Algorithms and Correctness Proofs for Visibility Polygon Computations

B.Joe
R.B. Simpson
Department of Computer Science

Research Report CS-87-03
February 1987

Algorithms and Correctness Proofs for Visibility Polygon Computations

B. Joe

Dept. of Computing Science
University of Alberta

R. B. Simpson

Dept. of Computer Science
University of Waterloo

ABSTRACT

We present a modification and extension of the (linear time) visibility polygon algorithm of Lee (1983) and a proof of its correctness. The algorithm computes the visibility polygon of a simple polygon from a viewpoint that is either interior to the polygon, or in its blocked exterior (the cases of viewpoints on the boundary or in the free exterior being simple extensions of the interior case). The algorithm is described by a procedural decomposition in pseudocode and the proof establishes pre- and post-conditions for the procedures.

We show by example that the original algorithm by Lee, and a more complex algorithm by El Gindy and Avis (1981), can fail for polygons that wind sufficiently. We present a second version of the algorithm, which does not extend to the blocked exterior case.

CR Categories : F.2.2

Keywords : computational geometry, visibility, algorithm correctness

1. Introduction

We consider the following problem: Given a simple polygon P and a viewpoint z in the plane, find all points on the boundary of P that are 'visible' from z. The position of the viewpoint z determines three cases: z in the interior or exterior of P, or on the boundary of P. The exterior case can be further categorized as free exterior or blocked exterior depending on whether there exists a ray from z which does not intersect P or there is no such ray. The boundary and free exterior cases can be handled as simple extensions of the interior case. So from an

This work was partially supported by grants from the Central Research Fund of the University of Alberta and the Natural Sciences and Engineering Research Council of Canada.

algorithmic viewpoint, there are basically two cases, the interior case and the blocked exterior case.

Two linear time algorithms for computing the visibility polygon of P from an interior viewpoint have been published, Lee (1983) and El Gindy and Avis (1981), and Lee also presented a modification for the blocked exterior case in his paper. Lee's algorithm is simpler in structure; in particular it requires only one stack which eventually yields the visibility polygon, as opposed to three in the El Gindy and Avis algorithm. However, all these algorithms can fail for polygons which wind sufficiently and examples of polygons for which they fail are given in Figures 1 and 2. The failure of Lee's algorithm is a technicality that can be readily corrected, once its origins are understood; the failure of the El Gindy and Avis algorithm seems far more fundamental and we do not believe it can be corrected.

In this paper, we present a modification of Lee's algorithm which handles both interior and blocked exterior cases, and a second modification that handles only the interior case (the boundary and free exterior cases being immediately accessible from the interior case). We also present new proofs of correctness for the first of these algorithms. Although some of the proofs are quite complex, we believe the past difficulties in formulating a correct algorithm and proof and the importance of this problem in computational geometry justify a rather lengthy, detailed exposition. Our description of the algorithm is organized in a procedural decomposition with pre- and post-conditions specified for the procedures that aid in the presentation of the proof. This decomposition also helps to make the modifications to the original algorithm by Lee more apparent.

For a viewpoint z in the interior or exterior of P, this problem was originally called the two-dimensional hidden line elimination problem in computer graphics (Freeman and Loutrel (1967)). Our interest in this problem arose in the application of finite element triangulation (Joe and Simpson (1986)), where z is a reflex vertex on the boundary of P and an extended version of the method of Schachter (1978) is used to decompose a simple polygon into convex polygons.

Other recent papers in the computational geometry literature which reference this problem include Avis and Rappaport (1985), Guibas et al (1986), and Suri and O'Rourke (1986).

In Section 2, we give our notation and definitions. In Section 3, we present pseudocode for the two modified versions of Lee's algorithm and mention where the algorithms of Lee and El Gindy/ Avis are incorrect. In Section 4, we prove the correctness of the first modified version and state how to similarly prove the correctness of the second modified version. The viewpoint is assumed to be interior or blocked exterior in Sections 3 and 4. In Section 5, we describe how the algorithms can also be used for boundary and free exterior viewpoints. In Appendix A, we illustrate the algorithms for the examples in Figures 1 and 2.

2. Notation and definitions

We first present our notation for line segments, rays, and chains. Let u and v be distinct points. The line segment joining u and v is denoted by uv, (uv), $(uv]$, or $[uv)$ depending on whether both endpoints are included, both endpoints are excluded, only u is excluded, or only v is excluded, respectively. The directed half-line (ray) originating at u and going through v is denoted by $\vec{uv}$.

A chain of connected line segments u_0u_1, $u_1u_2, \cdots, u_{m-1}u_m$ is denoted by $u_0u_1 \cdots u_m$. The chain is open if $u_0 \neq u_m$ and closed if $u_0 = u_m$. The chain is simple if it does not intersect itself, with the exception that $u_0 = u_m$ is allowed. If $C = u_0u_1 \cdots u_m$ and $C' = w_0w_1 \cdots w_r$ are two open chains such that $u_m = w_0$, then $C \,||\, C'$ denotes the chain $u_0 \cdots u_m w_1 \cdots w_r$ formed by concatenating the two chains. If chain $C = u_0u_1 \cdots u_m$ is closed and simple, we define $Left(C)$ and $Right(C)$ to be the regions to the left and right, respectively, of the chain as it is traversed from u_0 to u_m, and we define $Int(C)$ and $Ext(C)$ to be the bounded interior and unbounded exterior regions, respectively, determined by the chain. These two regions exist from the Jordan Curve Theorem and do not include C. Depending on the orientation of the chain,

either $Int(C)=Left(C)$ and $Ext(C)=Right(C)$, or $Int(C)=Right(C)$ and $Ext(C)=Left(C)$.

Let u, v, and w be distinct points. The chain uvw is said to be a *left turn* or *right turn* if w is to the left of or right of, respectively, the directed line from u to v; and uvw is said to be a *backward move* or *forward move* if u, v, w are collinear, and $[uv)$ and $(vw]$ overlap or do not overlap, respectively.

We now present our notation for the simple polygon P and the definitions of angular displacement and visibility from the viewpoint z. The boundary of P is a simple closed chain. We denote the boundary, interior, and exterior of P by $Bd(P)$, $Int(P)$, and $Ext(P)$, respectively, so that $P=Bd(P)\cup Int(P)$, $Int(P)$ is a bounded region, and $Ext(P)$ is an unbounded region. The viewpoint z may be in $Int(P)$, $Ext(P)$, or $Bd(P)$. If $z \in Ext(P)$, then z is said to be *blocked exterior* to P if all rays emanating from z intersect $Bd(P)$, and z is said to be *free exterior* otherwise. Without loss of generality, we assume that the coordinate system is translated so that z is at the origin. We denote the polar angle of a point v by $\theta(v)$ where $0 \le \theta(v) < 2\pi$.

Let $C=u_0u_1\cdots u_m$ be a simple chain which does not intersect z. We define the *angular displacement*, $\alpha(u_i)$, of u_i with respect to z as follows : $\alpha(u_0)=\theta(u_0)$;

for $1 \le i \le m$,

$$\alpha(u_i)=\begin{cases}\alpha(u_{i-1})+\text{angle}(u_i z u_{i-1}) & \text{if } zu_{i-1}u_i \text{ is a left turn}\\ \alpha(u_{i-1})-\text{angle}(u_{i-1}zu_i) & \text{if } zu_{i-1}u_i \text{ is a right turn}\\ \alpha(u_{i-1}) & \text{if } z, u_{i-1}, u_i \text{ are collinear,}\end{cases} \tag{1.1}$$

i.e. $\alpha(u_i)=\theta(u_i)+2\pi k$ where k is the integer determined so that $|\alpha(u_i)-\alpha(u_{i-1})|<\pi$. The definition of angular displacement can be extended to any point on the chain C. Note that the angular displacement measures the 'winding' of the chain in addition to the polar angle. We define the *angular displacement* of the chain C to be $\delta(C)=\alpha(u_m)-\alpha(u_0)$. If the chain is closed (i.e. $u_0=u_m$), then from classical complex integration results (Carrier, Krook, and Pearson (1966, Section 2.3), Henrici (1974, Section 4.6)),

$$\delta(C) = \begin{cases} 0 & \text{if } z \in Ext(C) \\ 2\pi & \text{if } z \in Int(C) = Left(C) \\ -2\pi & \text{if } z \in Int(C) = Right(C). \end{cases} \tag{1.2}$$

Freeman and Loutrel (1967), who presented a nonlinear-time algorithm for finding the visibility polygon, used the term *total angle* for angular displacement.

If $z \in Int(P) \cup Bd(P)$, then the point v is said to be *visible* from z with respect to P if (zv) is entirely in $Int(P)$. If $z \in Ext(P)$, then the point v is said to be *visible* from z with respect to P if (zv) is entirely in $Ext(P)$. The problem we are considering is to find a subset $\bar{V}(P,z)$ of points on $Bd(P)$ which are visible from z, i.e. $\bar{V}(P,z) = \{ v \mid v \in Bd(P)$ and v is visible from $z \}$. An equivalent problem is to find the star-shaped simple polygon, $V(P,z)$, the visibility polygon from z, which is the closure of the set $\{ u \mid u \in zv$ and $v \in \bar{V}(P,z) \}$. Note that some points on $Bd(V(P,z))$ are not visible from z, according to our definition, but they are included to enable $Bd(V(P,z))$ to be a simple closed curve (see Figures 1 and 2).

In the rest of this paper, except for Section 5, we assume that $z \in Int(P)$ or z is blocked exterior to P. Note that in both of these cases, all rays emanating from z intersect $Bd(P)$, so there exists exactly one visible boundary point at each polar angle. In Section 5, we will describe how the algorithms given in Section 3 can also be used for boundary and free exterior viewpoints.

If $z \in Int(P)$, we orient the vertices of P in counterclockwise order ($Int(P)$ is to the left as $Bd(P)$ is traversed), and label them $v_0, v_1, \cdots, v_{n-1}$, and $v_n = v_0$, where v_0 is the point on $Bd(P)$ which is on the positive x-axis and has the smallest x-coordinate. (The edges of $Bd(P)$ are v_0v_1, $v_1v_2, \cdots, v_{n-1}v_n$.) If z is blocked exterior to P, we orient the vertices of P in clockwise order ($Int(P)$ is to the right as $Bd(P)$ is traversed), and label them $v_0, v_1, \cdots, v_{n-1}$, and $v_n = v_0$, where v_0 is the point on $Bd(P)$ which is on the positive x-axis and has the smallest x-coordinate. Note that v_0 is visible from z in both cases. We consider v_0 and v_n to be 'distinct' points with v_0 on edge v_0v_1 and v_n on edge $v_{n-1}v_n$.

Let s and t be points on $Bd(P)$. s is said to *occur* before t if s appears before t in a traversal of $Bd(P)$ from v_0 to v_n. If s occurs before t then we denote the chain of $Bd(P)$ from s to t by $Ch[s,t]$, $Ch(s,t)$, $Ch(s,t]$, or $Ch[s,t)$ depending on whether both s and t are included, both s and t are excluded, only s is excluded, or only t is excluded, respectively. This definition also applies when $s=t$ in which case the chain degenerates to a single point or the empty set. Note that $Ch[v_0,v_n]=Bd(P)$.

Let v be a point on $Bd(P)$. If $z \in Int(P)$ [alternatively $Ext(P)$], we define v to be

(a) *CCW-oriented* if v lies on an edge which is oriented counterclockwise (forms a left turn) with respect to z and zv contains a nonempty subsegment $[uv)$ which is entirely in $Int(P)$ [$Ext(P)$],

(b) *CW-oriented* if v lies on an edge which is oriented clockwise (forms a right turn) with respect to z and zv contains a nonempty subsegment $[uv)$ which is entirely in $Ext(P)$ [$Int(P)$],

(c) *CL-oriented* if v lies on an edge which is collinear with z and zv contains a nonempty subsegment $[uv)$ which is entirely on $Bd(P)$.

See Figures 1 and 2 for examples of CCW-oriented and CW-oriented points.

From the definition of 'visible' and the orientation of $Bd(P)$, we have the following two lemmas.

Lemma 1 : If (zv) intersects $Bd(P)$, then v is not visible from z.

Lemma 2 : If v is CW-oriented or CL-oriented, then v is not visible from z.

3. The algorithms

In this section, we present the two modified versions of Lee's algorithm. We refer to them as Algorithm 1 and Algorithm 2. Algorithm 1 is the version which works correctly for both interior and blocked exterior viewpoints. Algorithm 2 is the version which works correctly for

interior viewpoints only. To reduce the number of cases in the algorithms and the correctness proof, we make the simplifying assumption that no two vertices v_i and v_j, $0 \le i < j < n$, have the same polar angle with respect to z (Lee (1983) and El Gindy and Avis (1981) also made this assumption). It is straightforward to add the extra cases to the algorithms when this assumption is removed. This is done in Joe and Simpson (1985) for Algorithm 2.

Algorithms 1 and 2 carry out a sequential scan of $Bd(P)$ starting from edge v_0v_1 and ending at edge $v_{n-1}v_n$, while manipulating a stack of boundary points $s_0, s_1, \cdots, s_t$ such that ultimately the chain $s_0s_1 \cdots s_t$ becomes $Bd(V(P,z))$. When processing the current edge v_iv_{i+1}, the operations that may be performed are:

(a) add boundary points to the top of the stack,

(b) delete boundary points from the top of the stack,

(c) scan edges $v_{i+1}v_{i+2}$, $v_{i+2}v_{i+3}$, $\cdots$ for the first edge v_kv_{k+1} to satisfy a certain condition.

Operation (c) is performed if v_iv_{i+1} enters a 'hidden' region where boundary points are not visible from z. There are four types of hidden regions, and the condition for exiting the scan in each of these regions is slightly different. When v_iv_{i+1} is not in a hidden region or upon exiting a hidden region with the new current edge v_iv_{i+1}, operation (a) or (b) is performed if zv_iv_{i+1} is a left turn or right turn, respectively. Note that if zv_iv_{i+1} is a left turn then the points on (v_iv_{i+1}) may be visible from z since they are CCW-oriented, and if zv_iv_{i+1} is a right turn then the points on (v_iv_{i+1}) are not visible from z since they are CW-oriented. Algorithms 1 and 2, as well as Lee's two algorithms for interior and blocked exterior viewpoints, differ only in the conditions for exiting the scan in the four types of hidden regions.

Algorithm 1 is given in the pseudocode below. It is decomposed into the six procedures LEFT, RIGHT, SCANA, SCANB, SCANC, and SCAND, along with a driver procedure VISPOL which repeatedly calls them. Procedure LEFT is called when the previous edge $v_{i-1}v_i$ is not in a

hidden region, $zv_{i-1}v_i$ is a left turn, and one or two boundary points have just been added to the stack. Procedure RIGHT is called when the previous edge $v_{i-1}v_i$ is not in a hidden region, $zv_{i-1}v_i$ is a right turn, and $v_{i-1}v_i$ is in front of the tail part of the chain $s_0s_1 \cdots s_t$ so that points must be deleted from the stack. Procedure SCANA, SCANB, SCANC, or SCAND is called when the previous edge $v_{i-1}v_i$ enters one of the four hidden regions. In the next section, we prove that the edges scanned in a hidden region are not visible from z, and that there exists an edge which satisfies the condition for exit from the scan.

The viewpoint z, the number of vertices n, and the vertices $v_0, v_1, \cdots, v_{n-1}, v_n = v_0$ of P are global variables in the six procedures. They all have input/output parameters *proc*, i, t, S, and w. *proc* is a string containing the name of the current or next procedure or FINISH. i is the index of the current edge v_iv_{i+1} ($Ch[v_0,v_i]$ has been processed so far). t is the index of the top stack point s_t. S is the chain of stack points $s_0s_1 \cdots s_t$. w is a point on $v_{i-1}v_i$ which is required only in SCANC and SCAND; it is used extensively in the correctness proof of the next section. On entering LEFT, SCANA, and SCANB, $w = v_i$; on entering RIGHT, SCANC, and SCAND, w is the point on $v_{i-1}v_i$ such that $\theta(w) = \theta(s_t)$.

The pseudocode is designed so that the following properties are satisfied by the stack points $s_0, s_1, \cdots, s_t$ on entrance to each of the six procedures. In these properties, i and t are the parameters described above. In this section and the next, $\alpha(s_j)$ denotes the angular displacement of s_j on the varying chain $s_0s_1 \cdots s_t$.

(S1) $0 \le t \le i$, $s_j \in Ch[v_0,v_i]$ for $0 \le j \le t$, $s_0 = v_0$, and s_j occurs before s_{j+1}.

(S2) $0 = \alpha(s_0) \le \alpha(s_1) \le \cdots \le \alpha(s_t) \le 2\pi$; at most two consecutive s_j have the same angular displacement.

(S3) If $\alpha(s_j) < \alpha(s_{j+1})$ then $s_js_{j+1} \subseteq Bd(P)$. If $\alpha(s_j) = \alpha(s_{j+1})$ then (s_js_{j+1}) is not on $Bd(P)$.

The various cases in the pseudocode below are labeled for reference in the figures and in the

the next section. The statement $(proc, i, t, S, w) :=$ (arg1, arg2, arg3, arg4, arg5) used in the procedures is short for the simultaneous execution of the statements $proc :=$ 'arg1'; $i :=$ arg2; $t :=$ arg3; $S :=$ arg4; $w :=$ arg5. A '#' symbol indicates that the rest of the line is a comment.

```
procedure VISPOL(z, n, P, t, V(P,z))        # Pseudocode for Algorithm 1
# Input : Viewpoint z in Int(P) or blocked exterior to P, and vertices v_0, v_1, ..., v_n
#         of P with labeling and orientation of vertices as described in Section 2.
# Output: Vertices s_0, s_1, ..., s_t of visibility polygon V(P,z).

# Assumption (V0): No two vertices v_i, v_j have the same polar angle with respect to z.

      # See Figure 3 for possible configurations of v_0v_1 and v_{n-1}v_n.
(V1)  if zv_0v_1 is a left turn then
            (proc,i,t,S,w) := (LEFT, 1, 1, v_0v_1, v_1)
(V2)  else # zv_0v_1 is a right turn
            (proc,i,t,S,w) := (SCANA, 1, 0, v_0, v_1)
      endif

      repeat
            # Properties (S1), (S2), (S3) are satisfied.
            case proc of
                  'LEFT'  : LEFT(proc, i, t, S, w)
                  'RIGHT' : RIGHT(proc, i, t, S, w)
                  'SCANA' : SCANA(proc, i, t, S, w)
                  'SCANB' : SCANB(proc, i, t, S, w)
                  'SCANC' : SCANC(proc, i, t, S, w)
                  'SCAND' : SCAND(proc, i, t, S, w)
            endcase

(V3)        if proc = 'LEFT' and (s_{t-1}s_t) intersects zv_n^→ then
            # α(s_{t-1}) < 2π < α(s_t), zv_{i-1}v_i is a left turn, and s_t = w = v_i.
                  # Replace s_t (shorten s_{t-1}s_t) so that α(s_t) = 2π.
                  s_t := intersection of s_{t-1}v_i and zv_n^→
                  proc := 'SCANB'
            endif
      until proc = 'FINISH'

      # Properties (S1), (S2), (S3) are satisfied, and s_t = v_n and α(s_t) = 2π.
end # VISPOL
```

procedure LEFT($proc, i, t, S, w$) # Previous case can be (V1), (L2), (R2), (A3), or (D1).
(L0) $zv_{i-1}v_i$ is a left turn, $s_t = w = v_i$, $s_{t-1} \in [v_{i-1}v_i)$, and either $\alpha(s_t) < 2\pi$ or $s_t = v_n$.

    # See Figure 4 for possible locations of $v_i v_{i+1}$.
(L1) if $i = n$ then
        $(proc, i, t, S, w) := (\text{FINISH}, n, t, s_0 \cdots s_t, v_n)$
(L2) else if $zv_i v_{i+1}$ is a left turn then
        $(proc, i, t, S, w) := (\text{LEFT}, i+1, t+1, s_0 \cdots s_t v_{i+1}, v_{i+1})$
(L3) else if $zv_i v_{i+1}$ is a right turn and $s_{t-1}v_i v_{i+1}$ is a right turn then
        $(proc, i, t, S, w) := (\text{SCANA}, i+1, t, s_0 \cdots s_t, v_{i+1})$
(L4) else # $zv_i v_{i+1}$ is a right turn and $s_{t-1}v_i v_{i+1}$ is a left turn
        $(proc, i, t, S, w) := (\text{RIGHT}, i+1, t, s_0 \cdots s_t, v_i)$
    endif
end # LEFT

procedure RIGHT($proc, i, t, S, w$) # Previous case can be (L4), (R1), (A1), (B2), or (C1).
(R0) $zv_{i-1}v_i$ is a right turn, $zs_t v_i$ is a right turn, $\alpha(s_{t-1}) < \alpha(s_t)$, and
either (i) $s_t = w = v_{i-1}$ and $s_{t-1}s_t v_i$ is a left turn,
or (ii) s_t is not on $v_{i-1}v_i$, $w \in [v_{i-1}v_i)$, and $zs_t w$ is a backward move.

    # See Figure 5 for possible locations of $v_{i-1}v_i$ and $v_i v_{i+1}$.
    Scan $s_t s_{t-1}, \cdots, s_1 s_0$ for the first edge $s_j s_{j-1}$ such that
        (RA) $zs_j v_i$ is a right turn and $zs_{j-1}v_i$ is a left turn, or
        (RB) $zs_{j-1}s_j$ is a forward move and $v_{i-1}v_i$ intersects $(s_{j-1}s_j)$
    Delete $s_t, s_{t-1}, \cdots, s_{j+1}$ from stack

    if case (RA) then # $zs_{j-1}s_j$ is a left turn
        # Replace s_j (shorten $s_{j-1}s_j$).
        $s_j :=$ intersection of $(s_{j-1}s_j)$ and $\overrightarrow{zv_i}$
        # $zs_j v_i$ is a backward move and $(s_j v_i)$ is not on $Bd(P)$.
(R1)   if $zv_i v_{i+1}$ is a right turn then
            $(proc, i, t, S, w) := (\text{RIGHT}, i+1, j, s_0 \cdots s_j, v_i)$
(R2)   else if $zv_i v_{i+1}$ is a left turn and $v_{i-1}v_i v_{i+1}$ is a right turn then
            $(proc, i, t, S, w) := (\text{LEFT}, i+1, j+2, s_0 \cdots s_j v_i v_{i+1}, v_{i+1})$
(R3)   else # $zv_i v_{i+1}$ is a left turn and $v_{i-1}v_i v_{i+1}$ is a left turn
            $(proc, i, t, S, w) := (\text{SCANC}, i+1, j, s_0 \cdots s_j, v_i)$
        endif
(R4) else # case (RB)
        $u :=$ intersection of $v_{i-1}v_i$ and $(s_{j-1}s_j)$ # $u \in (v_{i-1}v_i)$
        # Delete s_j from stack.
        $(proc, i, t, S, w) := (\text{SCAND}, i, j-1, s_0 \cdots s_{j-1}, u)$
    endif
end # RIGHT

procedure SCANA($proc, i, t, S, w$) # Previous case can be (V2) or (L3).
(A0) $zv_{i-1}v_i$ is a right turn, $s_t = v_{i-1}$, $\alpha(s_t) < 2\pi$, and $w = v_i$.
If $i > 1$ then $\alpha(s_{t-1}) < \alpha(s_t)$ and $s_{t-1}s_t v_i$ is a right turn.

  # See Figure 6 for possible exit cases.
(AS) Scan $v_i v_{i+1}, \cdots, v_{n-1}v_n$ for the first edge $v_k v_{k+1}$ to intersect $\overrightarrow{zs_t}$
  u := intersection of $v_k v_{k+1}$ and $\overrightarrow{zs_t}$ # $u \in (v_k v_{k+1})$
(A1) if $zv_k v_{k+1}$ is a right turn and $zs_t u$ is a backward move then
    # $\delta(s_t v_i \cdots v_k u) = -2\pi$
    $(proc, i, t, S, w)$:= (RIGHT, $k+1, t, s_0 \cdots s_t, u$)
(A2) else if $zv_k v_{k+1}$ is a right turn and $zs_t u$ is a forward move then
    # $\delta(s_t v_i \cdots v_k u) = -2\pi$
    $(proc, i, t, S, w)$:= (SCAND, $k+1, t, s_0 \cdots s_t, u$)
(A3) else if $zv_k v_{k+1}$ is a left turn and $zs_t u$ is a forward move then
    # $\delta(s_t v_i \cdots v_k u) = 0$
    $(proc, i, t, S, w)$:= (LEFT, $k+1, t+2, s_0 \cdots s_t u v_{k+1}, v_{k+1}$)
(A4) else # $zv_k v_{k+1}$ is a left turn and $zs_t u$ is a backward move
    # This case is not possible.
  endif
end # SCANA

procedure SCANB($proc, i, t, S, w$) # Previous case is (V3).
(B0) $zv_{i-1}v_i$ is a left turn, $\alpha(s_{t-1}) < \alpha(s_t) = 2\pi$, $s_t \in (v_{i-1}v_i)$, and $w = v_i$.

  # See Figure 7 for possible exit cases.
(BS) Scan $v_i v_{i+1}, \cdots, v_{n-1}v_n$ for the first edge $v_k v_{k+1}$ to intersect $(s_t v_n]$
  # $zv_k v_{k+1}$ must be a right turn. $\delta(s_t v_i \cdots v_k u)$ may be 0 or -2π.
  u := intersection of $v_k v_{k+1}$ and $(s_t v_n]$
(B1) if $u = v_{k+1} = v_n$ then
    $(proc, i, t, S, w)$:= (FINISH, $n, t+1, s_0 \cdots s_t v_n, v_n$)
(B2) else # $u \in (v_k v_{k+1})$
    $(proc, i, t, S, w)$:= (RIGHT, $k+1, t, s_0 \cdots s_t, u$)
  endif
end # SCANB

procedure SCANC($proc, i, t, S, w$) # Previous case is (R3).
(C0) $zv_{i-1}v_i$ is a left turn, $zv_{i-2}v_{i-1}$ is a right turn, $v_{i-2}v_{i-1}v_i$ is a left turn, $w = v_{i-1}$,
s_t is not on $v_{i-1}v_i$, $zs_t w$ is a backward move, and $\alpha(s_{t-1}) < \alpha(s_t) < 2\pi$.

  # See Figure 8 for possible exit cases.
(CS) Scan $v_i v_{i+1}, \cdots, v_{n-1}v_n$ for the first edge $v_k v_{k+1}$ to intersect $(s_t w)$
  # $zv_k v_{k+1}$ must be a right turn. $\delta(wv_i \cdots v_k u)$ may be 0 or 2π.
(C1) u := intersection of $v_k v_{k+1}$ and $(s_t w)$ # $u \in (v_k v_{k+1})$
  $(proc, i, t, S, w)$:= (RIGHT, $k+1, t, s_0 \cdots s_t, u$)
end # SCANC

procedure SCAND($proc, i, t, S, w$) # Previous case can be (R4) or (A2).
(D0) $zv_{i-1}v_i$ is a right turn, s_t is not on $v_{i-1}v_i$, $\alpha(s_t) < 2\pi$, $w \in (v_{i-1}v_i)$,
zs_tw is a forward move, and if $t \geq 1$ then $\alpha(s_{t-1}) < \alpha(s_t)$.

See Figure 9 for possible exit cases.
(DS) Scan $v_iv_{i+1}, \cdots, v_{n-1}v_n$ for the first edge v_kv_{k+1} to intersect (s_tw)
zv_kv_{k+1} must be a left turn. $\delta(wv_i \cdots v_ku)$ may be 0 or 2π.
(D1) u := intersection of v_kv_{k+1} and (s_tw) # $u \in (v_kv_{k+1})$
$(proc, i, t, S, w) := (\text{LEFT}, k+1, t+2, s_0 \cdots s_tuv_{k+1}, v_{k+1})$
end # SCAND

Algorithm 2 only differs from Algorithm 1 in when the exits from the four scan statements (AS), (BS), (CS), (DS) occur. In Algorithm 2, the exits in these four statements occur at the first edge v_kv_{k+1} which intersects the line segment or half-line and also contains a point u such that $\delta(xv_i \cdots v_ku) = 0$, where $x = s_t$ for (AS), (BS) and $x = w$ for (CS), (DS). So it is possible that the exit does not occur at the first edge to intersect the line segment or half-line, since the first intersecting point $\bar{u}$ may satisfy $\delta(xv_i \cdots v_k\bar{u}) = \pm 2\pi$ as seen in the above comments for Algorithm 1. For polygons that wind a lot, such as Figures 1 and 2, exits will occur at different points in the two algorithms. With the change to (AS), only case (A3) can occur in SCANA for Algorithm 2. Algorithm 2 does not work correctly for blocked exterior viewpoints, since the exit condition in SCANA is not guaranteed to be satisfied. In Appendix A, Algorithm 1 is illustrated for the examples in Figure 1 (interior viewpoint) and Figure 2 (blocked exterior viewpoint), and Algorithm 2 is illustrated for the example in Figure 1.

Lee's algorithm for interior viewpoints has the same exit condition in SCANA as Algorithm 2 and the same exit conditions in SCANB, SCANC, SCAND as Algorithm 1 (Lee's algorithm does not take into account that $\delta(xv_i \cdots v_ku) = \pm 2\pi$ may occur in SCANB, SCANC, SCAND). This inconsistency in the exit conditions causes his algorithm to fail for polygons that wind

sufficiently; either the exit condition of a scan statement is not satisfied or the algorithm terminates with an incorrect visibility polygon. Lee's modified algorithm for blocked exterior viewpoints is based on a modification of an incorrect algorithm, so it is also incorrect for polygons that wind sufficiently. In Lee's modified algorithm, the exit condition in SCANA is the same as Algorithm 1 until the first occurrence of case (A1) or (A2), then the exit condition changes to that of his original algorithm or Algorithm 2, i.e. (A3) only; the exit conditions in SCANB, SCANC, SCAND are the same as Algorithm 1 (again, Lee's algorithm does not take into account that $\delta(xv_i \cdots v_k u) = \pm 2\pi$ may occur).

The El Gindy and Avis algorithm also has a step which is similar to the scan statement in SCANA. But this algorithm neglects the fact that the first intersection of $\overrightarrow{zs_t}$ may occur after traversing a clockwise circle of angular displacement. Therefore it also fails for polygons that wind sufficiently. We do not believe that this algorithm can be corrected since El Gindy and Avis try to maintain the property 'chain $S = s_0 \cdots s_t$ contains the boundary points visible with respect to $Ch[v_0, v_i]$', which is different from the less restrictive property (S6) of the next section.

4. Correctness proof

In this section, we establish that Algorithm 1 is correct for both interior and blocked exterior cases, i.e. $S = Bd(V(P,z))$ at its termination. (At the end of this section, we state how the correctness of Algorithm 2 can be similarly established.) It is established by an induction proof that properties (S1), (S2), (S3) of the previous section, and properties (S4), (S5), (S6) below hold at all entrances to the six procedures and at the end of the algorithm. This proof is quite different from Lee's proof, and is based on keeping track of the processed part of the polygon boundary with respect to the stack points and its winding with respect to the viewpoint.

To describe the additional properties satisfied by the varying chain of stack points and the processed part of the polygon boundary, we assume that properties (S1), (S2), and (S3) of the previous section and condition (L0), (R0), (A0), (B0), (C0), or (D0) are satisfied on entering LEFT, RIGHT, SCANA, SCANB, SCANC, or SCAND, respectively. These additional properties require the definitions of chain C_j and region R_j. We first make some observations.

Recall that on entering a procedure, $Ch[v_0,v_i]$ is the part of $Bd(P)$ processed so far, $S = s_0 s_1 \cdots s_t$ is the chain of stack points, and $w \in v_{i-1}v_i$. From (S1), (L0), ..., (D0), $Ch[v_0,v_i]$ is partitioned into the subchains $Ch[s_0,s_1]$, $Ch[s_1,s_2]$, $\cdots$, $Ch[s_{t-1},s_t]$, $Ch[s_t,w]$, $Ch[w,v_i]$. For notational convenience, we denote w by s_{t+1} although s_{t+1} is not on the stack. From (S3), for $j < t$, $s_j s_{j+1} \subseteq Bd(P)$ if $\alpha(s_j) < \alpha(s_{j+1})$, and $(s_j s_{j+1})$ is not on $Bd(P)$ if $\alpha(s_j) = \alpha(s_{j+1})$, i.e. $zs_j s_{j+1}$ is a forward or backward move. From (L0), ..., (D0), if s_t is not on $v_{i-1}v_i$ (which is possible on entering RIGHT, SCANC, or SCAND), then $zs_t s_{t+1}$ is a forward or backward move and $(s_t s_{t+1})$ is not on $Bd(P)$.

We now define C_j to be the closed chain (which forms the boundary of a 'hidden' region)

$$C_j = Ch[s_j,s_{j+1}] \,||\, s_{j+1}s_j \tag{4.1}$$

if $s_j s_{j+1}$ is not on $Bd(P)$, $j \le t$. Otherwise C_j is not defined . We define R_j to be the simply-connected region

$$R_j = \{ u \mid u \in zv \text{ and } v \in s_0 s_1 \cdots s_j \} \tag{4.2}$$

for $j = 0, 1, ..., t$. (If $zs_{j-1}s_j$ is a forward move then R_j is simply-connnected but not simple because it contains the 'extra' edge $s_{j-1}s_j$ on its boundary.) We also define $Ext(R_j)$ to be the set of points which are not in R_j. From (S2),

$$R_0 \subseteq R_1 \subseteq \cdots \subseteq R_t \text{ and } Ext(R_t) \subseteq \cdots \subseteq Ext(R_1) \subseteq Ext(R_0). \tag{4.3}$$

Examples of C_j and R_j are illustrated in Figure 10.

We now state three additional properties which are satisfied on entrance to each of the six procedures.

(S4) For $0 \le j \le t$, $Ch(s_j,w) \subseteq Ext(R_j)$ unless ($j < t$ and zs_js_{j+1} is a backward move) or ($j = t$ and $w \neq s_t$ and zs_tw is a backward move) in which case $Ch(s_j,w) \subseteq Ext(R_j) \cup (zs_j)$. (Note that $Ch(v_0,s_j)$ may intersect R_j.)

(S5) For all $j \le t$ such that C_j is defined, C_j is simple (i.e. $Ch(s_j,s_{j+1})$ does not intersect s_js_{j+1}), $Ch(s_{j+1},w) \subseteq Right(C_j)$, $z \in Right(C_j)$, and $Left(C_j) \subseteq Ext(R_j)$. (It can also be shown that either $Int(C_j) = Left(C_j)$ and $\delta(C_j) = 0$ or $Int(C_j) = Right(C_j)$ and $\delta(C_j) = -2\pi$.)

(S6) If $v \in Ch[v_0,v_i)$ but v is not on $S = s_0s_1 \cdots s_t$, then v is not visible from z.

Before proving these properties by induction in Lemma 5, we need Lemmas 3 and 4.

Lemma 3 : Suppose

(a) s_j occurs before s_{j+1}, (s_js_{j+1}) is not on $Bd(P)$, zs_js_{j+1} is a forward or backward move,

(b) $C_j = Ch[s_j,s_{j+1}] \,||\, s_{j+1}s_j$ is simple, and

(c) there exists a nonempty segment $(s_{j+1}x]$ of $Bd(P)$ such that x occurs after s_{j+1} and $(s_{j+1}x]$ is entirely in $Left(C_j)$ [or alternatively $Right(C_j)$].

Then $Ch(x,v_n)$ remains in the same region (either $Left(C_j)$ or $Right(C_j)$) as $(s_{j+1}x]$ until it intersects (s_js_{j+1}). Furthermore if such an intersection occurs for the first time at point u on edge v_kv_{k+1} so that $Ch(x,u)$ is in the same region as $(s_{j+1}x]$, then either zv_kv_{k+1} is a left turn if

(i) zs_js_{j+1} is a forward move and $(s_{j+1}x] \subseteq Left(C_j)$, or

(ii) zs_js_{j+1} is a backward move and $(s_{j+1}x] \subseteq Right(C_j)$,

or zv_kv_{k+1} is a right turn if

(iii) $zs_j s_{j+1}$ is a forward move and $(s_{j+1}x] \subseteq Right(C_j)$, or

(iv) $zs_j s_{j+1}$ is a backward move and $(s_{j+1}x] \subseteq Left(C_j)$.

Proof : $Bd(P)$ is simple, $Ch[s_j, s_{j+1}] \subseteq Bd(P)$, and $(s_j s_{j+1})$ is not on $Bd(P)$ imply that $Ch[x, v_n)$ may intersect C_j only on $(s_j s_{j+1})$. Hence $Ch(x, v_n)$ remains in the same region as $(s_{j+1}x]$ until it intersects $(s_j s_{j+1})$. Suppose an intersection occurs for the first time at point u on edge $v_k v_{k+1}$ so that $Ch(x, u)$ is in the same region as $(s_{j+1}x]$. Then $zv_k v_{k+1}$ is either a left or right turn, with the direction as indicated in the cases (i) to (iv), since (see Figure 11) $zs_j s_{j+1}$ is either a forward or backward move, v_k is either to the left or right of $\overrightarrow{zs_j}$ and is in the same region as $(s_{j+1}x]$, and $v_k v_{k+1}$ intersects $(s_j s_{j+1})$. □

Lemma 4 : Suppose

(a) properties (S1), (S2), and (S3) are satisfied by $S = s_0 s_1 \cdots s_t$, $t \geq 1$,

(b) $zs_1 s_t$ is a forward move if $\alpha(s_t) = 2\pi$ and $\alpha(s_0) = \alpha(s_1)$,

(c) there exists a nonempty segment $(s_t x]$ of $Bd(P)$ such that x occurs after s_t and $(s_t x] \subseteq Ext(R_t)$, and

(d) for all $j < t$ such that C_j is defined, C_j is simple and $Ch(s_{j+1}, x] \subseteq Right(C_j)$ (this condition is similar to property (S5)).

If $Ch(x, v_n)$ intersects the boundary of R_t for the first time at point u on edge $v_k v_{k+1}$ so that $Ch(x, u) \subseteq Ext(R_t)$, then $zv_k v_{k+1}$ is a right turn and $u \in (ys_t)$, where $y = z$ if $\alpha(s_t) < 2\pi$, $y = s_0$ if $\alpha(s_t) = 2\pi$ and $\alpha(s_0) < \alpha(s_1)$, or $y = s_1$ if $\alpha(s_t) = 2\pi$ and $\alpha(s_0) = \alpha(s_1)$.

Proof : (See Figure 12.) Let $q = t-1$ if $zs_{t-1}s_t$ is a forward move, and $q = t$ otherwise. $Ch(x, v_n)$ does not intersect zs_0 since $s_0 = v_0$ is visible from z. Suppose $u \in (v_k v_{k+1})$ is the first point of intersection with S or (ys_t) (i.e. the boundary of R_t) in a traversal of $Bd(P)$ from x to v_n,

so that $Ch(x,u) \subseteq Ext(R_t)$. If $u \in (ys_q)$, then $zv_k v_{k+1}$ is a right turn since v_k must be to the left of $\overrightarrow{zs_t}$.

Suppose $u \in S$. Then u must be on $s_j s_{j+1}$ for an index j such that $\alpha(s_j) = \alpha(s_{j+1})$. From part (d), $Ch(s_{j+1},x] \subseteq Right(C_j)$. From Lemma 3, $zv_k v_{k+1}$ is a left turn if $zs_j s_{j+1}$ is a backward move, and $zv_k v_{k+1}$ is a right turn if $zs_j s_{j+1}$ is a forward move. But, if $zs_j s_{j+1}$ is a backward move, it is not possible for both $zv_k v_{k+1}$ to be a left turn and $v_k \in Ext(R_t)$ (if $v_k \in Ext(R_t)$ and $(v_k v_{k+1})$ intersects $(s_j s_{j+1})$ but does not intersect any other part of S then $zv_k v_{k+1}$ must be right turn). Similarly, if $zs_j s_{j+1}$ is a forward move and $j < t-1$, it is not possible for both $zv_k v_{k+1}$ to be a right turn and $v_k \in Ext(R_t)$. Therefore if $u \in S$, then $u \in (s_{t-1}s_t)$, $zs_{t-1}s_t$ is a forward move, and $zv_k v_{k+1}$ is a right turn. □

Let P_1, P_2, ... be the sequence of procedures called by VISPOL, where P_m may be LEFT, RIGHT, SCANA, SCANB, SCANC, or SCAND. P_m = FINISH is also allowed as the last procedure in the sequence to indicate the end of the algorithm.

Lemma 5 : For all m, on entering P_m,

(a) condition (L0), (R0), (A0), (B0), (C0), or (D0) is satisfied if P_m is LEFT, RIGHT, SCANA, SCANB, SCANC, or SCAND, respectively,

(b) properties (S1), (S2), and (S3) are satisfied,

(c) properties (S4), (S5), and (S6) are satisfied, and

(d) $(wv_i] \subseteq Left(C_t)$ if P_m is SCANC or SCAND.

Proof : We will use induction to show that (a), (b), (c), and (d) are true on entering P_m for all m. By assumption (V0) and the orientation of the vertices of P, there are only three possible configurations of the edges v_0v_1 and $v_{n-1}v_n$ (see Figure 3): (1) both zv_0v_1 and $zv_{n-1}v_n$ are left turns, (2) zv_0v_1 is a left turn, $zv_{n-1}v_n$ is a right turn, and $v_{n-1}v_0v_1$ is a right turn, or (3) zv_0v_1 is a

right turn, $zv_{n-1}v_n$ is a left turn, and $v_{n-1}v_0v_1$ is a right turn. Initially, P_1 is LEFT or SCANA if case (V1) or (V2) occur, respectively, and (L0) or (A0) and (S1) to (S6) are clearly satisfied on entering P_1 in both cases (in case (V2), (v_0v_1) is not visible from z by Lemma 2).

Suppose (a), (b), (c), and (d) are true on entering P_m. Then it is straightforward but tedious to show that (a) and (b) are true on entering P_{m+1} (which is allowed to be FINISH), so we shall leave the details of these two parts to the reader and concentrate on parts (c) and (d) which are more difficult. We will show that properties (S4), (S5), (S6), and (d) are satisfied on entering P_{m+1} for all the possible cases. In these cases, i, t, S, and w refer to the values of these variables on entering P_m. The reader should refer to the pseudocode and figures to see how these variables and (S4), (S5), (S6) are updated on entering P_{m+1}. Since (S4), (S5), and (S6) are satisfied on entering P_m, we mainly have to examine the location of $Ch(w,\hat{w})$, $Ch(v_i,\hat{v_i})$, and any newly defined C_j to show that (S4), (S5), and (S6) are satisfied on entering P_{m+1}, where $\hat{w}$ and $\hat{v_i}$ are the new values of w and v_i on entering P_{m+1}. Also, note that the inductive hypothesis implies that the conditions of Lemmas 3 and 4 are satisfied when we use these lemmas below.

(i) P_m = LEFT : $i=n$ (case (L1)) may occur for configurations (a) and (c) of Figure 3. If $i \neq n$, then zv_iv_{i+1} may be a left turn (case (L2)) or a right turn (cases (L3) and (L4)) by assumption (V0). In the latter case, $s_{t-1}v_iv_{i+1}$ may be a right turn (case (L3)) or a left turn (case (L4)).

Case (L1) : Since i, t, S, and w are not changed, (S4), (S5), and (S6) are satisfied on entering P_{m+1} = FINISH.

Case (L2) : First we consider the subcase where $\delta(s_0 \cdots s_t v_{i+1}) \leq 2\pi$ (by assumption (V0), $\delta(s_0 \cdots s_t v_{i+1}) = 2\pi$ can only occur if $v_{i+1} = v_n$). (S6) is clearly satisfied on entering P_{m+1} = LEFT. By Lemma 4, (v_iv_{i+1}) is in $Ext(R_t)$, so (S4) is satisfied on entering P_{m+1}. (v_iv_{i+1}) does not intersect S, so by Lemma 3, (S5) is satisfied on entering P_{m+1}.

Now we consider the subcase where $\delta(s_0 \cdots s_t v_{i+1}) > 2\pi$. The intersection of $s_t v_{i+1}$ and $z\vec{v_n}$ is added to the stack (replacing v_{i+1}) in (V3), so let this point be called s_{t+1}. By Lemma 4, $(v_i v_{i+1})$ is in $Ext(R_t)$, so $zs_1 s_{t+1}$ is a forward move if $\alpha(s_0) = \alpha(s_1)$, and $(s_{t+1} v_{i+1})$ is in $Ext(R_{t+1})$. This implies that (S4) is satisfied on entering P_{m+1} = SCANB. By Lemma 3, (S5) is satisfied on entering P_{m+1}. For $v \in (s_{t+1} v_{i+1})$, (zv) intersects a point of S which is also on $Bd(P)$, so by Lemma 1, (S6) is satisfied on entering P_{m+1}.

Case (L3) : By Lemma 4, $(v_i v_{i+1})$ is in $Ext(R_t)$, so (S4) is satisfied on entering P_{m+1} = SCANA. $(v_i v_{i+1})$ does not intersect S, so by Lemma 3, (S5) is satisfied on entering P_{m+1}. $(v_i v_{i+1})$ is not visible from z by Lemma 2, so (S6) is satisfied on entering P_{m+1}.

Case (L4) : S and w are not changed so (S4) and (S5) are satisfied on entering P_{m+1} = RIGHT. $(v_i v_{i+1})$ is not visible from z by Lemma 2, so (S6) is satisfied on entering P_{m+1}.

(ii) P_m = RIGHT : From (R0), a nonempty initial subsegment of $(wv_i]$ is in R_t. It is not possible that $v_i = v_n$, since $v_{i-1} v_i$ does not satisfy any of the three configurations of Figure 3. If $(wv_i]$ is entirely in R_t, then by assumption (V0), (RA) is satisfied for exactly one index j in the range $1 \le j \le t$ and (RB) is not satisfied for any j. Otherwise, since it is not possible for (wv_i) to intersect zv_0, (wv_i) intersects $(s_{j-1} s_j)$ for some j in which $zs_{j-1} s_j$ is a forward move, and in a backward scan of the edges $s_j s_{j-1}$, (RB) occurs first (note that if j is the largest index such that (wv_i) intersects $s_{j-1} s_j$ then $zs_{j-1} s_j$ cannot be a backward move since $zv_{i-1} v_i$ is a right turn).

Case (RA) : Let $\hat{s_j}$ be the intersection of $(s_{j-1} s_j)$ and $z\vec{v_i}$. A point v on subchain $\hat{s_j} s_j s_{j+1} \cdots s_t$ is not visible from z by Lemma 1, since (zv) intersects wv_i (if $s_t = v_{i-1}$ then s_t is not visible from z by Lemma 2). Hence S is shortened to $s_0 s_1 \cdots s_{j-1} \hat{s_j}$, and R_j is shrunk to $\hat{R_j}$ (which is defined as in (4.2) with $\hat{s_j}$ replacing s_j). From (S4), $Ch(s_j, w) \subseteq Ext(R_j) \cup (zs_j)$. This implies that $Ch(\hat{s_j}, w) \subseteq Ext(\hat{R_j})$ since $\hat{s_j} s_j \subseteq Bd(P)$. Also, $[wv_i) \subseteq Ext(\hat{R_j})$ and $v_i \in z\hat{s_j}$. So $Ch(\hat{s_j}, v_i)$ does not intersect $\hat{s_j} v_i$ and $C_j = Ch[\hat{s_j}, v_i] \,||\, v_i \hat{s_j}$ is simple. $Ch(\hat{s_j}, v_i) \subseteq Ext(\hat{R_j})$,

$zv_{i-1}v_i$ is a right turn, $zv_i\hat{s}_j$ is a forward move, and $z\hat{s}_j s_j$ is a left turn imply that $z \in Right(C_j)$ and $Left(C_j) \subseteq Ext(\hat{R}_j)$. $\hat{s}_j$ and $\hat{R}_j$ are renamed s_j and R_j below.

As mentioned above, it is not possible that $v_i = v_n$. By assumption (V0), $zv_i v_{i+1}$ may be a right turn (case (R1)) or a left turn (cases (R2) and (R3)). In the latter case, $v_{i-1}v_i v_{i+1}$ may be a right turn (case (R2)) or a left turn (case (R3)).

Case (R1) : From the above statements for case (RA), (S4) and (S5) are satisfied on entering P_{m+1} = RIGHT. $zv_{i-1}v_i$ and $zv_i v_{i+1}$ are both right turns imply that v_i is not visible from z by Lemma 2. The latter right turn implies that $(v_i v_{i+1})$ is not visible from z by Lemma 2. Hence, (S6) is satisfied on entering P_{m+1}.

Case (R2) : By Lemma 4, $(v_i v_{i+1}) \subseteq Ext(R_j)$. $v_{i-1}v_i v_{i+1}$ is a right turn and Lemma 3 imply that $(v_i v_{i+1}] \subseteq Right(C_j)$. From the above statements for case (RA) and for reasons similar to case (L2), (S4), (S5), and (S6) are satisfied on entering P_{m+1} = LEFT or SCANB.

Case (R3) : $v_{i-1}v_i v_{i+1}$ is a left turn and Lemma 3 imply that $(v_i v_{i+1}] \subseteq Left(C_j)$. $z \in Right(C_j)$ implies that all points in $Left(C_j)$ are not visible from z by Lemma 1; in particular $(v_i v_{i+1}]$ is not visible from z. Hence, from the above statements for case (RA), (S4), (S5), (S6), and (d) are satisfied on entering P_{m+1} = SCANC.

Case (RB)/(R4) : Subchain $s_j s_{j+1} \cdots s_t$ is not visible from z for the same reason as case (RA). $(s_{j-1}s_j)$, which is not on $Bd(P)$, is not visible from z by Lemma 1 since $s_{j-1} \in Bd(P)$. So S is shortened to $s_0 s_1 \cdots s_{j-1}$. From (S4) and (S5), $Ch(s_{j-1},w) \subseteq Ext(R_{j-1})$, $Ch(s_j,w) \subseteq Right(C_{j-1})$, $z \in Right(C_{j-1})$, and $Left(C_{j-1}) \subseteq Ext(R_{j-1})$. Let u be the intersection of (wv_i) and $(s_{j-1}s_j)$. $wu \subseteq Ext(R_{j-1})$ and $[wu)$ does not intersect $s_{j-1}s_j$ imply that $Ch(s_{j-1},u) \subseteq Ext(R_{j-1})$ and $Ch(s_j,u) \subseteq Right(C_{j-1})$. So $Ch(s_j,u)$ does not intersect $s_{j-1}s_j$ and $\hat{C}_j = Ch[s_j,u] \,||\, us_j$ is simple. Also, $Ch(s_{j-1},s_j]$ does not intersect $s_{j-1}u$, so $\hat{C}_{j-1} = Ch[s_{j-1},u] \,||\, us_{j-1}$ is simple. $z \in Right(\hat{C}_j)$ and $Left(\hat{C}_j) \subseteq Ext(R_{j-1})$ for reasons simi-

lar to case (RA). $Ch(s_j,u) \subseteq Right(C_{j-1})$ implies that $Left(\hat{C}_j) \subseteq Right(C_{j-1})$ which implies that $Left(\hat{C}_{j-1}) = Left(C_{j-1}) \cup Left(\hat{C}_j) \cup (s_j u)$ and $Right(\hat{C}_{j-1}) = Right(C_{j-1}) \cap Right(\hat{C}_j)$. Finally, this implies that $z \in Right(\hat{C}_{j-1})$ and $Left(\hat{C}_{j-1}) \subseteq Ext(R_{j-1})$. Since $(uv_i]$ does not intersect $s_{j-1}s_j$ and an initial nonempty subsegment of $(uv_i]$ is in $Left(C_{j-1})$, $(uv_i] \subseteq Left(C_{j-1}) \subseteq Left(\hat{C}_{j-1})$. From the above statements and with C_{j-1} set to $\hat{C}_{j-1}$, (S4), (S5), (S6), and (d) are satisfied on entering P_{m+1} = SCAND.

(iii) P_m = SCANA : There are two possible subcases, $i = 1$ or $i > 1$. If $i = 1$ then, in order to satisfy configuration (c) of Figure 3, $Ch[v_i,v_n)$ must intersect $\overrightarrow{zs_t} = \overrightarrow{zv_0}$ at a point u which is farther from z than v_0, so case (A2) or (A3) is satisfied by u.

Now suppose $i > 1$. From (A0), $0 < \alpha(s_t) < 2\pi$. From (S4), $(s_t v_i) \subseteq Ext(R_t)$. If $Ch[v_i,v_n)$ intersects (zs_t) for the first time on edge $v_k v_{k+1}$, then from Lemma 4, $zv_k v_{k+1}$ is a right turn so it is not possible for case (A4) to occur before case (A1). We will show by contradiction that $Ch[v_i,v_n)$ intersects $\overrightarrow{zs_t}$. Suppose $Ch[v_i,v_n)$ does not intersect $\overrightarrow{zs_t}$. Then there exists an interval of polar angles $[\theta(s_t),\gamma]$, where $\theta(s_t) < \gamma < 2\pi$, which contains no points from $Ch[v_i,v_n)$. From (S6), $Ch[v_0,v_i)$ does not contain any points with polar angle in the interval $(\theta(s_t),\gamma]$ which are visible from z. Therefore $Bd(P)$ does not contain any points with polar angle in $(\theta(s_t),\gamma]$ which are visible from z. This contradicts the fact that if $z \in Int(P)$ or z is blocked exterior to P then there exists exactly one point on $Bd(P)$ at each polar angle which is visible from z. Therefore $v_i \neq v_n$, $Ch[v_i,v_n)$ intersects $\overrightarrow{zs_t}$, and either case (A1) or (A2) or (A3) will occur for the first such intersection.

For $i = 1$ or $i > 1$, suppose the first intersection of $Ch[v_i,v_n)$ and $\overrightarrow{zs_t}$ occurs at point u on $(v_k v_{k+1})$. Then $C_t = Ch[s_t,u] \,||\, us_t$ is simple. If $i > 1$, then $Ch(s_t,u) \subseteq Ext(R_t)$ by Lemma 4. If $i = 1$, then $Ch(s_t,u) \subseteq Ext(R_t)$ since $R_t = zv_0$. $Ch[v_i,u)$ does not intersect S, so by Lemma 3, $Ch[v_i,u) \subseteq Right(C_j)$ for all $j < t$ such that C_j is defined. Since $zs_t v_i$ is a right turn and $Ch(s_t,u)$ does no intersect zv_0, $\delta(s_t v_i \cdots v_k u)$ must be -2π in cases (A1) and (A2), and 0 in case

(A3).

Case (A1) : $Ch(s_t,u) \subseteq Ext(R_t)$, $zs_t v_i$ is a right turn, $zv_k u$ is a right turn, and zus_t is a forward move imply that $z \in Right(C_t)$, $Left(C_t) \subseteq Ext(R_t)$, and $Right(C_t) = Int(C_t)$. Let v be a point on $Ch(s_t,u)$. If v is the closest point to z on $Ch(s_t,u)$ with polar angle $\theta(v)$ then v is CW-oriented since $Int(C_t) = Right(C_t)$ and $z \in Int(C_t)$, so v is not visible from z by Lemma 2. Otherwise (zv) intersects $Ch(s_t,u)$ so v is not visible from z by Lemma 1. Also, $[uv_{k+1})$ is not visible from z by Lemma 2. From the above statements, (S4), (S5), and (S6) are satisfied on entering P_{m+1} = RIGHT.

Case (A2) : For reasons similar to case (A1), $z \in Right(C_t)$, $Left(C_t) \subseteq Ext(R_t)$, and $Ch(s_t,v_{k+1})$ is not visible from z. $(uv_{k+1}]$ is to the left of $\overrightarrow{us_t}$ and Lemma 3 imply that $(uv_{k+1}] \subseteq Left(C_t)$. From the above statements, (S4), (S5), (S6), and (d) are satisfied on entering P_{m+1} = SCAND.

Case (A3) : $Ch(s_t,u) \subseteq Ext(R_t)$, $zs_t v_i$ is a right turn, $zv_k u$ is a left turn, and zus_t is a backward move imply that $z \in Right(C_t)$, $Left(C_t) \subseteq Ext(R_t)$, and $Left(C_t) = Int(C_t)$. As for case (A1), a point v on $Ch(s_t,u)$ is not visible from z because either v is the closest point to z on $Ch(s_t,u)$ with polar angle $\theta(v)$ and it is CW-oriented since $Int(C_t) = Left(C_t)$ and $z \in Ext(C_t)$, or (zv) intersects $Ch(s_t,u)$. $(uv_{k+1}]$ is to the right of $\overrightarrow{us_t}$ and Lemma 3 imply that $(uv_{k+1}] \subseteq Right(C_t)$. From the above statements and for reasons similar to case (L2), (S4), (S5), and (S6) are satisfied on entering P_{m+1} = LEFT or SCANB.

(iv) P_m = SCANB : $v_i \neq v_n$ since $\theta(v_i) \neq 0$. The scan for edge $v_k v_{k+1}$ must be successful since $i < n$ and $v_{n-1}v_n$ intersects $(s_t v_n]$ at v_n. There are two possible cases for the first intersection of $(s_t v_n]$: (B1) at v_n $(k = n-1)$, or (B2) on $(s_t v_n)$ and $v_k v_{k+1}$, $k < n-1$. Let u be this first intersection point. In case (B2), $zv_k v_{k+1}$ is a right turn, $Ch(s_t,u) \subseteq Ext(R_t)$, and $u \in (ys_t)$ by Lemma 4, where $y = s_0$ if $\alpha(s_0) < \alpha(s_1)$ and $y = s_1$ otherwise. In case (B1), $Ch(s_t,v_n) \subseteq Ext(R_t)$

by Lemma 4; in particular, $[v_{n-1}v_n) \subseteq Ext(R_t)$, so $zv_{n-1}v_n$ is a right turn (configuration (b) of Figure 3).

u is the first point from $Ch(s_t,v_n]$ to intersect $(s_tv_n]$ implies that $C_t = Ch[s_t,u] \,||\, us_t$ is simple. $Ch(s_t,u) \subseteq Ext(R_t)$, zs_tv_i is a left turn, zv_ku is a right turn, and zus_t is a forward move imply that $z \in Right(C_t)$, $Left(C_t) \subseteq Ext(R_t)$, and $\delta(s_tv_i \cdots v_ku)$ may be 0 or -2π only. $Ch[v_i,u)$ does not intersect S, so by Lemma 3, $Ch[v_i,u) \subseteq Right(C_j)$ for all $j < t$ such that C_j is defined. $Ch(s_t,u)$ is not visible from z by Lemma 1 since it is entirely in $Ext(R_t)$ and $\alpha(s_t) = 2\pi$. In case (B2), $[uv_{k+1})$ is not visible from z by Lemma 2. From the above statements, (S4), (S5), and (S6) are satisfied on entering P_{m+1} = FINISH or RIGHT.

(v) P_m = SCANC : From (S5), $z \in Right(C_t)$ so all points in $Left(C_t)$ are not visible from z by Lemma 1. From (d), $(wv_i] \subseteq Left(C_t)$. $v_i \neq v_n$ since v_i is not visible from z. v_n is visible from z and $0 < \alpha(s_t) < 2\pi$, so $Ch(v_i,v_n)$ must enter $Right(C_t)$. By Lemma 3 with $s_{t+1} = w$, $Ch(v_i,v_n)$ intersects (s_tw) for the first time at a point u on an edge v_kv_{k+1} such that zv_kv_{k+1} is a right turn (since zs_tw is a backward move). Also, $Ch[v_i,u) \subseteq Left(C_j)$ so $Ch[v_i,u)$ is not visible from z as mentioned above. $[uv_{k+1})$ is not visible from z by Lemma 2.

From (S4) and (S5), $Ch(s_t,w] \subseteq Ext(R_t) \cup (zw]$ and $Left(C_t) \subseteq Ext(R_t)$, so $Ch(w,u) \subseteq Ext(R_t)$, $Ch(s_t,u) \subseteq Ext(R_t) \cup (zw]$, and u is the only intersection of (s_tw) from $Ch[s_t,u]$, i.e. $\hat{C}_t = Ch[s_t,u] \,||\, us_t$ is simple. $Ch(w,u) \subseteq Ext(R_t)$, zwv_i is a left turn, zv_ku is a right turn, and zuw is a backward move imply $\delta(wv_i \cdots v_ku)$ may be 0 or 2π only. $Ch(s_t,w) \subseteq Ch(s_t,u)$ and $Ch(w,u) \subseteq Left(C_t)$ imply that $Left(\hat{C}_t) \subseteq Left(C_t)$ and $Right(C_t) \subseteq Right(\hat{C}_t)$ which imply that $z \in Right(\hat{C}_t)$ and $Left(\hat{C}_t) \subseteq Ext(R_t)$. $Ch[w,u)$ does not intersect S, so by Lemma 3, $Ch[w,u) \subseteq Right(C_j)$ for all $j < t$ such that C_j is defined. From the above statements and with C_t set to $\hat{C}_t$, (S4), (S5), and (S6) are satisfied on entering P_{m+1} = RIGHT.

(vi) P_m = SCAND : From (S5), $z \in Right(C_t)$ so all points in $Left(C_t)$ are not visible from z by Lemma 1. From (d), $(wv_i] \subseteq Left(C_t)$. $v_i \neq v_n$ since v_i is not visible from z. If $t > 0$ then $0 < \alpha(s_t) < 2\pi$, so $Ch(v_i, v_n)$ must enter $Right(C_t)$ since v_n is visible from z. If $t = 0$ (which is possible only if zv_0v_1 is a right turn) then, in order to satisfy configuration (c) of Figure 3, $Ch(v_i, v_n)$ must enter $Right(C_t)$. By Lemma 3 with $s_{t+1} = w$, $Ch(v_i, v_n)$ intersects $(s_t w)$ for the first time at a point u on an edge $v_k v_{k+1}$ such that $zv_k v_{k+1}$ is a left turn (since $zs_t w$ is a forward move). Also, $Ch[v_i, u) \subseteq Left(C_j)$ so $Ch[v_i, u)$ is not visible from z as mentioned above.

From (S5), $Left(C_t) \subseteq Ext(R_t)$, so $Ch[w, u) \subseteq Ext(R_t)$ and by Lemma 3, $Ch[w, u) \subseteq Right(C_j)$ for all $j < t$ such that C_j is defined. $Ch[w, u) \subseteq Ext(R_t)$, zwv_i is a right turn, $zv_k u$ is a left turn, and zuw is a forward move imply $\delta(wv_i \cdots v_k u)$ may be 0 or 2π only. u is the only intersection of $(s_t w)$ from $Ch[s_t, u]$, so $\hat{C}_t = Ch[s_t, u] \,||\, us_t$ is simple. $Ch(s_t, w) \subseteq Ch(s_t, u)$ and $Ch(w, u) \subseteq Left(C_t)$ imply that $Left(\hat{C}_t) \subseteq Left(C_t)$ and $Right(C_t) \subseteq Right(\hat{C}_t)$ which imply that $z \in Right(\hat{C}_t)$ and $Left(\hat{C}_t) \subseteq Ext(R_t)$. $(uv_{k+1}]$ is to the right of $\overrightarrow{us_t}$ and Lemma 3 imply that $(uv_{k+1}] \subseteq Right(\hat{C}_t)$. From the above statements and with C_t set to $\hat{C}_t$ and for reasons similar to case (L2), (S4), (S5), and (S6) are satisfied on entering P_{m+1} = LEFT or SCANB. □

Lemma 6 : Algorithm 1 terminates with $s_t = v_n$ and $\alpha(s_t) = 2\pi$.

Proof : In each procedure P_m called by VISPOL, i is increased by at least one, except in cases (L1) and (R4) for P_m = LEFT and RIGHT, respectively. In the former case, P_{m+1} = FINISH; in the latter case, P_{m+1} = SCAND. So i must eventually get to n and there are a finite number of procedure calls. By Lemma 5, VISPOL terminates as a result of case (L1) or (B1). In both cases, upon exiting the 'repeat' loop, $s_t = v_n$. In the former case, $zs_{t-1}s_t$ is a left turn and property (S2) imply $\alpha(s_t) = 2\pi$. In the latter case $\alpha(s_{t-1}) = 2\pi$ and $zs_{t-1}s_t$ is a backward move imply $\alpha(s_t) = 2\pi$. □

Theorem 1 : Algorithm 1 correctly computes $V(P,z)$ in $O(n)$ time where n is the number of vertices of P.

Proof : Let $S = s_0 s_1 \cdots s_t$ be the chain of stack points at the end of the algorithm. By Lemmas 5 and 6, properties (S1), (S2), (S3), and (S6) are satisfied with $i = n$, $s_0 = v_0$, $s_t = v_n$, and $\delta(S) = 2\pi$. If $v \in Ch[v_0, v_n] = Bd(P)$ but v is not on S, then v is not visible from z by (S6). Therefore if $v \in Bd(P)$ is visible from z, then $v \in S$. From (S2), S is a simple closed curve and $Int(S)$ is a star-shaped region containing z. The points of S can be partitioned into two disjoint sets V and W where $V = \{v \mid v$ is the point of S closest to z with polar angle ψ, $0 \le \psi < 2\pi\}$ and $W = S - V$. From (S1) and (S3), $V \subseteq Bd(P)$. If $v \in W \cap Bd(P)$ then either $v = s_{j+1}$ where $z s_j s_{j+1}$ is a forward move or $v = s_j$ where $z s_j s_{j+1}$ is a backward move, and v is not visible from z by Lemma 1 since zv intersects s_j or s_{j+1}, respectively. A point $v \in V$ is visible from z since there exists a point on $Bd(P)$ with polar angle $\theta(v)$ which is visible from z and all other points on $Bd(P)$ with polar angle $\theta(v)$ are not visible from z as mentioned above. Therefore V contains exactly the points of $Bd(P)$ visible from z and W contains the points which connects up the points of V to form a simple closed curve, i.e. the algorithm correctly computes $S = Bd(V(P,z))$.

The running time of the algorithm is $O(n)$ since the edges of $Bd(P)$ are sequentially scanned once in the algorithm and for each edge $v_i v_{i+1}$ processed, at most two points are added to the stack, which implies that at most $2n$ points are deleted from the stack in RIGHT. □

The correctness of Algorithm 2 for interior viewpoints can be established in a way similar to that for Algorithm 1. In Algorithm 1, properties (S4) and (S5) are needed to show that property (S6) is satisfied. In Algorithm 2, properties (S4) and (S5) are not always satisfied because the exits from the four scan statements (AS), (BS), (CS), and (DS) do not have to occur at the first point to intersect the line segment or half-line, so the closed chains C_j defined in (4.1) are not guaranteed to be simple. Recall that an additional exit condition is that $\delta(x v_i \cdots v_k u) = 0$ where $x = s_t$ or w. The following observations can be used to show that property (S6) is satisfied for

Algorithm 2. More details are given in Joe and Simpson (1985).

(1) $\delta(Ch[v_0,v_n])=2\pi$ by (1.2) since $z \in Int(P)$ and the boundary is oriented in the counterclockwise direction. Let $\bar{\alpha}(v)$ denote the angular displacement of a point v on $Bd(P)=Ch[v_0,v_n]$. Then $\alpha(v)=\bar{\alpha}(v)$ for $v \in Bd(P) \cap S$ where S is the varying chain of stack points. In particular, the previous statement is true when $S=Bd(V(P,z))$. This implies that if $v \in Bd(P)$ and either $\bar{\alpha}(v)<0$ or $\bar{\alpha}(v)>2\pi$ then v is not visible from z.

(2) The exit condition $\delta(xv_i \cdots v_k u)=0$ implies that $\delta(Ch[s_j,s_{j+1}])=0$ whenever zs_js_{j+1} is a forward or backward move. This means that it is possible to redefine C_j for these indices j as follows to make it simple:

$$C_j = Ch[s_j,s_{j+1}] \,||\, Path[s_{j+1},s_j] \tag{4.4}$$

where $Path[s_{j+1},s_j]=s_{j+1}s_j$ if $Ch[s_j,s_{j+1}]$ does not intersect (s_js_{j+1}), otherwise $Path[s_{j+1},s_j]$ is the path from s_{j+1} to s_j with 'detours' to the left of $\overrightarrow{s_{j+1}s_j}$ to avoid intersecting $Ch[s_j,s_{j+1}]$ (see example in Figure 13). C_j satisfies $Left(C_j)=Int(C_j)$ and $z \in Right(C_j)$.

(3) Suppose there exists a nonempty segment $(s_t x]$ of $Bd(P)$ such that x occurs after s_t and $(s_t x] \subseteq Ext(R_t)$. Let I be the interval $[0,\theta(s_t)]$ or $(0,2\pi]$ if $\alpha(s_t)<2\pi$ or $\alpha(s_t)=2\pi$, respectively. If point u on edge v_kv_{k+1} is the first point on $Ch(x,v_n)$ to intersect the boundary of R_t and also satisfy $\bar{\alpha}(u) \in I$, then zv_kv_{k+1} is a right turn and $u \in (ys_t)$ where y is defined in Lemma 4.

5. Extension

In this section, we indicate how Algorithms 1 and 2 can be used for boundary and free exterior viewpoints. For a boundary viewpoint z, we orient the vertices of P in counterclockwise order and label them $z, v_0, v_1, \cdots, v_{n-1}, v_n$, and z, where v_0 is the successor vertex of z and v_n is the predecessor vertex of z. We also assume that the coordinate system is translated and rotated so that z is at the origin and v_0 is on the positive x-axis. This implies that $\theta(v_n)<2\pi$ is the interior angle at z. No modifications are required in the pseudocode, but case (V3) and

procedure SCANB imply that $\alpha(s_t) \leq \theta(v_n)$ in property (S2). The visibility polygon $V(P,z)$ is $zs_0s_1 \cdots s_t z$ where $s_0s_1 \cdots s_t$ is the chain of stack points at the end of the algorithm. The reason why the algorithms are correct is that there are no boundary points with polar angle in the interval $(\theta(v_n),2\pi)$ which are visible from z.

If $z \in Ext(P)$, but it is not known whether z is blocked or free exterior, then the angular displacement of the boundary vertices can be used to classify z as in Freeman and Loutrel (1967). Suppose the vertices $v_0, v_1, \cdots, v_{n-1}$, and $v_n = v_0$ of P are oriented in clockwise order with v_0 being an arbitrary vertex. The angular displacement of the vertices, $\alpha(v_0), \alpha(v_1), \cdots, \alpha(v_n)$, as well as their maximum and minimum, can be computed in linear time using (1.1). Let $\alpha_{max} = \max\{\alpha(v_i)\}$ and $\alpha_{min} = \min\{\alpha(v_i)\}$. If $\alpha_{max} - \alpha_{min} \geq 2\pi$ then z is blocked exterior otherwise z is free exterior.

If z is free exterior to P, then the problem of computing $V(P,z)$ can be reduced to the following equivalent problem of computing $V(Q,z)$ with z on $Bd(Q)$ as in Lee (1983). Let v_j and v_k be the vertices closest to z with angular displacement α_{min} and α_{max}, respectively. $Bd(P)$ can be partitioned into the front chain $F = v_j v_{j+1} \cdots v_{k-1} v_k$ and the back chain $B = v_k v_{k+1} \cdots v_{j-1} v_j$ where the indices are taken modulo n. Since the vertices of P are oriented in the clockwise direction, chain F is in front of chain B, i.e. for every point $v \neq v_j$ or v_k on B, zv intersects F, so $B - \{v_j, v_k\}$ is not visible from z by Lemma 1 and can be ignored. v_j and v_k are both visible from z, so $V(P,z) = V(Q,z)$ where Q is the polygon with vertices z, v_j, v_{j+1}, $\cdots$, v_{k-1}, v_k, and z in counterclockwise order.

References

D. Avis and D. Rappaport (1985), Computing the largest empty convex subset of a set of points, *Proc. 1st ACM Symp. on Computational Geometry*, pp. 161-167.

G. F. Carrier, M. Krook, and C. E. Pearson (1966), *Functions of a Complex Variable*, McGraw-Hill.

H. El Gindy and D. Avis (1981), A linear algorithm for computing the visibility polygon from a point, *J. Algorithms*, 2, pp. 186-197.

H. Freeman and P. P. Loutrel (1967), An algorithm for the solution of the two-dimensional hidden-line problem, *IEEE Trans. on Electronic Computers*, EC-16, pp. 784-790.

L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan (1986), Linear time algorithms for visibility and shortest path problems inside simple polygons, *Proc. 2nd ACM Symp. on Computational Geometry*, pp. 1-13.

P. Henrici (1974), *Applied and Computational Complex Analysis, Vol. 1*, John Wiley & Sons.

B. Joe and R. B. Simpson (1985), Visibility of a simple polygon from a point, Technical Report CS-85-38, Dept. of Computer Science, Univ. of Waterloo.

B. Joe and R. B. Simpson (1986), Triangular meshes for regions of complicated shape, *Int. J. for Num. Meth. in Eng.*, 23, pp. 751-778.

D. T. Lee (1983), Visibility of a simple polygon, *Computer Vision, Graphics, and Image Processing*, 22, pp. 207-221.

B. Schachter (1978), Decomposition of polygons into convex sets, *IEEE Trans. on Comp.*, C-27, pp. 1078-1082.

S. Suri and J. O'Rourke (1986), Worst-case optimal algorithms for constructing visibility polygons with holes, *Proc. 2nd ACM Symp. on Computational Geometry*, pp. 14-23.

Appendix A

In this appendix, we illustrate Algorithms 1 and 2 for the polygon P and the interior viewpoint z in Figure 1. We also illustrate Algorithm 1 for the polygon P and the blocked exterior viewpoint z in Figure 2. The sequence of procedures P_m called by VISPOL, along with the parameter values i, t, S, and w on entering each procedure, are given in Tables 1, 2, and 3.

m	P_m	i	t	w	S
1	LEFT	1	1	v_1	v_0v_1
2	SCANA	2	1	v_2	unchanged
3	SCAND	6	1	a	unchanged
4	LEFT	24	3	v_{24}	$v_0v_1dv_{24}$
5	LEFT	25	4	v_{25}	$v_0v_1dv_{24}v_{25}$
6	LEFT	26	5	v_{26}	$v_0v_1dv_{24}v_{25}v_{26}$
7	LEFT	27	6	v_{27}	$v_0v_1dv_{24}v_{25}v_{26}v_{27}$
8	FINISH	27	6	v_{27}	unchanged

Table 1 Sequence of procedure calls for polygon P and interior viewpoint z in Figure 1 (Algorithm 1)

m	P_m	i	t	w	S
1	LEFT	1	1	v_1	v_0v_1
2	SCANA	2	1	v_2	unchanged
3	LEFT	12	3	v_{12}	$v_0v_1bv_{12}$
4	LEFT	13	4	v_{13}	$v_0v_1bv_{12}v_{13}$
5	RIGHT	14	4	v_{13}	unchanged
6	SCAND	14	1	c	v_0v_1
7	LEFT	24	3	v_{24}	$v_0v_1dv_{24}$
8	LEFT	25	4	v_{25}	$v_0v_1dv_{24}v_{25}$
9	LEFT	26	5	v_{26}	$v_0v_1dv_{24}v_{25}v_{26}$
10	LEFT	27	6	v_{27}	$v_0v_1dv_{24}v_{25}v_{26}v_{27}$
11	FINISH	27	6	v_{27}	unchanged

Table 2 Sequence of procedure calls for polygon P and interior viewpoint z in Figure 1 (Algorithm 2)

m	P_m	i	t	w	S
1	LEFT	1	1	v_1	v_0v_1
2	LEFT	2	2	v_2	$v_0v_1v_2$
3	SCANA	3	2	v_3	unchanged
4	RIGHT	8	2	d	unchanged
5	SCANC	9	2	v_8	v_0v_1c
6	RIGHT	16	2	e	unchanged
7	RIGHT	17	2	v_{16}	v_0v_1b
8	LEFT	18	3	v_{18}	$v_0av_{17}v_{18}$
9	SCANA	19	3	v_{19}	unchanged
10	LEFT	22	5	v_{22}	$v_0av_{17}v_{18}fv_{22}$
11	SCANA	23	5	v_{23}	unchanged
12	SCAND	27	5	g	unchanged
13	LEFT	43	7	v_{43}	$v_0av_{17}v_{18}fv_{22}hv_{43}$
14	RIGHT	44	7	v_{43}	unchanged
15	SCAND	44	3	p	$v_0av_{17}v_{18}$
16	LEFT	46	5	v_{46}	$v_0av_{17}v_{18}qv_{46}$
17	LEFT	47	6	v_{47}	$v_0av_{17}v_{18}qv_{46}v_{47}$
18	LEFT	48	7	v_{48}	$v_0av_{17}v_{18}qv_{46}v_{47}v_{48}$
19	LEFT	49	8	v_{49}	$v_0av_{17}v_{18}qv_{46}v_{47}v_{48}v_{49}$
20	FINISH	49	8	v_{49}	unchanged

Table 3 Sequence of procedure calls for polygon P and blocked exterior viewpoint z in Figure 2 (Algorithm 1)

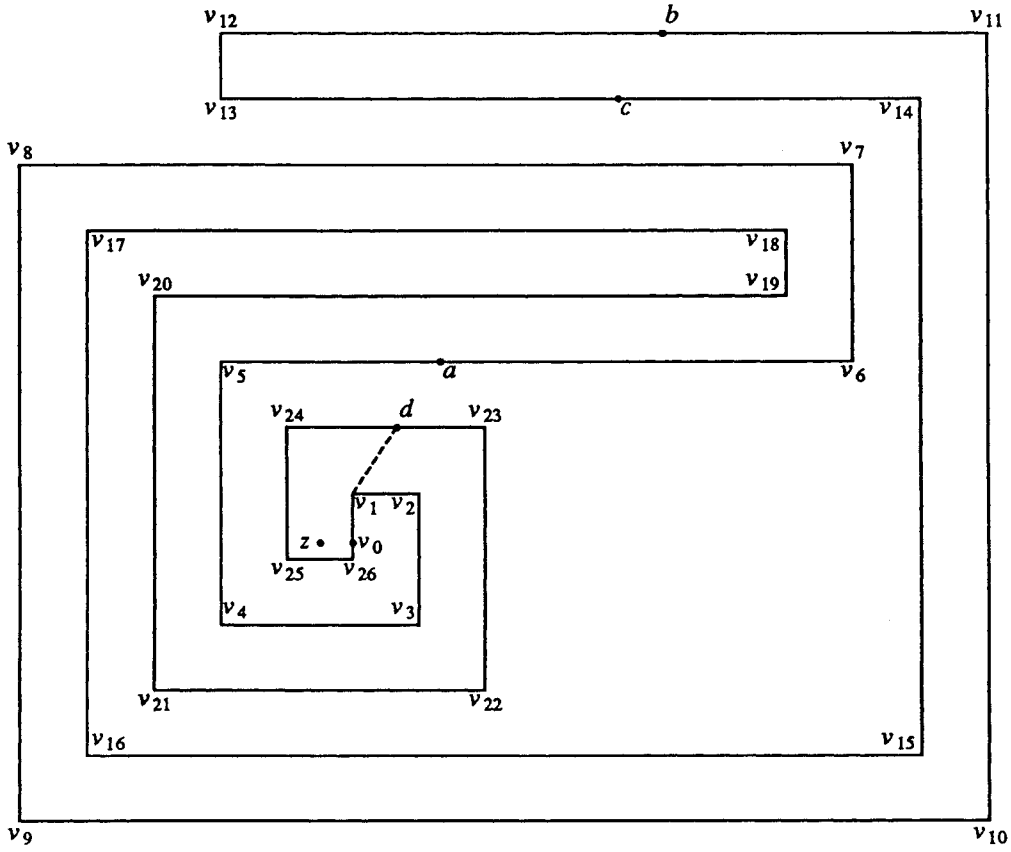

Figure 1 Example for which the algorithms of Lee and El Gindy/ Avis fail for an interior viewpoint z. The labels of vertices are in the exterior of the polygon. $Bd(V(P,z)) = v_0v_1dv_{24}v_{25}v_{26}v_0$. v_1 and v_7 are CCW-oriented points, v_2 and v_6 are CW-oriented points.

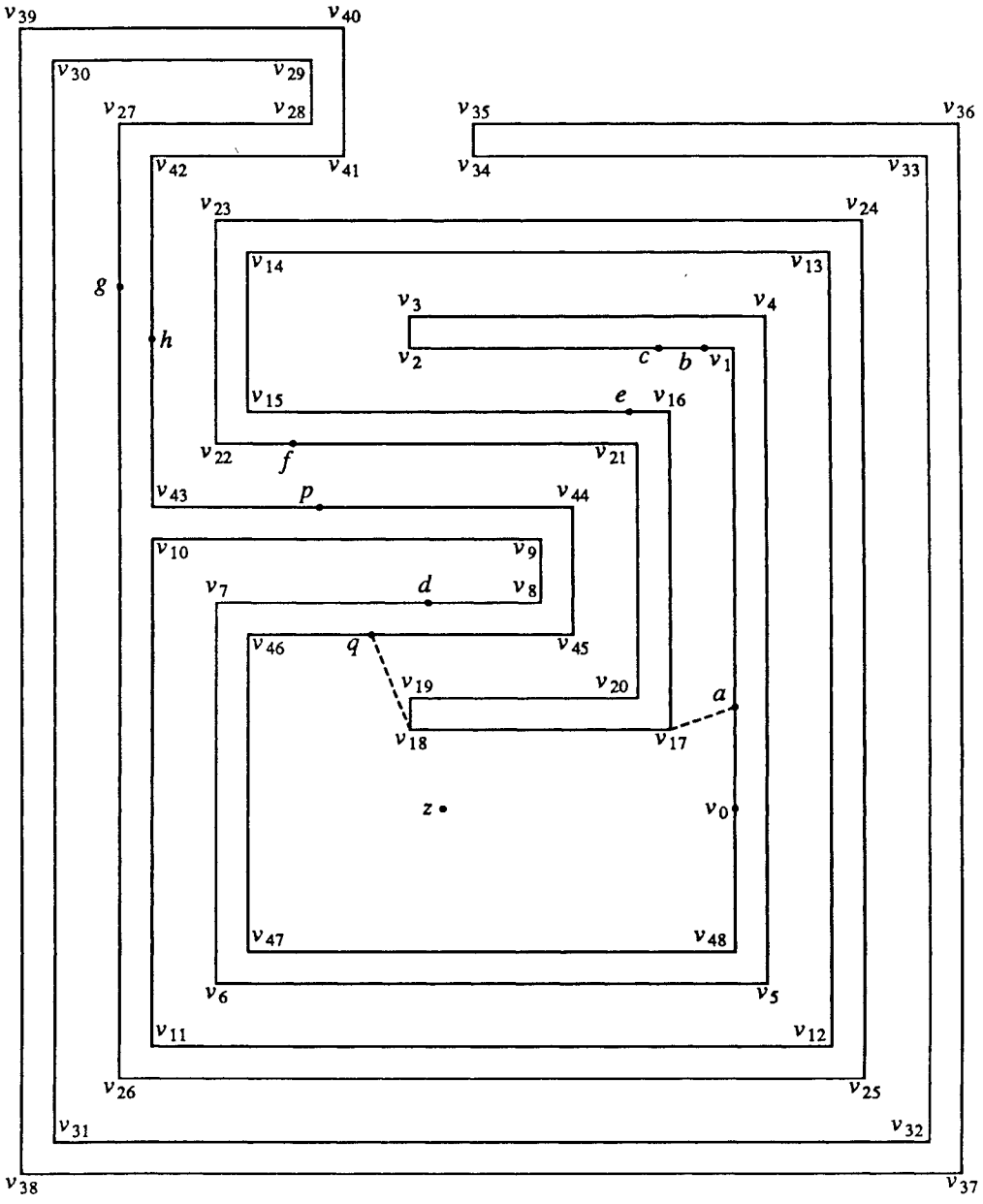

Figure 2 Example for which Lee's modified algorithm fails for a blocked exterior viewpoint z. The labels of vertices are in the exterior of the polygon. $Bd(V(P,z)) = v_0 a v_{17} v_{18} q v_{46} v_{47} v_{48} v_0$. v_1 and v_{18} are CCW-oriented points, v_4 and v_8 are CW-oriented points.

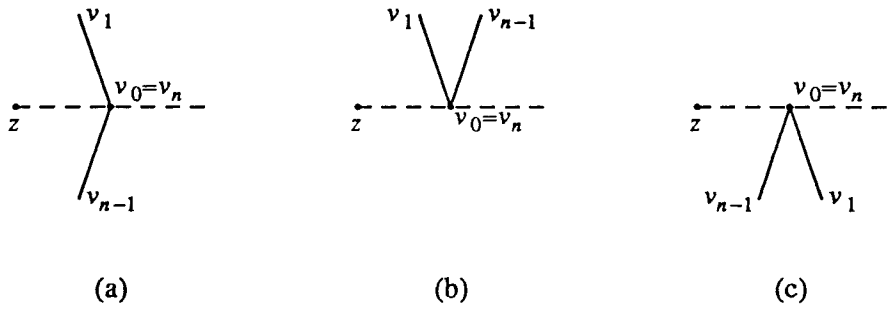

Figure 3 Illustration of possible configurations of edges v_0v_1 and $v_{n-1}v_n$. (a) and (b) are case (V1), (c) is case (V2).

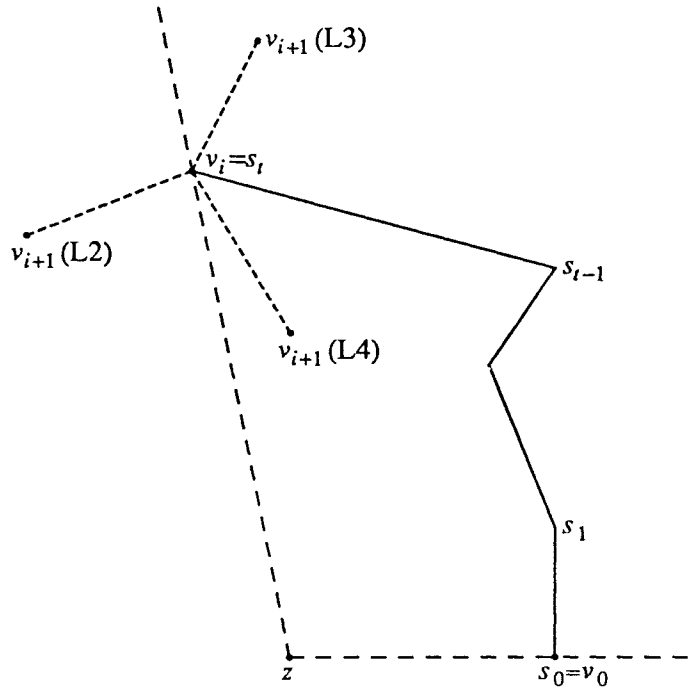

Figure 4 Illustration of chain $S=s_0s_1\cdots s_t$ and edge v_iv_{i+1} in cases (L2), (L3), and (L4) of LEFT.

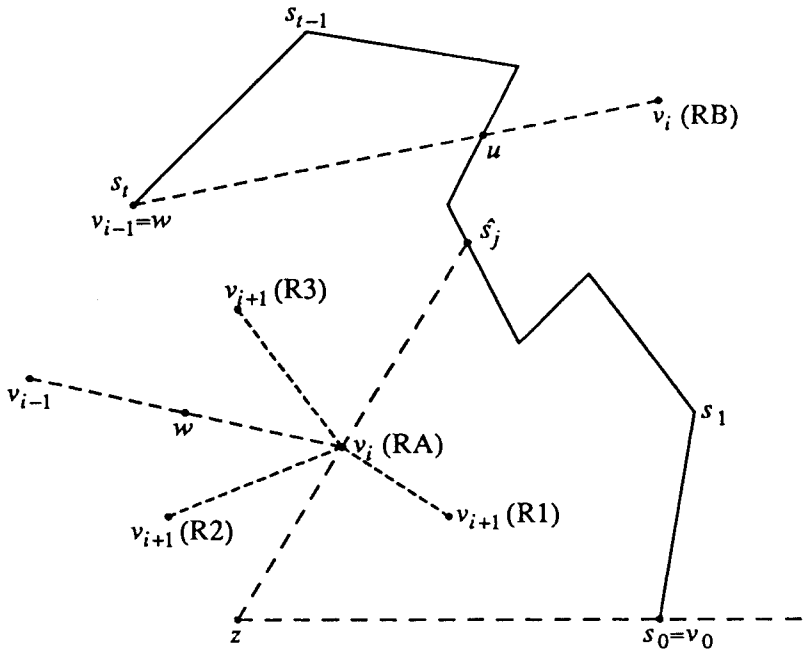

Figure 5 Illustration of chain S, edge $v_{i-1}v_i$ in cases (RA) and (RB), and edge $v_i v_{i+1}$ in cases (R1), (R2), and (R3) of RIGHT.

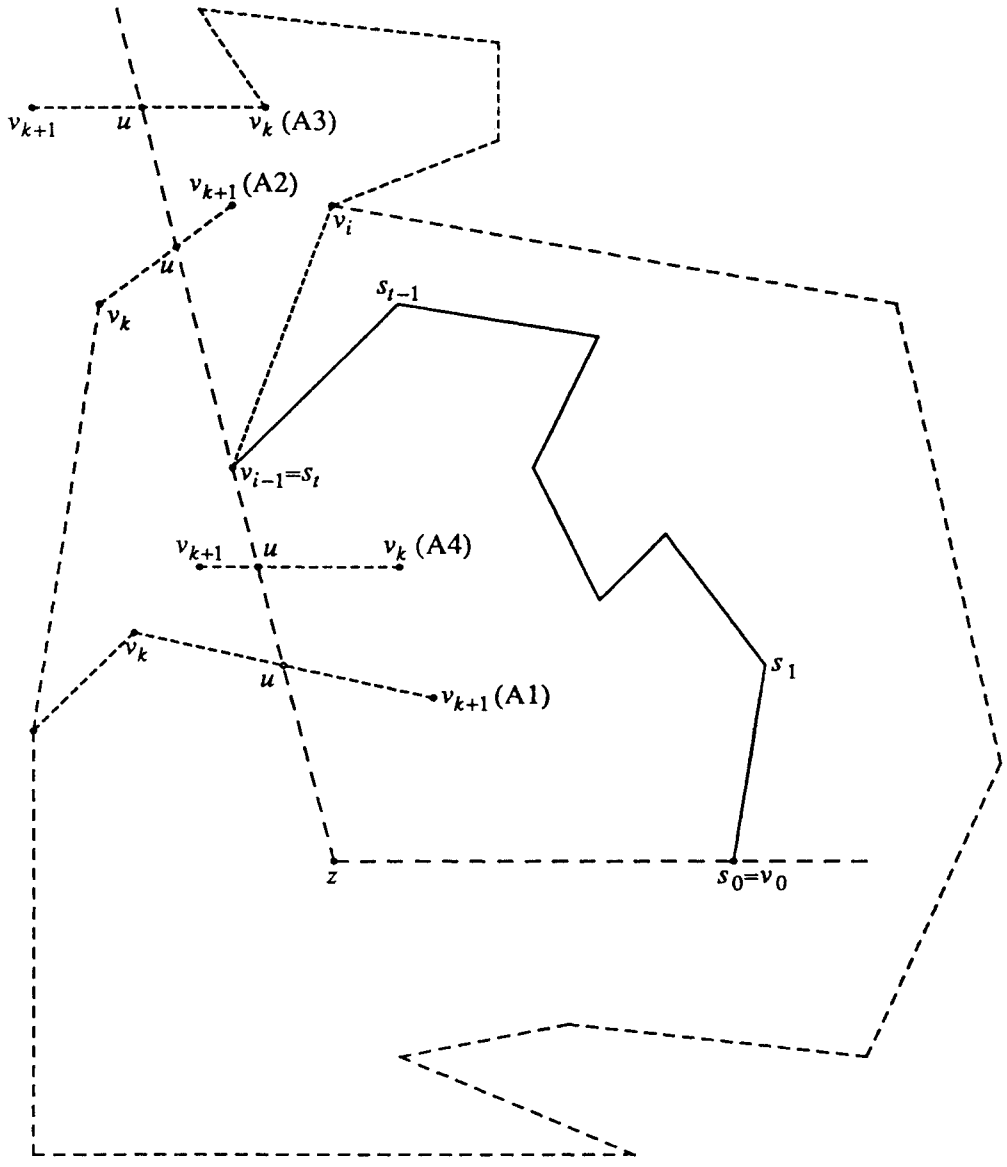

Figure 6 Illustration of chain S and scan for first edge $v_k v_{k+1}$ to intersect $\overrightarrow{zs_t}$ in cases (A1), (A2), and (A3) of SCANA; case (A4) is not possible. $\delta(s_t v_i \cdots v_k u)$ is -2π in cases (A1), (A2) and 0 in case (A3).

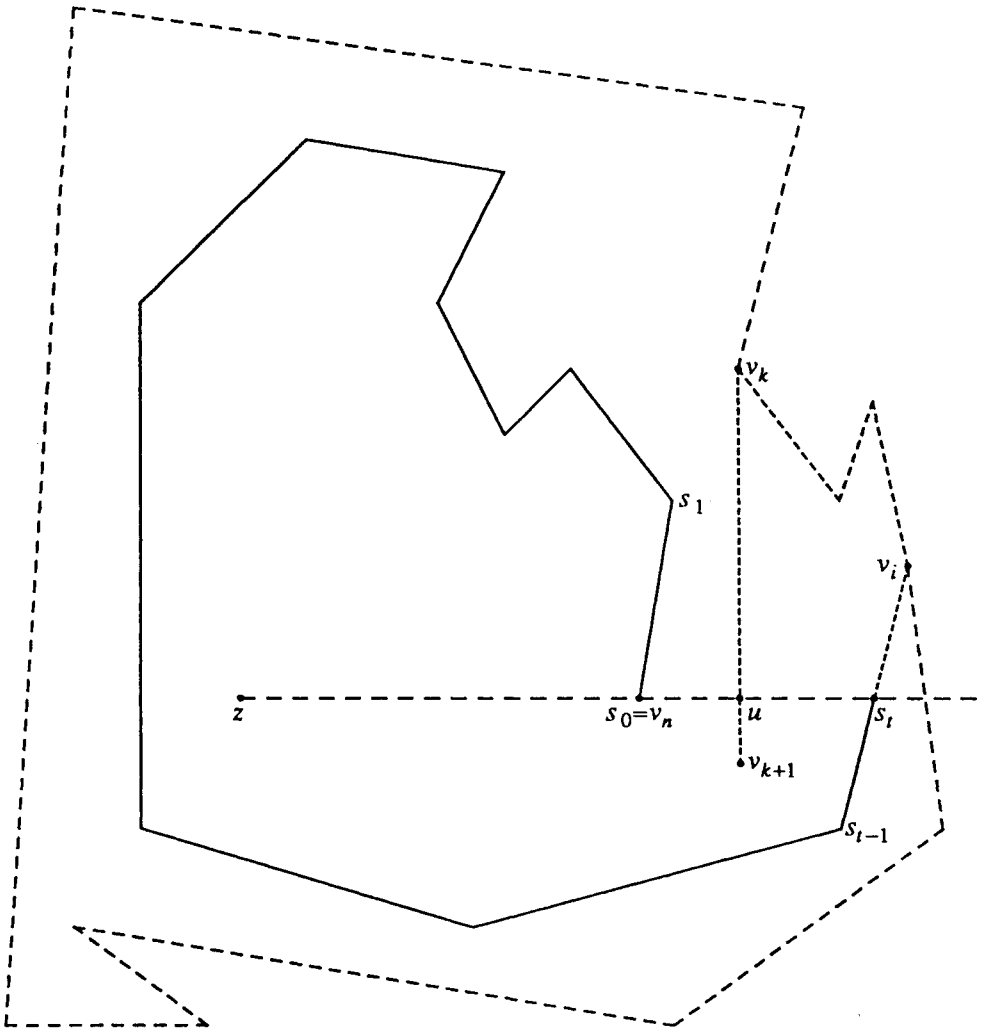

Figure 7 Illustration of chain S and scan for first edge $v_k v_{k+1}$ to intersect $(s_t v_n]$ in SCANB. $\delta(s_t v_i \cdots v_k u)$ may be 0 or -2π.

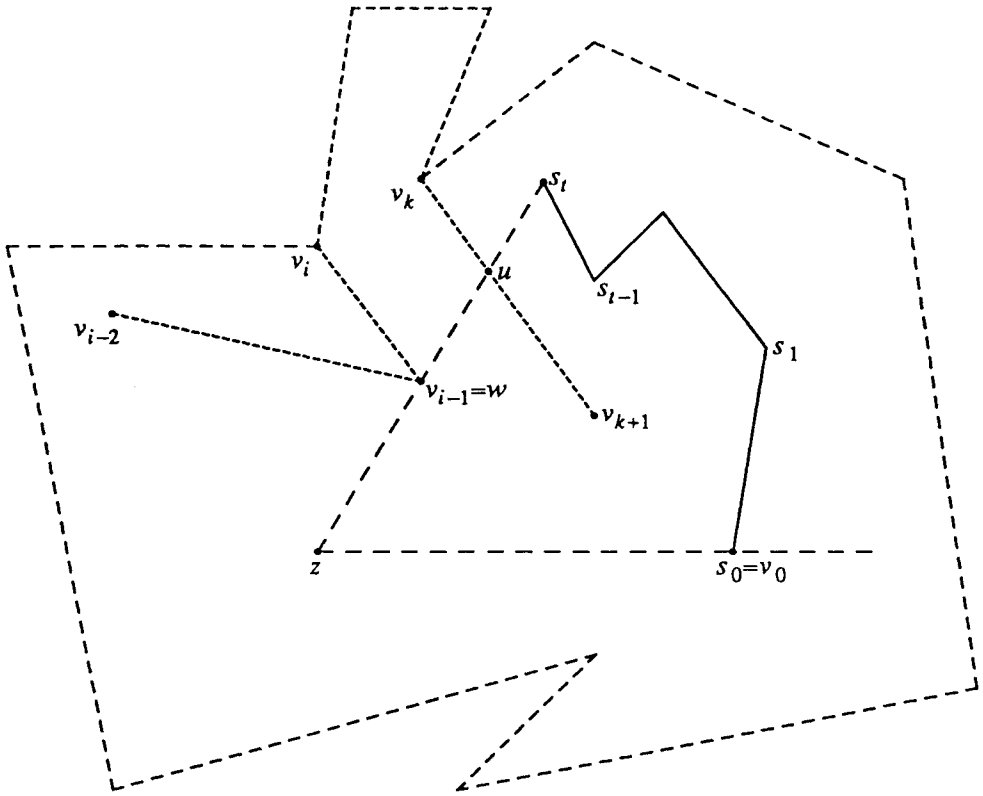

Figure 8 Illustration of chain S and scan for first edge $v_k v_{k+1}$ to intersect $(s_t w)$ in SCANC. $\delta(wv_i \cdots v_k u)$ may be 0 or 2π.

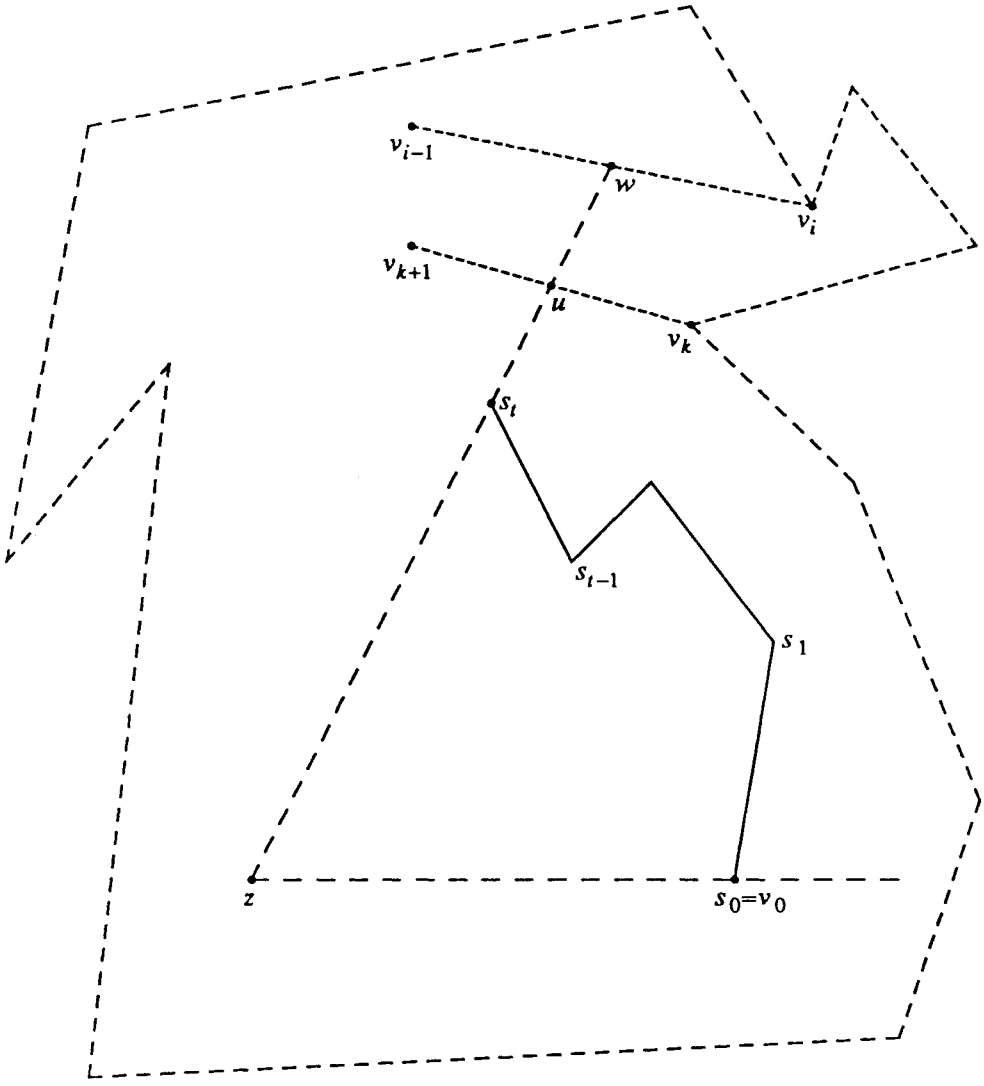

Figure 9 Illustration of chain S and scan for first edge $v_k v_{k+1}$ to intersect $(s_t w)$ in SCAND. $\delta(w v_i \cdots v_k u)$ may be 0 or 2π.

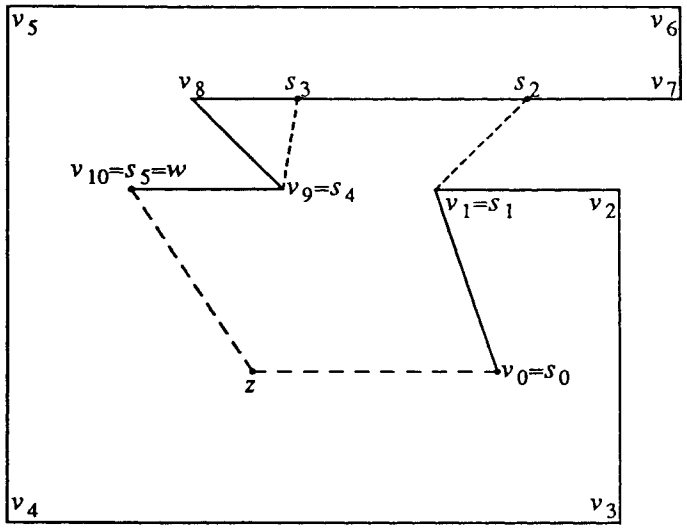

Figure 10 Illustration of C_j and R_j for $Ch[v_0,v_{10}]$ and $S=s_0s_1s_2s_3s_4s_5$:
$C_1=s_1v_2v_3v_4v_5v_6v_7s_2s_1$, $C_3=s_3v_8s_4s_3$, C_j is not defined for $j=0, 2, 4, 5$,
$R_0=zs_0$, $R_1=$ closed interior region bounded by zs_0s_1z,
$R_2=R_1\cup s_1s_2$, $R_3=R_4=$ closed interior region bounded by $zs_0s_1s_2s_3z$,
$R_5=$ closed interior region bounded by $zs_0s_1s_2s_3s_4s_5z$.

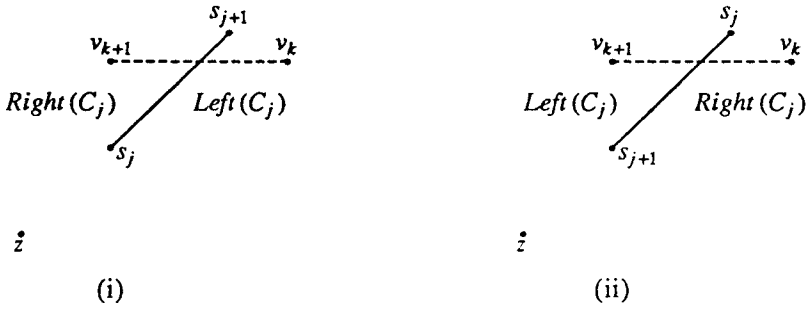

Figure 11 Illustration of s_js_{j+1} and v_kv_{k+1} in cases (i) and (ii) of Lemma 3; cases (iii) and (iv) are similar.

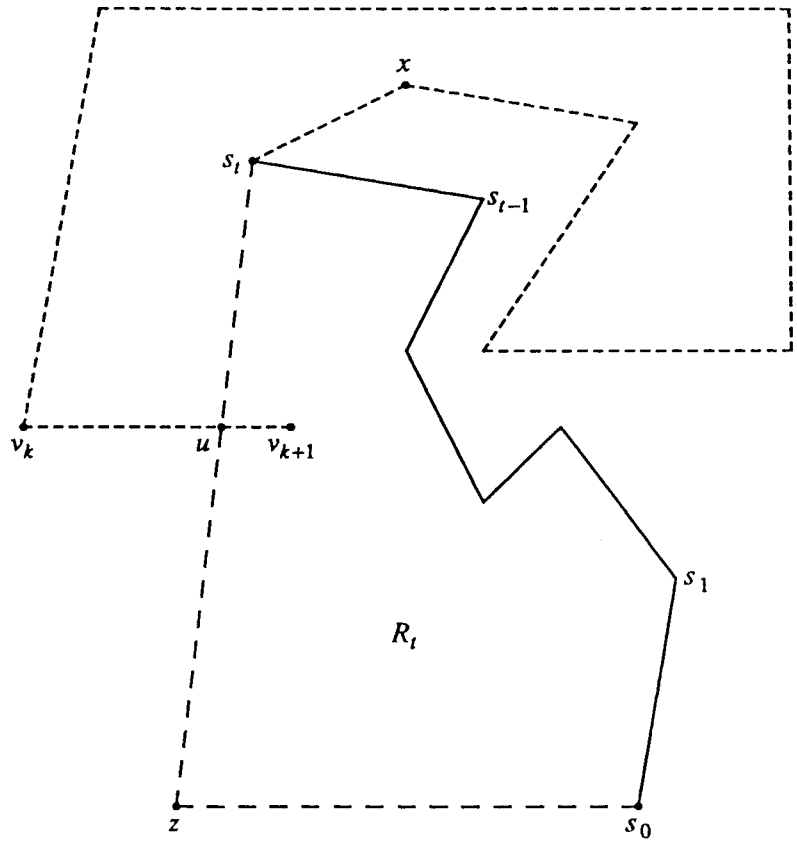

Figure 12 Illustration of chain S and $Ch[s_t, v_{k+1}]$ for Lemma 4.

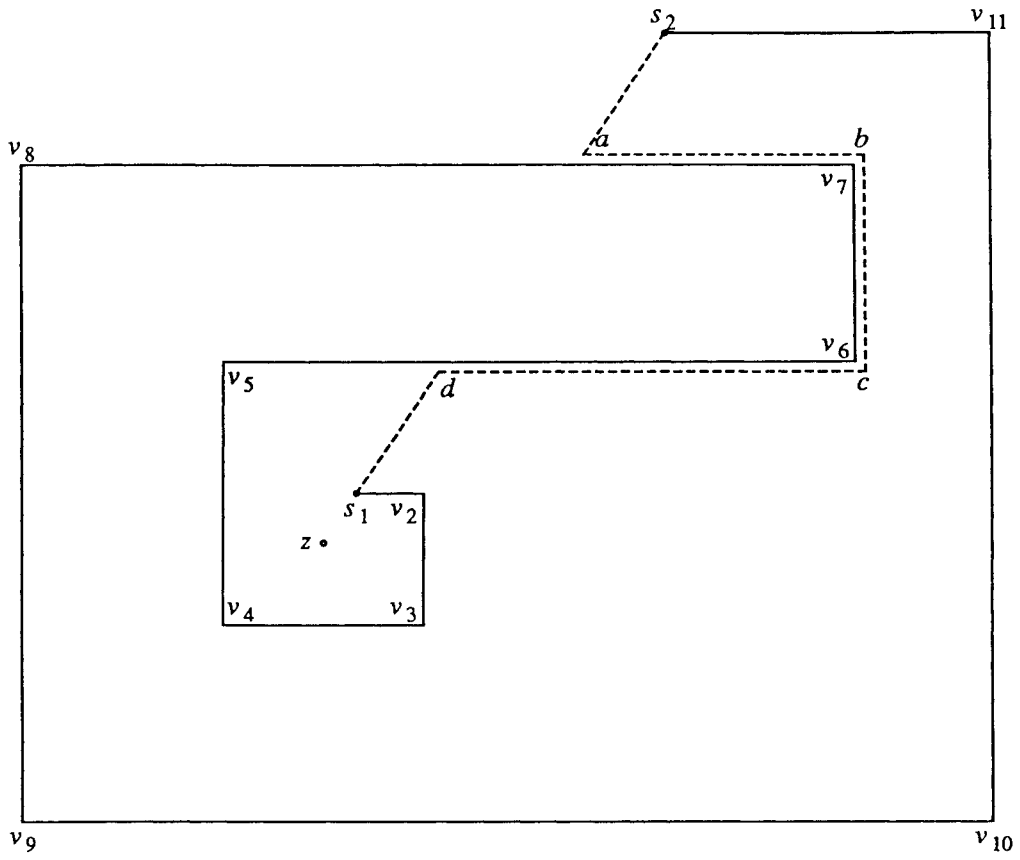

Figure 13 Illustration of $C_j = Ch[s_j, s_{j+1}] \,||\, Path[s_{j+1}, s_j]$ for $j = 1$.
$Ch[s_1, s_2] = s_1 v_2 v_3 \cdots v_{11} s_2$ is part of $Bd(P)$ in Figure 1.
$Path[s_2, s_1] = s_2 abcd s_1$ is the dashed chain; the 'detour' part, $abcd$, is arbitrarily close to the corresponding part of $Ch[s_1, s_2]$.