Algorithmica (1995) 14: 203–228

Algorithmica

A Linear-Time Algorithm for Constructing a Circular Visibility Diagram

Shuo-Yan Chou[1] and T. C. Woo[2]

Abstract. To computer circular visibility inside a simple polygon, circular arcs that emanate from a given interior point are classified with respect to the edges of the polygon they first intersect. Representing these sets of circular arcs by their centers results in a planar partition called the circular visibility diagram. An $O(n)$ algorithm is given for constructing the circular visibility diagram for a simple polygon with n vertices.

Key Words. Computational geometry, Circular visibility, Planar partition, Trapezoidal decomposition, Point visibility, Simple polygon, Amortized.

1. Introduction. One of the fundamental visibility problems is the computation of a point-visibility polygon: the portion of a polygon that is visible to an interior point. Joe and Simpson [13] develop a linear-time algorithm for constructing a point-visibility polygon inside a simple polygon. Another fundamental visibility problem is the computation of an edge-visibility polygon. Introduced by Avis and Toussaint [3], edge visibility is divided into three categories: complete, strong, and weak. Whether a polygon is completely or strongly visible to a given edge can be answered by the *kernel algorithm* developed by Lee and Preparata [16], and whether a polygon is weakly visible from an edge can be solved in linear time [3]. Guibas *et al.* [12] show that an edge-visibility polygon inside a triangulated simple polygon can be constructed in linear time. Suri and O'Rourke [20] show that an edge-visibility polygon inside a nonsimple polygon can be constructed in $\Omega(n^4)$ time. These linear visibility algorithms support many applications. The *art-gallery* problem [17] seeks the minimum number of points inside a polygon such that the point-visibility polygons of these points cover the entire polygon. The *minimum link path* between two points inside a simple polygon is solved optimally by constructing a sequence of visibility polygons [19]. The *shortest-path* problem can also be solved in linear time by utilizing visibility [12].

Whereas linear visibility is established by straight lines, circular visibility is established by arcs. Since straight lines can be considered as degenerate arcs, the realm of visibility is extended by considering circularity. The notion of circular visibility is illustrated in Figure 1, where a point q is *circularly visible* to a point

[1] Department of Industrial Management, National Taiwan Institute of Technology, 43 Keelung Road, Section 4, Taipei, Taiwan.
[2] Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109–2117, USA.

Received May 7, 1992; revised August 10, 1993. Communicated by D. T. Lee.

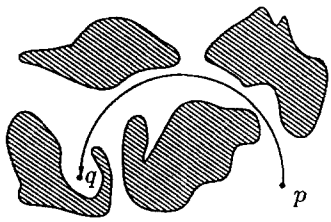

Fig. 1. Circularly visible points.

p if a circular arc can be drawn from p to q without intersecting any obstacle. Such a directed circular arc—clockwise or counterclockwise—is called a *visibility arc*, which can be uniquely defined by its center, endpoints, and direction. To simplify the discussion, we adopt the counterclockwise direction for visibility arcs. Under this convention, each visibility arc emanating from a fixed point can be uniquely represented by the center of the arc. The set of visibility arcs from the fixed point to a particular edge of the polygon can then be represented by a region consisting of their corresponding centers. This representation is utilized to solve the following problem:

PROBLEM CVD(p, Q) (Circular Visibility Diagram of a Simple Polygon Q).
Given: a simple polygon Q with edges $e_0, e_1, \ldots, e_n$, and a point p contained in Q.
Find: for each e_i, the circular arcs which emanate from p and intersect e_i before intersecting any other edge of Q.

Let $F_{e_i}^Q$ denote the set of centers corresponding to the visibility arcs from p to e_i, as shown in Figure 2. The notation F_A^B is used, throughout this paper, to

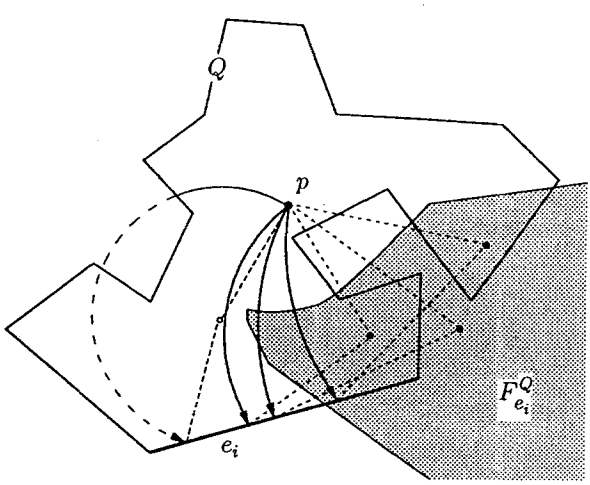

Fig. 2. Some counterclockwise visibility arcs of edge e_i.

represent the set of centers of the visibility arcs from the emanating point p to A in the presence of B. Since visibility arcs are represented by their centers, a solution to Problem CVD(p, Q) indicates a partition of the plane. The partition is represented as $\{F_{\varphi}^{Q}, F_{e_0}^{Q}, F_{e_1}^{Q}, \ldots, F_{e_n}^{Q}\}$, where F_{φ}^{Q} represents the set of centers for visibility arcs which emanate from p and do not intersect the boundary of Q. Such a partition is called the *circular visibility diagram* (CVD) of Q with respect to p.

An $O(n \log n)$ algorithm has been reported by Agarwal and Sharir [1] for computing the portion of a simple polygon which is circularly visible to a fixed interior point. As a corollary of the main result of this paper, the time for computing such a region can be reduced to $O(n)$ [8]. Agarwal and Sharir [2] also developed a data structure analogous to that of CVDs to solve a circle shooting problem. In that algorithm a given simple polygon is first preprocessed in $O(n \log^3 n)$ time into a data structure of size $O(n \log^3 n)$. For a query circle, the first intersection of the circle and the simple polygon can then be computed in $O(\log^4 n)$ time.

A sketch of the proposed algorithm for constructing a CVD is given in the next section. The development of the algorithm is detailed in the rest of the sections. The CVD of a single edge is first examined. Based on the CVD of an edge, algorithms are developed for constructing the CVDs of a star-shaped polygon and pockets. Finally, the circular visibility diagram of the simple polygon is constructed, and linearity in time for the construction is shown.

2. Sketch of the Overall Algorithm. The data structure of the CVD is similar to the dual space data structure used by Chazelle and Guibas [6] for solving a variety of linear visibility problems. In that paper a transform is employed, in which a line $ax + by + 1 = 0$ is represented by a point (a, b) [7], [15]. Points in the dual space are grouped into regions according to the edge whose corresponding visibility rays hit, resulting in a planar partition in the dual space.

In linear visibility a partial order in which visibility rays hit the edges of a polygon is crucial for the construction. In circular visibility, however, visibility arcs emanating from a point can hit two edges in either order, as shown in Figure 3. To overcome the apparent lack of a partial order, a polygon is decomposed into a star-shaped polygon and a set of pockets, each of which exhibits a partial order. The CVD of a simple polygon can be obtained by constructing the CVDs

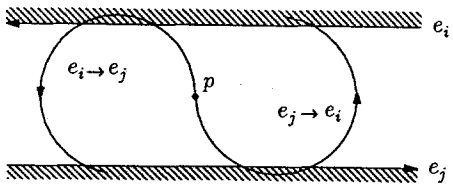

Fig. 3. The possible hitting order between two edges.

for the star-shaped polygon and then for every pocket. The outline of the CVD construction procedure is:

Algorithm (CVD_Simple_Polygon)
Input: a simple polygon Q;
Decompose Q: $Q = Q^* + P_1 + \cdots + P_m$;
Construct CVD(p, Q^*);
for $i = 1$ to m **do**
  Construct CVD(p, P_i);
Output CVD(p, Q);

The polygon Q^* is a star-shaped polygon in Q, and $P_1, \ldots, P_m$ are pockets. A polygon is said to be *star-shaped* if a point interior to the polygon exists such that the entire boundary of the polygon is linearly visible to the point. By definition, a point-visibility polygon is star-shaped. An open polygonal chain is said to be a *pocket* if the union of the chain and its *lid*, the line segment connecting its two endpoints, forms a simple polygon. Such a chain is said to be a pocket with respect to a point if the point is collinear with the lid but not on the lid. The collinearity of p with the lid exhibits an essential property, which will be explained shortly and is employed in the construction of the CVD for a pocket.

LEMMA 2.1. *Let p be a point collinear with a line segment $\overline{uv}$, where $u \in \overline{pv}$. Then every arc emanating from p will intersect $\overline{uv}$ at most once.*

PROOF. Suppose an arc emanating from p passes through $\overline{uv}$ at q. Since there can be at most two intersections between an arc and a line segment, and since the arc has already passed through $\overline{pv}$ at q and p, it cannot have another intersection with $\overline{pv}$. As $p \notin \overline{uv}$, $\overline{uv}$ contains exactly one intersection with the arc, namely q. □

It is observed that any simple polygon can be decomposed into a star-shaped polygon and a number of pockets, with respect to a point inside the polygon, as shown Figure 4. Let Q^* be the star-shaped polygon obtained by computing the linear point-visibility polygon with respect to p. The polygon Q^* can be constructed optimally in $O(n)$ time by the algorithm developed by Lee [14], where n is the number of vertices of Q. Taking the boolean difference between Q and Q^* results in a set of pockets $P_1, P_2, \ldots, P_m$ (if the difference exists). Since $Q^* \subset Q$, the boolean difference between them can be computed in $O(n)$ time.

That this algorithm correctly constructs the CVD of a simple polygon is easy to see. The fixed point p is, by definition, in Q^*. By Lemma 2.1, visibility arcs crossing the lid of a pocket cannot come back into Q^* through the lid and hit other edges of Q^*. Thus, the collection of visibility arcs that hit the edges of Q that are also the edges of Q^* are the same as those computed with respect to Q^*. On the other hand, visibility arcs that hit the lid of a pocket, which is also an edge of Q^*, constitute all the visibility arcs going into this pocket. The CVD for

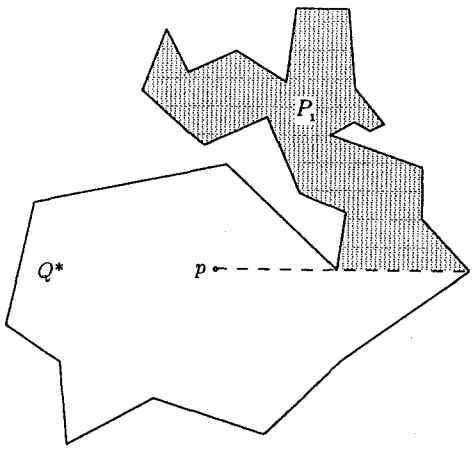

Fig. 4. The decomposition of a simple polygon Q.

edges of Q which are in a pocket can thus be computed by further decomposing the region containing all the centers about which visibility arcs hit the lid of the pocket.

3. Circular Visibility Diagram of an Edge. Let p be the fixed point from which visibility arcs emanate, and let $\overrightarrow{uv}$ be a directed edge from u to v. As a visibility arc emanates from p, the arc may miss $\overrightarrow{uv}$, hit the left side of $\overrightarrow{uv}$, or hit the right side of $\overrightarrow{uv}$. The differentiation of visibility arcs can be made by classifying the loci of the centers according to their distances to p and to $\overrightarrow{uv}$, as the partition shown in Figure 5(a). First, the bisector β_u of p and u, and the bisector β_v of p and v, are constructed respectively. Then a parabola $\beta_{\overleftrightarrow{uv}}$, with focus p and directrix coincident with $\overrightarrow{uv}$, is constructed. By construction, the parabola $\beta_{\overleftrightarrow{uv}}$ is tangent to β_u and β_v at u' and v'. Moreover, both $\overline{uu'}$ and $\overline{vv'}$ are perpendicular to $\overline{uv}$ [4]. Let $\beta'_{\overleftrightarrow{uv}}$ denote the portion of $\beta_{\overleftrightarrow{uv}}$ between u' and v'. Let β_u^+ denote the half-line of β_u which has C^1 continuity with $\beta'_{\overleftrightarrow{uv}}$ at u', and let β_v^+ denote the half-line of β_v

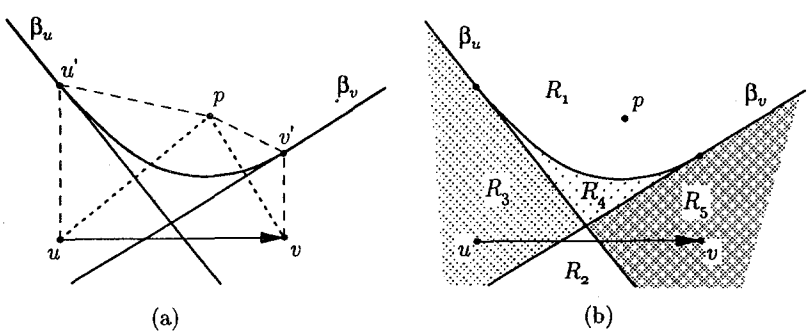

Fig. 5. A partition of counterclockwise visibility arcs with respect to edge $\overrightarrow{uv}$.

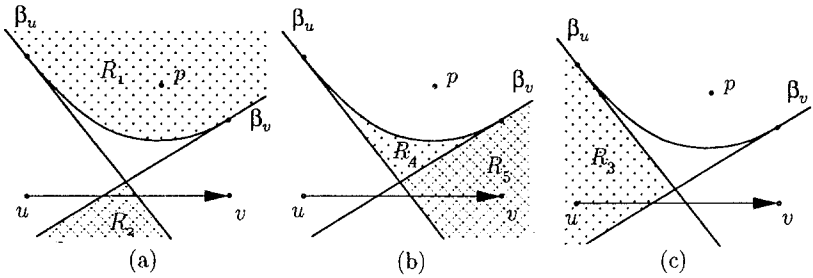

Fig. 6. The centers of visibility arcs which: (a) miss $\overrightarrow{uv}$, (b) hit the left side of $\overrightarrow{uv}$, and (c) hit the right side of $\overrightarrow{uv}$.

which has C^1 continuity with $\beta'_{\overleftrightarrow{uv}}$ at v'. (The other halves of β_u and β_v are denoted as β_u^- and β_v^-.) While β_u, β_v, and $\beta_{\overleftrightarrow{uv}}$ contain all the equidistant points between p and u, p and v, and p and $\overleftrightarrow{uv}$, respectively, the continuous curve $\beta_{\overline{uv}}$ consisting of β_u^+, $\beta'_{\overleftrightarrow{uv}}$, and β_v^+ contains all the equidistant points between p and $\overline{uv}$. This curve is also known [18] as the *Voronoi diagram* of p and $\overline{uv}$.

The curves β_u, β_v, and $\beta'_{\overleftrightarrow{uv}}$ partition the plane into five regions, R_1, R_2, ..., and R_5, as shown in Figure 5(b). The following theorem shows that counterclockwise arcs drawn from p about points in each region will all miss $\overrightarrow{uv}$, all hit the left side of $\overrightarrow{uv}$, or all hit the right side of $\overrightarrow{uv}$. These five regions can therefore be combined into three sets, as shown in Figure 6. It is noted that hitting the right side of $\overrightarrow{uv}$ is the same as hitting the left side of $\overrightarrow{vu}$.

THEOREM 3.1. *Let $F_\varphi^{\overrightarrow{uv}}$, $F_{\overrightarrow{uv}}$, and $F_{\overrightarrow{vu}}$ represent, respectively, the regions containing all the points about which visibility arcs emanating from p miss $\overrightarrow{uv}$, hit the left side of $\overrightarrow{uv}$, and hit the right side of $\overrightarrow{uv}$. Then:*

(i) $F_\varphi^{\overrightarrow{uv}} = R_1 \cup R_2$.
(ii) $F_{\overrightarrow{uv}} = R_4 \cup R_5$.
(iii) $F_{\overrightarrow{vu}} = R_3$.

PROOF. As R_1 lies on the side of $\beta_{\overline{uv}}$ that contains p, $d(x, p) < d(x, \overrightarrow{uv})$, for all $x \in R_1$, which means that visibility arcs from p centered at a point in R_1 will not intersect $\overrightarrow{uv}$. Similarly, since R_2 is the intersection of the half-planes defined by the bisectors β_u and β_v and not containing p, for all $x \in R_2$, $d(x, u) < d(x, p)$ and $d(x, v) < d(x, p)$. This implies that visibility arcs centered at a point in R_2 will not intersect $\overrightarrow{uv}$ either. Thus, (i) is true.

To show (ii) and (iii), first consider the points in R_4. As R_4 lies on the side of $\beta_{\overline{uv}}$ that does not contain p, for all $x \in R_4$, $d(x, \overrightarrow{uv}) < d(x, p)$. Since R_4 also lies in the two half-planes defined by the bisectors β_u and β_v and containing p, for all $x \in R_4$, $d(x, p) < d(x, u)$ and $d(x, p) < d(x, v)$. In other words, any circle centered at a point in R_4 will not contain the endpoints of $\overrightarrow{uv}$; yet, the distance between the center and $\overrightarrow{uv}$ is shorter than the radius of the circle, which means that such a

circle will intersect $\vec{uv}$ at two points. Since R_4 lies to the left of $\vec{uv}$, all such arcs will intersect $\vec{uv}$ from the left.

A similar reasoning shows that every arc drawn from point p about a point in R_3 or R_5 will intersect $\overline{uv}$ at one point. Suppose an arbitrary arc drawn from p intersects $\vec{uv}$ at q. The center of the arc $\widehat{pq}$, if in R_5, always lies to the right of $\vec{qp}$. This indicates that a counterclockwise $\widehat{pq}$ drawn from p and centered at a point in R_5 always hits $\vec{uv}$ from the left. On the other hand, if arc $\widehat{pq}$ is drawn from p and centered at a point in R_3, it always hits $\vec{uv}$ from the right. This completes the proof for (ii) and (iii). □

Assuming that the edges of the simple polygon are given in the counterclockwise order, all the first crossings of the visibility arcs will be from the left of the edges. Thus, only visibility arcs that hit the left side of an edge are of interest. Figure 7 gives the four possible cases of the CVD of an edge $\vec{uv}$. Figure 7(a) depicts a Type-L CVD, where p is to the *left* of $\vec{uv}$, and a Type-R CVD, where p is to the *right* of $\vec{uv}$. In both cases counterclockwise arcs drawn from p about a point in $F_{\vec{uv}}$ hit the left side of $\vec{uv}$. Figure 7(b) shows the limiting cases, Type-T and Type-A CVDs, in which p is collinear with $\vec{uv}$, and $\vec{uv}$ is directed either *toward* or *away* from p. We note that if $\vec{uv}$ is directed away from p, no visibility arc emanating from p will hit the left side of $\vec{uv}$.

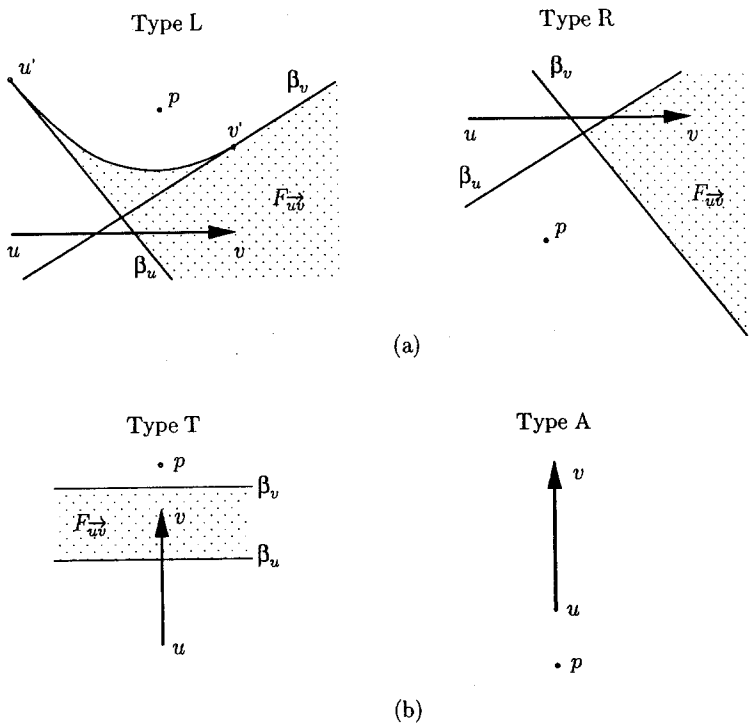

Fig. 7. Counterclockwise CVDs for p and $\vec{uv}$.

4. Circular Visibility Diagram of a Star-Shaped Polygon. Given a star-shaped polygon Q^* with respect to p, it is observed that cutting Q^* by a horizontal line passing through p yields two star-shaped chains—the upper chain C_U and the lower chain C_L—each of which spans 180° about p. In this section the CVDs for C_U and C_L are constructed individually, and then merged to obtain the CVD of Q^*.

4.1. Obtuse Star-Shaped Chain. The algorithm for computing star-shaped chains C_U and C_L is discussed. As noted earlier, edges of a polygonal chain may obstruct visibility arcs. The resolution of the obstruction and subsequently the construction of the CVD can be costly. However, if a partial order in which an arc hits the edges of such a chain can be established, the CVD can then be constructed efficiently.

A star-shaped chain is *obtuse* if the span of the polar angle θ of a point traversing the star-shaped chain is less than 180°, where θ is the angle measured counterclockwise from the polar axis. Both C_U and C_L are obtuse. Given an obtuse star-shaped chain with monotonically increasing θ with respect to p (with all the edges directed counterclockwise), the visibility arcs to the individual edges of the chain are to be computed. To determine efficiently the first edge that a visibility arc hits, a partial order in which circular arcs emanating from p hit the edges of the chain is established in the following lemma.

LEMMA 4.1. *Let $C = \{e_0, e_1, e_2, \ldots, e_m\}$ be an obtuse star-shaped chain which is monotonically increasing in θ. A counterclockwise arc drawn from p hits e_i before e_j only if $i < j$. Similarly, if θ is monotonically decreasing, then an arc can hit e_i before hitting e_j only if $j < i$.*

PROOF. For a chain that is monotonically increasing, a counterclockwise arc $\widehat{pq}$ drawn from p which hits e_i at a point q will always lie to the left of $\vec{qp}$. However, for all $j > i$, e_j always lies to the right of $\vec{qp}$ because the chain C is star-shaped about p and spans less than 180°. Since $\widehat{pq}$ does not intersect any succeeding edges of e_i in C, it hits another edge before hitting e_i only if the edge precedes e_i. □

Since constructing the CVD for an obtuse star-shaped chain with decreasing θ is analogous to that with increasing θ, only the latter case is presented. Based on Lemma 14.1, a recursive relationship between the CVD of an obtuse star-shaped chain and the CVDs of its constituent edges is established.

THEOREM 4.2. *The regions in a CVD of the monotonically increasing chain C can be computed by*

$$F_{e_i}^{C} = \begin{cases} F_{e_0} & \text{if } i = 0, \\ F_{e_i} \cap F_{\varphi}^{C_{i-1}} & \text{if } i > 0, \end{cases}$$

and

$$F_{\varphi}^{C_i} = \begin{cases} F_{\varphi}^{e_0} & \text{if } i = 0, \\ F_{\varphi}^{e_i} \cap F_{\varphi}^{C_{i-1}} & \text{if } i > 0, \end{cases}$$

where $C_i = \{e_0, e_1, \ldots, e_i\}$ denotes a subchain of C.

PROOF. Based on Lemma 4.1, visibility arcs about points in F_{e_i} may hit e_j, only if $j < i$. Therefore, for points to be in $F_{e_i}^{C_i}$, their corresponding circular arcs cannot hit e_j, for all $j < i$. As $F_{\varphi}^{C_i}$ represents the intersection of $F_{\varphi}^{e_j}$, for all $j < i$, the region $F_{e_i}^{C_i}$ is equal to the intersection of F_{e_i} and $F_{\varphi}^{C_{i-1}}$. □

Intersecting F_{e_i} and $F_{\varphi}^{C_{i-1}}$, for all $i > 0$, takes $O(n^2)$ time, which seems to indicate that an algorithm for constructing a CVD for an obtuse star-shaped chain would exceed linear time. However, by utilizing certain properties of the consecutive $F_{e_i}^{C}$'s, we can show that constructing a circular visibility diagram of an obtuse star-shaped chain is analogous to cutting a pie and removing it piece by piece in a sequential order. Also, by establishing that the time for computing the "cutting points" is amortized, such a pie-cutting procedure can be achieved in linear time. A similar linear-time cutting procedure is described by Edelsbrunner and Guibas [11] for computing a "bay" formed by lines sorted in slope order.

Before investigating the properties of the consecutive $F_{e_i}^{C}$'s, the portion of $F_{\varphi}^{e_i}$ that has no effect on the construction of the CVD is identified, and omitted from the subsequent analysis. Recall that $F_{\varphi}^{e_i}$ contains all the points about which arcs emanating from p miss e_i, and equals the union of regions R_1 and R_2 as shown in Figure 6(a). Let region R_2 computed with respect to e_i be denoted as $R_2^{e_i}$. (See Figure 8.) Lemma A.1 in the Appendix establishes that the intersection of $R_2^{e_i}$ and $F_{e_{i+1}}$ is always empty. Also, since the star-shaped chain is obtuse, the region $F_{e_{i+1}}$ will not intersect $R_2^{e_j}$, for $j \leq i$. Therefore, for the purpose of computing $F_{e_{i+1}}^{C}$, each $F_{\varphi}^{e_i}$ need only include points in R_1, which is the region of the Voronoi diagram of p and e_i containing p.

In addition to the reduction of $F_{\varphi}^{e_i}$, properties on the unbounded regions in the CVD need to be established. In a given CVD it is possible to distinguish bounded regions from the unbounded ones. As established in Lemma A.2 in the Appendix, the existence of unbounded regions in a CVD is closely related to the linear visibility of the edges: a region containing the centers of visibility arcs that hit a particular edge is unbounded if and only if some portion of the edge is

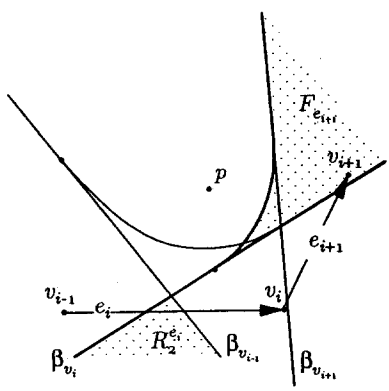

Fig. 8. The superimposition of $F_{e_{i+1}}$ and $R_2^{e_i}$.

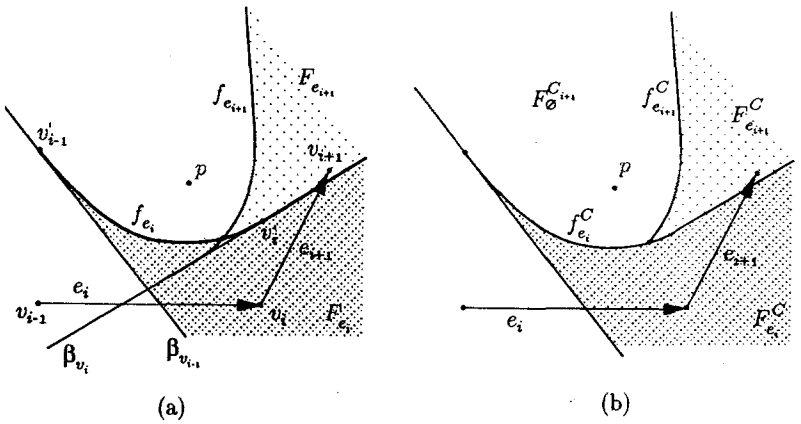

Fig. 9. (a) Overlapping the CVDs of two edges. (b) The circular visibility diagram of two consecutive edges.

linearly visible to p. Furthermore, it is shown in Lemma A.3 that these unbounded regions of a CVD are ordered about p.

With the order on the unbounded regions of the CVD, a property between the consecutive $F^C_{e_i}$'s can now be established, enabling the development of an efficient CVD construction algorithm. Without loss of generality, all the CVDs of individual edges of the obtuse star-shaped chain C are assumed to be of Type L. (Refer to Figure 7.) As such, F_{e_i} of edge e_i in C is bounded by three curves: $\beta^-_{v_{i-1}}$, β'_{e_i} (the portion of β_{e_i} between v'_{i-1} and v'_i), and $\beta^+_{v'_i}$, as shown in Figure 9(a). To simplify following discussions the concatenation of β'_{e_i} and $\beta^+_{v'_i}$ is denoted by a single curve f_{e_i}, as indicated in Figure 9(a). The portion of f_{e_i} remaining in the CVD of chain C is denoted as $f^C_{e_i}$, as shown in Figure 9(b).

Since C is a star-shaped about p, i.e., every point on e_i is linearly visible to p, the region $F^C_{e_i}$ (for all i) is unbounded and the order of $F^C_{e_i}$ about p is the same as that of e_i. Such an order of $F^C_{e_i}$ indicates an order for constructing the CVD of C. It is shown in the following that individual $F^C_{e_i}$'s can be constructed efficiently following this order. First, $F^C_{e_{i+1}}$ is shown to be bounded by $f_{e_{i+1}}$ and the boundaries of $F^{C_i}_{\varphi}$.

LEMMA 4.3. *Let C be an obtuse star-shaped chain, and let e_i and e_{i+1} be two consecutive edges in C. Then $F^C_{e_{i+1}}$ is bounded by $f_{e_{i+1}}$ and the boundaries of $F^{C_i}_{\varphi}$.*

PROOF. As shown in Theorem 4.2, $F^C_{e_{i+1}}$ is equal to the intersection of $F_{e_{i+1}}$ and $F^{C_i}_{\varphi}$. As $F_{e_{i+1}}$ is a Type-L region, by definition, it is bounded by $f_{e_{i+1}}$ and β_{v_i}. See Figure 9(a). Also, $F^{C_i}_{\varphi}$ is bounded by β_{v_i} since $F^{e_i}_{\varphi}$ is bounded by β_{v_i} and $F^{C_i}_{\varphi}$ is a subset of $F^{e_i}_{\varphi}$. Moreover, since both $F_{e_{i+1}}$ and $F^{C_i}_{\varphi}$ lie in the half-plane defined by β_{v_i} and containing p, the region $F^C_{e_{i+1}}$ is therefore bounded by $f_{e_{i+1}}$ and the boundary of $F^{C_i}_{\varphi}$. □

To achieve efficiency, it is also essential that $f_{e_{i+1}}$ intersects the boundaries of $F_\varphi^{C_i}$ at only one point, which is equivalent to saying that $f_{e_i}^C$ is continuous. This is shown in the following lemma.

LEMMA 4.4. *$f_{e_i}^C$ is continuous, for all i.*

PROOF. Given two consecutive edges, we show that $f_{e_{i+1}}$ will intersect f_{e_i} only once. This is because the tangents of the points on f_{e_i} are nondecreasing as f_{e_i} goes to infinity, and are bounded by the two perpendicular bisectors between p and the two endpoints of e_i. Thus, the tangents of the points on $f_{e_{i+1}}$ are greater than those on f_{e_i}. Also, since C is obtuse, $\beta_{v_{i+1}}$ cannot go around p and hit f_{e_i}. Therefore, the curve $f_{e_{i+1}}$ will only intersect f_{e_i} once. By using the same argument, we can show that $f_{e_{i+1}}$ can intersect the boundaries of $F_\varphi^{C_i}$ only once, which means that the portion of $f_{e_{i+1}}$ becoming $f_{e_i}^C$ is continuous. □

Lemma 4.3 indicates that $F_{e_{i+1}}^C$ can be constructed by intersecting $f_{e_{i+1}}$ with the boundaries of $F_\varphi^{C_i}$, and Lemma 4.4 indicates that they intersect at only one point. Subsequently, in each iteration of the algorithm, the curve $f_{e_{i+1}}^C$ partitions $F_\varphi^{C_i}$ into two regions: $F_{e_{i+1}}^C$ and $F_\varphi^{C_{i+1}}$, as depicted in Figure 9(b). The detailed steps for constructing the CVD of an obtuse star-shaped chain C are now discussed.

Algorithm (CVD_Obtuse_Star_Shaped_Chain)
  S: a stack maintaining the boundary of $F_\varphi^{C_i}$;
  Compute $F_{e_0}^C$ and $F_\varphi^{C_0}$;
  Output β_{v_0}; Output $f_{e_0}^C$;
  Push($f_{e_0}^C$, S);
  for $i = 1$ to m **do** /* Computing $F_{e_i}^C$ */
    while ($f_{e_i} \cap$ Top(S) $= \varnothing$) **do**
      Output Top(S);
      Pop(S);
    Compute $f_{e_i}^C$;
    Divide Top(S) into f^+ and f^-;
    Update Top(S) with f^+;
    Push($f_{e_i}^C$, S);
    Output f^-;
  Output S;

In the algorithm, $F_{e_0}^C$ and $F_\varphi^{C_0}$ are constructed first. This can be done in constant time. Then, in each iteration, the curve f_{e_i} is intersected with the boundaries of $F_\varphi^{C_{i-1}}$, resulting in $F_\varphi^{C_i}$ and $F_{e_i}^C$. After $f_{e_i}^C$ is computed, it replaces those boundaries[3] of $F_\varphi^{C_{i-1}}$ which lie in $F_{e_i}^C$ and becomes a boundary of $F_\varphi^{C_i}$.

[3] The particular $f_{e_j}^C$ that bounds $F_\varphi^{C_{i-1}}$ and intersects f_{e_i} may still contribute to the boundaries of $F_\varphi^{C_i}$.

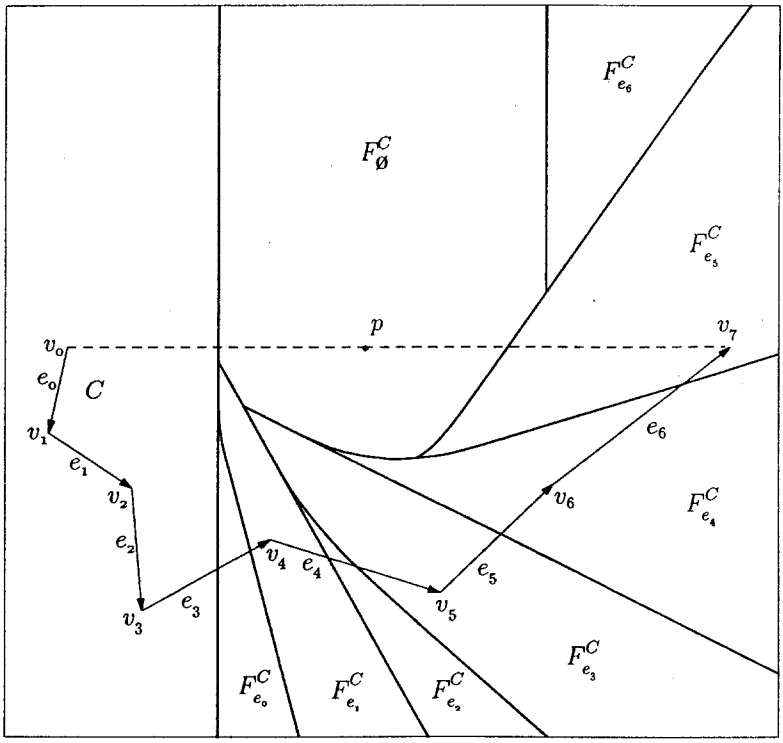

Fig. 10. The CVD of an obtuse star-shaped chain.

A stack is used to record $f^C_{e_j}$ that comprise the boundaries of $F^{C_i}_{\varphi}$. Every time an f_{e_i} is introduced to partition $F^{C_{i-1}}_{\varphi}$, those $f^C_{e_j}$ that do not intersect f_{e_i} are popped from the stack until one does is found. The curve $f^C_{e_i}$ is then computed and pushed to the top of the stack. Those $f^C_{e_j}$ that were popped, along with $f^C_{e_i}$, comprise the boundaries of $F^C_{e_i}$. At the end of the algorithm, the $f^C_{e_j}$'s remaining in the stack are identified as the boundaries of F^C_{φ}.

The time complexity of this algorithm is of the same order as the number of $f^C_{e_j}$ popped out of the stack, which is of the same order as the number of edges in the chain. Therefore, the CVD of an obtuse star-shaped chain C can be computed in $O(n)$ time, where n is the total number of edges in C. The CVD of an obtuse star-shaped chain is illustrated in Figure 10.

4.2. Merging. By using the algorithm described above, the CVDs of C_U and C_L can be constructed efficiently. However, to obtain the CVD of the point-visibility polygon Q^*, the overlapping of the two CVDs needs to be resolved to merge the two CVDs properly. Clearly, if a visibility arc does not hit the left sides of C_U and C_L, it will not intersect Q^*. If a visibility arc only hits the left side of one of the two chains, the visibility arc only hits the corresponding edge in Q^*. If a visibility arc hits both chains on the left side, its center will appear in

both of the CVDs of the two chains. To resolve such overlapping, the order in which the arc intersects the two chains needs to be determined.

Let L be the perpendicular line passing through p. Let F_{C_U} and F_{C_L} denote the regions containing the centers about which visibility arcs emanate from p and hit C_U and C_L, respectively. Let $F_{C_U}^{Q^*}$ and $F_{C_L}^{Q^*}$ be the regions containing the centers about which arcs emanate from p and intersect C_U and C_L, respectively, in the presence of Q^*. The following lemma resolves the overlapping between the CVDs of the two chains.

LEMMA 4.5. *Let q be a point in $F_{C_U} \cap F_{C_L}$. Then, if q lies to the left of L, $q \in F_{C_U}^{Q^*}$. If q lies to the right of L, $q \in F_{C_L}^{Q^*}$.*

PROOF. Arcs centered at a point to the right of L always go downward first from p. Therefore, if such arcs intersect both C_U and C_L, they must hit C_L first. Likewise, arcs centered at a point to the left of L must hit C_U first if they intersect both C_U and C_L. □

Lemma 4.5 indicates that, in the presence of Q^*, arcs emanating from p and centered at points to the right of L can hit C_U without being blocked by C_L only if these points also lie in $F_\varphi^{C_L}$. Similarly, arcs emanating from p and centered at points to the left of L can hit C_L without being blocked by C_U only if these points also lie in $F_\varphi^{C_U}$.

Without loss of generality, let $F_{e_j}^{Q^*}$, for all $e_j \in C_L$, be constructed first. The feasible area where $F_{e_j}^{C_L}$ can lie is the union of $F_\varphi^{C_U}$ and the half-plane to the right of L. $F_\varphi^{C_U}$ can be obtained by computing the CVD of C_U, as shown in Figure 11(a). The result of the union of $F_\varphi^{C_U}$ and the half-plane to the right of L is shown in Figure 11(b). $F_{e_j}^{Q^*}$ can therefore be constructed along the boundary of $F_\varphi^{C_U}$ with the procedure used for constructing obtuse star-shaped chains. After completing constructing the $F_{e_j}^{Q^*}$'s, $F_\varphi^{C_L}$ is obtained. $F_{e_i}^{Q^*}$, for all $e_i \in C_U$, can then be constructed similarly along the boundary of $F_\varphi^{C_L}$. Figure 12 shows the final result

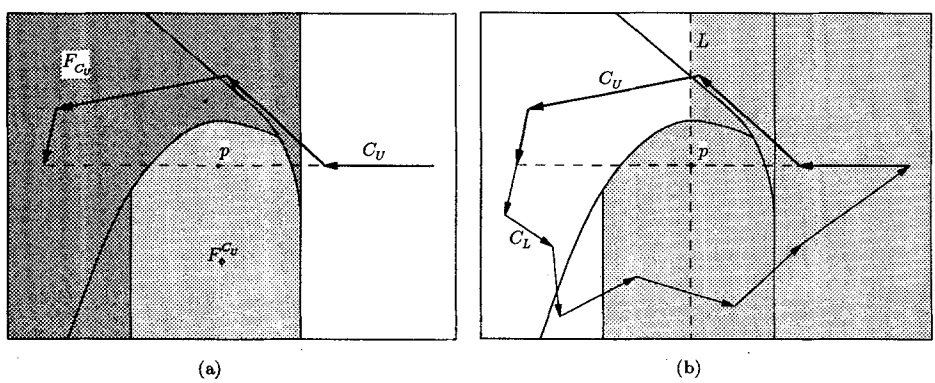

Fig. 11. Construction of the CVD for the upper chain of a star-shaped polygon.

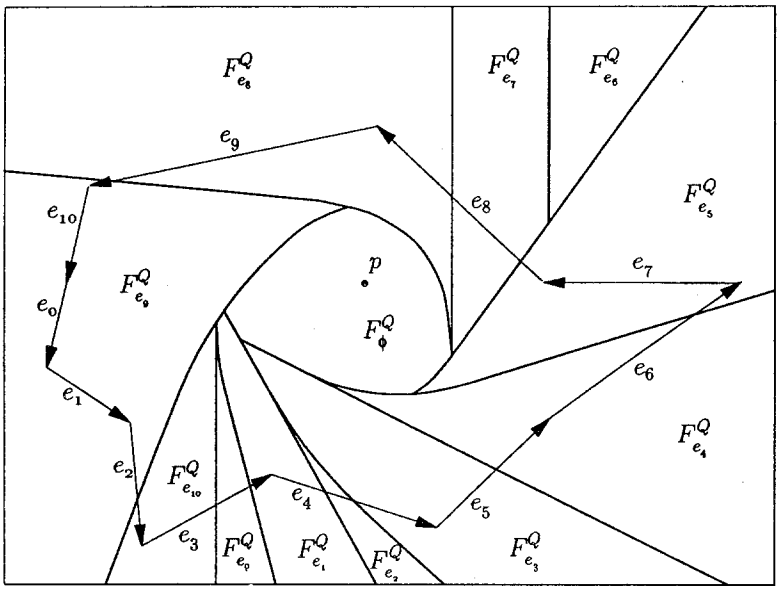

Fig. 12. The CVD of the star-shaped polygon.

of the CVD of the star-shaped polygon Q^*. An outline of this algorithm is as follows.

Algorithm (CVD_Star_Shaped_Polygon)
CVD_Obtuse_Star_Shaped_Chain(C_U);
$S \leftarrow F^{C_U}_{\varphi}$;
CVD_Obtuse_Star_Shaped_Chain(C_L);
$S \leftarrow F^{C_L}_{\varphi}$;
CVD_Obtuse_Star_Shaped_Chain(C_U);

It is noted that the construction of the CVD for Q^* is equivalent to constructing the CVDs for three obtuse star-shaped chains (one and a half rounds of the star-shaped polygon). Since the construction of $F^{Q^*}_{e_i}$, for all $e_i \in C_U$, and $F^{Q^*}_{e_j}$, for all $e_j \in C_L$, takes $O(n)$ time each, where n is the number of vertices of Q^*, the time required for constructing the CVD for a star-shaped polygon is $O(n)$.

5. Circular Visibility Diagram of a Pocket. In this section the construction of CVDs of pockets with respect to p is discussed. Each such pocket has its two endpoints collinear with p, but p does not lie on the lid. As adopted earlier, the edges of a pocket are oriented counterclockwise; only visibility arcs hitting the left side (or the inside) of the pocket are of interest. In the example depicted in Figure 13(a), only counterclockwise arcs emanating from p can hit the inside of the pocket without first hitting the outside of the pocket. Such pockets are called *CCW pockets*. Pockets on the opposite side of a lid, on the other hand, can only

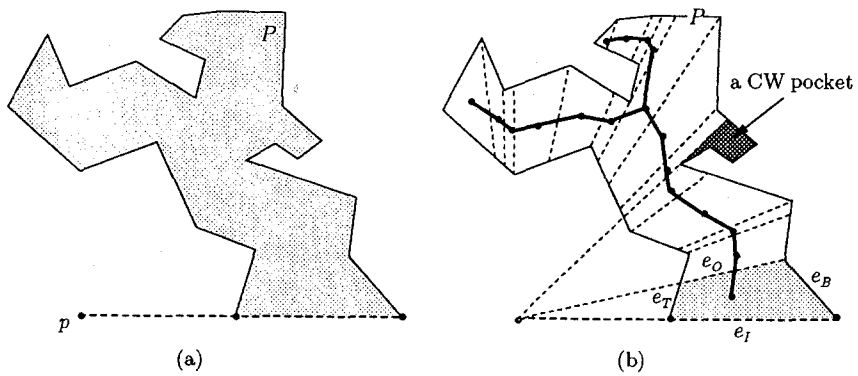

Fig. 13. (a) A counterclockwise pocket. (b) The trapezoidization of a pocket.

have clockwise visibility arcs hitting the inside of them and are thus called *CW pockets*. As illustrated in Figure 13(b), the darker shaded area is a CW pocket within a CCW pocket.

To construct the CVD for a pocket efficiently, a partial order in which arcs emanating from p hit the edges of the pocket is required. However, the edges of a pocket do not inherently possess such an order. In the following the edges of a pocket are decomposed into edges exhibiting partial ordering: a CVD is first computed with respect to the decomposed edges; and regions in such a CVD are then merged to obtain the CVD of the pocket with respect to the original (undercomposed) edges.

The decomposition of the edges of a pocket is done by utilizing *vertex–edge visible pairs* joined by dashed line segments, as shown in the pocket of Figure 13(b). A vertex–edge visible pair is a vertex and an edge which can be connected by a line segment lying entirely inside the pocket. By employing line segments whose extensions pass through p to connect all the vertex–edge pairs of the pocket, the interior of the pocket is decomposed into *trapezoids*,[4] as shown in Figure 13(b). Such a decomposition is also known as a trapezoidization [5], [18], [21]. Tarjan and van Wyk [21] show that a trapezoidization of a simple polygon can be done in $O(n \log\log n)$ time, where n is the total number of vertices in the simple polygon. As a recent development, the time complexity of the trapezoidization algorithm is reduced to $O(n)$ by Chazelle [5].

Since no counterclockwise visibility arc can reach the inside of a CW pocket, the trapezoids in CW pockets such as the one depicted in Figure 13(b) are discarded from further consideration. Aslo, since counterclockwise visibility arcs cannot reach portions of the pocket that extends to the other side of the line coincident with the lid, such portions are discarded as well. Consequently, the polar angles of the remaining in-edges span less than 180°. After removing these

[4] Lines originating from the same point can be viewed as in parallel.

two types of regions, it can be safely assumed that the trapezoid containing the lid of a pocket has the *smallest* θ with respect to p.

Each trapezoid in the pocket consists of four sides. The *in-edge* and the *out-edge* are connecting line segments of the vertex–edge visible pairs, where the *in-edge* is the side with the smaller θ. Both the *in-edge* and the *out-edge* are considered to be transparent. The *top-edge* and the *bottom-edge* are portions of the edges of the pocket, where the *top-edge* is closer to p than the *bottom-edge* is. Both the *top-edge* and the *bottom-edge* are considered to be opaque.

The circular arcs are now classified with respect to the side of a trapezoid they hit. Let e_I, e_O, e_T, and e_B denote the *in-edge*, *out-edge*, *top-edge*, and *bottom-edge*, respectively, of a trapezoid T_i. Let $F_{e_I^-}^{T_i}$ denote the region containing all the centers about which arcs emanating from p hit e_I from the outside (or the right side).[5] Since e_T and e_B are opaque and since only counterclockwise visibility arcs are employed, any arc which hits e_T, e_B, or e_O from the inside must first pass through e_I. After crossing e_I, an arc will then first hit e_T, e_B, or e_O. Those arcs that hit e_T or e_B are blocked. Those arcs that hit e_O, on the other hand, may pass through e_O (as e_O is transparent), come back through it, and hit either e_T or e_B. However, since p is collinear with e_O, be Lemma 2.1, those arcs that pass through e_O cannot come back into T_i through e_O. Therefore, $F_{e_I^-}^{T_i}$ can be partitioned into three mutually exclusive regions, $F_{e_T}^{T_i}$, $F_{e_B}^{T_i}$, and $F_{e_O}^{T_i}$, corresponding to arcs that pass through e_I and hit e_T, e_B, and e_O, respectively. Such a partition is illustrated in Figure 14.

A partial order in which arcs about points in $F_{e_I^-}^{T_i}$ intersect e_T, e_O, and e_B is utilized to compute $F_{e_T}^{T_i}$, $F_{e_B}^{T_i}$, and $F_{e_O}^{T_i}$.

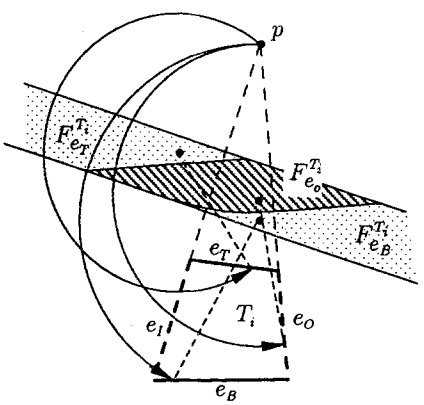

Fig. 14. $F_{e_I^-}^{T_i}$ is divided into $F_{e_T}^{T_i}$, $F_{e_B}^{T_i}$, and $F_{e_O}^{T_i}$.

[5] As adopted earlier, e_I is a directed edge such that the inside of the trapezoid is to the left of it. The notation e_I^- indicates e_I directed in the opposite direction. In other words, visibility arcs about points in $F_{e_I^-}^{T_i}$ will hit e_I^- from the left, consistent with the notation of F_A^B.

THEOREM 5.1. *Let F_{e_T}, F_{e_B}, and F_{e_O} be the regions containing all the points about which circular arcs hit e_T, e_B, and e_O, respectively. Then*

$$\begin{cases} F_{e_B}^{T_i} = F_{e_{\bar{I}}}^{T_i} \cap F_{e_B}, \\ F_{e_T}^{T_i} = F_{e_{\bar{I}}}^{T_i} \cap (F_{e_T} - F_{e_B}), \\ F_{e_O}^{T_i} = F_{e_{\bar{I}}}^{T_i} \cap (F_{e_O} - F_{e_B}). \end{cases}$$

PROOF. Since e_T is facing away from the emanating point p, visibility arcs which hit e_T immediately after crossing e_I will not intersect e_O or e_B afterward. Also, by Lemma 2.1, a visibility arc which hits e_O immediately after crossing e_I will not intersect e_B or e_T afterward. However, since e_B is facing toward p, a visibility arc which hits e_B immediately after crossing e_I may intersect e_O or e_T afterward. These three relations indicate that only e_B may obstruct arcs in hitting e_O or e_T.

Since e_O and e_T do not obstruct visibility arcs passing through e_I in hitting e_B, $F_{e_B}^{T_i} = F_{e_{\bar{I}}}^{T_i} \cap F_{e_B}$. On the other hand, since e_B may obstruct visibility arcs in hitting e_T or e_O, $F_{e_T}^{T_i} = (F_{e_{\bar{I}}}^{T_i} \cap F_{e_T}) - F_{e_B}^{T_i}$ and $F_{e_O}^{T_i} = (F_{e_{\bar{I}}}^{T_i} \cap F_{e_O}) - F_{e_B}^{T_i}$. By substituting $F_{e_B}^{T_i}$ with $(F_{e_{\bar{I}}}^{T_i} \cap F_{e_B})$,

$$\begin{aligned} F_{e_T}^{T_i} &= (F_{e_{\bar{I}}}^{T_i} \cap F_{e_T}) - (F_{e_{\bar{I}}}^{T_i} \cap F_{e_B}) \\ &= F_{e_{\bar{I}}}^{T_i} \cap (F_{e_T} - F_{e_B}). \end{aligned}$$

Similarly, $F_{e_O}^{T_i} = F_{e_{\bar{I}}}^{T_i} \cap (F_{e_O} - F_{e_B})$, which completes the proof. □

With the visibility arcs classified according to the side of a trapezoid they hit, the transition of the visibility arcs from one trapezoid to the next is now examined. The θ's of the *in-edges* exhibit a partial order in which the visibility arcs pass through these trapezoids. The same order also gives rise to the order in which arcs hit the *top-edges* and the *bottom-edges* of the trapezoids in a pocket. Such a partial order can be uniquely represented by the dual graph of the trapezoidized pocket, in which each node is associated with a trapezoid and each link with any two trapezoids sharing a side. This dual graph is clearly a partial-order tree with the root node representing the trapezoid containing the lid of the pocket,[6] as shown in Figure 13(b).

The construction of the CVD of a pocket starts from T_R, the trapezoid at the root of the partial-order tree. $F_{e_{\bar{I}}}^{T_R}$ is subsequently partitioned into three regions: $F_{e_T}^{T_R}$, $F_{e_B}^{T_R}$, and $F_{e_O}^{T_R}$, by utilizing Theorem 5.1. The region $F_{e_O}^{T_R}$ then becomes $F_{e_{\bar{I}}}^{T_1}$ of T_1, the *immediate descendant trapezoid* (*child*) of T_R. The CVD of a pocket is constructed by orderly partitioning $F_{e_{\bar{I}}}^{T_i}$ (or $F_{e_O}^{T_{i-1}}$) into $F_{e_T}^{T_i}$, $F_{e_B}^{T_i}$, and $F_{e_O}^{T_i}$

[6] There are trapezoids with three opaque sides and only one transparent side. However, since such trapezoids only appear at the leaves of the partial order tree, they will not affect the algorithm, and therefore are not discussed.

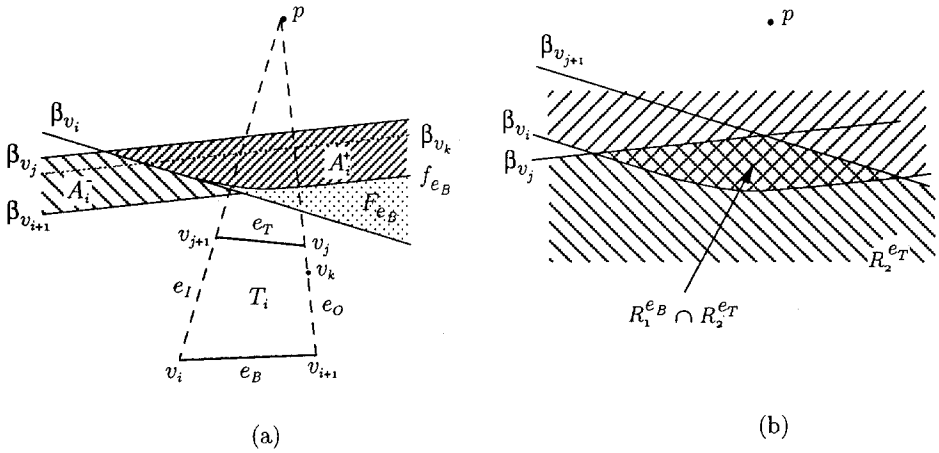

Fig. 15. (a) $(F_{e_O} - F_{e_B})$ is divided into A_i^+ and A_i^-. (b) $(R_1^{e_B} \cap R_2^{e_T})$.

throughout the partial-order tree. It is noted that arcs which hit e_T and e_B of T_i hit the same two edges in the pocket P where T_i resides, which means $F_{e_T}^P = F_{e_T}^{T_i}$ and $F_{e_B}^P = F_{e_B}^{T_i}$. Such a partition seems to require $O(n \log n)$ time to compute [10] for a pocket with n vertices. However, by showing that $F_{e_O}^{T_i}$ is convex and utilizing the counterclockwise order of the *in-edges* about p, the partition can be completed in linear time.

Before verifying the two properties indicated above, the detailed construction of $F_{e_O}^{T_i}$ is first examined. Let the trapezoid of interest, T_i, be depicted as in Figure 15(a). By Theorem 5.1, $F_{e_O}^{T_i}$ equals the intersection of $F_{e_I}^{T_i}$ and $(F_{e_O} - F_{e_B})$. Since F_{e_O} is a Type-T region bounded by β_{v_j} and $\beta_{v_{i+1}}$ and F_{e_B} is a Type-L region bounded by f_{e_B} and β_{v_i}, subtracting F_{e_B} from F_{e_O} always results in two separate regions, denoted as A_i^+ and A_i^-, as shown in Figure 15(a). It is noted that $A_i^- \subset R_3^{e_B}$, i.e., all the arcs centered at points in A_i^- will hit e_B from the outside. Since an arc cannot hit both e_I and e_B from the outside, the intersection of A_i^- and $F_{e_I}^{T_i}$ is therefore always empty. On the other hand, A_i^+ may contribute to $F_{e_O}^{T_i}$ since $(R_1^{e_B} \cap R_2^{e_T}) \subset A_i^+$, where $(R_1^{e_B} \cap R_2^{e_T})$, as illustrated in Figure 15(b), contains all the centers about which arcs emanating from p and not hitting either e_T or e_B from the inside occur. Therefore, the intersection of $F_{e_I}^{T_i}$ and $(F_{e_O} - F_{e_B})$ can be substituted with the intersection of $F_{e_I}^{T_i}$ and A_i^+. When e_O is adjacent to more than one trapezoid as shown in Figure 13(b), i.e., the current trapezoid has more than one child node, the region $F_{e_O}^{T_i}$ needs to be decomposed accordingly. This decomposition can be done easily by using bisectors of p and the points that separate e_O into e_I's of the descendants of the current trapezoid. Such bisectors are parallel to and lie between β_{v_j} and $\beta_{v_{i+1}}$. The dotted line β_{v_k} shown in Figure 15(a) cuts A_i^+ into two regions that correspond to visibility arcs going into two adjacent trapezoids whose *in-edges* are separated by v_k. The partition is then resumed for each resulting $F_{e_I}^{T_i}$. That $F_{e_O}^{T_i}$ is convex is shown by examining the intersection of $F_{e_I}^{T_i}$ and A_i^+.

THEOREM 5.2. *$F_{e_O}^{T_i}$ is convex, for all $T_i \in P$.*

PROOF. The initial $F_{e_I}^{T_R}$ is a stripe bounded by two parallel lines and is therefore convex. The subsequent $F_{e_O}^{T_i}$, which is equal to the intersection of A_i^+ and $F_{e_I}^{T_i}$, is convex since both A_i^+ and $F_{e_I}^{T_i}$ are convex. Therefore, by induction, all the $F_{e_O}^{T_i}$'s are convex. □

The computation for the intersection of $F_{e_I}^{T_i}$ and A_i^+, for all $T_i \in P$, is dominated by the computation of the intersection points between the boundary of $F_{e_I}^{T_i}$ and the boundary of A_i^+. It is shown that, by dividing the boundary of $F_{e_I}^{T_i}$ into two pieces and maintaining them with two stacks, all the intersections can be computed in linear time.

Since $F_{e_O}^{T_i}$ results from intersecting $F_{e_I}^{T_i}$ and A_i^+, the boundary of $F_{e_O}^{T_i}$ consists of portions of the β_{v_j}'s and the f_{e_i}'s contributing to the boundary of $F_{e_I}^{T_i}$, where v_j is a vertex of the *top-edge* of a trapezoid and e_i is the *bottom-edge* of a trapezoid. Since the *in-edges* of the pockets are ordered about p, the perpendicular bisectors of line segments between p and the vertices of the *top-edges* and the *bottom-edges* are also ordered, respectively, according to their normals (pointing toward p). Consequently, the β_{v_j}'s and the f_{e_i}'s of the A_i^+'s are in slope order, respectively, following the partial ordering of the pockets.[7] The construction of the CVD for a pocket by intersecting $F_{e_I}^{T_i}$ with A_i^+ is thus analogous to constructing an upper bay and a lower bay, as described in Section 4, simultaneously.

The upper bay, maintained by a stack S_U, consists of only the β_{v_j}'s whereas the lower bay, maintained by stack S_L, consists of only the f_{e_i}'s. Let μ and v denote the intersections between the upper bay and the lower bay, as shown in Figure 16(a). As the construction of the CVD proceeds, the intersection of $F_{e_I}^{T_i}$ and A_i^+ (with boundary β_{v_t} and f_{e_B}) is computed, as shown in Figure 16(b). The intersection points between β_{v_t} and the boundary of $F_{e_I}^{T_i}$ are sought by checking through S_U and S_L, respectively, starting from the end of the stacks containing μ. The β_{v_j}'s in S_U that do not intersect β_{v_t} are popped sequentially until one that does is found. Similarly, S_L is updated by popping out f_{e_i}'s which do not intersect β_{v_t} until one that does is found. The portion of β_{v_t} lying inside $F_{e_I}^{T_i}$ contributes a boundary curve to $F_{e_I}^{T_{i+1}}$, and is pushed into S_U. The region encompassed by this portion of β_{v_t}, and the boundary curves of $F_{e_I}^{T_i}$ between μ and the two intersections with β_{v_t} yields $F_{e_T}^{T_i}$. The boundary curves of $F_{e_I}^{T_i}$ between μ and the two intersections with β_{v_t} are the β_{v_j}'s and f_{e_i}'s being popped out of S_U and S_L when computing the intersections with β_{v_t}. Likewise, the intersection points between f_{e_B} and the boundary of $F_{e_I}^{T_i}$ can be identified by checking through S_U and S_L from v, the other end of the stacks. The portion of f_{e_B} lying inside $F_{e_I}^{T_i}$ contributes a boundary cuve to $F_{e_I}^{T_{i+1}}$, and is pushed into S_L. The region encompassed by this portion of f_{e_B} and the boundary curves of $F_{e_I}^{T_i}$ between v and the two intersections with f_{e_B} yields $F_{e_B}^{T_i}$. The boundary curves of $F_{e_I}^{T_i}$ between v and the two intersections with f_{e_B} are the β_{v_j}'s and f_{e_i}'s being popped out of S_U

[7] By construction, the slope on $f_{\overline{uv}}$ changes monotonically and is bounded by the slopes of β_u and β_v.

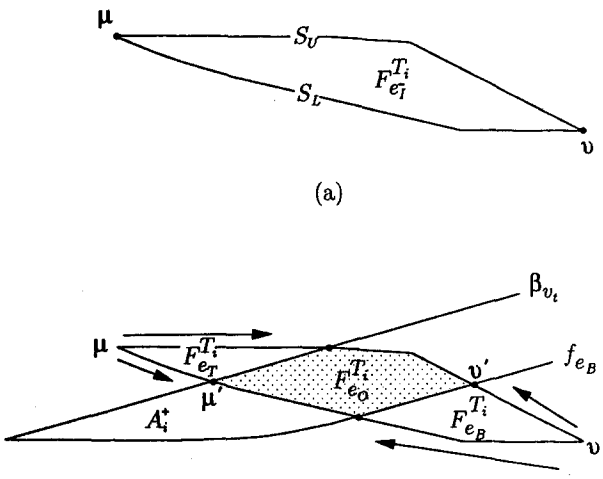

Fig. 16. The partition of $F_{e_I}^{T_i}$.

and S_L when computing the intersections with f_{e_B}. The curves remaining in S_U and S_L yield $F_{e_O}^{T_i}$. If neither β_{v_t} nor f_{e_B} intersects $F_{e_I}^{T_{i+1}}$, S_U and S_L remain the same. The partition continues until all the trapezoids are examined or when $F_{e_I}^{T_i} \cap A_i^+$ is empty; the latter indicates that all the visibility arcs are blocked. A procedure form of the algorithm is given.

Algorithm (CVD_Pocket).
  Trapezoidize P;
  Construct the dual of the trapezoidization;
  Compute $F_{e_I}^{T_R}$, $F_{e_T}^{T_R}$, $F_{e_B}^{T_R}$, $F_{e_O}^{T_R}$;
  if $(F_{e_I}^{T_R} \cap A_i^{*+}) \neq \varnothing$ **do**
    for every Child(T_R) **do**
      CVD_Trapezoid(T_R);
  Output S_U and S_L;

Procedure (CVD_Trapezoid(T_i)).
    /* Computing $F_{e_T}^P$ */
    while $(\beta_{v_t} \cap \text{Top}(S_U) = \varnothing)$ **do**
      Output Top(S_U); Pop(Top(S_U));
    while $(\beta_{v_t} \cap \text{Top}(S_L) = \varnothing)$ **do**
      Output Top(S_L); Pop(Top(S_L));
    $\beta_{v_t}^* \leftarrow (\beta_{v_t}$ between Top(S_U) and Top(S_L));
    Update Top(S_U) and Top(S_L);
    Output $\beta_{v_t}^*$; Push($\beta_{v_t}^*$, S_U);
    /* Computing $F_{e_B}^P$ */
    while $(f_{e_B} \cap \text{Bottom}(S_U) = \varnothing)$ **do**

Output Bottom(S_U); Pop(Bottom(S_U));
while ($f_{e_B} \cap$ Bottom(S_L) $= \varnothing$) **do**
Output Bottom(S_L); Pop(Bottom(S_L));
$f^*_{e_B} \leftarrow (f_{e_B}$ between Bottom(S_U) and Bottom(S_L));
Update Bottom(S_U) and Bottom(S_L);
Output $f^*_{e_B}$; Push($f^*_{e_B}$, S_L);
if ($F^{T_i}_{e_T} \cap A_i^+) \neq \varnothing$ **do**
for every Child(T_i) **do**
CVD_Trapezoid(T_i)

The time required for finishing all the partitioning can be shown to be linear in the number of vertices in the pocket. Let n_U and n_L be the number of β_{v_j}'s and f_{e_i}'s in S_U and S_L, respectively, and let $C(n_U, n_L)$ denote the total time required for partitioning the $F^{T_i}_{e_T}$ with S_U and S_L of size n_U and n_L, respectively. Suppose i_U, i_L, j_U, and j_L curves are checked, respectively, for identifying the intersections between β_{v_t} and S_U, β_{v_t} and S_L, f_{e_B} and S_U, and f_{e_B} and S_L. Then

$$C(n_U, n_L) = O(i_U) + O(i_L) + O(j_U) + O(j_L) + C(n_U - i_U - j_U, n_L - i_L - j_L).$$

By replacing $C(n_U, n_L)$ with $C(n_U + n_L)$,

$$C(n_U + n_L) = O(i_U + i_L + j_U + j_L) + C(n_U + n_L - i_U - j_U - i_L - j_L).$$

Let $(n_U + n_L)$ equal n and let $(i_U + j_U + i_L + j_L)$ equal k. By substituting n and k into the formula, a recurrence formula

$$C(n) = O(k) + C(n - k)$$

is obtained. The solution to this recurrence formula is clearly $O(n)$. In other words, the plane partition, which yields the CVD of a pocket, can be constructed in $O(n)$ time, where n is the number of vertices in the trapezoidized pocket.

It is clear that the number of vertices in the original pocket and the number of vertices in the trapezoidized pocket are of the same order. Therefore, given a pocket with n vertices, the CVD of the pocket with decomposed edges can be constructed in $O(n)$ time. The CVD of the pocket with the original edges can then be obtained by merging the regions corresponding to the decomposed edges of original edges, which can be computed easily in $O(n)$ time. Therefore, the total time complexity of the algorithm is still $O(n)$.

The CVD of the pocket P in Figure 13 is depicted in Figure 17. Figure 17(a) shows the CVD of P with the decomposed edges, and Figure 17(b) shows the CVD of P with the original edges.

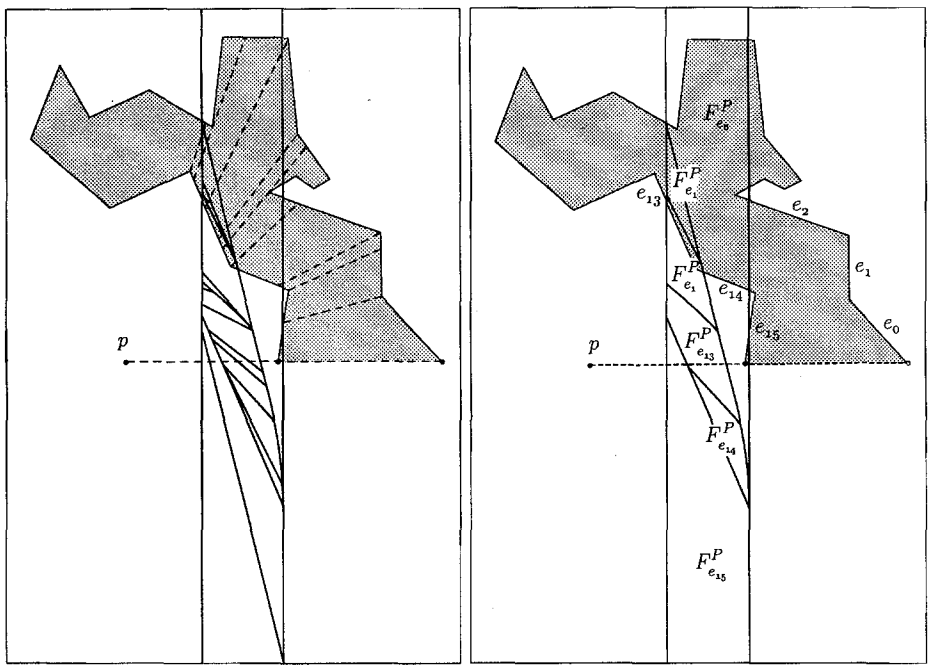

Fig. 17. (a) CVD of a pocket with decomposed edges. (b) CVD of a pocket with the original edges.

6. Analysis and Discussion. Given the linear-time algorithms for computing the CVDs for a star-shaped polygon and for a pocket, it is shown in the following that the CVD of a simple polygon Q can also be constructed in linear time. While the CVD of Q^* can be constructed in $O(n)$ time by directly applying the algorithm described in Section 4, computing the CVDs of the pockets in Q requires extra effort: Since some visibility arcs hitting the lids of the pockets in Q may be obstructed by other edges of Q^*, the CVD region of each of these lids will not be a simple Type-T region, as assumed in Section 5. By construction, the CVD region of a lid in Q is the intersection of a stripe, a region bounded by two parallel lines, and a region $F_\varphi^{C_i}$, as described in Section 4. Since both the stripe and $F_\varphi^{C_i}$ are convex, the CVD region of a lid in Q is always convex. With the CVD region of a lid being convex, the algorithm for computing the CVD of an independent pocket is therefore applicable to constructing the CVD for a pocket in Q. The time required for constructing the CVD for a pocket with m_i edges is $O(m_i + n_i)$, where n_i is the number of boundary curves of the region $F_{e_i}^Q$ (e_i being the lid of P_i). Since the sum of n_i is bounded by the total number of edges in Q^*, the total time required for constructing the CVDs for all the pockets is bounded by the total number of edges in Q.

The time complexity for constructing the CVD of a simple polygon is the sum of the time complexity for the individual processes, which are all bounded by $O(n)$.

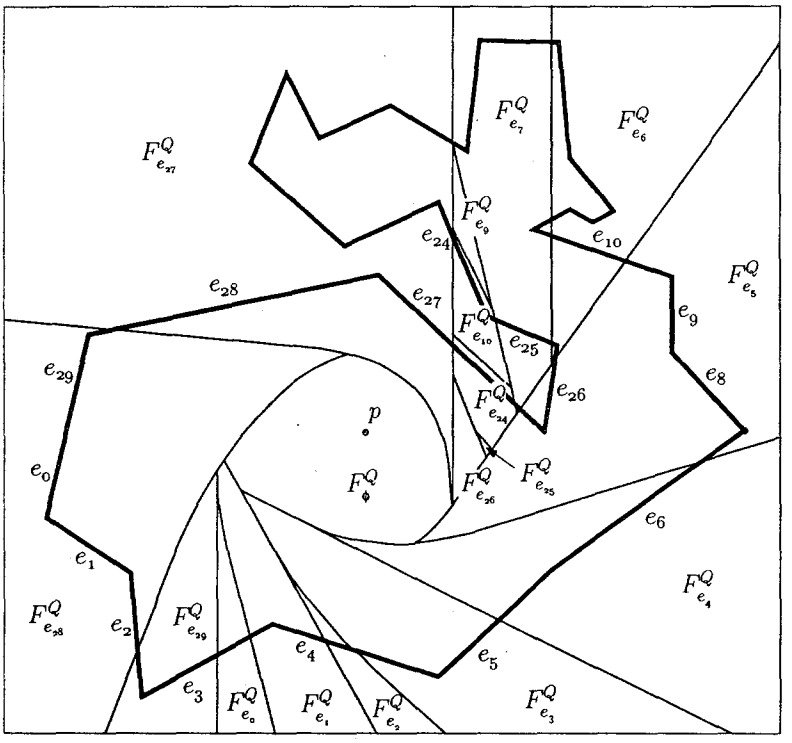

Fig. 18. The CVD of a simple polygon Q.

This algorithm thus computes the CVD of a simple polygon in linear time. The CVD of the polygon in Figure 4 is shown in Figure 18.

The partition curves of a CVD can be attributed to either the edges where the corresponding visibility arcs end, or the first "contact" between these visibility arcs and the boundary of Q. If visibility arcs passing through a vertex and remaining inside Q are considered to be blocked at the vertex, then the partition curve corresponding to such visibility arcs is attributed to the edge containing this vertex. On the other hand, if such visibility arcs are not considered to be blocked by the vertices, the partition curve is attributed to the edge where these visibility arcs end. Similarly, when visibility arcs tangent to an edge are considered to be blocked by the edge, the partition curve corresponding to such visibility arcs is attributed to the edge. If such visibility arcs are not considered to be blocked by the edges, the partition curve is attributed to the edge where these visibility arcs end.

The structure of CVDs enables many applications involving the notion of circular visibility to be solved efficiently. For example, the shortest circular arc from point p to a circularly visible point q can be identified directly from the CVD computed with respect to either point. CVDs can also be used for computing visibility hulls inside a simply polygon. Chou *et al.* [8] show that the circular visibility hull of a point inside a simple polygon can be obtained in linear time by

utilizing the CVD of the polygon computed with respect to the point. They also show that by computing a set of CVDs, the circular visibility hull of an edge inside a simple polygon can be obtained in $O(nk)$ time, where n is the total number of edges in the polygon and k is the number of CVDs computed. The value of k is bounded by the number of arcs and edges in the visibility hull that are not the edges of the polygon. In the worst case, k is of $O(n)$.

Circular visibility can be used to characterize the workspace of a stationary robot with rotary joints. A special case occurs in coordinate measurement where the locations of the rotary probe of a coordinate measuring machine need to be determined. By limiting the motion of the probe to one translation and one rotation (and in this order), probe paths for measuring a point on a polygonal object can be formulated as a combination of the linear visibility and circular visibility problem. The trajectory of the rotation of a probe tip, characterized by circular visibility, can be computed with the help of CVDs.

The idea of representing circular arcs emanating from a fixed point by their centers is also applicable to characterizing a set of parabolic curves emanating from a fixed point and having parallel axes. Whereas a circle can be viewed as having two coincident foci, a parabola can be viewed as having one of the two foci at infinity. By fixing the direction of the axes of parabolas, their foci at infinity are fixed. Therefore, such parabolas passing through a fixed point can be uniquely represented by their other foci. Chou *et al.* show that the parabolic visibility diagram of a simple polygon can also be computed in linear time [9]. It should be noted that the elements of such diagrams may not be straight lines and parabolic curves. For parabolic visibility diagrams, the partition curves consist of parabolic curves and hyperbolic curves.

Acknowledgment. The authors would like to thank Jan Wolter who initiated the work.

Appendix

LEMMA A.1. *Let C be an obtuse star-shaped chain, and let e_i and e_{i+1} be two consecutive edges in C. Then $F_{e_{i+1}} \cap R_2^{e_i} = \varphi$.*

PROOF. Let β_{v_i} represent the perpendicular bisectors of $\overline{pv}$, as shown in Figure 8. Since C is star-shaped, the fixed point p is to the left of e_i and e_{i+1}, implying that F_{e_i} and $F_{e_{i+1}}$ are both Type-L regions. The curve β_{v_i}, which is the perpendicular bisector of p and the vertex v_i joining edges e_i and e_{i+1}, divides the plane into two half-planes, one of which contains p. By definition, the region $F_{e_{i+1}}$ is on the side of the half-plane containing p, while R_2 of e_i is on the other half-plane. Therefore, $F_{e_{i+1}} \cap R_2^{e_i} = \varphi$. □

LEMMA A.2. *For some edge e_i of a polygon Q, the region $F_{e_i}^Q$, is unbounded if and only if some point on e_i is linearly visible from p.*

PROOF. A point q on e_i is linearly visible from p if and only if the line segment $\overline{pq}$ does not intersect any other edges of Q. A line segment is in fact a degenerate circular arc whose center lies on the perpendicular bisector of this line segment at infinity. A region corresponding to a linearly visible edge contains points to infinity and therefore is unbounded. The validity of the converse is easy to see. □

LEMMA A.3. *The counterclockwise order of the unbounded regions around p in the CVD is the same as that of the edges linearly visible from p.*

PROOF. The order of the linearly visible edges around p follows the order of the ray from p to the edges. Also, the order of the unbounded regions around p follows the order of the centers at infinity which in turn follows the order of the rays from p to the edge because the radii of the centers at infinity are perpendicular to the right of the rays from p. □

References

[1] Agarwal, P. K., and M. Sharir, Circular visibility from a point in a simple polygon, *Internat. J. Comput. Geom. Appl.*, **3**(1), 1–25, 1993.

[2] Agarwal, P. K., and M. Sharir, Circle shooting in a simple polygon, *J. Algorithms*, **14**, 69–87, 1993.

[3] Avis, D., and G. T. Toussaint, An optimal algorithm for determining the visibility of a polygon from an edge, *IEEE Trans. Comput.*, **30**, 910–914, 1981.

[4] Baker, W. M., *Algebraic Geometry: A New Treatise on Analytical Conic Sections*, Bell, London, 1906.

[5] Chazelle, B. M., Triangulating a simply polygon in linear time, *Discrete Computat. Geom.*, **6**, 485–524, 1991.

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

[7] Chazelle, B. M., L. T. Guibas, and D. T. Lee, The power of geometric duality, *BIT*, **25**(1), 76–90, 1985.

[8] Chou, S. Y., L. L. Chen, and T. C. Woo, Circular Visibility of a Simple Polygon, Working Paper 92–102, Department of Industrial and Manufacturing Systems Engineering, Iowa State University, 1992.

[9] Chou, S. Y., L. L. Chen, and T. C. Woo, Parabolic Visibility in the Plane, Working Paper 92–103, Department of Industrial and Manufacturing Systems Engineering, Iowa State University, 1992.

[10] Edelsbrunner, H., *Algorithms in Combinatorial Geometry*, Springer-Verlag, New York, 1987.

[11] Edelsbrunner, H., and L. J. Guibas, Topologically sweeping an arrangement, *J. Comput. System Sci.*, **38**, 165–194, 1989.

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

[13] Joe, B., and R. B. Simpson, Correction to Lee's visibility polygon algorithm, *BIT*, **27**, 458–473, 1987.

[14] Lee, D. T., Visibility of a simple polygon, *Comput. Vision Graphics Image process.*, **22**, 207–221, 1983.

[15] Lee, D. T., and Y. T. Ching, The power of geometric duality revisited, *Inform. Process, Lett.*, **21**, 117–122, 1985.

[16] Lee, D. T., and F. P. Preparata, An optimal algorithm for finding the kernel of a polygon, *J. Assoc. Comput. Mach.*, **26**, 415–421, 1979.
[17] O'Rourke, J., *Art Gallery Theorems and Algorithms*, Oxford University Press, Oxford, 1987.
[18] Preparata, F., and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.
[19] Suri, S., A linear time algorithm for minimum link paths inside a simply polygon, *Comput. Vision Graphics Image Process.*, **35**, 99–110, 1986.
[20] Suri, S., and J. O'Rourke, Worst-case optimal algorithms for constructing visibility polygons with holes, *Proc. 2nd ACM Symp. on Computational Geometry*, Yorktown Heights, NY, 1986, pp. 14–23.
[21] Tarjan, R. E., and C. van Wyk, An $O(n \log \log n)$-time algorithm for triangulating a simple polygon, *SIAM J. Comput.*, **17**(5), 143–178, 1985.