ポリゴン数ｎに依存しない凸図形間交差判定アルゴリズム

仁尾　都 [1)]　徳増　真司 [2)]　越智利夫 [1)]

計算機に安定的に実装可能な凸多面体間の交差判定アルゴリズムの計算量はO（ｎ）が現時点ではベストである．この論文では，平均計算量が図形のポリゴン数ｎに依存しないアルゴリズムを提案する．曲面を含む凸図形に対応したポリゴンメッシュマップを交差判定処理前に作成しておけば，精度高く，無条件で高速に交差判定の繰返しを行う事が出来る．本アルゴリズムは新しく提案する４つの手法，すなわち，極座標ニュートン法，ポリゴンメッシュマップ，交差スクリーニング法，ニュートン法の非収束判定法より構成される．実験によって，凸多面体間の交差判定のための平均計算時間はｎに従属せず，平均的なケースでの平均収束回数はわずかの５回以下である事を確認した．

N-free Algorithm for Intersection Detection between Convex Objects

Misato Nio[1)], Sinji Tokumasu[2)], Toshio Ochi[1)]

At present, the smallest order of the computational complexity of the intersection detection algorithm between convex polyhedra, which can be stably mounted in a computer, is O (n). In this paper, an algorithm is proposed, whose average computational complexity is independent of the sum of polygons of convex objects containing curved surfaces. Once the Polygon mesh map corresponded to each object is created as preprocessing, intersection detection can be performed repeatedly, at high speed, in high decision with no conditions of objects' movements. This algorithm consists of four technique proposed newly, i.e., the Polar-coordinate Newton method, Convex-corn close conditions of convergence for Newton method, the Polygon mesh map and the Intersection screening method. An experiment using a computer made sure that the average CPU time to judge if two polyhedra cross was not subordinate to n, and in a average case the average iteration times of convergence was less than only ５.

1. Introduction

Perhaps, the problem of intersection detection between objects is one which we must cope with first, when we must treat objects by a computer. Many intersection detection algorithms are now used abundantly by CAD, CG, Games, Robot / NC control, LSI design, 3D simulation applications, etc.

Former very many researches have been continued for the purpose of reduction of the order of the computational complexity expressed by n the sum of polygons of polyhedra. If we search such papers by a internet search engine with the keyword of intersection detection or collision detection, we can get easily one thousand or more papers published in last only 20 years. About the convex polygon, the intersection

1)明星大学情報学部経営情報学科
Meisei University, Faculty of Informatics
nio@mi.meisei-u.ac.jp
2)神奈川工科大学工学部情報工学科
Kanagawa Institute of Technology, Department of Information and Computer Sciences

detection algorithm of O (logn) which B.Chazelle and others announced in 1980 [1] is known as the fastest for the present. About convex polyhedra, some O (n) algorithms [2, 3] are proposed, because they can be mount stably in a computer.

The reason we use intersection detection algorithms frequently is that it is necessary to move objects or objects move in a computer. Therefore, it is a rare case that an intersection detection algorithm is used only 1 time or a decimal time in an application program. Moreover, in most cases the shapes of objects do not change. That is, in most cases an intersection detection algorithm is used repeatedly to the same objects. Many conventional algorithms are not always optimized for repetition use. As an algorithm which thought repetition use as important, Closest-feature Algorithm [4, 5, 6] is mentioned. They maintain the pair of the polygon which is always in the shortest distance of two objects during there continuous movements, using the coherence. Therefore, when a quick or big movement happens, the algorithm can trace the pear of polygon hardly.

In this paper, an algorithm for intersection detection of convex objects containing curved surfaces, whose average computational complexity is independent of the number n of polygons of objects, is proposed. Once Polygon mesh maps corresponded to each objects are created as preprocessing, intersection detection can be performed repeatedly, at high speed with no conditions of objects' movements. Moreover, since approximation of objects is omitted, highly precise intersection detection can be performed. This algorithm consists of four technique developed newly.

(1) The **Polar-coordinate Newton method** which enables intersection calculation between binary convex functions like convex objects.

(2) **Convex-corn close conditions of Newton method** which can close calculation to an early stage of the convergence of Newton method when objects do not cross.

(3) The Θ (1) algorithm by **Polygon mesh map** for the location problem on a planar curved graph.

(4) The Θ (1) **Intersection screening algorithm** of detecting intersection between two convex objects in the short distance.

Since Newton method is carried out on condition that it uses by the orthogonal coordinate system, it is inconvenient for performing intersection calculation between binary convex functions like convex objects. The bucket method [7] of Θ (1) to the location problem on a planar straight-line graph is announced. But, the method needs large size memory, and more larger memory if the graph is a planar curved graph. O(1) un-intersection screening algorithm using bounding sphere [8] which guarantees not crossing to two objects in the far distance is well known, but Θ (1) **Intersection screening algorithm** which guarantees crossing to two objects in the short distance is not proposed yet.

2. Polar-coordinate Newton Method and its application to intersection detection

The Polar-coordinate Newton Method and its application to intersection detection algorithm between convex close curved surfaces is proposed. Suppose that two differentiable, convex, closed, and curved surfaces C_i (i=1,2) are defined by two local θ - ψ -r polar-coordinate systems whose starting point is O_i (O_i is lapped in C_i), and there is an intersection between them, and BS (C_i) is the minimum sphere which includes C_i. d(p,q) means the distance of point p and q.

(1) If $d(O_1, O_2) \geqq$ radius of $BS(O_1)$+ radius

of BS(O_2), stop the procedure, noting that C_1 and C_2 do not intersect.

(2) (**Intersection screening algorithm**) If the following two conditions are not satisfied simultaneously, stop the procedure, noting that C_1 and C_2 intersect.

a. Intersection $P_{1,1}$=(segment line O_1O_2) $\cap C_1$ and intersection $P_{2,1}$= (segment line O_1O_2)$\cap C_2$ exist.

b. O_1, $P_{1,1}$, $P_{2,1}$, and O_2 are located in a line with the order.

(3) j= 1, and choose suitably the point K_j in the global coordinates where C_i are arranged, as an initial point.

(4) Let L_j be the intersection between the tangential planes $G_{i,j}$ of C_i at the intersection $P_{i,j}$ (= $O_iK_j \cap C_i$).

(5) If j=1 or j> 1 and one of the following close conditions of intersection convergence calculation is satisfied, stop the procedure.

a. It is a procedure end noting that two objects intersect, if $d(P_{1,j}, P_{2,j}) < \varepsilon$.

b. If j exceeds the maximum iteration count of convergence, and if times of reselection of initial point K_1 does not exceeds the maximum times of reselection of K_1, then it is a convergence end noting that the initial position K_1 is not suitably selected and go back to (3), else it is a procedure end noting that two objects do not intersect.

c. (**Convex-corn close conditions of Newton method**) The half-space of the direction where C_i exists between two half-space made by $G_{i,j}$ is set to HS ($G_{i,j}$), and V_j (Convex-corn) is set to HS($G_{1,j}$) $\cap$ HS ($G_{2,j}$). If one of following conditions is satisfied, it is a procedure end noting that they do not intersect.

c-1. $V_j \cap V_{j-1} = \phi$.

c-2. $V_j \cap V_{j-1} \cap BS(C_1) = \phi$ or $V_j \cap V_{j-1} \cap BS(C_2) = \phi$.

c-3. $BS(C_1) \cap HS(G_{2,j}) = \phi$ or $BS(C_2) \cap HS(G_{1,j}) = \phi$.

(6) If the foot point p of the perpendicular taken down from K_i to L_j is included in V_j, K_{j+1}=p, else set the point in V_j which is nearest from K_i to K_{j+1}.

(7) j =j+1, and back to (4).

The Intersection screening theorem of the following guarantees that the Intersection screening algorithm of (2) is right, because a convex object is a star-shaped object and any inner point of a convex object is the kernel point the convex object.

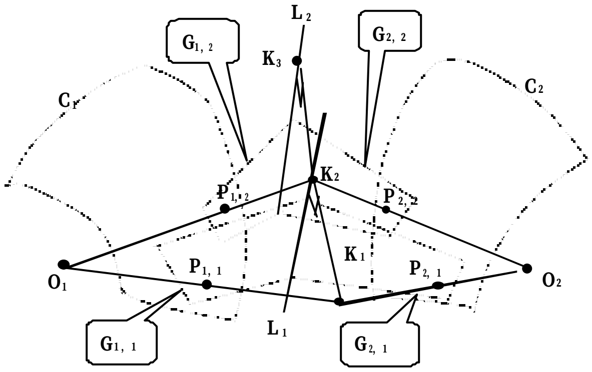

Fig.1 Procedure of detecting intersection between C_1 and C_2 by the Polar-coordinate Newton method

(**Intersection screening theorem**)

Supose the kernel of two star-shaped 3D objects Q_i (i=1,2) is K_i and ∂ (Q_i) is boundary of Q_i. If the following two conditions are not satisfied simultaneously, Q_1 and Q_2 intersect.

a. Intersection $R_1 \in$(segment line K_1K_2) $\cap \partial$ (Q_1) and intersection $R_2 \in$(segment line K_1K_2)$\cap \partial$ (Q2) exist.

b. K_1, R_1, R_2, and K_2 are located in a line with the order.

Proof: If R_1 and R_2 do not exist, K_2 is a inner point of Q_1 because K_1 is the kernel point of Q_1. If only R_2 exists, R_2 and K_2 are inner points of Q_1 because K_1 is the kernel point of Q_1. If R_1 and R_2 exist and K_1, R_2, R_1, and K_2 are located in a line with the order, R_2 is inner points of Q_1 because K_1 is the kernel point of Q_1. □

As explained after, if Q_i is composed of n curved surfaces and its Polygon map, a planar curved graph, is a star-shaped graph, the computational complexity of intersection detection algorithm is $\Theta(1)$ by using the Polygon mesh map maid form the polygon map .

3. Intersection Detection Algorithm between 3D convex objects composed of polygons

Suppose that 3D convex object Q_i (i= 1,2) has curved surfaces $C_{i,k}$ (= $f_{i,k}(\theta, \psi)$) (k= 1, 2, .., m_i) (called polygon) which were defined by the polar coordinate which makes O_i ($\in Q_i$) the starting point, and in which differential is possible.

In this case, the search processing for the polygon $C_{i,j}$ which O_iK_j intersects is necessary. (4) of Chapter 2 is rewritten as follows.

(4) The coordinates values of θ axis and ψ axis of K_j in the local coordinates which defines Q_i are referred to as $\theta_{i,j}$ and $\psi_{i,j}$. Search for $C_{i,k}$ which intersects O_iK_j and the intersection between the tangential planes $G_{i,j}$ of $C_{i,k}$ at the point $P_{i,j}$(=($\theta_{i,j}$, $\psi_{i,j}$, $f(\theta_{i,j}, \psi_{i,j})$) is set to L_j.

4. Polygon Mesh Map

Call a planar curved graph G (V, E, P) a star-shaped graph, whose P (polygons in G) is all star-shaped. The mesh map of a star-shaped graph G for $\Theta(1)$ algorithm for the location problem on G is proposed.

(1) The mesh map covers G.

(2) The mesh map has a tree structure, and a set of a child's mesh covers parents' mesh.

(3) The mesh lines are parallel to a rectangular-coordinates axis.

(4) The number of levels of the tree structure is below the constant t.

Using the mesh map, the average number of P which any mesh of the lowest level contains will be made to below a given number. When a star-shaped polygon P has many vertices, the computational complexity of whether q is contained in P can be set to Θ (1) by assigning the 1-dimensional mesh map of P. Therefore, the computational complexity of search for P including the query point in a star-shaped graph is $\Theta(1)$.

(Theorem of location problem on a star-shaped graph)

By using a 2D Polygon mesh map of a star-shaped graph G and a set of 1D Polygon mesh map given to each P of the graph, the computational complexity of a location problem on G is $\Theta(1)$.

The map which is obtained by mapping the ridgeline of the polygons of a 3D object onto the θ - ϕ orthogonal coordinate system is called a Polygon map, and the mesh map of the Polygon map is called a Polygon mesh map.

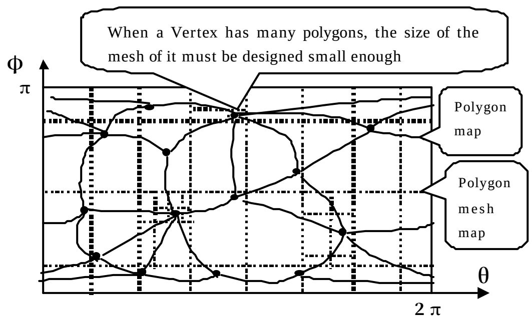

Fig. 2 Polygon map and its Polygon mesh map

A point-inclusion problem can be converted to a location problem. To solve a point-inclusion problem or a search problem for the polygon of polyhedron Q with a query point q in 3D space which intersects Oq (O is the origin point of Q, O$\in$Q, and O is starting point of Oq), P of the polygon map of Q is not necessary to be star-shaped, because we can judge that the half line Oq intersects the polygon(mapped to P) of Q in

3D space in time of $\Theta(1)$ by the 1D Polygon mesh map given to the polygon in 3D. Even if a polygon of Q is a curved surface, the point-inclusion problem is $\Theta(1)$ under the condition that the Polygon map of Q is star-shaped. Therefore, the intersection algorithm can be applied to convex objects composed of curved surfaces.

5. Performance Measurement Result

Fig.3 and Fig.4 show the performance measurement of the proposed intersection detection algorithm of two convex polyhedra mounted in a computer.

Two objects Q_1 and Q_2 are inscribed in a ellipse whose long axis is r_1 and short axis are r_2 and r_3. In order to simplify explanation, let the shape of Q_1 is the same of that of Q_2 , $r_1=1$, and $r_1 \geqq r_2 \geqq r_3$. Q_i is expressed with the following formula.

$x^2/r_1^2+y^2/r_2^2+z^2/r_3^2=1$

O_i is set to the center of the ellipse. O_1 is fixed and O_2 is distributed uniformly in the sphere of radius=2. So, any un-intersection screening by bounding sphere is not effective.

The short axis r_2 and r_3 and the number of vertices of two ellipses can be separately inputted interactively, and the deflection angles and zenithal angles of all vertices and layout arguments of Q_1 and Q_2 are automatically given by the uniform random number. Pentium4 (2Mhz) was used. Whenever the value of n was changed, the average time was measured after random 200,000 times of trials of intersection detection.

Fig.5 and Fig.6 show the results in the case of 2D. In 2D, the condition of c-2 and c-3 of Convex-corn close conditions of Newton method (Chapter 2) were not amounted. In the case of 2D, Visual Basic Application for Excel was used as a programming language, and in the case of 3D, C++.

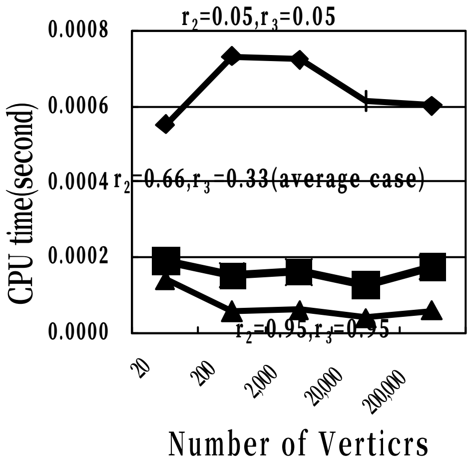

Fig.3 CPU time (3D)

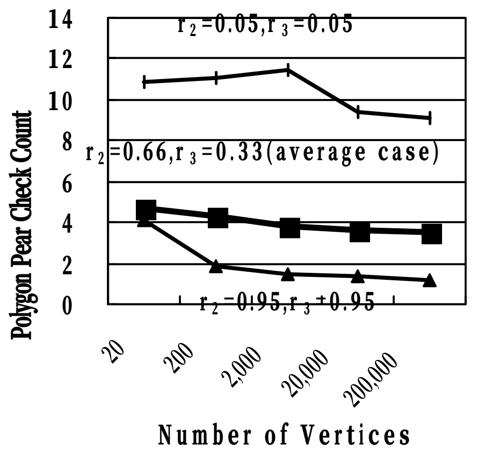

Fig.4 Polygon Pare Check Count (3D)

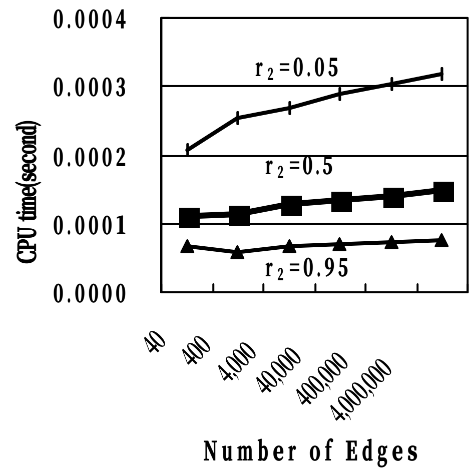

Fig.5 CPU time (2D)

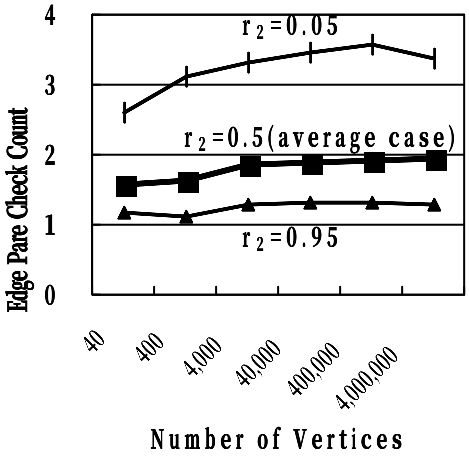

Fig.6 Edge Pare Check Count (2D)

When intersection screening is successful, the polygon pair check counter is set to 1, and in the case of others, the counter is set to the iteration counts of the Polar-coordinate Newton method. The computational complexity of this algorithm is almost equal to the polygon pair check counter, because one Intersection screening operation needs one pare of polygon, and the same pare is reused as first pare for the Polar-coordinate Newton Method, and each search of one pare needs two solutions of location problem whose computational complexity is $\Theta(1)$. Fig. 3,4,5,6 show that the computational complexity of this algorithm is not dependent on n but on the shape of object (ratio of r_1,r_2,r_3). As r1/r3 is smaller, angle of two polygons are larger, so the convergence of Newton method is more difficult. But, it must be emphasized that the polygon (edge) pair check count was less than 5 in the average case of 3D.

6. Examination of Result

6.1 Dependability to n of Computational Complexity and Calculation Time

If r_2=1 and r_3=1, the Intersection screening algorithm becomes always effective. As r_2 and r_3 becomes smaller, or $d(O_1,O_2)$ becomes larger, the percentage of success of the Intersection screening algorithm becomes smaller, and the polygon pair check counter larger.

But, as r_2 and r_3 becomes much smaller, or $d(O_1,O_2)$ becomes much larger, the probability of intersection becomes smaller, and Convex-corn close conditions become more effective, the polygon pair check counter becomes smaller, and finally converges to 2 near the effective zone of un-intersection screening by bounding sphere. The change is shown in Fig. 7 (in the case of a 3-dimensional intersection detection experiment (in the case of number of vertices n=2,000).

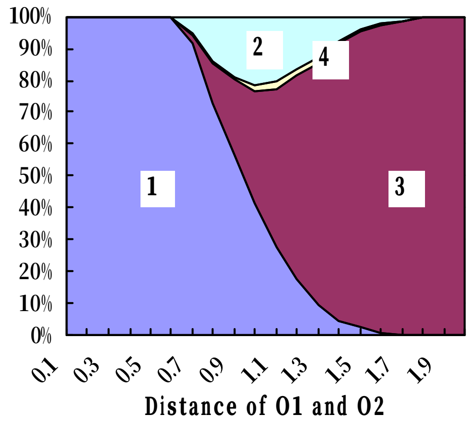

Fig.7 Percentages of Close Condition applied. (r_2=0.66,r_3=0.33)

Area of 1 in Fig.7 means the percentage of the cases of intersection detected by Intersection screening, 2 that of intersection detected by successful convergence of Newton method, 3 that of un-intersection detected by Convex-corn close conditions, and 4 that of un-intersection detected by iteration counter over.

The average of Fig. 7 becomes Fig. 4. Fig. 7 means large contribution of the Intersection screening algorithm and Convex-corn close conditions to the iteration counts of the Polar-coordinate Newton method. It is a surprising thing that the percentage of case in which the intersection detection by the Newton method was successful (area number is 2) was only 6%. Though the average iteration counts of the Polar-coordinates Newton method depends on n in small degree (it may state later, total average of polygon pair check counts does not depend on n (fig.4,6).

6.2 Comparison of Polar-coordinate Newton Method and Newton Method

From the position of the Polar-coordinate Newton method Newton can be interpreted as follows.

Newton method is for calculating the intersection of the straight line (x-axis) and a convex object, that is, a single-valued convex function y=f (x). The x-axis corresponds to the angle axis of a polar-coordinate system, and y-axis to the radius vector axis, and the starting point of the object f(x) is (0, +∞), or (0, -∞).

The feature of the Polar-coordinate Newton method is that it can ask for the intersection of between two closed curves defined by two local polar-coordinate systems.

The algorithm has also proposed a new intersection calculation technique between the curved surfaces. The conventional method used in the CAD field needs to search for the point $P_{i,j}$ on the surface as the foot point of perpendicular passing K_j onto C_i. The Polar-coordinate Newton method recommends $P_{i,j}$ as the intersection of OK_j and the surface.

6.3 Computational Complexity of Intersection Detection

Since the Polar-coordinate Newton method is enhanced method of Newton method, its convergence is perhaps secondary convergence. The convergence speed in the case of asking for the equal root falls down. When a group of ridge-lines whose length becomes shorter by the fixed ratio a (<1) in a row near the contacting point of a polygon and a line (Fig. 9), computational complexity of the Polar-coordinate Newton method must be set to O (n) or the larger order.

Though its average computational complexity is very small like Newton method, it depends on n in small degree. The reason is that if n becomes larger, the probability of density of polygons nearby the contact point between two polygons (polyhedra) becomes larger.

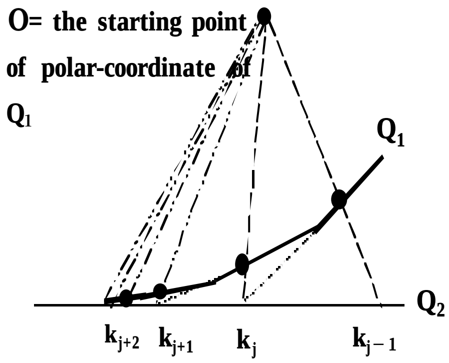

Fig. 9 the worst case of contact between two objects Q_1,Q_2

6.4 Memory size of Polygon Mesh Map

The average ratio of memory size of the 1-dimensional polygon mesh map of a polygon to the (x,y) data of vertices of the polygon was about 0.86, even if the hierarchy depth was 2 and the number of restrictions of the edges which overlaps a mesh was 2. And the standard deviation of the memory size was only 0.01. So, it can be said that the memory size is Θ(n) under the condition that a deflection angle is uniform distribution. The preprocessing time was Θ(n).

The average ratio of memory size of the 2D polygon mesh map of the polyhedra was 2.2, that is Θ(n),and preprocessing time wasΘ(nlogn).The room of reduction of the ratio is still left behind greatly. But, the memory size of the polygon mesh map will be much smaller than that of the voxels table which divide 3D Boronoi Region into 3D voxels 〔6〕,because the polygon mesh map is maid by division of 2D space into 2D meshes. Research of the optimal division technique of the polygon map which minimizes the amount of memory is a future subject, subject to a restricted number of a polygon which overlaps a mesh. However, since a polygon map is the concave figure with the curve, the optimal mesh division problem will be difficult to solve.

6.5 Data Structure of Polyhedron

This algorithm does not need the contiguity relation between polygons, so the polygon mesh map holds only the polygon number of a polyhedron. Therefore, the polyhedron which consists of polygon soup can be treated, so this algorithm is easy to use for many applications.

7. Conclusion

The algorithm for intersection detection of convex objects whose average computational complexity is independent on the number of polygons is proposed. An experiment using a computer made sure that the average CPU time to judge if two polyhedra cross was not subordinate to n the sum of polygons, and in a average case the average iteration times of convergence was less than only 5.

8. Acknowledgement

Great cooperation of Mr. Eiji Kamiya of the incorporated company ibis was obtained in mounting of 3D algorithm to a computer, gratitude is expressed here.

References

1. B.Chazelle and D.Dobkin. DETCTION IS EASIER THAN COMPUTATION, ACM 0-8791-017-6(1980)146-152.
2. E.G.Gilbert, D.W.Johnson and S.S.Keerthi. Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space, IEEE Journal of Robotics and Automation, Vol.4, No.2, April (1988)193-203.
3. D.Dobkin, J.Hershberer, D.Kirkpatrick. Computing the Intersection-Depth Polyhedra, (1993),http://citeseer.nj.nec.com/caschedpage/145776.
4. M.C.Lin and J.F.Canny. Efficient algorithm for incremental distant computation. In IEEE Conference on Robotics and Automation,1991.
5. B.Mir.tich. V-Clip:Fast and robust polyhedral collision detection. Technical Report TR-97-05, MERL, July 1997.
6. Katsuaki Kawauchi and Hiromasa Suzuki. Distance Computation between Non-convex Polyhedra at Short Range Based on Discrete Voronoi Regions.
7. H.Imai and K.Imai. Computational Geometry, Kyoritu-Syuppan, (1994).
8. Sean Quinlan. Efficient Distance Computation between Non-Convex Objects,(1994),http://plan9.bell-labs.com/cm/who/seanq/icra94.ps