Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A new fast algorithm for computing the distance between two disjoint convex polygons based on Voronoi diagram[*]

YANG Cheng-lei[†], QI Meng, MENG Xiang-xu, LI Xue-qing, WANG Jia-ye
(*School of Computer Science and Technology, Shandong University, Jinan 250100, China*)
[†]E-mail: chl_yang@sdu.edu.cn

Received May 20, 2006; revision accepted June 21, 2006

Abstract: Computing the distance between two convex polygons is often a basic step to the algorithms of collision detection and path planning. Now, the lowest time complexity algorithm takes $O(\log m+\log n)$ time to compute the minimum distance between two disjoint convex polygons P and Q, where n and m are the number of the polygons' edges respectively. This paper discusses the location relations of outer Voronoi diagrams of two disjoint convex polygons P and Q, and presents a new $O(\log m+\log n)$ algorithm to compute the minimum distance between P and Q. The algorithm is simple and easy to implement, and does not need any preprocessing and extra data structures.

Key words: Computational geometry, Polygon, Voronoi diagram, Distance computation
doi:10.1631/jzus.2006.A1522 **Document code:** A **CLC number:** TP202

INTRODUCTION

To compute the minimum distance between two convex polygons or polyhedrons is often a main step of many applications, such as collision detection (Choi *et al*., 2006; Li *et al*., 2003), path planning. In order to reduce the time complexity of the algorithm as much as possible, the convex property must be applied fully.

Edelsbrunner (1985) proposed an algorithm for computing the minimum distance between two disjoint convex polygons. The algorithm takes $O(\log m+\log n)$ time, and it is the lowest time complexity algorithm existing so far. Dobkin and Kirkpatrick (1990) took advantage of hierarchical representation of a convex polyhedron to obtain the minimum distance between two convex polyhedrons in $O(\log m\log n)$ time and got the minimum distance between two convex polygons in $O(\log m+\log n)$ time. Whereas, this algorithm needs $O(m+n)$ preprocessing time to establish the hierarchical representation of convex polygon or polyhedron, and is complex to implement.

Now collision detection using Voronoi diagram has become one of the most effective methods. Voronoi diagram is one of the most important geometrical structures and research topics in discrete computing geometry, which divides the space into several regions called Voronoi region (VOR) according to the nearest attribute of given objects such as points, line segments, circular arcs, and so on. The outer Voronoi diagram of a polygon is the union of all Voronoi regions outside the polygon (Fig.1b). The outer Voronoi diagram of a convex polygon is very special: each Voronoi edge is perpendicular to one of the convex polygon edges (Fig.1a).

Lin and Canny (1991) first calculated the minimum distance between two disjoint convex polyhedrons using adjacent properties of outer Voronoi diagram of convex polyhedrons. Their method is a feature-based method and well suited to repetitive distance computing as the polyhedrons move in a

[*] Project supported by the National Nature Science Foundation of China (Nos. 60473103 and 60473127) and the Natural Science Foundation of Shandong Province (No. Y2005G03), China

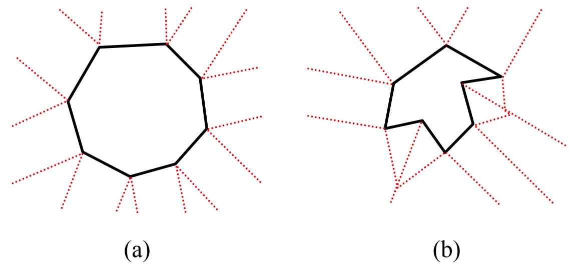


Fig.1 Polygon (solid line) and its outer Voronoi diagram (dot line). (a) Convex polygon; (b) Simple polygon

sequence of small, discrete steps. Once initialized, the excepted running time of the algorithm is constant. Some improvements were made later (Lin *et al*., 1994; Cohen *et al*., 1995; Ponamgi *et al*., 1997; Hudson *et al*., 1997; Mirtich, 1998), such as stability, which makes the method become one of the most effective methods for solving the collision detection problem at the present time. However, the time complexity of this category of algorithms is $O(m+n)$. Guibas *et al*.(2000) combined the ideas of feature-based methods and hierarchical data structures, and presented an algorithm for computing the minimum distance between two disjoint convex polyhedrons, which takes $O(\log m+\log n)$ time. However, just as Dobkin and Kirkpatrick (1990)'s method, it costs $O(m+n)$ preprocessing time to establish the hierarchical representation of a convex polyhedron or polygon.

This paper analyzes the properties of outer Voronoi diagram of two disjoint convex polygons and their mutual location relation, and presents a new algorithm for computing the minimum distance between two disjoint convex polygons, which can be completed in $O(\log m+\log n)$ time. The algorithm is simple and easy to implement, and does not need any preprocessing and extra data structures.

ALGORITHM DESCRIPTION

Let P and Q be disjoint convex polygons, whose vertex or edge is called object in the paper. The vertices of P are $p_1, p_2, \ldots, p_n$ in anticlockwise direction, and the vertices of Q are $q_1, q_2, \ldots, q_m$ in anticlockwise direction, where n and m are the number of vertices of P and Q respectively. If suffix is beyond n or m, compute the module. $C(p',p'')$ indicates the polygon chain between two objects p' and p'' in anticlockwise direction.

Definition 1 Assume $p\in P$ and $q\in Q$. If the distance between p and q is the minimum distance between P and Q, then $<p,q>$ is called a Minimum Distance Object Pair (MDOP) of P and Q.

The MDOP $<p,q>$ of P and Q has three types:

(1) <vertex, vertex> (Fig.2a);

(2) <vertex, edge> (Fig.2b);

(3) <edge, edge>: p and q are edges, and $p//q$. This case must contain the cases of <vertex, vertex> or <vertex, edge> (Figs.2c~2f). Hence, we just take the first and second types into consideration.

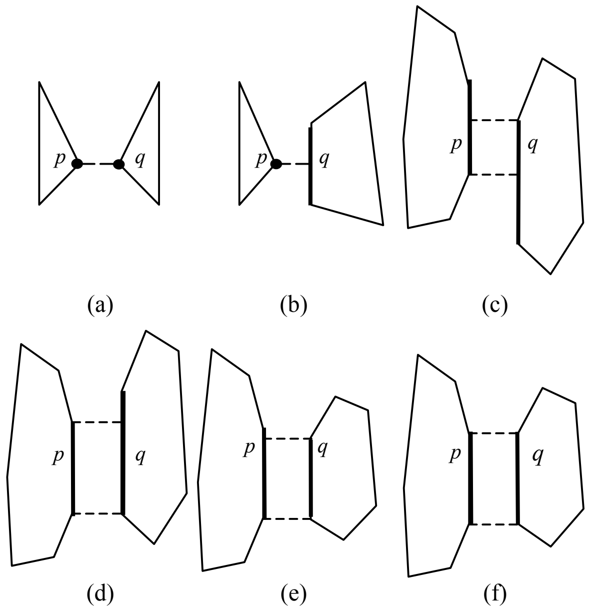


Fig.2 Three types of MDOP <*p*,*q*>
(a) p, q are vertices; (b) p is vertex, q is edge; p, q are edges in (c)~(f); (c) contains <vertex, edge>; (d) contains <vertex, vertex> and <vertex, edge>; (e) contains <vertex, vertex> and <vertex, edge>; (f) contains <vertex, vertex>

If p and q are vertices, the distance between them is their Euclid distance. If p is a vertex and q is an edge, the line perpendicular to q passing through p intersects with the edge q at q^*, the distance between the vertex p and q^* is the distance between p and q. p and q^* are called realizing point pair. Similarly, we can define the distance between two parallel edges p and q, but the number of realizing point pair (p^*, q^*) is infinite.

Our minimum distance algorithm is mainly for finding the MDOP of P and Q. The first step of the algorithm is to decide the initial search ranges containing the MDOP by binary search technique, and then shorten the search ranges gradually also using binary search technique till an MDOP is found.

According to the example shown in Fig.3, the process of finding an MDOP $<p,q>$ is described briefly as follows:

(1) First compute the initial search ranges (two polygon chains) $P'=C(p',p'')$ and $Q'=C(q'',q')$, where P' and Q' include the objects p and q of the MDOP respectively.

(2) Find the middle vertices p_a and p_b on P' and Q' respectively, and halve P' and Q' into sub-polygon chains: $P_1''=C(p', p_a)$ and $P_2''=C(p_a, p'')$, $Q_1''=C(q_b, q')$ and $Q_2''=C(q'', q_b)$;

(3) According to the location relation of p_a, q_b and their Voronoi regions, we can determine at least one sub-chain among P_1'', P_2'', Q_1'' and Q_2'', which does not contain any object of MDOP, and discard it. The searching procedure is carried out the remaining chains that contain the MDOP objects.

(4) Repeat the processes above until the MDOP is found.

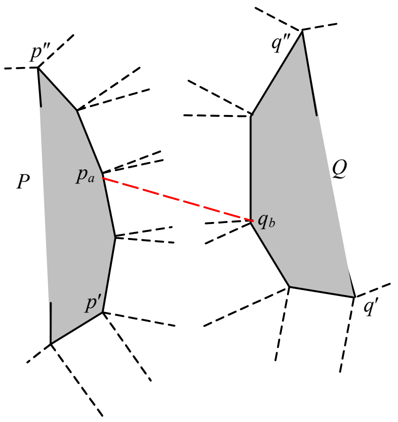


Fig.3 Example of two disjoint convex polygons

ANALYSIS OF LOCATION RELATIONS

The key of the algorithm is how to compute the location relation of p_a and q_b. In this section, we first give some related notions, analyze some related properties, and then introduce the method of computing the location relation of p_a and q_b.

Basic notions and properties

Definition 2 For an MDOP $<p,q>$ of P and Q, let p^* and q^* be the realizing point pair of minimum distance between p and q, then the perpendicular bisector of the line segment p^*q^* must be a separate line of P and Q, where P is on its left and Q is on its right. This separate line is called median line l (Fig.4).

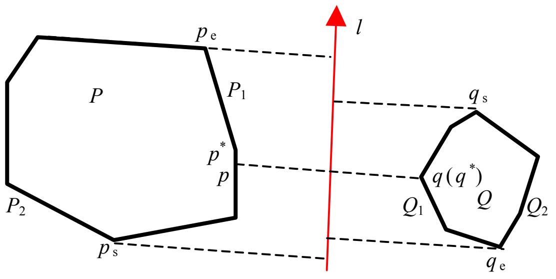


Fig.4 Median line l of P and Q

The median line l defined here is used to analyze the properties of outer Voronoi diagrams of P and Q. In fact, it does not need to be computed in the algorithm of finding the MDOP of P and Q.

Suppose that p_e and p_s are the highest and lowest vertices in the direction l (O'Rourke, 1994). P_1 is the chain formed by the objects from p_s to p_e on P. P_1 is monotone with respect to the median line l. Similarly, the objects from the highest vertex q_s in the direction l to the lowest vertex q_e on Q form a monotone chain Q_1. It is obvious that P_1 and Q_1 contain the MDOP objects of P and Q (Fig.4).

P_1 is divided into three parts by p of MDOP $<p, q>$: the upper part P_u, the middle part $P_m=\{p\}$ and the lower part P_d. Similarly, Q_1 can be divided by q to Q_u, $Q_m=\{q\}$, and Q_d.

To describe it conveniently, for any vertex object $o=p_i$ on P, let $VOR(o)$ denote the Voronoi region of o, and $VE(o)_1$, $VE(o)_2$ denote its two Voronoi edges respectively, where $VE(o)_1$ is the ray from p_i perpendicular to $p_{i-1}p_i$, $VE(o)_2$ is another ray from p_i perpendicular to p_ip_{i+1}. For any edge object $o=p_{i-1}p_i$, $VE(o)_1$ and $VE(o)_2$ denote its two Voronoi edges, where $VE(o)_1$ and $VE(o)_2$ are the rays perpendicular to $p_{i-1}p_i$ from p_{i-1} and p_i respectively (Fig.5). Let $\boldsymbol{l}_u$, $\boldsymbol{l}_{i1}$, $\boldsymbol{l}_{i2}$ and $\boldsymbol{v}_u^*$ respectively denote unit vectors of l, $VE(o)_1$, $VE(o)_2$ and $v^*=q^*-p^*$. "$A\times B>0$" denotes that the projection of cross product of A and B in the direction of Z is more than zero.

Lemma 1 If p of MDOP $<p,q>$ is a vertex object, then for any vertex object $o\in P_1$, there are (Fig.5a):

(1) If $o\in P_u$, then

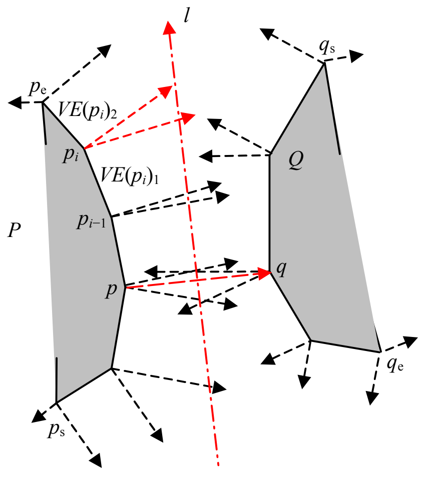
$$\boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u>\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u>0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}>\boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}>0;$$

(2) If $o\in P_d$, then

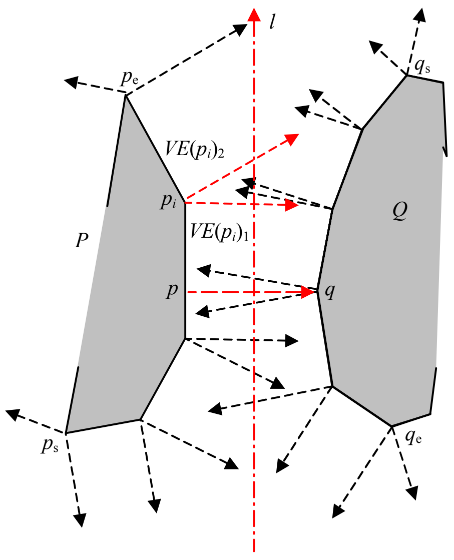
$$\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u<\boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u<0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}<\boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}<0;$$

(3) If $o=p\in P_m$, then

$$\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u<0 \text{ and } \boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u>0,\ \boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}<0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}>0.$$



(a)



(b)

Fig.5 Relations of Voronoi edges and median line *l*
(a) p, q are vertices; (b) p is edge, q is vertex

Lemma 2 If p of MDOP $<p,q>$ is an edge object, then for any vertex object $o\in P_1$, there are (Fig.5b):

(1) If $o\in P_u$ and o is not the end-point of p, then

$$\boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u>\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u>0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}>\boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}>0;$$

(2) If $o\in P_d$ and o is not the start-point of p, then

$$\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u<\boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u<0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}<\boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}<0;$$

(3) If o is the start-point of p, then

$$\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u<0 \text{ and } \boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u=0,\ \boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}<0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}=0;$$

(4) If o is the end-point of p, then

$$\boldsymbol{l}_{i1}\cdot\boldsymbol{l}_u=0 \text{ and } \boldsymbol{l}_{i2}\cdot\boldsymbol{l}_u>0,\ \boldsymbol{v}_u^*\times\boldsymbol{l}_{i1}=0 \text{ and } \boldsymbol{v}_u^*\times\boldsymbol{l}_{i2}>0.$$

Terminate condition of searching process

In the following, Theorem 1 shows the method of deciding whether $<p,q>$ is an MDOP of P and Q, which gives a terminate condition of searching the MDOP (Lin and Canny, 1991; Mirtich, 1998).

Theorem 1 P and Q are two convex polygons. (p^*, q^*) is the realizing point pair, $p^*\in p\in P$ and $q^*\in q\in Q$. $p^*\in VOR(q)$ and $q^*\in VOR(p)$, if and only if $<p,q>$ is the MDOP of P and Q.

Compute new search range

Let two vertex objects $p_a\in P_1$, $q_b\in Q_1$ are the middle vertices of the current search range P' and Q', and divide P', Q' into equal sub-polygon chains $P_1''=C(p', p_a)$, $P_2''=C(p_a, p'')$ and $Q_1''=C(q_b, q')$, $Q_2''=C(q'', q_b)$. If we can make sure that $p_a\in P_u$, P_m or P_d, or $q_b\in Q_u$, Q_m or Q_d, the new search range can be determined as follows:

(1) If $p_a\in P_m$ and $q_b\in Q_m$ (Fig.6a), then $<p_a,q_b>$ is an MDOP $<p,q>$ (by Theorem 1);

(2) If $p_a\in P_m$ and $q_b\notin Q_m$ (Fig.6b), then p_a is the object of MDOP $<p,q>$. We search the object q in Q' by dividing Q' into two sub-polygon chains by its middle vertex q_b repeatedly, such that $p=p_a\in VOR(q)$ and $q^*\in VOR(p)$, where (p^*,q^*) is realizing point pair.

(3) If $q_b\in Q_m$ and $p_a\notin P_m$ (Fig.6c), then q_b is one object of MDOP $<p,q>$, we also search the object p in P' by binary search technique, such that $p^*\in VOR(q)$ and $q\in VOR(p)$;

(4) For the other cases (Figs.6d~6g), we must determine the half chains P_u or P_d and Q_u or Q_d, to which p_a and q_b belong respectively. If $p_a\in P_u$ (or $p_a\in P_d$), then for the next search P_1 is taken place by P_d (or P_u). The replacement of Q_1 is determined similarly.

In order to describe conveniently, we define that q_b is on the right side of $VOR(p_a)$ when q_b is on the right of $VE(p_a)_1$; q_b is on the left of $VOR(p_a)$ when q_b is on the left of $VE(p_a)_2$; q_b is in the interior of

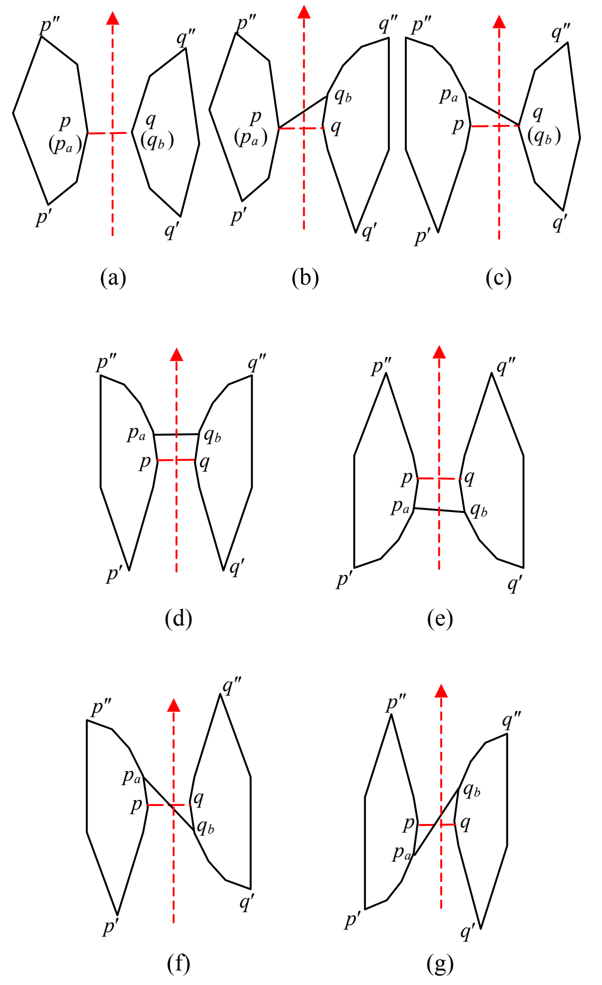


Fig.6 Location relations of p_a and p_b about pq
(a) $p_a \in P_m$ and $q_b \in Q_m$; (b) $p_a \in P_m$ and $q_b \notin Q_m$; (c) $q_b \in Q_m$ and $p_a \notin P_m$; (d) $p_a \in P_u$ and $q_b \in Q_u$; (e) $p_a \in P_d$ and $q_b \in Q_d$; (f) $p_a \in P_u$ and $q_b \in Q_d$; (g) $p_a \in P_d$ and $q_b \in Q_u$

$VOR(p_a)$ when $q_b \in VOR(p_a)$. The relation of p_a and $VOR(q_b)$ can be defined similarly. For example, in Fig.7, q_4 is on the right of $VOR(p_2)$; p_2 is on the left of $VOR(q_4)$ but not on the left of $VOR(q_3)$.

The location relation of objects $p_a \in P_1$, $q_b \in Q_1$ and P_u, P_m, P_d, Q_u, Q_m, Q_d can be classified in the following cases:

(1) q_b is on the right of $VOR(p_a)$, there are three cases:

(1.1) p_a is on the left of $VOR(q_b)$ (as p_2 and q_4 in Fig.7): We can obtain $p_a \in P_u$, $q_b \in Q_u$ by Theorem 2.

Theorem 2 Suppose q_b is on the right of $VE(p_a)_1$, p_a is on the left of $VE(q_b)_2$, then $p_a \in P_u$, $q_b \in Q_u$.

Proof Suppose $p_a \notin P_u$. Since q_b is on the right of $VE(p_a)_1$, by Lemma 1 and Lemma 2, we know $p^*q^* \times VE(p_a)_1 \leq 0$ and $p^*q^* \times p_a q_b < 0$, q_b is on the right of p^*q^*, namely $q_b \in Q_d$. Because $p^*q^* \times VE(q_b)_1 < 0$ and $p^*q^* \times VE(q_b)_2 < 0$, we can get that p_a is on the right of $VE(q_b)_1$ and $VE(q_b)_2$. This contradiction proves $p_a \in P_u$. Similarly, we can prove $q_b \in Q_u$. This completes the proof.

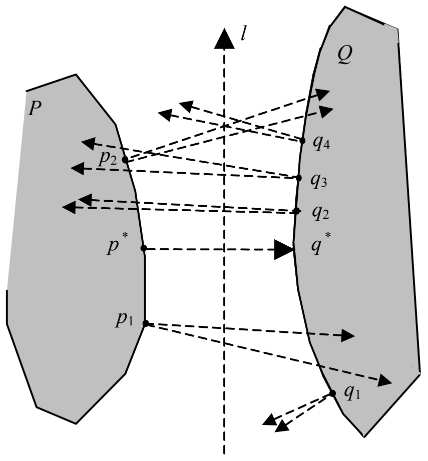


Fig.7 Case 1: Location relations *of* p_a and q_b

In the following, the proof of Cases 1.2, 2.1, 2.3, 3.2 and 3.3 is similar to that of Case 1.1.

(1.2) p_a is in the interior of $VOR(q_b)$ (as p_2 and q_3 in Fig.7): then $p_a \in P_u$, $q_b \in Q_u$;

(1.3) p_a is on the right of $VOR(q_b)$ (as p_2 and q_1, q_2; p_1 and q_1 in Fig.7): By Theorem 3,

If $VE(p_a)_1 \times VE(q_b)_1 \geq 0$, then $p_a \in P_u$;

If $VE(p_a)_1 \times VE(q_b)_1 < 0$, then $q_b \in Q_d$.

Theorem 3 Suppose q_b is on the right of $VE(p_a)_1$, p_a is on the right of $VE(q_b)_1$,

(1) If $VE(p_a)_1 \times VE(q_b)_1 \geq 0$, then $p_a \in P_u$;

(2) If $VE(p_a)_1 \times VE(q_b)_1 < 0$, then $q_b \in Q_d$.

Proof (1) Suppose $VE(p_a)_1 \times VE(q_b)_1 \geq 0$, but $p_a \notin P_u$. By Lemma 1 and Lemma 2, $p^*q^* \times VE(p_a)_1 \leq 0$. Since q_b is on the right of $VE(p_a)_1$ and p_a is on the right of $VE(q_b)_1$, we obtain $q_b \in Q_d$, and $p^*q^* \times VE(q_b)_1 < 0$. By Lemma 1 and Lemma 2, we have $VE(p_a)_1 \times VE(q_b)_1 < 0$. This contradicts the theorem condition. Hence, the original proposition is true. Similarly, we can prove Case 2.

(2) q_b is in the interior of $VOR(p_a)$, there are three cases:

(2.1) p_a is on the left of $VOR(q_b)$ (as p_2 and q_4, p^* and q_3 in Fig.8): Then $p_a \in P_u$ or $p_a \in P_m$, $q_b \in Q_u$;

(2.2) p_a is in the interior of $VOR(q_b)$ (as p^* and q^* in Fig.8): We can obtain $p_a \in P_m$, $q_b \in Q_m$ by Theorem 1.

(2.3) p_a is on the right of $VOR(q_b)$ (as p_1 and q_1,

p^* and q_2 in Fig.8): there are $p_a \in P_d$ or $p_a \in P_m$, $q_b \in Q_d$;

(3) q_b is on the left of $VOR(p_a)$, there are three cases:

(3.1) p_a is on the left of $VOR(q_b)$ (as p_2 and q_5; p_1 and q_3, q_5 in Fig.9):

If $VE(q_b)_2 \times VE(p_a)_2 > 0$, then $p_a \in P_d$.

If $VE(q_b)_2 \times VE(p_a)_2 \leq 0$, then $q_b \in Q_u$.

The proof of Case 3.1 is similar to that of Case 1.3.

(3.2) p_a is in the interior of $VOR(q_b)$ (as p_1 and q_2 in Fig.9): We can obtain $p_a \in P_d$, $q_b \in Q_d$;

(3.3) p_a is on the right of $VOR(q_b)$ (as p_1 and q_1 in Fig.9): We can obtain $p_a \in P_d$, $q_b \in Q_d$.

Table 1 summarizes the rules of determining the location relation of $p_a \in P_1$ and $q_b \in Q_1$.

FIND INITIAL SEARCHING RANGES

In this section, a method of computing the initial search ranges P' and Q' is discussed. The method is to find the end points of P' and Q', such that $P' \subseteq P_1$, $Q' \subseteq Q_1$, and $p \in P'$, $q \in Q'$, where $<p,q>$ is the MDOP of P and Q.

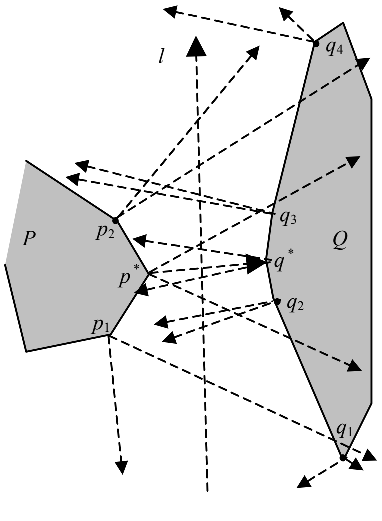


Fig.8 Case 2: Location relations of p_a and q_b

According to Fig.10, the idea of the method is briefly described as follows:

We first find two vertices $p_1 \in P$ and $q_1 \in Q$ by binary search technique, such that $q_1 \in VOR(p_i)$ or $q_1 \in VOR(p_i p_{i+1})$, and $p_1 \in VOR(q_j)$ or $p_1 \in VOR(q_j q_{j+1})$, where $1 \leq i \leq n$ and $1 \leq j \leq m$. On the another side of $p^* q^*$ find p_2 and q_2 such that $p_2 \in VOR(q_r)$ or $p_2 \in VOR(q_r q_{r+1})$, and $q_2 \in VOR(p_k)$ or $VOR(p_k p_{k+1})$, where $1 \leq k \leq n$ and $1 \leq r \leq m$. We obtain the initial searching ranges $P'=C(p_1, p_2)$, $Q'=C(q_2, q_1)$ or $P'=C(p_2, p_1)$, $Q'=C(q_1, q_2)$.

The steps of the method are briefly described as follows:

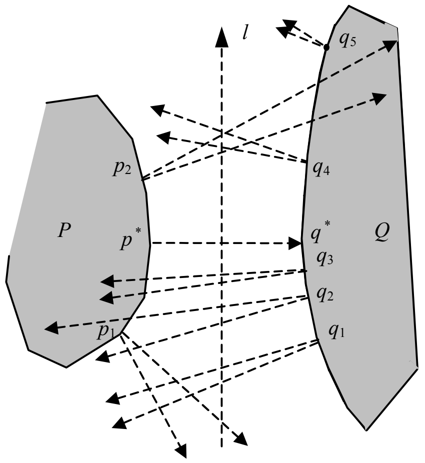


Fig.9 Case 3: Location relations of p_a and q_b

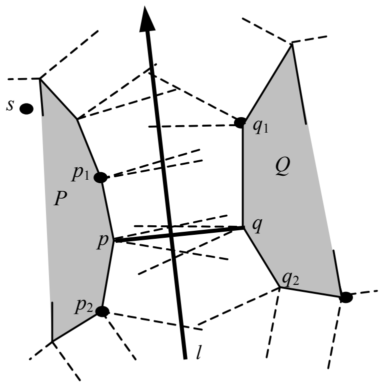


Fig.10 Find initial ranges

Table 1 The rules of determining the location relation of $p_a \in P_1$ and $q_b \in q_1$

Relation of q_b and $VOR(p_a)$	Relation of q_a and $VOR(p_b)$		
	Left	Inner	Right
Right	$p_a \in P_u$, $q_b \in Q_u$	$p_a \in P_u$, $q_b \in Q_u$	$p_a \in P_u$ or $q_b \in Q_d$
Inner	$p_a \in P_u$ or $p_a \in P_m$, $q_b \in Q_u$	$p_a \in P_m$, $q_b \in Q_m$	$p_a \in P_d$ or $p_a \in P_m$, $q_b \in Q_d$
Left	$p_a \in P_d$ or $q_b \in Q_u$	$p_a \in P_d$, $q_b \in Q_d$	$p_a \in P_d$, $q_b \in Q_d$

(1) Get any vertex $s\in P$, search o_1 ($o_1=q_j$ or q_jq_{j+1}) in Q by binary search technique, where $s\in VOR(o_1)$. And we let $q_1=q_j$.

(2) Search o_2 ($o_2=p_i$ or p_ip_{i+1}) in P by binary search technique, where $q_1\in VOR(o_2)$. Let $p_1=p_i$.

(3) Decide p_1 and q_1 are on which side of p^*q^* (i.e. decide $p_1\in P_\mathrm{u}$, P_m or P_d, $q_1\in Q_\mathrm{u}$, Q_m or Q_d):

Let

$$\tilde{p}=\begin{cases} p_i, & \text{if } o_2 \text{ is a vertex object;}\\ \text{projection of } q_1 \text{ on } o_2, & \text{if } o_2 \text{ is an edge object.}\end{cases}$$

Because $q_1\in VOR(o_2)$, we have

(i) If $\tilde{p}\in VOR(q_1)$, by Theorem 1, o_2 and q_1 is an MDOP;

(ii) If $\tilde{p}$ is on the left of $VE(q_1)_1$ and $VE(q_1)_2$, by Lemma 1 and Lemma 2, $p_1\in P_\mathrm{u}$ or P_m, $q_1\in Q_\mathrm{u}$ or Q_m;

(iii) If $\tilde{p}$ is on the right of $VE(q_1)_1$ and $VE(q_1)_2$, by Lemma 1and Lemma 2, $p_1\in P_\mathrm{d}$ or P_m, $q_1\in Q_\mathrm{d}$ or Q_m.

(iv) For the remaining cases, compute the intersection point a of $VE(q_1)_1$ and $VE(p_1)_1$, let t_1 and s_1 be the relevant parameters of a on $VE(q_1)_1$ and $VE(p_1)_1$ respectively; and compute the intersection point b of $VE(q_1)_2$ and $VE(p_1)_1$, let t_2 and s_2 be the relevant parameters of b on $VE(q_1)_2$ and $VE(p_1)_1$ respectively.

If ($t_2<0$ and $s_2>0$), and ($t_1>0$ and $s_1>0$ and $s_1>s_2$) or ($t_1<0$ and $s_1<0$) or $VE(q_1)_1//VE(p_1)_1$, then we can easily prove that $p_1\in P_\mathrm{u}$ or P_m, $q_1\in Q_\mathrm{u}$ or Q_m; or else, $p_1\in P_\mathrm{d}$ or P_m, $q_1\in Q_\mathrm{d}$ or Q_m.

(4) If $q_1\in Q_\mathrm{m}$, $q_2=q_1$; if $q_1\in Q_\mathrm{u}$, we find q_2 in Q by binary search technique, such that q_2 is on the left of $VE(q_1)_2$, and there exists a line l_1, that $l_1//VE(q_1)_2$, and q_2 is on l_1, and Q is on the right of l_1. It can be proved that $q_2\in Q_\mathrm{d}$ or $q_2\in Q_\mathrm{m}$ (for example, in Fig.11, $l_1//VE(q_1)_2$, $l_2//VE(q'')_2$, $l_3//p^*q^*$, q is on l_2, q' is on l_3, by Lemma 1 and Lemma 2, we can prove that q_2 must be a vertex of $C(q, q')$).

If $q_1\in Q_\mathrm{d}$, we can get $q_2\in Q_\mathrm{u}$ or Q_m similarly.

(5) Search o_3 ($o_3=p_k$ or p_kp_{k+1}) in P by binary search technique, till $q_2\in VOR(o_3)$. Let $p_2=p_k$.

By Theorem 4, we can get that p_2 and q_2 are on the same side of p^*q^*. Because p_1, $q_1$and p_2, q_2 are on different sides of p^*q^*, the polygon chains P' ($C(p_1, p_2)$ or $C(p_2, p_1)$) and Q' ($C(q_2, q_1)$ or $C(q_1, q_2)$) must contain MDOP $<p,q>$, and can be the initial search ranges.

Theorem 4 Let object $o_1\in P$, vertex object $o_2\in Q$, $o_2\in VOR(o_1)$, $i\in\{d, u\}$. If $o_2\in Q_i$ or Q_m, then $o_1\in P_i$ or P_m.

By Lemma 1 and Lemma 2, Theorem 4 can be proved.

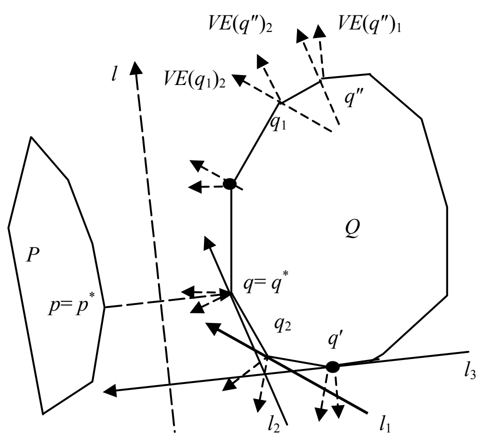


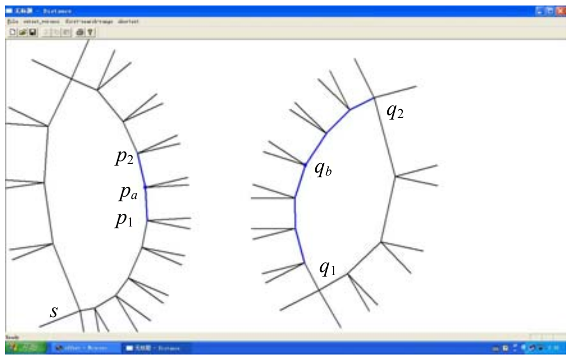
Fig.11 Find q_2

ANALYSIS AND IMPLEMENTATION

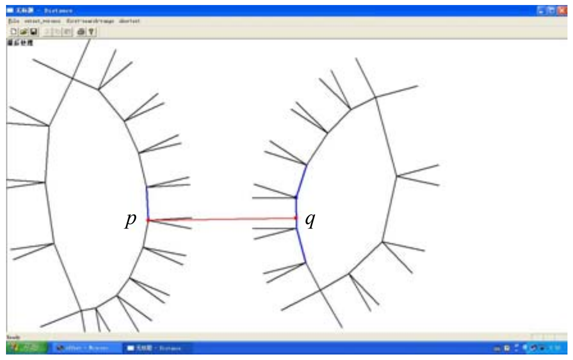
In Section 4, binary search technique used in Steps 1 and 4 costs $O(\log m)$ and $O(\log n)$ in Steps 2 and 5. As Step 3 costs $O(1)$ time, we can get that the time complexity of finding initial searching ranges is $O(\log n+\log m)$.

Since the VOR of a object of a convex polygon only has two Voronoi edges, with each Voronoi edge being perpendicular to an edge of the polygon, it only costs $O(1)$ time to compute a Voronoi edge in the algorithm, without any preprocessing and extra data structures. And binary search technique has also been used to find an MDOP $<p,q>$ in the initial searching ranges, hence, the time complexity of the algorithm in this paper is $O(\log n+\log m)$.

We have implemented our algorithm by VC++6.0 under Window XP/2000. Fig.12 gives two snapshots of computing the minimum distance between two disjoint convex polygons by our algorithm. Fig.12a shows the initial search ranges $C(p_1, p_2)$ and $C(q_2, q_1)$, where s is a vertex used to find q_1 in Q. p_a and q_b are middle vertices of $C(p_1, p_2)$ and $C(q_2, q_1)$. Fig.12b shows the MDOP $<p,q>$ that we obtained at last.



(a)



(b)

Fig.12 Samples of computing the minimum distance between two disjoint convex polygons by our algorithm. (a) Initial search ranges are obtained; (b) MDOP <*p*,*q*> is found

CONCLUSION

Computing minimum distance between two polygons based on Voronoi diagram is one of the basic and efficient methods of collision detection and path planning. Now, the lowest time complexity algorithm can compute the minimum distance between two disjoint convex polygons P and Q in $O(\log m+\log n)$ time. The paper presented a new $O(\log m+\log n)$ algorithm for computing the minimum distance between two disjoint convex polygons P and Q according to the location relations of their outer Voronoi diagrams.

The algorithm is simple and easy to implement, with no preprocessing and extra data structures being needed. It is also one part of our research on Voronoi diagram and its applications.

References

Choi, Y.K., Li, X.Q., Rong, F.G., Wang, W.P., Cameron, S., 2006. Computing the Minimum Directional Distance between Two Convex Polyhedra. HKU CS Tech Report TR-2006-01, University Of Hong Kong. Http://www.cs.hku.hk/research/techreps/document/TR-2006-01.pdf.

Cohen, J.D., Lin, M.C., Manocha, D., Ponamgi, M.K., 1995. I-collide: An Interactive and Exact Collision Detection System for Large Scale Environments. Proceedings of ACM International 3D Graphics Conference, ACM Press, New York, **1**:189-196.

Dobkin, D., Kirkpatrick, D., 1990. Determining the separation of preprocessed polyhedra—A unified approach. *Lecture Notes in Computer Science*, **443**(ICALP-90):400-413.

Edelsbrunner, H., 1985. Computing the extreme distance between two convex polygons. *J. Algorithms*, **6**(2):213-224. [doi:10.1016/0196-6774(85)90039-2]

Guibas, L., Hsu, D., Zhang, L., 2000. A hierarchical method for real-time distance computation among moving convex bodies. *Computational Geometry: Theory and Applications*, **15**(1-3):51-68.

Hudson, T.C., Lin, M.C., Cohen, J.D., Gottschalk, S., Manocha, D., 1997. V-collide: Accelerated Collision Detection for VRML. Proceeding of VRML 1997: Second Symposium on the Virtual Reality Modeling Language. ACM Press, New York, p.119-125.

Li, X.Q., Meng, X.X., Wang, C.Y., Wang, W.P., Chung, K., Yiu, S.M., 2003. Detect collision of polytopes using a heuristic search for separating vectors. *Chinese Journal of Computer*, **26**(7):837-847.

Lin, M.C., Canny, J.F., 1991. A Fast Algorithm for Incremental Distance Calculation. Proceeding of the IEEE International Conference on Robotics and Automation. IEEE Computer Society Press, Sacramento, CA, **2**:1008-1014.

Lin, M.C., Manocha, D., Canny, J.F., 1994. Fast Contact Determination in Dynamic Environments. Proceeding of the IEEE International Conference on Robotics and Automation. IEEE Computer Society Press, San Dieg, CA, **1**:602-608.

Mirtich, B., 1998. V-clip: fast and robust polyhedral collision detection. *ACM Trans. on Graphics*, **17**(3):177-208. [doi:10.1145/285857.285860]

O'Rourke, J., 1994. Computational Geometry in C. Cambridge University Press, New York, p.260.

Ponamgi, M.K., Manocha, D., Lin, M.C., 1997. Incremental algorithms for collision detection between polygonal models. *IEEE Trans. on Visualization and Computer Graphics*, **3**(1):51-64. [doi:10.1109/2945.582346]