ON COMPUTING DEPTH OF COLLISION IN TWO DIMENSIONS

S.S.Keerthi
Department of Computer Science & Automation
I.I.Sc.,Bangalore
India

K.Sridharan
Room No. 8114
Center for Industrial Innovation
Rensselaer Polytechnic Institute
Troy, NY-12180, U.S.A.
email:rbtpfp@pawl.rpi.edu

ABSTRACT
The problem of computing the depth of collision between intersecting convex objects is considered in this paper. Two measures are given and algorithms are developed for computing the measures.

1.INTRODUCTION

Given two static objects A and B, there are several scalar measures to describe their spatial relationships. These measures are useful for applications like planning in robotics(1) and VLSI layout(2). The focus to this day has been primarily on computing the minimum distance between non intersecting objects(3),(4) .

In this paper, we concern ourselves with a problem on intersecting objects. We attempt to characterize the intensity of penetration between two intersecting objects. We define what is called depth of collision, discuss prior work on measures for depth of collision and then present a new measure. We present algorithms for computing the measures when the two objects are convex polygons.

The problem of depth of collision finds application in collision detection. Collision detection may be viewed as a sub step for path planning algorithms. Detecting the exact instances at which a collision occurs and reporting all the instances in a generate and test procedure for planning is facilitated by computing the intensity of penetration between the colliding objects.

In the next section, we define our problem and discuss a measure introduced by Cameron and Culley(5). In section 3, we present an algorithm for computing the measure defined in (5). In section 4, we develop the new measure and give some of its properties. In section 5, we give the algorithmic details for computation of the new measure. We give some ideas on computing D(d;A,B) between simple polygons in section 6. We conclude in section 7 indicating scope for further work.

2.DEFINITION OF DEPTH OF COLLISION

Let A and B be convex, compact sets in R^2 such that $A \cap B \neq \phi$. How deep is the collision between A and B?

An intuitive way of quantifying depth would be as the diameter of the largest circle contained in $A \cap B$. This measure is however inadequate. A reasonable way of defining depth of collision would be as the least distance by which B has to be translated so that it just separates from A.

A natural measure that results from this definition is the one defined by Cameron and Culley, called as MTD(Minimum Translational Distance). The authors have presented an algorithm for computing the MTD function between two convex polyhedra. The basic idea is to recast the problem of computing MTD as a configuration space problem and forming the configuration space obstacle.

Buckley and Leifer(6) have also defined the same measure independently. They report a time bound of $O((m+n)^2 \log(m+n))$ to compute the measure in three dimensions when the objects are convex polyhedra with a total of (m+n) vertices.

We will call the measure defined by Cameron and Culley as I(A,B).In the next section, we will sketch our algorithm for computing I(A,B). It turns out that computing I(A,B) is expensive in three and higher dimensions . We have given a general method based on the Fourier Elimination technique(7) to compute I(A,B). However, our method also has exponential time complexity and is not suitable when the dimension is large. We have therefore defined a new measure that is easier to compute.

3.COMPUTATION OF I(A,B)

We need the following definition:

Definition 1:Given two sets, A and B their Minkowski sum is given by

$A \oplus B=\{a+b: a \in A, b \in B\}$

The above operation can be viewed as convolution(8). Minkowski difference, $A \ominus B$, is also defined similarly.

Our algorithm is based on the linear time result of Lozano-Perez(9) and Guibas et al(8) to compute the convolution when the objects are convex polygons.

We assume the coordinates of vertices of the polygons are available. Let A and B have m and n vertices respectively.

Algorithm compute_l(A,B):
1. Find A ⊖ B in linear time using the algorithm of Lozano-Perez, as an ordered set of vertices, $u_0, \ldots, u_k$.
2.For i=1,...,k: let L_i be the line passing through u_{i-1} and u_i (indices are modulo k). L_i contains the i-th edge of A ⊖ B. Determine x_i , the point on L_i nearest to the origin and set $r_i = ||x_i||_2$.
3. Compute $l(A,B) = \min\{r_i : 1 \leq i \leq k\}$ and find an index g such that $r_g = l(A,B)$. Stop.

It is easy to see that the algorithm runs in O(m+n) time and it is optimal.

4.THE NEW MEASURE AND ITS PROPERTIES

We define the new measure, D(d; A,B) as follows:

Given $X \in R^2$, $d \in R^2$ and $\beta \in R$, let

$$X(d, \beta) = \{ x + \beta d : x \in X\},$$

and

$$X(d) = \bigcup_{\beta \in R} X(d,\beta)$$

Then

$$D(d; A,B) = \begin{cases} \max \{ \beta: A \cap B(d,\beta) \neq \phi \} , \text{ if } A \cap B(d) \neq \phi; \\ -\infty , \text{ otherwise .} \end{cases}$$

When $A \cap B(d) \neq \phi$, D(d; A,B) denotes the maximum amount by which B has to be translated along d, while still remaining in contact with A.

The properties of D(d; A,B) are summarized by the following lemma. We omit the proof since it is easy.

Lemma 1:
(i) $D(d; A,B) = D(d; A \oplus B, \{0\})$
(ii) $D(d; A,B) = D(-d; B,A)$
(iii) $D(\alpha d; A,B) = (1/\alpha)\, D(d; A,B), \ \forall\ \alpha \geq 0$

In the next section, we present an algorithm for computing D(d; A,B).

5. COMPUTATION OF D(d; A,B) BETWEEN TWO CONVEX POLYGONS

We will first introduce some notation and review some known results which we will use later.

Definition 2: A ray starting from a point a in the direction d is given by $R(a,d)=\{a+ \lambda d: \lambda \geq 0\}$.

Definition 3 : Let $\alpha, \beta \in R$ and $-\infty < \alpha \leq \beta < \infty$. Define $[\alpha,\beta] = \{ t: \alpha \leq t \leq \beta \}$; $(\alpha, \beta) = \{ t: \alpha < t < \beta \}$. Consider functions of the form $f:[\alpha,\beta] \longrightarrow R$. Then f is piecewise affine if it is continuous and there exists a finite set of points, $\{ \alpha_i \}_{i=1}^{r}$ such that
(i) $\alpha = \alpha_1 < \alpha_2 < \ldots < \alpha_r = \beta$;
(ii) the restriction of f to $[\alpha_i, \alpha_{i+1}]$ is an affine function for each $i=1,\ldots,r-1$.
The points α and β, together with those points in (α,β) at which the slopes of f has jumps are called the breakpoints of f. The set of breakpoints of f can be taken to be the same as $\{\alpha_i\}_{i=1}^{r}$. Let $f_i = f(\alpha_i)$, $i=1,\ldots,r$. Then the set $\{(\alpha_i, f_i)\}_{i=1}^{r}$ is called the sorted representation of f.

We will now present an algorithmic result. This requires a knowledge of unimodal and bimodal sequences(The reader is referred to (10)).

Proposition 1: The extrema of a bimodal sequence $\{ \delta_k \}_{k=0}^{p-1}$ can be computed in O(log p) time which involves atmost an O(log p) number of δ_k evaluations
Proof: Chazelle and Dobkin(10) give an algorithm and this complexity result

We associate bimodal sequences to the vertices of the polygons and then use the result of Proposition 1 to get the time bound for our algorithm.

Our approach to find D(d;A,B) is based on a formulation of the problem in terms of maximization of a piecewise affine concave function(The reader may refer any standard text on convexity for definitions, for instance Rockafeller(11)). We will now present several results on piecewise affine functions.

Proposition 2: A piecewise affine function f attains the maximum at one of its breakpoints.

Proposition 3: Suppose f is piecewise affine with the sorted representation, as given in definition 3. Then, given any $t \in [\alpha, \beta]$, f(t) can be computed in O(log r) time.

Proposition 4: Suppose f^1 is piecewise affine and concave with its breakpoint set given by $B\_1 = \{ \alpha_i^1 \}_{i=1}^{r_1}$. Also let f^2 be piecewise affine and convex with its breakpoint set $B\_2 = \{ \alpha_i^2 \}_{i=1}^{r_2}$. Then $f = f^1 - f^2$ is piecewise affine and concave, and its break point set , B is given by $B = B\_1 \cup B\_2$.

Proposition 5: Consider a piecewise affine concave function f with sorted representation as in definition 3. Let $\{ t_j \}_{j=1}^{p}$ be any sorted subset of $\{ \alpha i \}_{i=1}^{r}$. Then the sequence $\{ f(tj) \}_{j=1}^{p}$ is unimodal.

We will now present the algorithm to maximize the difference between a piecewise affine concave function f^1 and a piecewise affine convex function f^2. Let the sorted representations of f^1 and f^2 be $\{ (\alpha_i^1 , f_i^1) \}_{i=1}^{r_1}$ and $\{ (\alpha_i^2 , f_i^2) \}_{i=1}^{r_2}$ respectively. Let us call the result as f^*.

Algorithm max_diff:
1. Compute the maximum of the unimodal sequence, $\{f(\alpha_j^1)\}_{j=1}^{r'}$ using Kiefer's algorithm(12). Denote the maximum by F. Whenever a $f(\alpha_j^1)$ is needed by Kiefer's algorithm, evaluate it as $f(\alpha_j^1) = f_j^1 - f^2 (\alpha_j^1)$.
2. Find max $\{f(\alpha_j^2): 1 \leq j \leq r^2\}$ by a similar procedure and denote it by G.
3.Set f^*=max(F, G) and stop.

The above algorithm computes f^* in O(log r^1 x log r^2) time.

The algorithm to find D(d;A,B) takes O(log m x log n) time and is crucially based on computing f^*. The additional step is to transform our polygons A and B into unbounded polyhedra A' and B' using the following step:

A' =A +R(0,-d)
B' = B +R(0,d)
The unbounded polyhedra can be derived in logarithmic time.
Remark: The special cases when an edge of A' is grazing an edge of B' can be handled easily and D(d;A,B) will take constant time. Our algorithm can also be extended to the case $A \cap B = \phi$. In this case , D(d;A,B) denotes the minimum distance by which B has to be translated so that it moves to the 'other side' of A.

6.COMPUTING D(d;A,B) BETWEEN SIMPLE POLYGONS

The problem of computing D(d;A,B) is more difficult when the objects involved are simple polygons. This is because a ray from any point p in the plane along a direction d can intersect a convex polygon in atmost two points. But this is not true in the case of a simple polygon and there can be O(n) intersections with a simple n-gon. So a naive algorithm to compute D(d; A,B) will take O(mn) time where A and B have m and n vertices respectively.

However, it is not necessary to compute all intersections of a ray with the simple polygon for computing D(d;A,B). It is adequate if we first determine the 'last' or 'rightmost' intersection in the direction d with A (of rays from the vertices of B) and then the 'farthest' intersection of rays from the vertices of A along -d with the polygon B. So our time bound can be improved if we look closely into the work of Chazelle and Guibas(13) where the case of computing a single intersection has been considered. We believe therefore that D(d; A,B) can be computed in O(mlog n + nlog m) time.

7.CONCLUSIONS

The problem of computing the depth of collision between intersecting objects has been considered in this paper. We have defined depth of collision, given measures and also developed algorithms to compute the measures.

A natural question is computation of depth in three dimensions. We have considered this problem in (14) and given algorithms. An interesting open problem is efficient computation of I(A,B) in three

and higher dimensions. It would also be worthwhile to consider other applications of the max_diff algorithm.

REFERENCES

(1)E.G.Gilbert and D.W.Johnson, Distance functions and their application to robot path planning in the presence of obstacles, IEEE Journal of Robotics and Automation,Vol. RA-1,1985, pp.21-30.
(2)J.L.Bentley,D.Haken and R.Hon, Fast geometric algorithms for VLSI tasks, Proceedings of the Computational Conference, 1981, pp.88-92.
(3)D.P.Dobkin and D.G.Kirkpatrick, A linear algorithm for determining the separation of convex polyhedra, Journal of Algorithms, Vol. 6, 1985,pp.381-392.
(4) E.G.Gilbert, D.W.Johnson and S.S.Keerthi, A fast procedure for computing the distance between complex objects in three dimensional space, IEEE Journal of Robotics and Automation, Vol. RA-4, 1988, pp.193-203.
(5)S.A.Cameron and R.K.Culley,Determining the minimum translational distance between two convex polyhedra, Proc. IEEE Int. Conf. Robotics and Automation, 1986, pp.591-596.
(6)C.E.Buckley and L.J.Leifer, A proximity metric for continuum path planning, Proc. Ninth Int. Conf. on Artificial Intelligence, 1985, pp.1096-1102.
(7)S.S.Keerthi and K.Sridharan, Solution of parametrized linear inequalities by Fourier Elimination and its applications, Journal of Optimization Theory and Applications, Vol. 65, April 1990.
(8)L.Guibas, L.Ramshaw and J.Stolfi, A kinetic framework for computational geometry, Proc. 24th IEEE Symposium on Foundations of Computer Science, 1983, pp.100-111.
(9)T.Lozano-Perez, Spatial planning:a configuration space approach, IEEE Transactions on Computers, Vol. C-32, 1983, pp.108-120.
(10) B.Chazelle and D.P.Dobkin,Intersection of convex objects in two and three dimensions, Journal of the ACM, Vol. 34, 1987, pp.1-27.
(11)R.T.Rockafeller, Convex Analysis, Princeton University Press, Princeton, New Jersey, 1970.
(12)J.Kiefer, Sequential minimax search for a maximum, Proceedings of the American Mathematical Society, Vol.4,1953,pp.502-506.
(13)B.Chazelle and L.Guibas, Visibility and intersection problems in plane geometry, Proc. ACM Symposium on Computational Geometry, 1985, pp.135-146.
(14)S.Sathiya Keerthi and K.Sridharan, Measures of intensity of collision between convex objects and their efficient computation, Proceedings of the International Symposium on Intelligent Robotics, Bangalore, India, Jan. 1991, pp. 266-275.