COMPUTING VISIBILITY INFORMATION
IN AN INACCURATE SIMPLE POLYGON

LEIZHEN CAI
Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong
Email: leizhen@cs.cuhk.edu.hk

and

J. MARK KEIL
Department of Computer Science, University of Saskatchewan
Saskatoon, Saskatchewan, Canada S7N 5A9
Email: keil@cs.usask.ca

Received 1 February 1994
Revised 20 February 1996
Communicated by G. T. Toussaint

ABSTRACT

This paper is concerned with the problem of capturing meaningful and useful visibility information inside a simple polygon given only an inaccurate representation of the vertices of the polygon. We introduce a notion of a visibility skeleton of an inaccurate representation of a simple polygon. We show that in most cases the visibility skeleton of a representation can be computed efficiently; furthermore, the visibility skeleton can be used to plan a collision-free path inside the polygon whose length approximates the length of a shortest such path to within a constant factor (independent of the number of vertices in the polygon).

Keywords: visibility graph, collision-free path, distance approximation, linear algorithm

1. Introduction

Given a simple polygon \( P \) in the plane, the visibility graph \( G \) of \( P \) has the vertices of \( P \) as its vertex set such that two vertices \( p \) and \( q \) are adjacent in \( G \) iff they see each other in \( P \), i.e., the line segment between \( p \) and \( q \) does not intersect the exterior of \( P \). Such a visibility graph has been well studied (see Ref. [9] for a survey) and can be used to plan shortest collision-free paths inside \( P \) between its vertices.

In this paper we assume that a simple polygon \( P = (p_1, p_2, \ldots, p_n) \) is specified by the coordinates of its vertices. Due to various reasons, such as measurement errors and round-off errors, these coordinates are normally only approximations of their true values. In fact, it may be impossible to obtain precise values of coordinates
in a dynamic environment where vertices may shift positions from time to time, or in an unfamiliar environment where only rough positions of vertices are available. Therefore an input for $P$ is usually an approximation of $P$, i.e., a polygon $P' = (p'_1, p'_2, \ldots, p'_n)$ in which the coordinates of $p'_i$ may differ from those of $p_i$. In general, we may assume that there is a real number $\epsilon \geq 0$ such that each $p'_i, 1 \leq i \leq n$, is within distance $\epsilon$ of vertex $p_i$. Thus an input for $P$ can be regarded as a pair $(P', \epsilon)$, called a representation, that satisfies $||p_i, p'_i|| \leq \epsilon$ for every $1 \leq i \leq n$, where $||p_i, p'_i||$ denotes the Euclidean distance between $p_i$ and $p'_i$. We call $P'$ an $\epsilon$-approximation polygon of $P$.

Clearly, when an $\epsilon$-approximation polygon $P'$ is used as the input for $P$, the visibility graph of $P$ may not truly reflect the visibility information of $P$. This raises the question whether it is possible to obtain meaningful and useful visibility information of $P$ from its approximation polygon $P'$. To answer this question, we introduce the following notion: Given a representation $(P', \epsilon)$ of $P$, the visibility skeleton $G$ of $(P', \epsilon)$ is a graph whose vertices correspond to vertices of $P'$ such that two vertices are adjacent in $G$ if their corresponding vertices see each other in any $\epsilon$-approximation simple polygon $P''$ of $P'$. By this definition, it is clear that any two adjacent vertices in $G$ are guaranteed to see each other in $P$ since $P$ is also an $\epsilon$-approximation simple polygon of $P'$. We will show in this paper that the visibility skeleton can be constructed efficiently in most cases. Furthermore, we will give a potential application of a visibility skeleton in planning a collision-free path inside $P$ whose length approximates the length of a shortest such path to within a constant factor (independent of the number of vertices in $P$).

The idea and input model in this paper bear some resemblance to those in several general frameworks concerning the implementation of geometric algorithms robust to inaccurate numeric input and/or finite precision computation. However, our emphasis is on the information itself rather than the robustness of the computation of the information. We want the visibility information obtained from a simple polygon to remain meaningful when the vertices of the polygon are arbitrarily perturbed within an $\epsilon$-disk. Therefore, we may say that such information is robust to inaccurate input.

Throughout the paper, we use $\beta$ to denote the **minimum vertex-to-edge distance** of $P$, i.e., the minimum distance in $P$ between any vertex and any edge not incident with that vertex, and $\gamma$ to denote the **minimum vertex-to-vertex distance** of $P$, i.e., the minimum distance in $P$ between any two distinct vertices. We should note that $\beta \leq \gamma$. For a point $p$ and a number $r \geq 0$, we use $D_r(p)$ and $D_r[p]$ respectively to denote the open and closed $r$-disks centred at $p$, i.e., $D_r(p) = \{ x : ||p, x|| < r \}$ and $D_r[p] = \{ x : ||p, x|| \leq r \}$. Likewise, for any two points $p$ and $q$, we use $L(p, q)$ and $L[p, q]$ respectively to denote the open and closed line segments between $p$ and $q$. For any number $\delta \geq 0$, we define the $\delta$-strip of two points $p$ and $q$, denoted $S_\delta(p, q)$, to be the region covered by $D_\delta(p)$ when its centre moves along the line segment $L[p, q]$ from $p$ to $q$ (see Fig. 1), i.e., $S_\delta(p, q) = \{ x : ||x, L[p, q]|| < \delta \}$; and similarly the closed $\delta$-strip $S_\delta[p, q] = \{ x : ||x, L[p, q]|| \leq \delta \}$. For convenience, we hereafter will use $P'$ to denote an actual polygon and $(P, \epsilon)$
to denote its representation, since the representation is usually the subject under consideration.

The rest of the paper is organized as follows: We make a reasonable assumption in Section 2 about a representation \((P, \varepsilon)\) by introducing the notion of a simple representation. In Section 3 we present an algorithm for computing the visibility skeleton of a simple representation \((P, \varepsilon)\) whose running time is proportional to the size of the visibility graph of \(P\). We demonstrate in Section 4 that for most simple representations \((P, \varepsilon)\) of \(P'\), the visibility skeleton can be used to find a collision-free path inside \(P'\) whose length is bounded by a constant, which depends only on the ratio between \(\varepsilon\) and the minimum vertex-to-vertex distance in \(P\), times the length of an actual shortest collision-free path inside \(P'\). (Proofs for the results in Section 4 are given in the appendix.) We give a brief summary and some further research suggestions in Section 5.

2. Simple Representations

In order for the notion of visibility skeleton to be useful, we require that a representation \((P, \varepsilon)\) satisfy the property that any \(\varepsilon\)-approximation polygon \(P'\) of \(P\) is a simple polygon; such a representation will be called a simple representation. Henceforth in the paper, unless specified otherwise, \((P, \varepsilon)\) is always a simple representation. The following theorem characterizes a simple representation in terms of the minimum vertex-to-edge distance in \(P\).

**Theorem 2.1** A representation \((P, \varepsilon)\) is a simple representation iff the minimum vertex-to-edge distance \(\beta\) in \(P\) is greater than \(2\varepsilon\).

**Proof.** Suppose \(\beta > 2\varepsilon\). Let \(P'\) be an arbitrary \(\varepsilon\)-approximation polygon of \(P\). We want to show that \(P'\) is a simple polygon. To do so, we only need to show that any two nonadjacent edges of \(P'\) do not intersect and that any two adjacent edges of \(P'\) have only one point in common. Let \(p'q', x'y'\) be two arbitrary edges of \(P'\), and \(pq, xy\), respectively, be their corresponding edges of \(P\). Consider the two closed \(\varepsilon\)-strips \(S_\varepsilon[p, q]\) and \(S_\varepsilon[x, y]\). It is easy to see that \(L[p', q'] \subseteq S_\varepsilon[p, q]\) and \(L[x', y'] \subseteq S_\varepsilon[x, y]\) as \(P'\) is an \(\varepsilon\)-approximation polygon of \(P\). Thus if \(pq\) and \(xy\) are nonadjacent edges then by the assumption that \(\beta > 2\varepsilon\) we easily deduce \(||L[p, q], L[x, y]|| > 2\varepsilon\), which implies \(S_\varepsilon[p, q] \cap S_\varepsilon[x, y] = \emptyset\). Therefore \(L[p', q'] \cap L[x', y'] = \emptyset\) in this case. Otherwise, \(pq\) and \(xy\) share a vertex, say \(p = x\). Since \(||y, L[p, q]|| > 2\varepsilon\), we have \(D_\varepsilon[y] \cap S_\varepsilon[p, q] = \emptyset\), implying that \(y'\) is not on the line segment \(L[p', q']\). Similarly, we can show that \(q'\) is not on the line segment \(L[x', y']\). Hence \(y'\) is the only point in \(P'\) shared by \(L[p', q']\) and \(L[x', y']\). This confirms that \(P'\) is a simple polygon.

Conversely suppose \(\beta \leq 2\varepsilon\). Then in \(P\) there is a vertex \(p\) and an edge \(xy\) which is not incident with \(p\) such that \(||p, L[x, y]|| \leq 2\varepsilon\). Obviously we can construct an
\(\epsilon\)-approximation polygon \(P'\) of \(P\) by shifting \(x, y, p\) to \(x', y', y'\), respectively, by at most distance \(\epsilon\) so that \(y'\) is on the line segment \(L[x', y']\). Then \(P'\) is not a simple polygon. This completes the proof.

We remark that this simple representation assumption is weaker than the minimum feature separation assumption of Hoffman, Hopcroft and Karasick\(^7\) where \((P, \epsilon)\) is required to satisfy \(\beta \geq 3\epsilon\) and other conditions.

Let \(\text{Int}(P)\) and \(\text{Ext}(P)\) denote the interior and exterior of polygon \(P\) respectively. Let \(\text{Int}_\epsilon(P)\) be the set of points in the interior of \(P\) that are at least distance \(\epsilon\) away from \(P\), i.e.,

\[
\text{Int}_\epsilon(P) = \{x : x \in \text{Int}(P) \text{ and } ||x, P|| > \epsilon\}.
\]

And similarly let \(\text{Ext}_\epsilon(P)\) be the set of points in the exterior of \(P\) that are at least distance \(\epsilon\) away from \(P\), i.e.,

\[
\text{Ext}_\epsilon(P) = \{x : x \in \text{Ext}(P) \text{ and } ||x, P|| > \epsilon\}.
\]

Let \(B_\epsilon(P) = \{x : ||x, P|| \leq \epsilon\}\). If we imagine that the centre of a closed \(\epsilon\)-disk moves along \(P\), then \(B_\epsilon(P)\) is the region covered by this \(\epsilon\)-disk, and \(\text{Int}_\epsilon(P)\) and \(\text{Ext}_\epsilon(P)\), respectively, are the finite and infinite regions of the plane after removing \(B_\epsilon(P)\). See Fig. 2 for an illustration.

![Fig. 2. \(\text{Int}_\epsilon(P), \text{Ext}_\epsilon(P)\) and \(B_\epsilon(P)\) of \(P\)](image)

Since for a simple representation \((P, \epsilon)\), \(\beta > 2\epsilon\) and every \(\epsilon\)-approximation polygon \(P'\) of \(P\) is a simple polygon, it is easy to see the following facts:

**Proposition 2.2** Let \((P, \epsilon)\) be a simple representation. Then

(a) both \(\text{Int}_\epsilon(P)\) and \(\text{Ext}_\epsilon(P)\) are nonempty and connected; and

(b) for any \(\epsilon\)-approximation polygon \(P'\) of \(P\), \(\text{Int}_\epsilon(P) \subseteq \text{Int}(P')\), \(\text{Ext}_\epsilon(P) \subseteq \text{Ext}(P')\), and \(P' \subseteq B_\epsilon(P)\).

3. An Efficient Algorithm

How can we efficiently compute the visibility skeleton \(G_{VS}\) of a simple representation \((P, \epsilon)\)? From the definition of \(G_{VS}\), it is not even clear whether there is an algorithm for computing it, since it is defined in terms of an infinite number of
polygons. To answer this question, we first introduce a notion of δ-visibility graphs and then use it to compute \( G_{\delta} \).

Let \( P \) be a simple polygon and \( \delta \) be a nonnegative number. Then two vertices \( p \) and \( q \) of \( P \) are δ-visible \(^a\) iff they either are the endpoints of a side of \( P \) or see each other in \( P \) and there is no other vertex inside the δ-strip \( S_\delta(p,q) \). The δ-visibility graph \( G_\delta \) of \( P \) is a graph whose vertices correspond to vertices of \( P \) and where two vertices are adjacent iff their corresponding vertices in \( P \) are δ-visible. Fig. 3 shows an example of a δ-visibility graph, where each dashed circle centred at a vertex indicates the δ-disk of that vertex and two dashed strips depict the δ-strips of \( \{1,4\} \) and \( \{2,5\} \) respectively.

![Fig. 3. A δ-visibility graph of \( P \)](image)

A δ-visibility graph \( G_\delta \) of \( P \) is clearly a spanning subgraph of the visibility graph \( G \) of \( P \) and becomes \( G \) when \( \delta = 0 \). Therefore the visibility information captured by \( G_\delta \) approximates the visibility information contained in \( G \). In addition, one can control the quality of the approximation by changing the value of \( \delta \): decrease \( \delta \) to get a better approximation, and increase \( \delta \) to get a rougher approximation but a sparser graph. Because of this, δ-visibility graphs seem to be an interesting subject in its own right. In any rate, they are quite useful in dealing with visibility skeletons as indicated by the following fundamental relation between the visibility skeleton of a simple representation \((P,\epsilon)\) and the 2ε-visibility graph of \( P \).

**Theorem 3.1** A graph is the visibility skeleton of a simple representation \((P,\epsilon)\) iff it is the 2ε-visibility graph of \( P \).

**Proof.** We only need to show that for any two nonadjacent vertices \( p \) and \( q \) of \( P \), \( pq \) is an edge of the 2ε-visibility graph \( G_{2\epsilon} \) of \( P \) iff for every \( \epsilon \)-approximation simple polygon \( P' \) of \( P \), \( p'q' \) is an edge of the visibility graph \( G' \) of \( P' \).

Suppose that \( pq \) is an edge of \( G_{2\epsilon} \). Then in polygon \( P \), \( p \) sees \( q \) and the 2ε-strip \( S_{2\epsilon}(p,q) \) contains no vertex other than \( p \) and \( q \), implying \( ||u,L(p,q)|| \geq 2\epsilon \) for any other vertex \( u \) of \( P \). Since \( p \) sees \( q \), \( pq \) divides \( P \) into two simple polygons \( P_1 \) and \( P_2 \). If \( ||u,L(p,q)|| \geq 2\epsilon \) then since \( \beta > 2\epsilon \) (by Theorem 2.1), we know that both \((P_1,\epsilon)\) and \((P_2,\epsilon)\) are simple representations. Thus in any \( \epsilon \)-approximation polygon \( P' \) of \( P \), the two polygons \( P'_1 \) and \( P'_2 \), which correspond to \( P_1 \) and \( P_2 \) respectively, are simple polygons. If we regard the construction of \( P' \) as a continuous transformation

\(^a\)A variation of δ-visibility is to disallow other edges of \( P \) in \( S_\delta(p,q) \) as well. Such a variation is equivalent to the one here when \( \delta \) is less than the minimum vertex-to-edge distance \( \beta \) of \( P \) and may be useful in planning collision-free paths inside \( P \) for an object with diameter less than \( \delta \).
from $P$, then we easily see that no edge of $P'$ ever crosses $p'q'$ in the transformation. This implies that $L(p', q')$ belongs to the interior of $P'$ and intersects no edge of $P'$. Thus $p'$ sees $q'$ in $P'$ in this case. Otherwise there is some vertex $v$ of $P$ for which $\|v, L(p, q)\| = 2\epsilon$. In this case it is easy to show, similar to the proof of Theorem 2.1, that although $P'_1$ ($P'_2$, respectively) is not guaranteed to be a simple polygon (vertices may lie on edges to which they are not incident), the open line segments corresponding to any two nonadjacent edges of $P'_1$ ($P'_2$, respectively) do not intersect. This implies that $L(p', q')$ does not intersect the exterior of $P'$ and thus $p'$ also sees $q'$ in this case.

Conversely, suppose that $pq$ is not an edge of $G_{2\epsilon}$. Then there is a vertex $r$ inside the $2\epsilon$-strip $S_{2\epsilon}(p, q)$. Construct an $\epsilon$-approximation polygon $P'$ of $P$ by shifting vertices $p, q, r$ to $p', q', r'$, respectively, such that $r'$ is as close to $L[p, q]$ as possible and $L[p', q']$ is as close to $r$ as possible (see Fig. 4 for an example). Since $(P, \epsilon)$ is a

![Fig. 4.](image)

simple representation, it follows from Theorem 2.1 that $P'$ is a simple polygon. It is easy to see that $p'$ does not see $q'$ in $P'$; and thus $p'q'$ is not an edge of $G'$. □

Theorem 3.1 enables us to construct the visibility skeleton of a simple representation $(P, \epsilon)$ by computing the $2\epsilon$-visibility graph of $P$. Indeed, for any $\delta$ smaller than the minimum vertex-to-edge distance in $P$, the $\delta$-visibility graph $G_\delta$ of $P$ can be constructed efficiently from the visibility graph $G$ of $P$.

**Theorem 3.2** Given a simple polygon $P$ and a nonnegative number $\delta$ which is less than the minimum vertex-to-edge distance $\beta$ of $P$, the $\delta$-visibility graph $G_\delta$ of $P$ can be constructed from the visibility graph $G$ of $P$ in $O(m + n)$ time, where $m$ is the number of edges in $G$ and $n$ is the number of vertices in $P$.

Because the visibility graph $G$ of $P$ can be constructed in time $O(m + n)$ by combining the output-sensitive visibility graph algorithm of Hershberger and the linear time triangulation algorithm of Chazelle, Theorem 3.2 clearly implies the following result:

**Corollary 3.3** Given a simple polygon $P$ and a nonnegative number $\delta$ which is less than the minimum vertex-to-edge distance $\beta$ of $P$, the $\delta$-visibility graph $G_\delta$ of $P$ can be constructed from $P$ in $O(m + n)$ time, where $m$ is the number of edges in the visibility graph $G$ of $P$ and $n$ is the number of vertices in $P$.

By Theorem 2.1, any simple representation $(P, \epsilon)$ satisfies $\beta > 2\epsilon$. Therefore
Corollary 3.3 implies that the 2ε-visibility graph of P can be constructed in O(m+n) time, which, together with Theorem 3.1, gives us the following result:

**Corollary 3.4** The visibility skeleton of any simple representation \((P, ε)\) can be constructed from P in \(O(m+n)\) time, where m is the number of edges in the visibility graph G of P and n is the number of vertices in P.

In the remainder of this section, we will prove Theorem 3.2 by presenting an \(O(m+n)\) algorithm for computing a \(δ\)-visibility graph \(G_δ\) of a simple polygon P from P and its visibility graph G. Hereafter, we assume that the minimum vertex-to-edge distance \(β\) of P is greater than \(δ\).

As noted before, any \(δ\)-visibility graph \(G_δ\) of P is a spanning subgraph of the visibility graph G of P. Therefore we can construct \(G_δ\) from G by deleting edges. To facilitate efficient construction of \(G_δ\), we consider the edges incident to a vertex collectively. Let \(v\) be an arbitrary vertex of G. Without loss of generality, we may assume that \(v_1, v_2, \ldots, v_{deg(v)}\), where \(deg(v)\) is the degree of \(v\) in G, are the neighbors of \(v\) in G that are arranged in clockwise order in P with \(vv_1\) and \(vv_{deg(v)}\) being the two edges of P incident with \(v\). Let \(S^v_i\) denote the \(δ\)-strip \(S^v_δ(v, v_i)\). Then the line passing through \(v\) and \(v_i\), which is oriented from \(v\) to \(v_i\), divides \(S^v_i\) into two disjoint strips (see Fig. 5): the one on the right hand side of the line (including the portion of the line inside \(S^v_i\)) is the **right \(δ\)-strip** and denoted by \(R^v_i\), and the other one is the **left \(δ\)-strip** and denoted by \(L^v_i\).

![Fig. 5. The right and left \(δ\)-strips of \(S^v_i\)](image)

For an edge \(pq\) of G, its \(δ\)-strip \(S^v_δ(p, q)\) in P is **encroached** if there is a third vertex inside \(S^v_δ(p, q)\); then edge \(pq\) is also said to be encroached. We can define encroached right (left, respectively) \(δ\)-strips in a similar fashion. It is obvious that a \(δ\)-strip \(S^v_i\) is encroached in P iff either its right \(δ\)-strip \(R^v_i\) or left \(δ\)-strip \(L^v_i\) is encroached in P. Thus, we may consider right \(δ\)-strips and left \(δ\)-strips separately. To simplify the presentation of our algorithm, we will only consider edges incident with vertex \(v\) and divide the algorithm into the following three major steps:

1. **Step 1.** Find all encroached right \(δ\)-strips in \(\{R^v_1, \ldots, R^v_{deg(v)}\}\);
2. **Step 2.** Find all encroached left \(δ\)-strips in \(\{L^v_1, \ldots, L^v_{deg(v)}\}\);
3. **Step 3.** Delete all encroached edges in \(\{vv_2, \ldots, vv_{deg(v)}\}\) from G.

To carry out these three steps efficiently, we would like to execute each of the above three steps in time proportional to the degree \(deg(v)\) of \(v\). This can be easily done for Step 3. Because of the symmetry between Step 1 and Step 2, we will only
discuss Step 1 by giving an \( O(\deg(v)) \) time procedure. First we have the following observation:

**Lemma 3.5** If \( R_i^v, 2 \leq i \leq \deg(v) - 1 \), is encroached then there is a vertex \( u \) inside \( R_i^v \) that sees both \( v \) and \( v_i \).

**Proof.** If \( R_i^v \) is encroached then \( R_i^v \) contains vertices other than \( v \) and \( v_i \). Let \( u \) be a vertex inside \( R_i^v \) that is closest to the line segment \( L[v, v_i] \). Consider the triangular region \( T \) enclosed by line segments \( L[u, v], L[v, v_i] \) and \( L[v_i, u] \). Then by the choice of \( u \), \( T \) contains no other vertices of \( P \). Suppose that there is a side \( S \) of \( P \) that intersects \( T \). Since \( v \) sees \( v_i \) in \( P \), \( S \) must lie in between \( u \) and \( L[v, v_i] \) (see Fig. 6). This would imply that \(|u, S| < \delta\), contradicting \( \delta < \beta \). Therefore \( T \) does not intersect any side of \( P \), and since \( L(v, v_i) \in \text{Int}(P) \), \( T \in \text{Int}(P) \). Hence \( u \) sees both \( v \) and \( v_i \). \( \square \)

![Fig. 6.](image)

In fact, Lemma 3.5 can be strengthened to the following result, which allows us to find encroached edges in an orderly fashion and thus to save time.

**Lemma 3.6** A right \( \delta \)-strip \( R_i^v \), \( 2 \leq i \leq \deg(v) - 1 \), is encroached iff there is a vertex \( v_j, i < j \leq \deg(v) \), inside \( R_i^v \).

**Proof.** The sufficiency is trivial. We only need to show the necessity. Suppose that \( R_i^v \) is encroached. Then by Lemma 3.5, there is a vertex \( u \) inside \( R_i^v \) that sees \( v_i \). Therefore \( u = v_j \) for some \( 1 \leq j \leq \deg(v) \). If \( j < i \) then the line segment \( L(v, v_{d\deg(v)}) \) lies in between \( L[v, v_i] \) and \( v_j \) (see Fig. 7). This would imply that \( ||v_j, L[v, v_{d\deg(v)}]|| < \delta \), contradicting the assumption that \( \beta > \delta \). Hence \( j > i \). \( \square \)

![Fig. 7.](image)

In light of the above lemma, Step 1 proceeds as a clockwise angular sweep through the neighbourhood of \( v \). After stage \( i \), the subset of right \( \delta \)-strips \( \{ R_j^v : 2 \leq j < i \} \) that are not encroached by any vertex in \( \{v_3, \ldots, v_i\} \) are kept in a
stack RS. At stage i, if vertex \( v_i \) does not encroach \( R^v_{i-1} \) then \( R^v_{i-1} \) is pushed onto the stack RS; otherwise the right \( \delta \)-strips in RS are popped out until \( v_i \) no longer encroaches the right \( \delta \)-strip on the top of the stack. A PASCAL-like description of the whole algorithm is as follows:

**Algorithm:** \( \delta \)-VISIBILITY GRAPH

**Input:** A simple polygon \( P \), its visibility graph \( G = (V, E) \), and a number \( \delta \) that is less than the minimum vertex-to-edge distance \( \beta \) of \( P \).

**Output:** The \( \delta \)-visibility graph \( G_\delta = (V, E_\delta) \) of \( P \).

**begin**

1. \( E_\delta = E; \)
2. for each vertex \( v \) of \( G \) do
3. \{Step 1. Find all encroached right \( \delta \)-strips in \( \{R^v_2, \ldots, R^v_{\text{deg}(v)-1}\} \}\}
4. \( \text{right} = R^v_2; \) RS = \( \emptyset \);
5. for \( i = 3 \) to \( \text{deg}(v) \) do
6. \( \text{while} \ v_i \in \text{right} \) do
7. \( \text{if} \ RS \neq \emptyset \)
8. \( \text{then} \ \text{pop(right, RS)} \)
9. \( \text{else goto right} \)
10. \( \text{end if}; \)
11. \( \text{end while}; \)
12. \( \text{push(right, RS)}; \)
13. \( \text{R: right} = R^v_i; \)
14. end for;
15. \{Step 2. Find all encroached left \( \delta \)-strips in \( \{L^u_2, \ldots, L^u_{\text{deg}(v)-1}\} \}\}
16. \( \text{left} = L^u_{\text{deg}(v)-1}; \) LS = \( \emptyset \);    
17. for \( i = \text{deg}(v) - 2 \) downto \( 1 \) do
18. \( \text{while} \ v_i \in \text{left} \) do
19. \( \text{if} \ LS \neq \emptyset \)
20. \( \text{then} \ \text{pop(left, LS)} \)
21. \( \text{else goto left} \)
22. \( \text{end if}; \)
23. \( \text{end while}; \)
24. \( \text{push(left, LS)}; \)
25. \( \text{L: left} = L^u_i; \)
26. end for;
27. \{Step 3. Delete all encroached edges in \( \{vv_2, \ldots, vv_{\text{deg}(v)-1}\} \) from \( G \}. \)
28. Mark each vertex that is contained in both RS and LS
29. for \( i = 2 \) to \( \text{deg}(v) - 1 \) do
30. \( \text{delete edge} vv_i \) from \( E_\delta \) whenever vertex \( v_i \) is not marked
31. end for
32. end for;

**end algorithm.**
We now analyze the above algorithm. For this purpose, we use "after stage $i$" to mean the moment after the completion of the $(i - 2)$-th iteration of the “for” loop (lines 5-14) of the algorithm. To prove the correctness of the algorithm, we first notice the following invariant:

**Lemma 3.7** After stage $i$, $3 \leq i \leq \text{deg}(v)$, an arbitrary right $\delta$-strip $R^v_j$, $2 \leq j \leq i - 1$, is contained in the stack $RS$ if it is not encroached by any vertex in $\{v_1, \ldots, v_i\}$.

**Proof.** We use induction on $i$. It is easy to verify the lemma for $i = 3$. Assume that the lemma is true after stage $i$ and consider stage $i + 1$. Let $N_i = \{v_1, \ldots, v_i\}$.

Suppose that $R^v_j$, $2 \leq j \leq i$, is not encroached by any vertex in $N_i$. If $j = i$ then right = $R^v_j$ after stage $i$. Since $v_{i+1} \notin R^v_j$, $R^v_j$ is put onto the stack $RS$ at line 12 of stage $i + 1$. Otherwise $j \leq i - 1$ and by the induction hypothesis, $R^v_j$ is contained in $RS$ after stage $i$. If there is an unencroached right $\delta$-strip lying above $R^v_j$ in $RS$, then at stage $i + 1$ $RS$ stops popping out elements after this right $\delta$-strip is popped out. Therefore $R^v_j$ remains in $RS$ after stage $i + 1$. Otherwise, every right $\delta$-strip lying above $R^v_j$ in $RS$ is encroached by $v_{i+1}$ and thus will be popped out from $RS$ at stage $i + 1$. After the right $\delta$-strip immediately above $R^v_j$ is examined, $R^v_j$ will first be popped out of $RS$ at line 8 and then put back onto $RS$ at line 12 since $v_{i+1} \notin R^v_j$. Therefore $R^v_j$ is contained in $RS$ after stage $i + 1$.

Conversely, suppose that $R^v_j$, $2 \leq j \leq i$, is encroached by some vertex of $N_{i+1}$. If $j = i$ then again right = $R^v_j$ after stage $i$ and by Lemma 3.6 $R^v_j$ is encroached by $v_{i+1}$. Also note that $R^v_j$ is not in $RS$ after stage $i$. If $RS$ is empty after stage $i$ then stage $i + 1$ terminates by executing lines 9 and 13 as $v_{i+1} \in R^v_j$, which will not put $R^v_j$ onto $RS$; otherwise, the value of right is changed to the top element of $RS$ at line 8 and thus $R^v_j$ will not be put onto $RS$ at stage $i + 1$. The remaining case is $j \leq i - 1$. If $R^v_j$ is encroached by some vertex in $N_i$ then by the induction hypothesis $R^v_j$ is not contained in $RS$ after stage $i$. Since right = $R^v_k \neq R^v_j$ after stage $i$, $R^v_j$ will not be put onto $RS$ at stage $i + 1$. Otherwise, $R^v_k$ is encroached by $v_{i+1}$. Let $R^v_k$ be an arbitrary right $\delta$-strip in $RS$ that lies above $R^v_j$. Then $j < k \leq i$. Since $v_k$ does not encroach $R^v_j$ and each right $\delta$-strip has the same width, it is easy to see that $v_{i+1} \in R^v_k$ (see Fig. 8). Therefore every $\delta$-strip in $RS$ that lies above $R^v_j$,

![Diagram](image)

**Fig. 8.**

as well as $R^v_j$, will be successively removed from $RS$ at stage $i + 1$. Therefore $R^v_j$ is not in $RS$ after stage $i + 1$. This completes the proof. \(\square\)
From Lemma 3.7, it is obvious that a right $\delta$-strip $R_j^v$, $2 \leq j \leq \deg(v) - 1$, is contained in stack RS after the termination of Step 1 iff $R_j^v$ is not encroached in $P$. This implies the correctness of Step 1. The correctness of Step 2 can be shown in a similar manner, and the correctness of Step 3 is obvious. Therefore the algorithm correctly computes $G_\delta$ from $G$.

In terms of the complexity, we first note that the neighbourhoods of all vertices can be sorted collectively with respect to their orderings in $P$ in $O(m + n)$ time by standard techniques. In Step 1, checking whether a vertex is inside a right $\delta$-strip takes $O(1)$ time. Furthermore, the time spent in each iteration of the “while” loop (lines 6-11) is proportional to the number of encroached right $\delta$-strips removed from the stack (the last one popped out from RS is not considered to be removed) plus one. Since no right $\delta$-strip will be re-examined once it is removed, it is clear that the total time for Step 1 is $O(\deg(v))$. Similarly, Step 2 takes $O(\deg(v))$ time, and Step 3 can be easily implemented in $O(\deg(v))$ time. Therefore our algorithm takes $O(m + n)$ time. This completes the proof of Theorem 3.2.

We remark that it may be possible to modify the existing output-sensitive visibility graph algorithms to derive an output-sensitive algorithm for constructing $G_\delta$ and thus the visibility skeleton $G_{VS}$ of $(P, \epsilon)$. We leave this as an open problem for the reader to ponder.

4. Distance Approximation

The visibility graph $G$ of a simple polygon $P$ is a useful structure for planning collision-free paths inside $P$ between its vertices. Unfortunately, this method fails when the input of $P$ is inaccurate. Suppose that $(P, \epsilon)$ is a simple representation of a simple polygon $P'$. If $\epsilon > \frac{1}{2}$ then for some vertices $p$ and $q$ of $P$, a $(p, q)$-path in $G$ may not correspond to a collision-free $(p', q')$-path inside $P'$ at all, which essentially prevents us from using $G$ to plan collision-free paths inside $P'$. On the other hand, any $(p, q)$-path in the visibility skeleton $G_{VS}$ of $(P, \epsilon)$ corresponds to a collision-free $(p', q')$-path inside $P'$. Therefore we can first find a shortest $(p, q)$-path $Q$ in $G_{VS}$ and then take its corresponding $(p', q')$-path $Q'$ in $P'$ to plan a collision-free $(p', q')$-path. Such a strategy is feasible because at the corners the robot normally must slow down to turn and thus can deal with inaccuracies at the vertices with tactile sensors. Of course, for this approach to be effective, the length of $Q'$ should be close to the length of a shortest collision-free $(p', q')$-path inside $P'$. Indeed, as shown by the following two theorems, whose proofs are given in the appendix, in most cases the length of $Q'$ is at most a constant times the length of the shortest collision-free $(p', q')$-path inside $P'$.

**Theorem 4.1** Let $(P, \epsilon)$ be a simple representation of a simple polygon $P'$ with $\beta > 4\epsilon$ and $\gamma > (4\sqrt{3}+6)\epsilon/3 \approx 4.31\epsilon$, where $\beta$ and $\gamma$, respectively, are the minimum vertex-to-edge and vertex-to-vertex distances in $P$. For any two vertices $p'$ and $q'$ of $P'$, let $d(p', q')$ be the length of a shortest collision-free $(p', q')$-path inside $P'$, and let $D_{VS}(p', q')$ be the length of the collision-free $(p', q')$-path inside $P'$ that corresponds
to a shortest \((p, q)\)-path in the visibility skeleton of \((P, \epsilon)\). Then

\[
\frac{D_{VS}(p',q')}{d(p',q')} < \frac{1-4\rho^2}{(\sqrt{1-4\rho^2}-\rho)(2\rho+2\sqrt{1-4\rho}-1)} \quad \text{where } \rho = \epsilon/\gamma.
\]

The parameter \(\rho\) in the above theorem measures the relative precision of the input data of \((P, \epsilon)\), and its reciprocal indicates the separation of vertices in \(P\) with respect to \(\epsilon\). The theorem shows that whenever \(\beta > 4\epsilon\) and \(\rho < (2\sqrt{3}-3)/2 \approx 0.232\), \(D_{VS}(p',q')\) is within a constant factor times \(d(p',q')\). In many applications, one could expect that \(\rho\) is quite small (perhaps \(\leq 0.01\)). Some typical values of the upper bound in the above theorem are shown in Table 1. We remark that the upper bound in the theorem is not tight.

<table>
<thead>
<tr>
<th>(\rho = \epsilon/\gamma)</th>
<th>0.232</th>
<th>0.125</th>
<th>0.10</th>
<th>0.05</th>
<th>0.01</th>
<th>0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>upper bound</td>
<td>10.72</td>
<td>3.98</td>
<td>2.35</td>
<td>1.46</td>
<td>1.18</td>
<td>1.03</td>
</tr>
</tbody>
</table>

Table 1. Typical values of the upper bound in Theorem 4.1

As stated in Theorem 3.1, the visibility skeleton \(G_{VS}\) of a simple representation \((P, \epsilon)\) is the same as the \(2\epsilon\)-visibility graph \(G_{2\epsilon}\) of \(P\). Although the ratio in Theorem 4.1 is affected by the perturbation of the vertices of \(P\), it is largely determined by the ratio between the length of a shortest \((p, q)\)-path in \(G_{2\epsilon}\) and that in \(G\) since \(\epsilon\) is normally quite small compared to vertex-to-vertex distances in \(P\). From this perspective, Theorem 4.1 can be viewed as an implication that \(G_{2\epsilon}\) approximates \(G\) well in terms of distances. Indeed, most \(\delta\)-visibility graphs behave well in such approximations.

**Theorem 4.2** Let \(P\) be a simple polygon and \(\delta\) be a real number that satisfies \(0 < \delta < \min\{\beta, \sqrt{3}/2\}\), where \(\beta\) and \(\gamma\) are the minimum vertex-to-edge and vertex-to-vertex distances in \(P\) respectively. For any two vertices \(p\) and \(q\) of \(P\), let \(d(p, q)\) and \(d_{\delta}(p, q)\) denote the \((p, q)\)-distances in the visibility graph \(G\) and the \(\delta\)-visibility graph \(G_{\delta}\) of \(P\) respectively. Then

\[
\frac{d_{\delta}(p, q)}{d(p, q)} < \frac{1}{2\sqrt{1-\tau^2}-1} \quad \text{where } \tau = \delta/\gamma.
\]

The above theorem confirms that a \(\delta\)-visibility graph approximates the visibility graph of \(P\) quite well in terms of distances when \(\delta < \beta\) and \(\tau < \sqrt{3}/2 \approx 0.866\). Table 2 lists some typical values of the upper bound in Theorem 4.2.

<table>
<thead>
<tr>
<th>(\tau = \delta/\gamma)</th>
<th>0.886</th>
<th>0.55</th>
<th>0.333</th>
<th>0.2</th>
<th>0.1</th>
<th>0.05</th>
</tr>
</thead>
<tbody>
<tr>
<td>upper bound</td>
<td>5.00</td>
<td>2.04</td>
<td>1.37</td>
<td>1.04</td>
<td>1.01</td>
<td>1.003</td>
</tr>
</tbody>
</table>

Table 2. Typical values of the upper bound in Theorem 4.2

Note that for accurately known polygons, the funnel method allows for linear time determination of shortest paths in a polygon.
5. Concluding Remarks

We have considered the problem of capturing meaningful and useful visibility information of a simple polygon given only inaccurate input of its vertices. In particular, we have obtained an efficient algorithm for computing the visibility skeleton of a simple representation \((P, e)\) of a polygon \(P'\). We have also demonstrated an application of the visibility skeleton in planning collision-free paths by showing that one can use the visibility skeleton to find a collision-free path inside \(P'\) whose length closely approximates the length of such a shortest path inside \(P'\).

The notion of visibility skeletons can be extended to capture meaningful and useful visibility information in more general settings, such as the endpoint visibility graph of a set of nonintersecting line segments in the plane and the visibility graph of a set of nonintersecting polygonal obstacles in the plane. The results in this paper appear to be easily extendible to such general settings and their proofs are very similar to the proofs in the paper. On the other hand, it seems that we require new ideas to deal with such visibility problems in higher dimensions. We leave the technical details of the former and the exploration of the latter to the interested readers.

Visibility skeletons may also be useful in dealing with the tolerance of a visibility graph. Abellanas \textit{et al.}\textsuperscript{1} define the tolerance of a geometric structure to be the smallest amount which the input points can be shifted without changing the structure. We also expect that the idea behind the notion of visibility skeletons can be applied to some other geometric problems, such as intersection problems, facility location problems and range searching problems, to yield satisfactory results in dealing with inaccurate input data. Finally, we point out that in some cases it may be also possible to incorporate the error bound obtained from backward error analysis of a computation into a representation \((P, e)\) to cope with problems caused by finite precision arithmetic.

Acknowledgements

This research was supported in part by an Earmarked Research Grant from the Research Grants Council of Hong Kong and a research grant from the Natural Sciences and Engineering Research Council of Canada. The authors thank referees for valuable comments and the first author also thanks Derek Corneil and Andrzej Proskurowski for fruitful discussions.

References


Appendix A. Proofs of Theorem 4.1 and Theorem 4.2

We prove Theorem 4.1 and Theorem 4.2 in this appendix. The central part of our proofs is an upper bound on the length of a shortest \((p, q)\)-path in a \(\delta\)-visibility graph \(G_{\delta}^\square\) of \(P\). Henceforth we assume that \(P\) is a simple polygon with \(\delta < \min\{\beta, \sqrt{3\gamma}/2\}\) and will regard \(G_{\delta}^\square\) as a weighted graph where the length \(l(pq)\) of an edge \(pq\) equals the Euclidean distance \(\|p, q\|\).

First we define a couple of terms (see the illustration in Fig. A.1). The \((\gamma, \delta)\)-strip of two vertices \(p\) and \(q\), denoted \(T_{\gamma, \delta}(p, q)\), is the part of their \(\delta\)-strip \(S_{\delta}(p, q)\) that is not covered by the two \(\gamma\)-disks \(D_{\gamma}[p]\) and \(D_{\gamma}[q]\), i.e.,

\[
T_{\gamma, \delta}(p, q) = S_{\delta}(p, q) - (D_{\gamma}[p] \cup D_{\gamma}[q]).
\]

Since \(\gamma\) is the minimum vertex-to-vertex distance in \(P\), it is clear that an edge \(pq\) of \(G\) is encroached, i.e., \(S_{\delta}(p, q)\) contains other vertices, iff \(T_{\gamma, \delta}(p, q)\) contains other vertices. To measure the size of \(T_{\gamma, \delta}(p, q)\), we define the following parameter

\[
g(p, q) = \begin{cases} 
\|p, q\| - 2\sqrt{\gamma^2 - \delta^2} & \text{if } T_{\gamma, \delta}(p, q) \neq \emptyset \\
0 & \text{otherwise}
\end{cases}
\]

Fig. A.1. A \((\gamma, \delta)\)-strip (shaded area) and its gap.
which will be referred to as the gap of $T_{\gamma, \delta}(p, q)$. For convenience, we will use \( \zeta \) to denote \( \sqrt{\gamma^2 - \delta^2} \) hereafter. It is important to note that $T_{\gamma, \delta}(p, q) \neq \emptyset$ iff \( g(p, q) > 0 \), i.e., \( ||p, q|| > 2\zeta \).

By the above remarks and a proof similar to that of Lemma 3.5, we notice the following result:

**Lemma A.1** If an edge $pq$ of $G$ is encroached, then \( ||p, q|| > 2\zeta \) and there is at least one vertex $r$ of $P$ that lies inside the \( (\gamma, \delta) \)-strip $T_{\gamma, \delta}(p, q)$ and sees both $p$ and $q$ in $P$.

Let $pq$ be an encroached edge of the visibility graph $G$ of $P$. We first consider the \( (p, q) \)-distance $d_{\delta}(p, q)$ in $G_{\delta}$. For this purpose, we will construct a special \( (p, q) \)-walk in $G_{\delta}$ and use it to get an upper bound of $d_{\delta}(p, q)$. In light of Lemma A.1, such a \( (p, q) \)-walk $W$ can be constructed by using the following procedure WALK($p, q; W$), which operates on $G$ and gradually extends \( (p, q) \)-walks in $G$.

**procedure** WALK($p, q; W$);
\{ $G$ is the input graph and $pq$ is an encroached edge of $G$ \}

**begin**

$W := pq$;

while $W$ contains encroached edges do

Arbitrarily choose an encroached edge $uv$ and a vertex $u$

that lies inside \( (\gamma, \delta) \)-strip $T_{\gamma, \delta}(u, v)$ and sees both $u$ and $v$;

$W := W[u, v] \cdot w \cdot W[v, q]$;

**end while**;

end WALK.

In the above procedure, we observe that the initial \( (p, q) \)-walk $W$ consists of a single edge $pq$, and that after each iteration, $W$ is extended to a longer one by the addition of a vertex. Therefore if $W_i$ denotes the value of $W$ after the $i$-th iteration of the “while” loop of WALK($p, q; W$), then WALK($p, q; W$) defines a sequence of \( (p, q) \)-walks $W_0, W_1, \ldots, W_i, \ldots$ in $G$. It should be noted that WALK($p, q; W$) may not terminate if $\tau = \delta/\gamma \geq \sqrt{3}/2$. Nevertheless, when WALK($p, q; W$) terminates, $W$ is a \( (p, q) \)-walk in $G_{\delta}$ since every edge of $W$ is an unencroached edge of $G$ and hence an edge of $G_{\delta}$.

**Lemma A.2** If WALK($p, q; W$) terminates then $W$ is a \( (p, q) \)-walk of $G_{\delta}$.

It is clear that upon the termination of WALK($p, q; W$), the length $l(W)$ of $W$ is an upper bound of the \( (p, q) \)-distance in $G_{\delta}$. Because of this, we first estimate the length $l(W_i)$ of $W_i$ before considering the termination condition of WALK($p, q; W$).

**Lemma A.3** For any $i \geq 1$, $l(W_i) < ||p, q|| + 2(\gamma - \zeta)i$.

**Proof.** Let $\Delta l_j = l(W_{j+1}) - l(W_j), 0 \leq j \leq i - 1$. Then

$$l(W_i) = ||p, q|| + \sum_{j=0}^{i-1} \Delta l_j.$$  \hspace{1cm} (A.1)
To determine $\Delta l_j$, we consider the construction of $W_{j+1}$ from $W_j$: in the $(j + 1)$-th iteration of the “while” loop of $WALK(p, q; W)$, $W_{j+1}$ is formed from $W_j$ by inserting vertex $w$ in between two consecutive vertices $u$ and $v$ of $W_j$, where $u, v$ and $w$ are the vertices chosen in the $(j + 1)$-th iteration. Let $d$ denote the Euclidean distance between $u$ and $v$, i.e., $d = ||u, v||$. Then

$$l(W_{j+1}) = l(W_j) - d + ||u, w|| + ||w, v||.$$  
(A.2)

Without loss of generality, we may assume that the coordinates of points $u$, $v$ and $w$ are $(0, 0)$, $(d, 0)$ and $(x, y)$, respectively. Then by the assumption $\delta < \min\{\beta, \sqrt{3}\gamma/2\}$ and the fact $w \in T_{\gamma, \delta}(u, v)$, we have $\zeta < x < d - \zeta$ and $|y| < \delta$ (see Fig. A.2).

![Diagram](image)

Fig. A.2. The coordinates of points $u$, $v$ and $w$

Now we use (A.2) to obtain

$$\Delta l_j = ||u, w|| + ||v, w|| - d$$
$$= \sqrt{x^2 + y^2} + \sqrt{(d - x)^2 + y^2} - d$$
$$< \sqrt{\zeta^2 + \delta^2} + \sqrt{(d - \zeta)^2 + \delta^2} - d$$
(since $\zeta < x < d - \zeta$ and $|y| < \delta$)
$$= \gamma + \sqrt{(d - \zeta)^2 + \delta^2} - d$$

Notice that $\sqrt{(d - \zeta)^2 + \delta^2} - d$ is a monotonically decreasing function of $d$. Also notice that since $uw$ is an encroached edge, we have $d > 2\zeta$ by Lemma A.1. Therefore

$$\sqrt{(d - \zeta)^2 + \delta^2} - d < \gamma - 2\zeta,$$

and thus

$$\Delta l_j < 2(\gamma - \zeta).$$  
(A.3)

Substituting (A.3) into (A.1), we obtain the required inequality.

To bound $l(W)$ we still need to determine the maximum number of iterations of the “while” loop of $WALK(p, q; W)$. For this purpose we introduce the following line segment deletion game (LSDG) specified by a quadruple $\langle L, a, b, c \rangle$, where $L$ is a set of a finite number of line segments and $a, b, c > 0$ are positive real numbers.

The game is played by a single player and finishes when $L$ is empty. At each move, whenever $L$ is not empty, the player may choose a line segment $x$ from $L$ and make one of the following three moves, where $l(x)$ denotes the length of $x$:

**Remove:** If $l(x) \leq a$ then remove $x$ from $L$.

**Prune:** If $l(x) \geq b$ then replace $x$ by a line segment $x'$ with $l(x') \leq l(x) - b$.

16
Split: If \( l(x) \geq c \) then replace \( x \) by two line segments \( y \) and \( z \) with \( l(y) + l(z) \leq l(x) - c \).

As we shall see shortly, the construction of \( W \) by \( \text{WALK}(p, q, W) \) can actually be regarded as a \( LSDG \); furthermore, the maximum number of moves a player can possibly make in the \( LSDG \) determines the maximum number of iterations of the “while” loop of \( \text{WALK}(p, q; W) \). First we determine the maximum number of moves in a \( LSDG \).

**Lemma A.4** Given a \( LSDG \) \((L, a, b, c)\), a player can make at most
\[
|L| + \frac{\sum_{x \in L} l(x)}{\min\{b, c/2\}}
\]

moves.

**Proof.** Let \( n \) and \( l \) denote the number and the total length, respectively, of line segments of \( L \). Let \( A, B \) and \( C \) be the number of “remove”, “prune”, and “split” moves in a \( LSDG \). Then we clearly have
\[
l \geq Bb + Cc \quad \text{(A.4)}
\]
since “prune” and “split” reduce the total length of \( L \) by at least \( b \) and \( c \) respectively.

We also note
\[
A = n + C \quad \text{(A.5)}
\]
since “split” increases the number of line segments in \( L \) by one.

If \( c \leq 2b \) then we deduce from (A.4) the following relation:
\[
l \geq Bb + \frac{C}{2}c + \frac{C}{2}c = Bb + \frac{C}{2}c + \frac{A - n}{2}c \quad \text{(substitution by (A.5))}
\]
\[
= (A + B + C)\frac{c}{2} + B(b - \frac{c}{2}) - \frac{n c}{2}.
\]
This implies
\[
A + B + C \leq \frac{2l + n + B(c - 2b)}{c} \leq n + \frac{2l}{c} \quad \text{(A.6)}
\]
since \( c - 2b \leq 0 \).

If \( c > 2b \) then we deduce from (A.4) the following relation:
\[
l > Bb + 2Cb
\]
\[
= Bb + Cb + (A - n)b \quad \text{(substitution by (A.5))}
\]
\[
= (A + B + C)b - nb.
\]
This gives us
\[
A + B + C < n + \frac{l}{b} \quad \text{(A.7)}
\]
Combining (A.6) and (A.7), we obtain the required result. \( \Box \)

We are now ready to give a sufficient condition for \( \text{WALK}(p, q; W) \) to terminate and use Lemma A.4 to obtain an upper bound on the number of iterations of its “while” loop.
Lemma A.5 If $0 < \delta < \min\{\beta, \sqrt{3}\gamma/2\}$ then the procedure $WALK(p, q; W)$ terminates after at most

$$\frac{||p, q|| \cdot \gamma}{2\zeta - \gamma}$$

iterations of the “while” loop.

Proof. Let $g(W_i)$ denote the sum of the gaps of all $(\gamma, \delta)$-strips (repetition counts) associated with $W_i$. Then from the construction of $W$ it is clear that the procedure terminates iff there exists an $i$ for which $g(W_i) = 0$. First consider $\Delta g_i = g(W_i) - g(W_{i+1})$ for each $i \geq 0$.

As in the proof of Lemma A.3, let $u, v$ and $w$ be the three vertices chosen in the $(i + 1)$-th iteration of the “while” loop of $WALK(p, q; W)$ (thus $W_{i+1} = W_i[p, u] \cdot w \cdot W_i[v, q]$). Again let $d$ be the Euclidean distance between $u$ and $v$. Furthermore, without loss of generality, we assume that the coordinates of points $u$, $v$ and $w$ are $(0, 0), (d, 0)$ and $(x, y)$, respectively (refer to Fig. A.2). Then we have

$$\Delta g_i = g(W_i) - g(W_{i+1}) = g(u, v) - (g(u, w) + g(w, v))$$

(A.8)

There are three cases (see Fig. A.3 for an illustration) to be considered depending on whether the $(\gamma, \delta)$-strips of $\{u, w\}$ and $\{w, v\}$ are empty or not:

**Case 1:** $T_{\gamma, \delta}(u, w) = \emptyset$ and $T_{\gamma, \delta}(w, v) = \emptyset$. In this case, $g(u, w) = 0$ and $g(w, v) = 0$, and thus $||u, w|| \leq 2\zeta$ and $||w, v|| \leq 2\zeta$. Therefore $\Delta g_i = g(u, v) = d - 2\zeta$. Combining with the fact that $uv$ is an encroached edge, we deduce $2\zeta < d \leq 4\zeta$, implying

$$0 < \Delta g_i \leq 2\zeta.$$  

(A.9)

**Case 2:** $T_{\gamma, \delta}(u, w) = \emptyset$ and $T_{\gamma, \delta}(w, v) \neq \emptyset$ (likewise $T_{\gamma, \delta}(u, w) \neq \emptyset$ and $T_{\gamma, \delta}(w, v) = \emptyset$). In this case, we have $g(u, w) = 0$ and $g(w, v) \neq 0$, implying $||u, w|| \leq 2\zeta$ and $||w, v|| > 2\zeta$. Therefore

$$\Delta g_i = g(u, v) - g(w, v)$$

Case 3: $T_{\gamma, \delta}(u, w) \neq \emptyset$ and $T_{\gamma, \delta}(w, v) \neq \emptyset$. In this case, we have $g(u, w) \neq 0$ and $g(w, v) \neq 0$, implying $||u, w|| \leq 2\zeta$ and $||w, v|| > 2\zeta$. Therefore

$$\Delta g_i = g(u, v) - g(w, v)$$

Fig. A.3. The three cases
\[
= (d - 2\zeta) - (||w, v|| - 2\zeta) \\
= d - ||w, v|| \\
= d - \sqrt{(d - x)^2 + y^2} \\
> d - \sqrt{(d - \zeta)^2 + \delta^2} \quad \text{(since } x > \zeta \text{ and } |y| < \delta) \\
\]

Notice that \(d - \sqrt{(d - \zeta)^2 + \delta^2}\) is a monotonically increasing function of \(d\) and that \(d > 2\zeta\) as \(uv\) is an encroached edge. Therefore \(d - \sqrt{(d - \zeta)^2 + \delta^2} > 2\zeta - \gamma\) and thus

\[
\Delta g_i > 2\zeta - \gamma > 0 \quad \text{(A.10)}
\]

since \(\delta < \min\{\beta, \sqrt{3\gamma}/2\}\).

**Case 3:** \(T_{\gamma, \delta}(u, w) \neq \emptyset\) and \(T_{\gamma, \delta}(w, v) \neq \emptyset\). In this case, we have \(g(u, w) \neq 0\) and \(g(w, v) \neq 0\). This implies \(||u, w|| > 2\zeta\) and \(||w, v|| > 2\zeta\). Then

\[
\Delta g_i = g(u, v) - (g(u, w) + g(w, v)) \\
= (d - 2\zeta) - (||u, w|| - 2\zeta) + (||w, v|| - 2\zeta)) \\
= d + 2\zeta - (\sqrt{x^2 + y^2} + \sqrt{(d - x)^2 + y^2}) \\
> d + 2\zeta - (\gamma + \sqrt{(d - \zeta)^2 + \delta^2})
\]

since \(\sqrt{x^2 + y^2} + \sqrt{(d - x)^2 + y^2} < \gamma + \sqrt{(d - \zeta)^2 + \delta^2}\) as \(x > \zeta\) and \(|y| < \delta\). As with Case 2, \(d - \sqrt{(d - \zeta)^2 + \delta^2} > 2\zeta - \gamma\). Therefore

\[
\Delta g_i > 2(2\zeta - \gamma) > 0 \quad \text{(A.11)}
\]

since \(\delta < \min\{\beta, \sqrt{3\gamma}/2\}\).

We now regard the construction of \(W\) as a LSGD \((\{x\}, 2\zeta, 2\zeta - \gamma, 2(2\zeta - \gamma))\), where \(x\) is a line segment of length \(||p, q|| - 2\zeta\). This connection can be seen by observing the following: First we notice that WALK\((p, q; W)\) terminates whenever \(g(W) = 0\). We also note that the initial gap of \(W\) equals \(||p, q|| - 2\zeta\). Furthermore, in each iteration of the “while” loop, exactly one of the situations in Cases 1-3 happens (refer to the thick edges in Fig. A.3), and Cases 1-3 correspond to moves “remove”, “prune”, and “split”, respectively, of the LSGD. Therefore we can use Lemma A.4 to get the required bound.

Because \(l(W)\) is an upper bound of the \((p, q)\)-distance \(d_\delta(p, q)\) in \(G_\delta\) upon the termination of WALK\((p, q; W)\), we easily deduce from Lemma A.3 and Lemma A.5 the following result for an encroached edge \(pq\) of \(G\).

**Proposition A.6** Let \(P\) be a simple polygon and \(\delta\) be a real number that satisfies \(0 < \delta < \min\{\beta, \sqrt{3\gamma}/2\}\). Then for any encroached edge \(pq\) of \(G\), the \((p, q)\)-distance \(d_\delta(p, q)\) in \(G_\delta\) satisfies

\[
d_\delta(p, q) \leq \frac{\gamma}{2\zeta - \gamma} ||p, q|| - \frac{2\gamma(\gamma - \zeta)}{2\zeta - \gamma}, \quad \text{where} \quad \zeta = \sqrt{\gamma^2 - \delta^2}.
\]
With an upper bound of $d_\delta(p, q)$ in hand, we are now ready to prove Theorem 4.2.

**Proof of Theorem 4.2.** Let $P$ be a simple polygon and $\delta$ be a number that satisfies $0 < \delta < \min\{\beta, \sqrt{3}/2\}$. For any two vertices $p$ and $q$ of $P$, we want to estimate an upper bound of $d_\delta(p, q)/d(p, q)$, where $d_\delta(p, q)$ and $d(p, q)$ are the $(p, q)$-distances in $G_\delta$ and $G$ respectively.

Let $Q = p_1 p_2 \ldots p_k$, where $p_1 = p$ and $p_k = q$, be a shortest $(p, q)$-path in $G$. If $p_i p_{i+1}$ is also an edge of $G_\delta$ then $d_\delta(p_i, p_{i+1}) = d(p_i, p_{i+1})$. Otherwise $p_i p_{i+1}$ is an encroached edge. Since $0 < \delta < \min\{\beta, \sqrt{3}/2\}$ it follows from Proposition A.6 that

$$
d_\delta(p_i, p_{i+1}) \leq \frac{2\gamma}{2\zeta - \gamma} ||p_i, p_{i+1}|| - \frac{2\gamma (\gamma - \zeta)}{2\zeta - \gamma}
$$

since $2\gamma (\gamma - \zeta)/(2\zeta - \gamma) > 0$ as both $\gamma - \zeta$ and $2\zeta - \gamma$ are positive. Notice $l(p_i p_{i+1}) = ||p_i, p_{i+1}|| = d(p_i, p_{i+1})$ and thus $l(Q) = \sum_{i=1}^{k-1} l(p_i p_{i+1}) = \sum_{i=1}^{k-1} ||p_i, p_{i+1}|| = d(p, q)$ since $Q$ is a shortest $(p, q)$-path in $G$. We obtain

$$
d_\delta(p, q) \leq \sum_{i=1}^{k-1} d_\delta(p_i, p_{i+1})$$

$$
\leq \frac{\gamma}{2\zeta - \gamma} \sum_{i=1}^{k-1} ||p_i, p_{i+1}||$$

$$
= \frac{\gamma}{2\zeta - \gamma} d(p, q).
$$

Substituting $\zeta$ by $\sqrt{\gamma^2 - \delta^2}$ and $\delta/\gamma$ by $\tau$, we get $d_\delta(p, q)/d(p, q) < 1/(2\sqrt{1 - \tau^2})$.
\[\square\]

We now turn to Theorem 4.1. Recall that for any two vertices $p'$ and $q'$ of $P'$, $d(p', q')$ is the length of a shortest collision-free $(p', q')$-path inside $P'$ (which equals the $(p', q')$-distance in the visibility graph $G'$ of $P'$), and that $D_{VS}(p', q')$ is the length of a collision-free $(p', q')$-path inside $P'$ that corresponds to a shortest $(p, q)$-path in the visibility skeleton $G_{VS}$ of $(P, \epsilon)$. To estimate $D_{VS}(p', q')/d(p', q')$, we need to relate a shortest $(p', q')$-path in $G'$ with a $(p, q)$-path in $G_{VS}$. Such a relationship can be established by using the following connection between the $2\epsilon$-visibility graph $G_{2\epsilon}$ of $P'$ and the visibility graph $G$ of $P$.

**Proposition A.7** Let $(P, \epsilon)$ be a simple representation of a simple polygon $P'$ with $\beta > 4\epsilon$ and $\gamma > (4\sqrt{3} + 6)\epsilon/3$. If $(p', q')$ is an edge of the $2\epsilon$-visibility graph $G_{2\epsilon}$ of $P'$ then $pq$ is an edge of the visibility graph $G$ of $P$.

**Proof.** Let $\beta'$ be the minimum vertex-to-edge distance in $P'$. Then $\beta' > \beta - 2\epsilon > 2\epsilon$ since $P'$ is an $(\epsilon)$-approximation polygon of $P$. Thus $(P', \epsilon)$ is a simple representation. By Theorem 3.1, $G_{2\epsilon}$ is the same as the visibility skeleton of $P'$. Since $P$ is an $(\epsilon)$-approximation polygon of $P'$, the lemma immediately follows from the definition of visibility skeletons.
\[\square\]

Equipped with the above results, we are now in a position to prove Theorem 4.1. First with the aid of Proposition A.7, we may construct a $(p, q)$-walk $W$ in $G_{VS}$
as follows (see Fig. A.4): first let $Q'$ (thick line) be a shortest $(p', q')$-path in $G'_{2\epsilon}$, then we apply Proposition A.7 to obtain a $(p, q)$-path $Q$ (dashed thick line) in $G$, and finally for each edge $uv$ of $Q$ we use $\text{WALK}(u, v; W_{uv})$ to construct a $(u, v)$-walk $W_{uv}$ in $G_{VS}$ and link these walks together to get a $(p, q)$-walk $W$ (dashed thin curve). Now by the definition of $G_{VS}$, $W$ corresponds to a $(p', q')$-walk $W'$ (thin curve) in $G'$. Using this information, we can prove Theorem 4.1 as follows:

**Proof of Theorem 4.1** Let $(P, \epsilon)$ be a simple representation of a simple polygon $P'$ with $\beta > 4\epsilon$ and $\gamma > (4\sqrt{3} + 6)\epsilon / 3$, where $\beta$ and $\gamma$ are the minimum vertex-to-edge and vertex-to-vertex distances in $P$ respectively. Let $p'$ and $q'$ be two arbitrary vertices of $P'$. Let $Q = p'_1 p'_2 \ldots p'_k$, where $p'_i = p'$ and $p'_k = q'$, be a shortest $(p', q')$-path in the $2\epsilon$-visibility graph $G'_{2\epsilon}$ of $P'$. Since $(P, \epsilon)$ is a simple representation with $\beta > 4\epsilon$ and $\gamma > (4\sqrt{3} + 6)\epsilon / 3$, it follows from Proposition A.7 that each $p_i p_{i+1}$, $1 \leq i \leq k - 1$, is an edge of the visibility graph $G$ of $P$ and thus $Q = p_1 p_2 \ldots p_k$ is a $(p, q)$-path in $G$. We will show that there exist $\lambda_1$ and $\lambda_2$ such that

$$l(Q') < \lambda_1 d(p', q') \tag{A.12}$$

and

$$D_{VS}(p'_i, p'_{i+1}) < \lambda_2 \|p'_i, p'_{i+1}\| \text{ for every } 1 \leq i \leq k - 1. \tag{A.13}$$

Then it follows that

$$D_{VS}(p', q') \leq \sum_{i=1}^{k-1} D_{VS}(p'_i, p'_{i+1})$$

$$< \lambda_2 \sum_{i=1}^{k-1} \|p'_i, p'_{i+1}\| \quad \text{(by (A.13))}$$

$$= \lambda_2 l(Q)$$

$$< \lambda_1 \lambda_2 d(p', q') \quad \text{(by (A.12))}$$

which implies $D_{VS}(p', q')/d(p', q') < \lambda_1 \lambda_2$. So we only need to find $\lambda_1$ and $\lambda_2$.

Consider $\lambda_1$ first. Let $\delta' = 2\epsilon$. Let $\beta'$ and $\gamma'$ be the minimum vertex-to-edge and vertex-to-vertex, respectively, distances in the visibility graph $G'$ of $P'$. Then
\[ \beta' \geq \beta - 2\varepsilon > 2\varepsilon \] and \[ \gamma' \geq \gamma - 2\varepsilon > 4\varepsilon / \sqrt{3} \] since \( P' \) is an \( \varepsilon \)-approximation polygon of \( P \). Then \( \delta' < \min\{\beta, \sqrt{3}\gamma'/2\} \). It follows from Theorem 4.2 that

\[ d_{g'}(p', q') < \frac{d(p', q')}{\sqrt{1 - \tau'^2} - 1}, \]

where \( d_{g'}(p', q') \) and \( d(p', q') \) are the \( (p', q') \)-distances in \( G_{2\varepsilon} \) and \( G' \) respectively, and \( \tau' = \delta'/\gamma' \). Since \( l(Q') = d_{g'}(p', q') \), we obtain

\[ l(Q') < \frac{d(p', q')}{\sqrt{1 - \tau'^2} - 1} \]

\[ \leq \frac{1}{\sqrt{1 - \left(\frac{2\varepsilon}{\gamma' - \delta'}\right)^2} - 1} d(p', q') \quad \text{(since } \gamma' \geq \gamma - 2\varepsilon \text{ and } \delta' = 2\varepsilon) \]

\[ = \frac{1 - 2\rho}{2\rho + 2\sqrt{1 - 4\rho} - 1} d(p', q'), \text{ where } \rho = \varepsilon/\gamma. \]

Thus \( \lambda_1 = (1 - 2\rho)/(2\rho + 2\sqrt{1 - 4\rho} - 1) \).

Now we consider \( \lambda_2 \). Let \( \delta = 2\varepsilon \). Then by Theorem 3.1, the visibility skeleton \( G_{\lambda 2} \) of \((P, \varepsilon)\) is precisely the \( \delta \)-visibility graph \( G_\delta \) of \( P \). As noted before, \( p_ip_{i+1} \) is an edge of \( G \). If it is also an edge of \( G_{\lambda 2} \) then \( D_{\lambda 2}(p_i, p_{i+1}) = d(p_i, p_{i+1}) \); otherwise \( p_ip_{i+1} \) is an encroached edge, and we can use \textsc{WALK}(\( p_i, p_{i+1}; U_i \)) to construct a \( (p_i, p_{i+1}) \)-walk \( U_i \) in \( G_{\lambda 2} \). By the assumption that \( \beta > 4\varepsilon \) and \( \gamma > (4\sqrt{3} + 6)\varepsilon/3 \approx 4.31\varepsilon \), it is clear that \( \delta < \min\{\beta, \sqrt{3}\gamma/2\} \). Therefore by Lemma A.5, \textsc{WALK}(\( p_i, p_{i+1}; U_i \)) terminates; and upon its termination, \( U_i \) corresponds to a collision-free \((p_i', p_{i+1}')\)-walk \( U'_i \) inside \( P' \). Therefore the length \( l(U'_i) \) of \( U'_i \) is an upper bound of \( D_{\lambda 2}(p_i', p_{i+1}') \). Note that for any two vertices \( u \) and \( v \) of \( P \), we have

\[ ||u', v'|| - \delta \leq ||u, v'|| \leq ||u', v'|| + \delta \]

as \( P' \) is an \( \varepsilon \)-approximation polygon of \( P \) and \( \delta = 2\varepsilon \). It follows that

\[ d(p_i', p_{i+1}') = ||p_i', p_{i+1}'|| \geq ||p_i, p_{i+1}|| - \delta \quad \text{(A.14)} \]

and

\[ D_{\lambda 2}(p_i', p_{i+1}') \leq l(U'_i) \leq l(U_i) + \delta k_i, \quad \text{(A.15)} \]

where \( k_i \) is the number (repetition counts) of edges in \( U_i \). Combining (A.15) with Lemma A.3 and Lemma A.5, we get

\[ D_{\lambda 2}(p_i', p_{i+1}') < ||p_i, p_{i+1}|| + 2(\gamma - \zeta)||p_i, p_{i+1}|| - \gamma + ((||p_i, p_{i+1}|| - \gamma)/ 2\zeta - \gamma) \delta, \quad \text{(A.16)} \]

where \( \zeta = \sqrt{\gamma^2 - \delta^2} \). Therefore we can deduce from (A.14) and (A.16) the following inequality:

\[ \frac{D_{\lambda 2}(p_i', p_{i+1}')}{d(p_i', p_{i+1}')} < \gamma + \delta (1 + 2\zeta + \delta - 2\gamma)/(||p_i, p_{i+1}|| - \delta). \quad \text{(A.17)} \]

It is easy to show that \( 2\zeta + \delta - 2\gamma > 0 \) if \( \gamma > 2.5\varepsilon \). Therefore from the assumption that \( \gamma > (4\sqrt{3} + 6)\varepsilon/3 \approx 4.31\varepsilon \) we have \( 2\zeta + \delta - 2\gamma > 0 \) and \( 2\zeta - \gamma > 0 \). Recall
that edge $p_ip_{i+1}$ is an encroached edge, implying $\|p_i, p_{i+1}\| > 2\zeta$ by Lemma A.1. Therefore (A.17) yields the following:

$$
\frac{D_{VS}(p_i, p_{i+1})}{d(p_i, p_{i+1})} < \frac{\gamma + \delta}{2\zeta - \gamma} (1 + \frac{2\zeta + \delta - 2\gamma}{2\zeta - \delta}) = \frac{2(\gamma + \delta)}{2\zeta - \delta} = \frac{1 + 2\rho}{\sqrt{1 - 4\rho^2} - \rho},
$$

where $\rho = \varepsilon / \gamma$.

So we let $\lambda_2 = (1 + 2\rho)/(\sqrt{1 - 4\rho^2} - \rho)$.

Putting the above results together, we get the upper bound in Theorem 4.1.

It is worth noting the meanings of the two numbers $\lambda_1$ and $\lambda_2$ in the above proof. Let $D(p', q')$ denote the length of the $(p', q')$-path (as noted previously, this path may not be a collision-free path in $P'$) corresponding to a shortest $(p, q)$-path in the visibility graph $G$ of $P$. Then the above proof implies that for a simple representation with $\beta > 4\varepsilon$ and $\gamma > (4\sqrt{3} + 6)\varepsilon / 3$, $D(p', q')/d(p', q') < \lambda_1$ and $D_{VS}(p', q')/D(p', q') < \lambda_2$. Table 3 gives some typical values of $\lambda_1$ and $\lambda_2$. It should also be noted that in deriving $\lambda_1$ the $(p', q')$-path $Q'$ is a collision-free path inside $P'$. Therefore $G$ actually contains a $(p, q)$-path $Q$ whose corresponding $(p', q')$-path $Q'$ is a collision-free path inside $P'$ which may be shorter than a $(p', q')$-path obtained from $G_{VS}$ (compare the upper bound in Table 1 with that in row "$\lambda_1$" of Table 3). Unfortunately, we do not know how to find such a $Q'$; in fact, it may be impossible to find such a path in $G$ without knowing the precise input of $P'$ since the visibility skeleton $G_{VS}$ is the largest graph that guarantees visibility amongst vertices of $P'$.

### Table A.1. Typical values of $\lambda_1$ and $\lambda_2$

<table>
<thead>
<tr>
<th>$\rho = \varepsilon / \gamma &lt; 0.232$</th>
<th>2/9</th>
<th>1/5</th>
<th>1/6</th>
<th>1/10</th>
<th>1/20</th>
<th>1/100</th>
<th>1/1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_1$</td>
<td>5.60</td>
<td>2.94</td>
<td>1.37</td>
<td>1.07</td>
<td>1.01</td>
<td>1.000</td>
<td>1.0000</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>2.14</td>
<td>1.95</td>
<td>1.72</td>
<td>1.36</td>
<td>1.16</td>
<td>1.031</td>
<td>1.0030</td>
</tr>
</tbody>
</table>