $\mathcal{I}$ is defined by two generating lines $\ell(x_\alpha)$ and $\ell(y_\alpha)$, where x_α and y_α represent the endpoints of this interval. According to Lemma 5, these lines $\ell(x_\alpha)$ and $\ell(y_\alpha)$ are determined by two points in $V \cup T(\alpha)$. During the sweep of α, the combinatorial change of the interval system in $\mathcal{I}$ arises in three different ways:



(a) (b) (c)

Figure 6: Illustration of interval events.

- Type of $f_\alpha(S, v)$ changes.
  - Transition between Type A and Types B or C happens when α is equal to the direction of one of the edges e_1 and e_2 incident to v. There are $O(n)$ such directions.
  - Transition from Type B to Type C occurs as follows: Assume that v is above S and $\ell(e_1)$ intersects ∂P at edge e, as in Figure 6(a). Let $p_v := \ell(e_1) \cap e$ (i.e., p_v is the first intersection of the ray from v along $\ell(e_1)$ with ∂P). Let w be the top owner of S and $t_w(\alpha) := \ell_\alpha(w) \cap e$. Let α' be the direction of $\ell(w, p_v)$. As α approaches to α' in counterclockwise direction, $t_w(\alpha)$ approaches p_v. At the moment that $\alpha = \alpha'$, i.e., that $t_w(\alpha)$ arrives at p_v, the type of $f_\alpha(S, v)$ changes from Type B to Type C.

The angle α' is defined by the owner w of S and the intersection p_v between $\ell(e_1)$ and e. Since there are at most n owners and at most n intersection points, the number of such transitions is $O(n^2)$.

- Combinatorial description of $\ell(x_\alpha)$ and $\ell(y_\alpha)$ changes while the type of $f_\alpha(S, v)$ remains the same.
  - $f_\alpha(S, v)$ is of Type A: $\ell(x_\alpha)$ and $\ell(y_\alpha)$ are extended lines of the edges incident to v, so they do not change as long as the interval type remains the same.
  - $f_\alpha(S, v)$ is of Type B: $\ell(x_\alpha)$ (and $\ell(y_\alpha)$) is a tangent between S and $R(S, v)$, where $\ell(x_\alpha)$ passes through two tangential points $p(x_\alpha) \in \mathcal{CH}(S)$ and $q(x_\alpha) \in \mathcal{CH}(R(S, v))$. The point $p(x_\alpha)$ can be either a vertex of P or a point of $T(\alpha)$ whereas $q(x_\alpha)$ is a vertex of P. We have three different situations:
    - See Figure 6(b). While α rotates from α_1 to α_3, $\mathcal{CH}(S)$ remains unchanged. For $\alpha \leq \alpha_2$, $p(x_\alpha)$ is the vertex u, but $p(x_\alpha)$ becomes a point of $T(\alpha)$ for $\alpha \geq \alpha_2$, so the combinatorial description of $\ell(x_\alpha)$ changes at $\alpha = \alpha_2$. The direction α_2 is determined by the owner of S and the intersection $\ell(q(x_\alpha), u) \cap e$. Since e is fixed for a component S during $\alpha_1 \leq \alpha < \alpha_3$, there are $O(n^2)$ directions for each S and thus $O(n^3)$ events in total.
    - $\ell(x_\alpha)$ can change combinatorially when $\mathcal{CH}(S)$ changes combinatorially. See Figure 6(c). While α rotates from α_1 to α_3, $\mathcal{CH}(S)$ has a new edge e' on its boundary. So the combinatorial description of $\ell(x_\alpha)$ at $\alpha = \alpha_2$ has changed. However, this event is also detected in the component event caused by lid-endpoint changes. (The direction α_2 is defined by a component owner and a vertex, so there are $O(n^2)$ such directions for a single component and thus $O(n^3)$ in total.)
    - $\ell(x_\alpha)$ can change when $\mathcal{CH}(R(S, v))$ changes combinatorially. It is easy to see that a change of $\mathcal{CH}(R(S, v))$ for a fixed pair (S, v) happens $O(n)$ times, according to the directions defined by v and the other vertices in $R(S, v)$. Since there are $O(n^2)$ pairs (S, v), we have $O(n^3)$ such events. In addition, we update the information of $R(S, v)$ and $\mathcal{CH}(R(S, v))$ in $\mathcal{C}$ whenever they have changes.
  - $f_\alpha(S, v)$ is of Type C: a similar analysis as in Type B applies.
- The order of intervals changes in $\mathcal{I}$. This event happens when the endpoints of two intervals change their relative (cyclic) order. To update the interval system $\mathcal{I}$, we first delete both intervals from $\mathcal{I}$ and then insert two new intervals whose endpoints reflect the new order. Whenever an interval is inserted or deleted in $\mathcal{I}$ (e.g., a new component causes $O(n)$ insertion of $R(S, v)$ into $\mathcal{C}$ and thus $O(n)$ insertion of new intervals into $\mathcal{I}$), we need to compute all the moments α that the order between the new interval and the others of $\mathcal{I}$ is changed. There are $O(n^2)$ intervals in $\mathcal{I}$, so insertion/deletion of an interval generates $O(n^2)$ ordering change moments and we need to push these α into $\mathcal{Q}$. We have seen that $O(n^3)$ insertion or deletion of intervals in $\mathcal{I}$ can happen by component events and the above two interval events, so we may need $O(n^5)$ ordering change events in $\mathcal{I}$.

In total, during the whole sweep there are $O(n^3)$ operations in $\mathcal{C}$ and $O(n^5)$ operations in $\mathcal{I}$ and $\mathcal{Q}$.

Overall algorithm. After the preprocessing step, we initialize the data structures $\mathcal{I}$, $\mathcal{C}$ and $\mathcal{Q}$, and push all directions α_{ij} combined with the pre-computed information into $\mathcal{Q}$. We set

α to be the horizontal direction and compute all types of events for this configuration. While $\mathcal{Q}$ is not empty, we repeat this process: we extract the smallest angle α from $\mathcal{Q}$, and update the data structures according to the event type of α in $O(\log^2 n)$ time. After that, we test in $O(1)$ time if the updated intervals cover the whole angle space. The total number of operations needed in the whole process is $O(n^5)$, so the algorithm runs $O(n^5 \log^2 n)$ time.

This completes the proof of Theorem 2.

4 Monotonicity for $k \geq 3$ directions

This section is devoted to the proof of Theorem 3: for fixed $k \geq 3$ and given simple polygon P with n vertices, compute in $O(n^{3k+2})$ time all sets of k directions for which P is k-monotone.

The general strategy is that we divide the space of all *direction k-tuples* into cells, where we can show that in each cell, either all direction k-tuples cover the polygon in this way, or none of them does. Then we just have to test a sample direction k-tuple in each cell to decide whether there is some k-tuple that covers the polygon. Our cell decomposition is generated by $O(n^3)$ hyper-surfaces in that space of k-tuples; they divide this k-dimensional space into $O(n^{3k})$ cells. By this, we reduce the existence problem of a direction k-tuple covering the polygon to $O(n^{3k})$ decision problems, each deciding in $O(n^2)$ time if a specific direction k-tuple covers the polygon. So the total complexity of the algorithms will be $O(n^{3k+2})$.

We define three types of hyper-surfaces for the subdivision of the k-dimensional space of direction k-tuples:

- For $a \in \{1, \dots, k\}$, S_{abc} is the set of $(\varphi_1, \dots, \varphi_k)$ where the direction φ_a coincides with the direction of the line through the polygon vertices v_b and v_c.
- For $b, d \in \{1, \dots, k\}$, T_{abcd} is the set of $(\varphi_1, \dots, \varphi_k)$ where the line through v_a with direction φ_b intersects the line through v_c with direction φ_d in a point on the polygon boundary.
- For $b, d, f \in \{1, \dots, k\}$, R_{abcdef} is the set of $(\varphi_1, \dots, \varphi_k)$ where the line through v_a with direction φ_b intersects the line through v_c with direction φ_d in a point on the line through v_e with direction φ_f.

There are $O(n^2 k)$ hyper-surfaces of the first type, $O(n^2 k^2)$ of the second type, and $O(n^3 k^3)$ of the third type, so for fixed k the total number of surfaces in our arrangement is $O(n^3)$.

We now have to show that within each cell of the arrangement, the polygon is either covered for all $(\varphi_1, \dots, \varphi_k)$, or not covered for any $(\varphi_1, \dots, \varphi_k)$. Suppose that the polygon is covered for $\Phi = (\varphi_1, \dots, \varphi_k)$, and not covered for $\Psi = (\psi_1, \dots, \psi_k)$ in the same cell. Consider any path from Φ to Ψ. Along this path there is a last stage in which the entire polygon is covered, and a first point u that will be uncovered. There are several possibilities how a point can become uncovered. Each boundary of a region covered by a direction is either an edge of the polygon or a line in that direction through a vertex of the polygon. So the uncovered region around u must be bounded by lines, which are either polygonal edges or lines in one of the directions through a vertex of the polygon. We can distinguish the following cases:

- If u is a vertex, then in the moment u becomes uncovered the line through u goes through another vertex, so the path from Φ to Ψ crosses a surface of the first type.
- If u is a point on an edge of the polygon, and

  - the region becoming uncovered is bounded by a line parallel to the polygonal edge, moving away from it, then the path from Φ to Ψ again crossed a surface of the first type, or
  - the region becoming uncovered is bounded by two lines intersecting the polygonal edge, then in the moment that u becomes uncovered, these two lines intersect each other on the polygonal boundary, so the path from Φ to Ψ crosses a surface of the second type.

- If u is an interior point of the polygon, and
  - the region becoming uncovered is bounded by two parallel lines moving away from each other, then the path from Φ to Ψ crosses a surface of the first type, or
  - the region becoming uncovered is bounded by three lines, then in the moment that u became uncovered, these three lines intersect one another in a point, so the path from Φ to Ψ crosses a surface of the third type.

Therefore it is sufficient to check one sample point from each cell of this arrangement. To test whether a given k-tuple of directions actually covers a given n-gon, we just construct the arrangement of all the lines of these directions through all polygon vertices. This arrangement has $O(n^2k^2)$ cells and can be constructed in that time, and all potential boundaries of uncovered regions are among these lines. So we just have to check whether each cell is covered.

This finishes the proof of Theorem 3.

References

[1] P. Bose and M. van Kreveld. Generalizing monotonicity: On recognizing special classes of polygons and polyhedra by computing nice sweeps, *International Journal of Computational Geometry and its Applications* **15(6)**, 591–608, 2005.

[2] P. Bose, P. Morin, M. Smid and S. Wuhrer. Rotationally Monotone Polygons, *Proceedings of the 18th Canadian Conference on Computational Geometry*(2006), 105–108.

[3] S. W. Cheng and R. Janardan. Efficient maintenance of the union of intervals on a line, with applications, *Journal of Algorithms* **12**, 57–74, 1991.

[4] J. S. Ha, K. H. Yoo and J. K. Hahn. Characterization of polyhedron monotonicity, *Computer-Aided Design* **38(1)**, 48–54, 2006.

[5] J. Hershberger and S. Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. *Journal of Algorithms* **18**, 403–431, 1995.

[6] M. Overmars and J. van Leeuwen. Maintenance of configurations in the plane, *Journal of Computer and System Sciences* **23**, 166–204, 1981.

[7] F. Preparata and K. Supowit. Testing a simple polygon for monotonicity, *Information Processing Letters* **12(4)**, 161–164, 1981.

[8] D. Rappaport and A. Rosenbloom. Moldable and castable polygons, *Computational Geometry: Theory and Applications* **4**, 219–233, 1994.

[9] G. T. Toussaint. Movable separability of sets. In *Computational Geometry* (G. T. Toussaint, Editor), 335–375, North-Holland, Netherlands, 1985.