1-1-1998

The Vertex-Edge Visibility Graph of a Polygon

Joseph O'Rourke
Smith College, jorourke@smith.edu

Ileana Streinu
Smith College, istreinu@smith.edu

ELSEVIER

Computational Geometry 10 (1998) 105–120

Computational Geometry
Theory and Applications

The vertex–edge visibility graph of a polygon ☆

Joseph O'Rourke *, Ileana Streinu

Department of Computer Science, Smith College, Northampton, MA 01063, USA

Communicated by J.-R. Sack; submitted 9 July 1996; accepted 18 November 1996

Abstract

We introduce a new polygon visibility graph, the vertex–edge visibility graph G_{VE}, and demonstrate that it encodes more geometric information about the polygon than does the vertex visibility graph G_V. © 1998 Elsevier Science B.V.

Keywords: Visibility graphs; Visibility complex

1. Introduction

The polygon vertex visibility graph has been studied extensively, often with the goal of characterization. However, progress has only been achieved by restriction of either the class of polygons or the class of graphs [9]. In this paper we introduce a polygon visibility graph G_{VE} that contains more information than the vertex visibility graph G_V (at least for polygons in general position), and so might be an easier target for characterization. Indeed in other work [10,11] we characterize G_{VE} in a "pseudo-visibility" context; in this paper, however, we consider only straight-line polygons. Our goal is to demonstrate that information is derivable from G_{VE} that is not available from G_V, thus establishing that G_{VE} is a "richer" combinatorial structure. The emphasis is on structural relationships and not on the algorithms for constructing the structures.

1.1. Definitions of visibility graphs

Let P be a polygon (a closed region of the plane), V its set of vertices, and E its set of edges. Let $V = (v_0, v_1, \ldots, v_{n-1})$ with indices increasing in counterclockwise (ccw) order, and let $E = (e_0, e_1, \ldots, e_{n-1})$ with $e_i = v_i v_{i+1}$, an open segment (excluding both endpoints).[1] Define two points

☆ Supported by NSF grant CCR-9421670.

* Corresponding author. E-mail: {orourke,streinu}@cs.smith.edu.

[1] All index arithmetic is mod n throughout the paper.

PII S0925-7721(97)00011-4

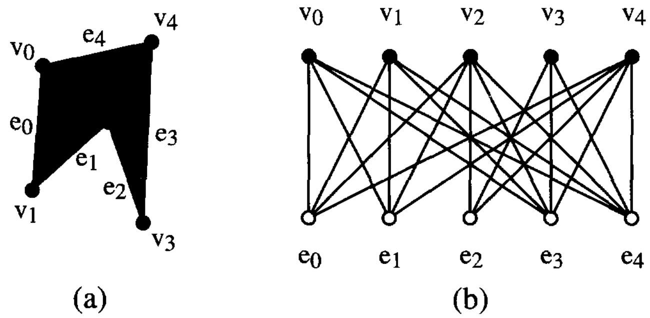

Fig. 1. (a) P. (b) $G_{VE}(P)$.

x and y of P to be *visible* to one another if the segment xy is nowhere exterior to P. The *vertex visibility graph* $G_V(P)$ of P has node set V, with an arc between two vertices iff they are visible to one another; we will not mention P if it is evident from the context. Note the definition of visibility implies that every edge of P is included as an arc of G_V. Also note that G_V is labeled by the indices of the vertices, which are given in ccw order. Therefore the Hamiltonian cycle corresponding to the polygon boundary is known from the labeling of the vertices/nodes.

Define two objects α and β to be *visible* to one another if there are points $x \in \alpha$ and $y \in \beta$ such that x sees y. This is the notion of *weak visibility* [2], but we will drop the modifier. For our purposes, the two objects will be a vertex and an edge of P.

Define $G_{VE}(P)$ to be a bipartite graph with node set $V \cup E$, with an arc between $v \in V$ and $e \in E$ iff v can see the (open) edge e. Note that v must see a point interior to e because we define edges as open. We explored the graph resulting from treating edges to include their endpoints, and found it much less useful.[2] We will use "$\rightarrow$" synonymous with "sees", so that "$(v, e) \in G_{VE}$" may be abbreviated $v \rightarrow e$. Again we consider the nodes labeled by the vertex and edge indices in ccw order, and again we will drop P when convenient. See the example in Fig. 1. Although this graph has been mentioned in the literature,[3] it seems not to have been studied systematically.

G_{VE} may be constructed in $O(n + k)$ time for a polygon with n vertices and k visibility edges, by a slight modification of Hershberger's algorithm that constructs G_V in this time bound [8] from a polygon triangulation. Supplementing with Chazelle's linear-time triangulation algorithm achieves the claimed bound.

A polygon is in *general position* (g.p.) if no three of its vertices lie on a line. Many of our results only hold for g.p. polygons. Collinearities present a significantly more complex situation for visibility graphs, which we explore briefly in Section 10.

1.2. Summary of results

We establish that for g.p. polygons, G_{VE} determines which vertices are reflex and which convex (Section 2), determines G_V uniquely (Section 3), determines for each vertex v the visibility polygon

[2] For example, Lemma 1 does not hold if edges include their endpoints.

[3] [5, p. 909]: "...from this the vertex–edge weak visibility graph can be derived (where a vertex and edge are adjacent if the vertex can see at least one point of the edge)." We have not located any other references.

$A(v)$, the "partial local sequence" $\sigma(v)$, and the shortest path tree $\tau(v)$ (Sections 4–6). We introduce the edge–edge visibility graph in Section 7 and show G_{VE} contains the same information. Relatedly, we sketch in Section 8 an argument that G_{VE} and the incidence structure of the "visibility complex" of Pocchiola and Vegter are derivable from one another. We also show that the external vertex–edge visibility graph G^e_{VE} determines the convex hull vertices (Section 9). In none of these cases does G_V determine the same information. We conclude in Section 10 with a discussion of collinearities.

2. Reflex vertex determination

We will use the symbol $\Rightarrow$ to mean "determines".

Let $V_r \subset V$ be the set of reflex vertices, vertices at which the interior angle is strictly greater than π.

Lemma 1. *$v_i \in V_r$ iff $v_{i-1} \not\to e_i$.*

Proof. Suppose v_i is reflex. Then v_{i-1} cannot see e_i because e_i, being open at v_i, lies entirely right of the extension of $v_{i-1}v_i$. Suppose on the other hand that v_i is not reflex: its internal angle is flat or strictly convex. Then v_{i-1} must be able to see a point of e_i in a neighborhood of v_i.

Note this holds true even without a general position assumption, since even if $\{v_{i-1}, v_i, v_{i+1}\}$ are collinear, v_{i-1} sees (every point of) $e_i = (v_i, v_{i+1})$. The lines of sight could only be completely blocked by violating the simplicity of the polygon. □

Theorem 1. *$G_{VE} \Rightarrow V_r$.*

Proof. Follows immediately from Lemma 1. □

Lemma 2. *$G_V \not\Rightarrow V_r$.*

Proof. This is established by the polygons in Fig. 2. □

3. Determination of G_V

Lemma 3. *$G_V \not\Rightarrow G_{VE}$.*

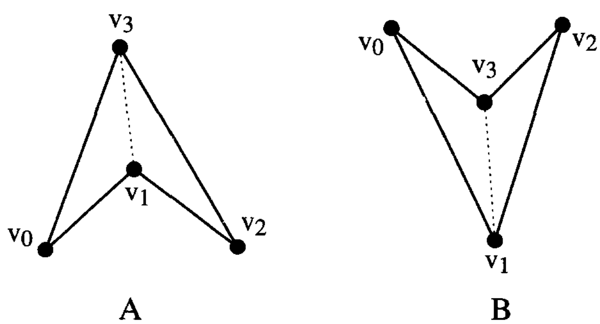

Fig. 2. $G_V(A) = G_V(B)$, but v_1 is reflex in A and v_3 is reflex in B.

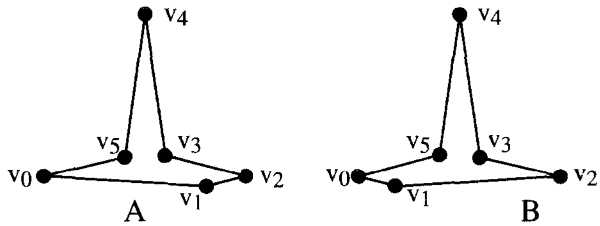

Fig. 3. $G_V(A) = G_V(B)$, but v_4 sees different edges in A and B, so $G_{VE}(A) \neq G_{VE}(B)$.

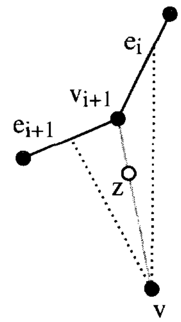

Fig. 4. If v sees e_i and e_{i+1}, it must also see v_{i+1}.

Proof. Fig. 3 shows two polygons with identical vertex visibility graphs: $G_V(A) = G_V(B)$. But in A, v_4 sees e_0 but not e_1, whereas in B, v_4 sees e_1 but not e_0. So $G_{VE}(A) \neq G_{VE}(B)$. Non-isomorphic A and B with the same property may also be found. □

To establish that $G_{VE} \Rightarrow G_V$, we first prove some lemmas that bridge between vertex–edge and vertex–vertex visibility.

Lemma 4. *If v sees e_i and e_{i+1}, it must also see v_{i+1}.*

Proof. The segment vv_{i+1} lies within the region bounded by a segment from v to an interior point of e_i, a segment from v to an interior point of e_{i+1}, and e_i and e_{i+1} (see Fig. 4).

Thus any exterior point z on vv_{i+1} is enclosed by points of P, a contradiction to the simply connectedness of P. □

We use the following notation to specify parts of the polygon boundary: $P[i,j]$ is the closed subset of the polygon boundary ccw from v_i to v_j. $P(i,j]$ excludes v_i; $P[i,j)$ excludes v_j; and $P(i,j)$ excludes both. Let l_{ij} be the line through v_i and v_j. Let $r_j^i \subset l_{ij}$ be the ray along l_{ij} starting at and including v_j, directed away from (and therefore excluding) v_i.

Lemma 5. *For g.p. polygons P, if v_i sees v_j, then v_i sees one of the two edges incident to v_j: either e_{j-1} or e_j (or both). More precisely, v_i sees both when they lie on opposite sides of the line l_{ij}, and just one when they lie on the same side: e_{j-1} when they lie to the right, and e_j when they lie to the left.*

Proof. This follows easily from the general position assumption. □

Lemma 6. *If v_k sees two distinct vertices, it must see an edge between them: if v_k sees v_i and v_j, $v_k \in P[j,i]$, then v_k sees an edge $e_m \in P[i,j]$.*

Proof. By Lemma 5, v_k sees at least one of the two edges incident to v_i. If v_k sees $e_i \in P[i,j]$, the lemma is satisfied. So suppose v_k sees e_{i-1} but not e_i. By Lemma 5, this occurs when e_{i-1} and e_i lie to the right of l_{ki}. But then the line of sight along l_{ki} must exit P somewhere in $P[i,j]$. The general position assumption prevents exit at a vertex; so it must exit interior to an edge $e_m \in P[i,j]$. □

A counterpart to the preceding lemma is the following, which is equally straightforward.

Lemma 7. *If v_k sees two distinct edges, it must see a vertex between them. In particular, if v_k sees edges e_i and e_j and no edge between, $v_k \in P[j+1,i]$, then v_k sees either v_{i+1} or v_j.*

Proof. We discuss only the more particular claim. One of the two edges can block the view from v_k of one of the two endpoints v_{i+1} or v_j. Only the chain $P(i+1,j)$ could block the view of the other endpoint. But if this chain did, v_k would see an edge between, in contradiction to the assumption of the lemma. □

The preceding three lemmas lead up to a key property that we will use often in the sequel.

Lemma 8. *If v_k sees non-adjacent edges e_i and e_j and no edge between, $v_k \in P[j+1,i]$, then exactly one of Cases A or B holds (see Fig. 5):*
Case A

(1) *v_k sees v_{i+1} but not v_j; and*
(2) *v_{i+1} sees e_j, but v_j does not see e_i.*

Case B

(1) *v_k sees v_j but not v_{i+1}; and*
(2) *v_j sees e_i, but v_{i+1} does not see e_j.*

Proof. We first illustrate the lemma with the example in Fig. 1. Let $v_k = v_0$, which sees $e_i = e_1$ and $e_j = e_3$ and no edge between. This falls under Case A of the lemma: v_0 sees v_2 but not v_3, and v_2 sees e_3 but v_3 does not see e_1. We now proceed with the proof.

We know from Lemma 7 that v_k sees at least one of v_{i+1} and v_j. The accuracy of the case partition between A(1) and B(1) is guaranteed: for if v_k saw both v_{i+1} and v_j, it must see an edge between by Lemma 6, violating the no-edge-between assumption. Let us then consider Case A, where v_k sees v_{i+1}, and prove A(2).

Note that because v_k sees e_i but not e_{i+1}, Lemma 5 shows that both these edges are right of $l_{k,i+1}$, as illustrated in Fig. 5A. The ray r^k_{i+1} from v_{i+1} along $l_{k,i+1}$ must exit on e_j, and therefore v_{i+1} sees e_j. Since v_j is in the shaded pocket, it cannot see e_i, and we have established A(2).

Case B of the lemma is the same as Case A under mirror reflection of the polygon (and appropriate relabeling). □

We prove in [10] that v_{i+1} and v_j are articulation points in the subgraphs induced by $P[i,j]$ and $P[j+1,i]$ in Cases A and B, respectively; other authors have called these "blocking" vertices [1,4].

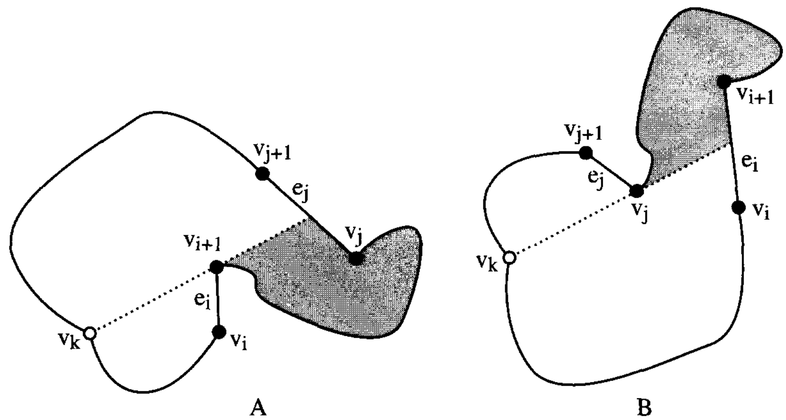

Fig. 5. Cases in the proof of Theorem 2: v_k sees e_i followed by e_j.

We call the shaded regions in Fig. 5 the *far pockets* of the $v_k \to e_j$ visibility relation. More precisely, the chains $P(i+1, j]$ and $P[i+1, j)$ are the far pockets in Cases A and B, respectively. It is important that these chains can be identified from G_{VE} according to Lemma 8.

Let $E(v)$ be the list of edges visible from v (i.e., the neighbors of v in G_{VE}) sorted ccw about v. Let $|G_{VE}|$ be the size (number of arcs) in G_{VE}. It is clear that $|G_V| = \Theta(|G_{VE}|)$.

Theorem 2. *For g.p. polygons P, $G_{VE} \Rightarrow G_V$, and G_V can be constructed from G_{VE} with an algorithm linear in $|G_{VE}|$.*

Proof. For each $v_k \in V$, scan the list of edges $E(v_k)$. For each pair of edges e_i and e_j consecutive in $E(v_k)$, consider two cases.
(1) e_i and e_j are adjacent edges of P, i.e., $i + 1 = j$. Then by Lemma 4, v_k must see v_{i+1}, so $(v_k, v_{i+1}) \in G_V$.
(2) e_i and e_j are not adjacent. Then Lemma 8 applies, and we know exactly one of Cases A or B holds (Fig. 5). Case A holds iff $v_{i+1} \to e_j$ (A(2)), in which case we have $(v_k, v_{i+1}) \in G_V$ and $(v_k, v_j) \notin G_V$ (A(1)). Case B holds iff $v_j \to e_i$ (B(2)), in which case we have $(v_k, v_j) \in G_V$ and $(v_k, v_{i+1}) \notin G_V$ (B(1)).

We now argue that this procedure uniquely determines G_V. Consider an arc $(v_k, v_{i+1}) \in G_V$. By Lemma 5, v_k sees at least one of the two edges incident to v_{i+1}. If it sees both, then we fall into (1) above. If it only sees one, suppose it only sees e_i without loss of generality. Let e_j be the edge in $E(v_k)$ following e_i. Now we fall into (2) above. Therefore every arc of G_V is identified by the algorithm.

Moreover, it is clear that the algorithm only identifies true vertex visibilities, so no arc will be identified for a pair of invisible vertices. □

4. Visibility polygon determination

Let Vis(x) be the region of points in P visible from x. This is often called the *visibility polygon* from x, but it might not be a polygon if x is collinear with two or more vertices. For general positions P, however, Vis(x) is always a polygon. Let $V(v)$ be the (circular) list of vertices visible from v, a list readily available from G_V. (Recall that G_V is labeled with vertex indices.) This is a partial representation for Vis(v). A slightly more informative representation includes the edge labels: let $\Lambda(v)$ be a list of the labels of the vertices and edges encountered in a boundary traversal of Vis(v).

Theorem 3. *$G_{VE} \Rightarrow \Lambda(v)$ for each vertex v of P; each list of labels can be constructed from G_{VE} with an $O(n)$ algorithm.*

Proof. $\Lambda(v)$ is just a merging of the two lists $V(v)$ and $E(v)$. □

Lemma 9. $G_V \not\Rightarrow \Lambda(v)$.

Proof. This is established by the two polygons with equal G_V's in Fig. 3. $\Lambda(v_4)$ is different: $(v_4, e_4, v_5, e_0, v_3, e_4)$ in A but $(v_4, e_4, v_5, e_1, v_3, e_4)$ in B. □

5. Partial local sequence determination

We now examine a structure slightly richer than the visibility polygon. Define the *partial local sequence*[4] $\sigma(x)$ of a point $x \in P$ to be a list of lists of vertex labels, as follows. Let a directed line pass through x, and record, as L spins from 0 to π about x ccw, the ordered sets of vertices (and x) visible to x that lie on L. For $x = v_0$ a vertex of a g.p. polygon, only one other vertex may lie on L at any orientation, and then we define $\sigma(v_0)$ to be this list of vertices. For example, for the polygon shown in Fig. 6,

$$\sigma(v_0) = (v_1, v_5, v_{10}, v_{11}, v_6, v_7, v_{12}, v_{13}, v_9).$$

The partial local sequence $\sigma(v)$ contains more information than $\Lambda(v)$, since the former rotates a *line* through v, while the latter effectively rotates a *ray* through v. We have found $\sigma(v)$ useful in our work on pseudo-visibility.

Before proving that G_{VE} determines local sequences, we need a technical lemma.

Lemma 10. *G_{VE} uniquely determines, for each pair of visible vertices (v_i, v_j), the label of the vertex or edge at which the ray r_j^i first exits P.*

Proof. Recall that $r_j^i \subset l_{ij}$ is the ray directed from v_j excluding v_i. If v_i sees both edges incident to v_j, then they lie on opposite sides of l_{ij} by Lemma 5, and r_j^i exits at v_j. If v_i sees only one edge incident to v_j, let it see e_{j-1} without loss of generality, again according to Lemma 5. Then r_j^i exits on the first edge e_m ccw of e_{j-1} visible from v_i. □

Theorem 4. *$G_{VE}(P)$ uniquely determines the partial local sequence $\sigma(v)$ of each vertex v of P, and each of these sequences can be constructed from G_{VE} with an $O(n)$ algorithm.*

[4] This is a specialization of Goodman and Pollack's "local i-sequences" for point configurations [6]. We call our sequences "partial" because ours contain no information about invisible vertices.

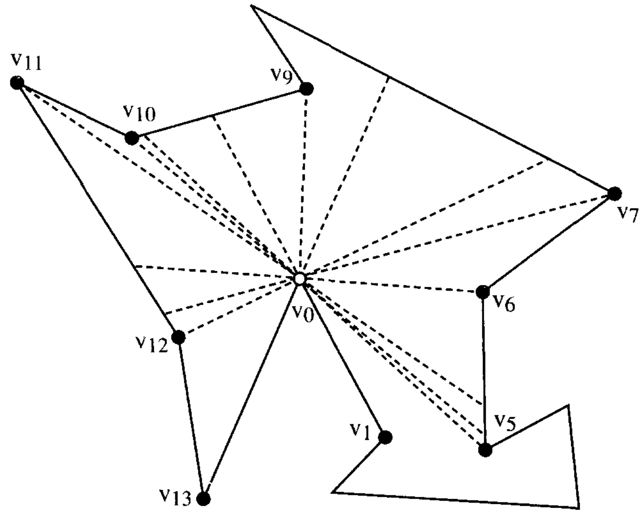

Fig. 6. Definition of partial local sequence.

Proof. We seek to insert the "backwards" ray projections r_k^i, into $V(v_k)$. Consider each vertex v_i ccw of v_k in turn. If r_k^i exits P at v_k (as does r_0^9 in Fig. 6), then label i is not altered in $V(v_k)$. If on the other hand r_k^i exits P on an edge e_m and $e_m \in P[k,i]$, then i is deleted from its original location, and inserted between labels i_1 and i_2 if $e_m \in P[i_1, i_2]$. Where r_k^i exits is determined according to Lemma 10. If there are several labels already moved between i_1 and i_2, because several rays project to the same edge, then i is placed according to the ccw order of the vertices. The process ceases when a complete pass through the vertices has been made. □

For example, for $i = 5$ in Fig. 6, r_0^5 exits on e_9, but because $e_9 \notin P[0,5]$, no change is made. For $i = 12$, r_0^{12} exits on e_7, and because $e_7 \in P[0,12]$, v_{12} is moved to reside between v_7 and v_9. Continuing, v_{13} is placed after v_{12} between v_7 and v_9, because v_{13} is ccw of v_{12}.

Lemma 11. $G_V \not\Rightarrow \sigma(v)$.

Proof. In Fig. 2, $\sigma(v_1) = (v_2, v_0, v_3)$ in A but $\sigma(v_1) = (v_2, v_3, v_0)$ in B. □

6. Shortest path trees

Perhaps our most interesting result is that G_{VE} determines the "shortest path tree" from each vertex, a notion introduced in [7]. What is surprising is that these trees can be captured by a combinatorial structure containing no metric information.

Let $\pi(v_i, v_j)$ be the shortest path between two vertices of P that is nowhere exterior to P. Define $\tau(v) = \bigcup_i \pi(v, v_i)$. $\tau(v)$ is a plane tree rooted at v, called the *shortest path tree* for v.

For example, let $v = v_0$ be the root of visibility in Fig. 7(a); then $\tau(v_0)$ is as shown in Fig. 7(b). The children of v_0 in $\tau(v_0)$ are just those vertices directly visible to v:

$$V(v_0) = (v_1, v_2, v_8, v_9, v_{11}, v_{12}).$$

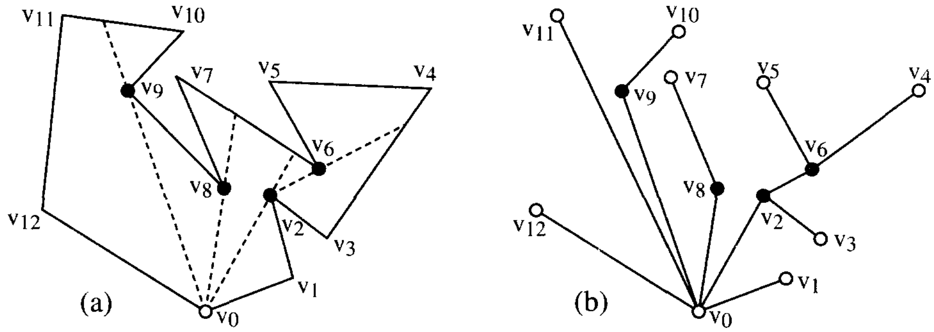

Fig. 7. (a) A polygon with lines of visibility (dashed) from v_0 through the articulation vertices (marked). (b) The shortest path tree rooted at v_0.

We identify which of these themselves have children in $\tau(v_0)$ as follows. Call a vertex an *articulation vertex* (with respect to v) iff it lies between two non-adjacent edges in the list $\Lambda(v)$. For v_0,

$$\Lambda(v_0) = (e_0, v_1, e_1, v_2, e_6, v_8, e_8, v_9, e_{10}, v_{11}, e_{11}, v_{12}, e_{12}).$$

Let $A(v) \subseteq V(v)$ be the list of the articulation vertices.[5] Thus $A(v_0) = (v_2, v_8, v_9)$ are the articulation vertices in the example: v_2 because it lies between e_1 and e_6, v_8 because it lies between e_6 and e_8, v_9 because it lies between e_8 and e_{10}. To move further down the tree we need to employ the notion of "far pockets".

If v_i is an articulation vertex with respect to v, let $F(v, v_i)$ be the vertices in the far pocket (Section 3) of v incident to v_i. In Fig. 7(a), $F(v_0, v_2) = (v_3, v_4, v_5, v_6)$, $F(v_0, v_8) = v_7$ and $F(v_2, v_6) = (v_4, v_5)$. It is clear that the shortest path from v to some vertex that is in the far pocket of an articulation vertex v_j, must pass through v_j. This is incorporated into Step 3 of the algorithm below.

(1) For a root v, the children of v in $\tau(v)$ are the vertices in $V(v)$.
(2) Each of these nodes itself has children in $\tau(v)$ only if it is in $A(v)$.
(3) Let a node v_i of $\tau(v)$ have a child v_j. Then v_j has children iff $v_j \in A(v_i)$. In that case, its children are the vertices in $V(v_j) \cap F(v_i, v_j)$.

We illustrate the use of this procedure to construct one branch of the tree for the polygon in Fig. 7(a). The children of v_0 in $\tau(v_0)$ are $V(v_0)$ with all but $A(v_0)$ leaves. Now let $v_i = v_0$ and $v_j = v_2$. Then v_2 has children because $v_2 \in A(v_0)$.

$$F(v_i, v_j) = F(v_0, v_2) = (v_3, v_4, v_5, v_6),$$

$$V(v_2) = (v_0, v_1, v_3, v_6, v_7, v_8, v_{12}),$$

$$V(v_2) \cap F(v_0, v_2) = (v_3, v_6).$$

Continuing, let $v_i = v_2$ and $v_j = v_6$. v_6 has children because $v_6 \in A(v_2)$.

$$F(v_2, v_6) = (v_4, v_5),$$

$$V(v_6) = (v_2, v_3, v_6, v_7, v_8),$$

$$V(v_6) \cap F(v_2, v_6) = (v_4, v_5).$$

[5] It will be convenient sometimes to treat lists as sets, so that subset notion and set intersection make sense.

Theorem 5. *$G_{VE}(P)$ uniquely determines the shortest path tree from each vertex of P, and each of these trees can be constructed from G_{VE} with an $O(|G_{VE}|)$ algorithm.*

Proof. The construction only requires identifying $\Lambda(v_i)$, $V(v_i) \subseteq \Lambda(v_i)$, and $A(v_i) \subseteq V(v_i)$ for each vertex. These lists are then restricted to far pockets (identified by Lemma 8) of a parent node to obtain the children. Repeating the process yields the complete shortest path tree.

The cost is dominated by the cost of constructing G_V in time linear in $|G_{VE}|$ via Theorem 2. $\Lambda(v_0)$ is obtained in $O(n)$ time via Theorem 3. $A(v_0)$ is identified by a single scan through $\Lambda(v_0)$; the far pockets lists $F(v_0, v_i)$ can be constructed in the same scan. Because subsequent processing is restricted to these far pockets, each intersection $V(v_j) \cap F(v_i, v_j)$ can be computed in time proportional the size of this intersection, by scanning the sublists of $V(v_j)$ and $E(v_j)$ (available from G_V and G_{VE}, respectively, or equivalently, from $\Lambda(v_j)$) between the endpoints of the far pocket, noting whether each vertex is in $A(v_j)$ and creating the relevant subpockets along the way. The total processing will then not exceed the number of children nodes identified. Because $|\tau(v_0)| = O(n)$, the claim that the total cost is dominated by constructing G_V is now established. □

We will not explore the possibility of efficiencies in the computation of all the shortest path trees.

If desired, it is a simple matter to add notation to each non-root, non-leaf node v_j of τ to indicate whether its children lie in the right or left far pocket with respect to (v_i, v_j), where v_i is the parent of v_j. For example, in Fig. 7, v_2's children lie in the right pocket and v_8's children in the left pocket. The right/left determination is made according to which of Cases A or B in Fig. 5 holds.

Lemma 12. *$G_V \not\Rightarrow \tau(v)$.*

Proof. This is established by Fig. 2: e.g., $\tau(v_0)$ differs in the two polygons. □

7. The edge visibility graph

Define the *edge visibility graph* G_E of a polygon P to have a node for each edge of P, and an arc $(e_i, e_j) \in G_E$ iff e_i sees e_j, i.e., iff there is a point x on the (open) edge e_i and a point y on the (open) edge e_j such that x sees y. We show in this section that G_{VE} determines G_E and vice versa.

Lemma 13. *$(e_i, e_j) \in G_E$ iff $\pi(v_i, v_{j+1})$ and $\pi(v_{i+1}, v_j)$ are disjoint from one another.*

Proof. If e_i and e_j can see one another, it was proven in [7] that the indicated paths are disjoint. They named the region delimited by these chains and the edges the *hourglass* for the edge pair; see Fig. 8.

In the other direction, assume $\pi_1 = \pi(v_i, v_{j+1})$ and $\pi_2 = \pi(v_{i+1}, v_j)$ are disjoint. We first argue that these two are both reflex chains.[6] Suppose π_1 is not reflex: so when directed from v_i to v_{j+1}, it turns right at some vertex v_k. Then it must be that $v_k \in P[i+1, j]$ (touching the chain from "above" in the orientation of Fig. 8). But then π_2 must also pass "below" v_k (actually, it must pass through v_k), and therefore π_1 and π_2 are not disjoint.

[6] Called "outward convex" in [7].

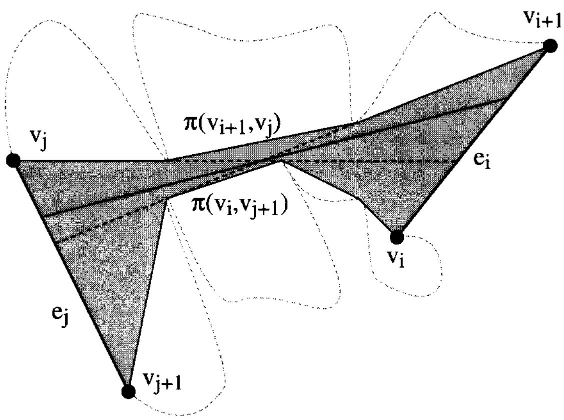

Fig. 8. The hourglass for a pair of edges.

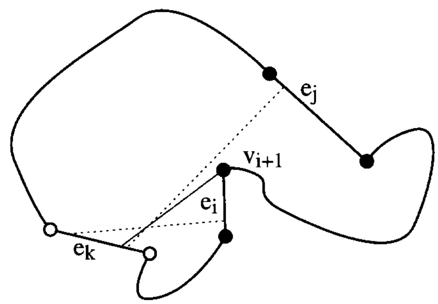

Fig. 9. Case A: $v_{i+1} \rightarrow e_k$.

Since they are both reflex chains, they support two tangents that cross in the hourglass, as shown in the figure. Then any line through their crossing point with slope between that of the two tangents is a line of sight between points interior to e_i and e_j. □

Theorem 6. $G_{VE} \Rightarrow G_E$.

Proof. Construct from G_{VE} the shortest path trees for every vertex of P according to Theorem 5. For each pair of edges e_i and e_j, extract the paths $\pi_1 = \pi(v_i, v_{j+1})$ and $\pi_2 = \pi(v_{i+1}, v_j)$ from $\tau(v_i)$ and $\tau(v_j)$. Checking whether they share any vertices then is easy. It only remains to argue that if the paths are not disjoint, they must share a vertex. But if they do cross properly, it is easy to infer that the polygon includes exterior points, a contradiction.

Invoking Lemma 13, we may use disjointness to infer visibility between each edge pair. □

For completeness we mention the derivation can run the other way in this case.

Theorem 7. $G_E \Rightarrow G_{VE}$.

Proof. We will only sketch a proof, as no new techniques are employed. If e_k sees adjacent edges e_i and e_{i+1}, then the shared vertex v_{i+1} can see e_k. If e_k sees non-adjacent edges e_i and e_j and no edge between, $e_k \in P[j+1, i]$, then we have two cases A and B, just as in Lemma 8. Here we will illustrate just Case A; see Fig. 9. The two segments of visibility between e_k and e_i and e_j either do or do not cross. In the crossing case (illustrated), the line through v_{i+1} and the crossing point shows that $v_{i+1} \to e_k$. The noncrossing case is similar.

Additionally it must be shown that all vertex–edge visibilities may be identified as above, a claim we do not prove in this sketch. □

8. The visibility complex

An important new representation of visibility among pairwise disjoint convex objects, the *visibility complex*, was introduced in [12]. It is a two-dimensional cell complex embedded in a three-dimensional space. Each point of the complex dually corresponds to a ray, with each face the collection of rays seeing the same object forward and backward. A precise definition is technically complicated and will not be repeated here.

Although the visibility complex was originally defined for disjoint, strictly convex objects, one can extend the notion to the visibility complex of the edges of a polygon, which (following [12]) we call $X(P)$. Here the objects are the individual edges. Because the original definitions had no reason to distinguish between internal and external visibility, we must additionally stipulate that only points of the complex corresponding to visibility rays inside P are included in $X(P)$. We now sketch an argument that the incidence structure [7] of $X(P)$ is determined by G_{VE}.

The correspondences between the visibility graphs considered in this paper and the cells of various dimensions of the visibility complex, are as follows.

(1) Each face of X corresponds to an arc of G_E, representing all rays that see e_i and e_j forward and backward. A second face of X, seeing the same pair of edges backward and forward, respectively, also corresponds to the same arc of G_E. There is a two-to-one correspondence between elements of X and visibility graph elements due to the directed nature of X contrasting with the undirected graphs.

(2) Each edge of X corresponds to an arc of G_{VE}, representing rays that are tangent to exactly one edge, i.e., which see an edge endpoint (a vertex) forward and an edge backwards (or vice versa).

(3) Each vertex of X corresponds to an arc of G_V, representing rays tangent to two edges, i.e., lines of sight between two vertices, the "bitangents" of [12].

For example, in Case A of Lemma 8 illustrated in Fig. 5A, the neighborhood of one of the two vertices of X corresponding to $(v_k, v_{i+1}) \in G_V$ includes five faces of X, corresponding to the following arcs of G_E:

$$(e_{k-1}, e_i), \quad (e_k, e_i), \quad (e_{k-1}, e_j), \quad (e_k, e_j), \quad (e_{i+1}, e_j).$$

These faces meet at the X vertex along four edges of X, corresponding to these arcs of G_{VE}:

$$(v_k, e_i), \quad (v_k, e_j), \quad (v_{i+1}, e_k), \quad (v_{i+1}, v_{k-1}).$$

Since we have seen in Theorems 2 and 6 that G_{VE} determines both G_V and G_E, we can determine the complete incidence structure of X from G_{VE}. For example, two of the faces of X corresponding

[7] [12, Section 2.4]: "[This] incidence structure [is] the basis for our choice of a data structure representing the visibility complex."

to the arcs (e_k, e_i) and (e_k, e_j) of G_E meet along one of the edges of X corresponding to the arc (e_k, v_{i+1}) of G_{VE}. Thus continuously moving a ray that sees e_k backward and e_i forward, until it changes to see e_j forward, corresponds to moving a point between these two faces of X, which necessarily transitions at the X edge with a ray that sees v_{i+1} forward. A detailed justification would be tedious, and we will leave determination of X by G_{VE} a claim. We also claim that the reverse holds as well: X determines all the visibility graphs.

9. External vertex–edge visibility graph

It was established in [3] that the external vertex visibility graph of P does not uniquely determine which vertices are on the convex hull of P. Here we show that the external vertex–edge visibility graph G^e_{VE} does.[8] We define $(v, e) \in G^e_{VE}(P)$ iff there is a point x on the (open) edge e such that the segment vx is nowhere interior to P.

Lemma 14. *Two vertices v_i and v_j of P are on the hull of P iff deletion of v_i and v_j from G^e_{VE} disconnects it into two components.*

Proof. We first handle the easier case when v_i and v_j are adjacent vertices, $i + 1 = j$, in which case v_iv_{i+1} is both a hull edge and a polygon edge. Such "polygon hull edges" are the only edges of P visible just from its endpoints.

Certainly if an edge e is a polygon hull edge, it is only externally visible from its endpoints, whose deletion isolates e in G^e_{VE}. Conversely, any edge e isolated by deletion of its endpoints must have no vertices in the exterior halfplane determined by e, which implies e is a polygon hull edge.

In the remainder of the proof we assume that v_i and v_j are not adjacent. Let $C_1 = P(i, j)$ and $C_2 = P(j, i)$ be the two chains delimited by v_i and v_j; both contain at least one vertex. If v_i and v_j are on the hull, it is clear that no vertex of C_2 can see an edge of C_1 and vice versa. So $G^e_{VE} - \{v_i, v_j\}$ consists of two components, H_1 induced by C_1 and H_2 induced by C_2. This establishes the lemma in the easy direction.

Now let us assume that deletion of v_i and v_j separates G^e_{VE} into at least two components, H_1 and H_2. First we argue that (without loss of generality) $H_1 \subseteq G^e_{VE}(C_1)$ and $H_2 \subseteq G^e_{VE}(C_2)$, where $G^e_{VE}(C_i)$ is the subgraph of G^e_{VE} induced by the vertices and edges of C_i. For suppose H_1 contained nodes corresponding to elements in both subchains. Then regardless of where H_2 lies, one of its vertices or edges must be connected by the polygon's boundary to a vertex or edge in H_1, the connection lying entirely in either $P(i, j)$ or $P(j, i)$. But these chains constitute paths in $G^e_{VE} - \{v_i, v_j\}$, contradicting the assumption that H_1 and H_2 are separate components.

So we may assume that $H_i \subseteq C_i$, $i = 1, 2$. Imagine any embeddings of C_i that realize G^e_{VE}. Let L^1_i be the line of sight of a point on C_1 that sees v_i which is most clockwise about v_i, and let L^1_j be the most counterclockwise such line through v_j; see Fig. 10. Then no point of C_2 may fall outside the exterior halfplanes delimited by these two lines. For imagine a point of C_2 did; then some vertex $x_2 \in C_2$ could see a point $x_1 \in C_1$. But these points x_i are connected by their chains C_i to all other points in those chains, and so there would be a path between H_1 and H_2 via x_1x_2 and the chains, contradicting the disconnectedness of H_1 and H_2.

---

[8] We thank Estie Arkin for posing this question.

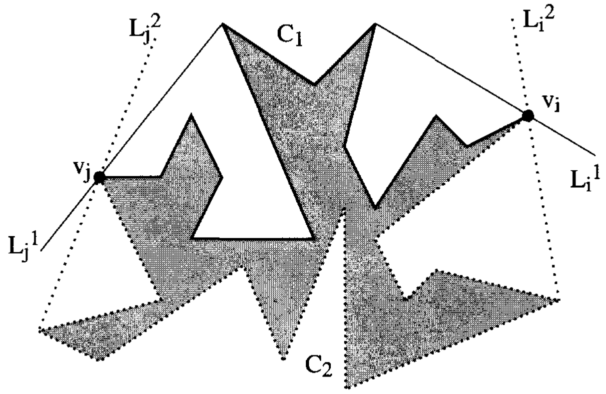

Fig. 10. Lemma 14 establishes that v_i and v_j must be hull vertices.

Similarly no point of C_1 may fall outside the exterior halfplanes delimited by the lines L_i^2 and L_j^2, as in the figure. The two lines through v_i can now be threaded by a third through v_i, supporting the entire polygon to one side. Thus v_i is a hull vertex, and a symmetric argument applies to v_j. □

Theorem 8. $G^e_{VE} \Rightarrow$ *hull vertices.*

Proof. For each v_i, remove v_i from G^e_{VE}, and then see if removal of v_{i+k}, $k = 1, 2, \ldots$, disconnects according to Lemma 14. If v_j is found to disconnect, then (v_i, v_j) is a hull edge, and the search loop can be restarted with $i \leftarrow j$.

A naive implementation of this is $O(n^4)$, a bound which no doubt can be improved. □

10. Remarks on collinearities

We would like to indicate here briefly why the general position assumption we have maintained throughout is more than just an assumption of convenience. First, Theorem 2 is false if collinear vertices are permitted.

Lemma 15. $G_{VE} \not\Rightarrow G_V$.

Proof. Consider the two polygons shown in Fig. 11, with $\{v_0, v_4, v_2\}$ collinear. $G_{VE}(A) = G_{VE}(B)$: moving v_4 down in B does not block any lines of sight between vertices and points interior to edges. For example, v_0 cannot see e_2 in both A and in B. But v_0 can see v_2 in A but not in B. Therefore $G_V(A) \neq G_V(B)$. □

Note that this example also demonstrates that $G_{VE} \not\Rightarrow$ collinearities: which vertices are collinear is not determined by G_{VE}. The next lemma shows that throwing in G_V doesn't help.

Lemma 16. $G_{VE} + G_V \not\Rightarrow$ *collinearities.*

Proof. In Fig. 12, $G_{VE}(A) = G_{VE}(B) = K_{5,5}$, and $G_V(A) = G_V(B) = K_5$, but B has collinearities and A does not. □

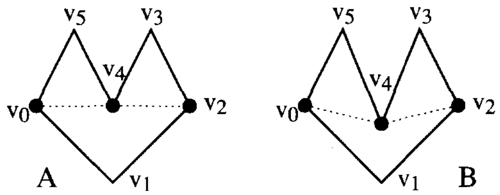

Fig. 11. $G_{VE}(A) = G_{VE}(B)$ but $G_V(A) \neq G_V(B)$.

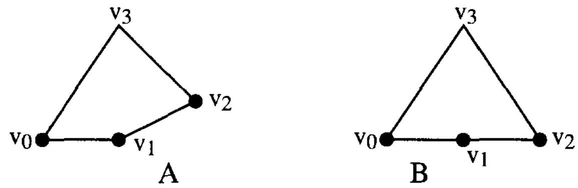

Fig. 12. r_1^0 exits at different vertices in A and B.

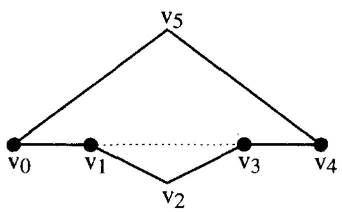

Fig. 13. The graph $G_{VE}(P)$ is not realized by any g.p. polygon.

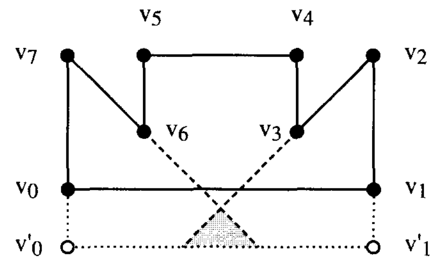

Fig. 14. $G_{VE} \not\Rightarrow$ star-shapedness. Moving the v_0v_1 edge down does not change the visibility graph but does alter the polygon to become star-shaped.

Lemma 10 fails in Fig. 12: in A, r_1^0 exits at v_1, while in B it exits at v_2.

Although G_{VE} does not determine collinearities, it does carry some information about collinearities, in the following sense.

Lemma 17. *There is a graph $G = G_{VE}(P)$ that is not realized by any general position P.*

Proof. Consider the polygon P in Fig. 13. Because $v_0 \to e_3$, v_0 must lie on or above $l_{4,3}$. Knowing that v_5 is not reflex (by Lemma 1), only v_1 can block v_0 from seeing e_2 to ensure that $v_0 \not\to e_2$. This requires v_0 to lie on or below $l_{1,3}$. Together these conditions force $l_{4,3}$ to pass below $l_{1,3}$ left of their point of intersection v_3. But if the lines cross properly at v_3, then $v_4 \to e_1$, which is not the case. Therefore $l_{4,3} = l_{1,3}$, forcing collinearity. □

We note that this is an open problem for vertex visibility graphs.[9]

Conjecture 1. *For any polygon P, there is a general position polygon P' such that $G_V(P') = G_V(P)$.*

Finally, we mention one among the many characteristics of a polygon *not* determined by G_{VE}: whether or not the polygon is star-shaped (visible from one point). Fig. 14 shows a polygon $P = (v_0, \ldots, v_7)$ that is not star-shaped. Moving the v_0v_1 edge downward to $v_0'v_1'$ creates a polygon P' that is star-shaped: any point in the shaded triangle (the kernel) can see all of P'. But $G_{VE}(P) = G_{VE}(P')$.

[9] Posed by the authors at the Ottawa Geometry Day, January 1995.

References

[1] J. Abello, K. Kumar, Visibility graphs and oriented matroids, in: R. Tamassia, I.G. Tollis (Eds.), Graph Drawing (Proc. GD '94), Lecture Notes in Computer Science 894, Springer, New York, 1995, pp. 147–158.
[2] D. Avis, G.T. Toussaint, An optimal algorithm for determining the visibility of a polygon from an edge, IEEE Trans. Comput. 30 (1981) 910–1014.
[3] H. Everett, A. Lubiw, J. O'Rourke, Recovery of convex hulls from external visibility graphs, in: Proc. 5th Canad. Conf. Comput. Geom., Waterloo, Canada, 1993, pp. 309–314.
[4] S.K. Ghosh, On recognizing and characterizing visibility graphs of simple polygons, Discrete Comput. Geom. 17 (2) (1997) 143–162.
[5] S.K. Ghosh, D.M. Mount, An output-sensitive algorithm for computing visibility graphs, SIAM J. Comput. 20 (1991) 888–910.
[6] J.E. Goodman, R. Pollack, Semispaces of configurations, cell complexes of arrangements, J. Combin. Theory Ser. A 37 (1984) 257–293.
[7] L.J. Guibas, J. Hershberger, D. Leven, M. Sharir, R.E. Tarjan, Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons, Algorithmica 2 (1987) 209–233.
[8] J. Hershberger, An optimal visibility graph algorithm for triangulated simple polygons, Algorithmica 4 (1989) 141–155.
[9] J. O'Rourke, Computational geometry column 18, Internat. J. Comput. Geom. Appl. 3 (1993) 107–113; also: SIGACT News 24 (1) (1993) 20–25.
[10] J. O'Rourke, I. Streinu, Pseudo-visibility graphs in pseudo-polygons: Part I, Technical Report 041, Dept. Comput. Sci., Smith College, Northampton, MA, January 1996; revised April 1996.
[11] J. O'Rourke, I. Streinu, Pseudo-visibility graphs in pseudo-polygons: Part II, Technical Report 042, Dept. Comput. Sci., Smith College, Northampton, MA, April 1996.
[12] M. Pocchiola, G. Vegter, The visibility complex, Internat. J. Comput. Geom. Appl. 6 (3) (1996) 279–308.