Judging the Intersection of Convex Polygons by Bracket Method

Ying Chen
Department of Basic Sciences
Beijing Electronic Science and Technology Institute
Beijing, P.R. China
ychen@besti.edu.cn

Yaogang Du
Department of Basic Sciences
Beijing Electronic Science and Technology Institute
Beijing, P.R. China
duyaogang@besti.edu.cn

***Abstract*-In this paper, we study a basic problem based on bracket manipulations in computational geometry: how to judge if two solid convex polygons intersect or not. A key idea in our criteria is that the signs of some brackets of the homogeneous coordinates of the vertices of the two convex bodies are all we need in carrying out the judgment. Experiments show that the uniformity of representation by bracket is significant and efficient in practical computation.**

Keywords- Bracket, Computer graphics, Convex polygons Intersection

I. INTRODUCTION

Judging whether two given convex polygons intersect is a basic task in planar computational geometry and computer graphics. As we know, this problems can be easily described theoretically. The most popular method with complexity $O(\log^2 n)$ is given by Dobkin & Kirkpatrick [1, 2], Muller & Preparata [3], Shamos & Hoey [4, 5]. However, from the application point of view, the efficiency of the above criteria still bears much concern, and finding more robust detection method is an active research topic nowadays.

One interesting thing is that if two planar convex polygons are separate, then there exists a line between them but not touching any of them. This idea leads to a separation-searching algorithm. In the other side, a concept, namely bracket, is put forward in this paper. By introducing bracket [6], we can apply it to our separation-searching algorithm. Moreover, the simple and uniform algebra expression is obtained for the computation of geometric objects.

The rest of this paper is arranged as follows. In Section 2, some notations and preliminaries are introduced. In Section 3, the details of our algortithm are discussed. In Section 4, Experimental results are showed. In Section 5, conclusions are summarized.

II. NOTATIONS

Definition 1. In R^2, three points x_1, x_2, and x_3 with the coordinates form $x_i=(x_{i1}, x_{i2})$ for i=1, 2, 3, their **bracket** $[x_1\ x_2\ x_3]$ is defined as follows:

$$[x_1\ x_2\ x_3]=\begin{vmatrix} x_{11} & x_{21} & x_{31} \\ x_{12} & x_{22} & x_{32} \\ 1 & 1 & 1 \end{vmatrix}.$$

Obviously, once x_1, x_2, and x_3 are counterclockwise, the sign of $[x_1\ x_2\ x_3]$ is nonnegative and is zero if the three points are collinear. Another interesting thing is that the bracket $[x_1\ x_2\ x_3]$ can be regarded as twice the signed area of the trangle $\triangle x_1x_2x_3$. By the convexity of covex polygon, we have the following definition.

Definition 2. A convex polygon **12⋯n** is **positive orientation**, if and only if [**i(i+1)k**]>0 for any **i**, **k**=**1**, **2**, **⋯**, **n**, where **k** is not equal **i** or **i+1**.

In this paper, we always assume that the given convex polygon **12⋯n** is positive orientation. For a vertex **i** of a convex polygon **12⋯n**, the two vertices joined to **i** are denoted by **i-1** and **i+1** respectively. Such indices are always modulo **n**. Often, we drop the bold face notation for points.

Definition 3. A discrete real function f(x) where x=1, 2, ⋯, n is a **discrete unimodal function** if for some value m , it is weakly monotonically decreasing (or increasing) for $x \le m$ and weakly monotonically increasing (or decreasing) for $x \ge m$.

Figure 1 shows the example of discrete unimodal function.

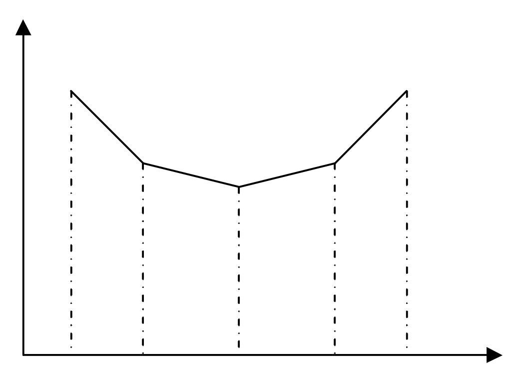

Figure 1. Discrete unimodal function

Definition 4. A discrete real function f(x) where x=1, 2, ⋯, n is a **discrete bimodal function** if for some value m , f(m), f(m+1), ⋯, f(n), f(1), f(2), ⋯, f(m-1) form a discrete unimodal function.

Figure2 shows the example of discrete bimodal function.

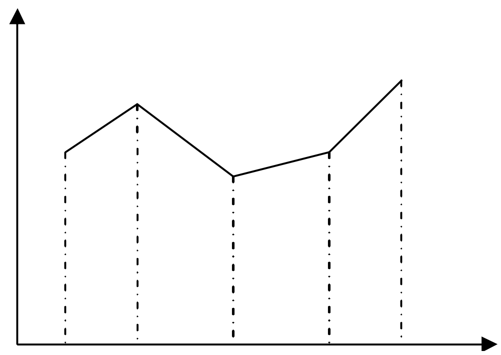

Figure 2. Discrete bimodal function

The signification of discrete bimodal function is that we can easily find its maximum value and minimum value with complexity O(log n) by [7]. Then the following lemma holds.

Lemma 1. For a convex polygon **12⋯n** and a line segment **1`2`**, then the signed distance d(**x**) where **x=1**, **2**, ⋯, **n** from vertex **x** to line segment **1`2`** is a discrete bimodal function.

Proof. Without loss of generality, we assume that all the vertices are on the same side of line segment **1`2`**, as shown in Figure 3. Clearly, d(**x**) have its maximum value and minimum value. Let d(**k**) be the maximum value and d(**i**) be the minimum one. We can see that d(**i**), d(**i-1**), ⋯, d(**k**), d(**k-1**), d(**k-2**), ⋯, d(**i+1**) form a discrete unimodal function. □

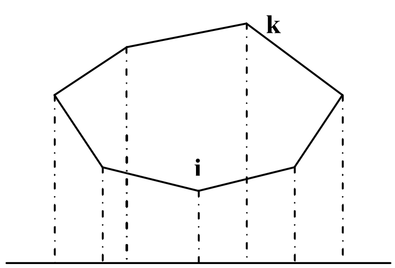

Figure 3. Proof of lemma 1

Since [**1`2`i**] can be regarded as twice the signed area of the trangle △ **1`2`i**, the signed distance d(**i**) is proportional to [**1`2`i**].

III. SEPARATION-SEARCHING ALGORITHM

For two convex polygons in one plane, the following lemma is true.

Lemma 2. Two convex polygons do not intersect if and only if there exists a line passing through one edge of a polygon such that the two polygons are on different sides of the line and one polygon does not touch the line at all.

Let **12⋯n** and **1`2`⋯m`** be two convex polygons. We have m vectors like that $M_i$=([**i`(i+1)`1**], [**i`(i+1)`2**], ⋯, [**i`(i+1)`n**]) for **i**=**1**, **2**, ⋯, **m** and n vectors like that $M`_k$=([**k (k+1) 1`**], [**k(k+1) 2`**], ⋯, [**k(k+1) m`**]) for **k**=**1**, **2**, ⋯, **n**. Based on the bracket, the above lemma can be written as:

Lemma 3. For two convex polygons **12⋯n** and **1`2`⋯m`**, they do not intersect if and only if at least one vector of $M_i$ or $M`_k$ is composed of negative elements.

Obviously, the scalars of M_i or $M`_k$ is a discrete bimodal function. That is to say, we can obtain an algorithm to judge intersection of two polygons with complexity O(nlog n).

Algorithm: Separation-searching Detection

Input: The coordinates of points **1**, **2**, ⋯, **n** and **1`**, **2`**, ⋯, **m`** respectively.
Output: "Intersection" or "No Intersection".
Step 1: Set **i**=**1**.
Step 2: If i≤**m**,
scan M_i, find the maximum value of M_i and let it be q.
If q<0,
output "No intersection" and exit.
If i>**m**,
goto **Step 4**.
Step 3: Set **i**=**i**+**1**, goto **Step 2**.
Step 4: Set **i**=**1**.
Step 5: If i≤**n**,
scan $M`_k$, find the maximum value of $M`_k$ and let it be q.
If q<0,
output "No intersection" and exit.
If i>**n**,
output "Intersection" and exit.
Step 6: Set **i**=**i**+**1**, goto **Step 5**.

IV. EXPERIMENT

In our experiments, we take one convex polygon as inscribed in a circle and the other convex polygon as inscribed in a branch of a hyperbola. The total number of tests is p_1+p_2, where p_1 is the number of intersection and p_2 is the number of separation, and each time we randomly choose n and m points sequentially from the circle and the huperblola respectively to form the vertices of our convex polygons. On a 2.40GHz CPU and 2.00GB RAM PC with the operating system of Windows 7, the tests were implemented in Matlab 6. Tab. 1 shows the performance of the different methods respectively. Though the complexity of our algorithm is more than the other methods, it can be see that our algorithm use less time in most situation from Tab.1.

V. CONCLUSION

In this paper, we present a novel approach based on bracket to solve some planar computational geometric problem. It is how to judge whether two convex polygons intersect. Although the complexity of our algorithm is O(nlog n) more than the one of traditional methods O($\log^2$ n), it based on bracket has simple expression and steady robustness. It deserves to be specially noted that only bracket manipulation in our algorithm, which can be more easily realized in hardware programming. That is to say, algorithm based on bracket has still much room for improvement in application and this is one focal point of our future works.

ACKNOWLEDGEMENT

The authors would like to thank many other workmates for their valuable suggestion. This work is supported by the National Natural Science Foundation of China (Grant No.10971217).

REFERENCES

[1] D. P. Dobkin and D. G. Kirpatrick, Fast Detection of Polyhedral Intersection, Theor. Comput. Sci., 27: pp. 241-253, 1983.
[2] D. P. Dobkin and D. G. Kirpatrick, Determining the Separation of Preprocessed Polyhedra - A Unified Approach, Proc. 17th Internat. Colloq. Automata Lang. Program., Vol. 443 of Lecture Notes Comp. Sci., pp. 401-413, Springer Verlag, 1990.
[3] D. E. Muller and F. P. Preparata, Finding the Intersection of Two Convex Polyhedra, Theor. Comput. Sci., 7: pp. 217-236, 1978.
[4] M. I. Shamos, Computational Geometry, Ph. D. Thesis, Dept. of Comput. Sci., Yale Univ., 1978.
[5] M. I. Shamos and D. Hoey, Geometric Intersection Problems, In Proceedings of the 17th Annual IEEE Symposium on Foundations of Computer Science, IEEE, New York, pp. 208-215, 1976.
[6] H. Li, M. T. Cheng, Proving Theorems in Elementary Geometry with Cliford Algebric Method, Advances in Mathematics, Vol. 26, No. 4, pp. 357-371, 1997.
[7] P. Hodge, Algorithmic Geometry, Cambridge, 1952.

TABLE 1. PERFORMANCE WITH DIFFERENT METHODS

	Time of Dobkin & Kirkpatrick (second)	**Time of Muller & Preparata (second)**	**Time of Shamos & Hoey (second)**	**Time of separation-searching (second)**
p_1=50, p_2=50, n=5, m=10.	0.1306	0.1235	0.1015	0.1232
p_1=50, p_2=50, n=10, m=15.	0.1472	0.1452	0.1278	0.1275
p_1=50, p_2=50, n=15, m=20.	0.1711	0.1679	0.1586	0.1612
p_1=50, p_2=50, n=20, m=25.	0.1892	0.1862	0.1715	0.1763

Published by Atlantis Press, Paris, France.