Enhancing Collision Detection Method Based on Enclosed Ellipsoidby Facial Areas Splitting

Wei-Kun Su , Jing-Sin Liu, I-Fan Chen

Institute of Information Science 20

Academia Sinica

Nankang, Taipei 115, Taiwan, R.O.C.

Email: liu@iis.sinica.edu.tw

Abstract

This paper is aimed at introducing a faster collision detection algorithm for convex polyhedral in a three-dimensional workspace. The problem of collision detection has been researched in the literature of computer graphics, robotics, computational geometry, computer animation, and physically-based modeling. It has been regarded as a computationally demanding task and is often treated as an advanced feature. In this paper, we present a faster way to solve the collision detection problem by using the enclosed ellipsoid method. Enclosed ellipsoid method gets fewer collision error report than the standard bounding volume method, such as boxes or spheres. We split ellipsoids into facial areas and ignore the areas which are always the heavier computational load but impossible to be collided with. There are some computer simulations showing how the facial split method increases the efficiency of the detection in comparison with the original enclosed ellipsoid method.

1. Introduction

Computing distance between two objects is an important problem in various fields, including computer graphics, animation and simulation. Using mathematical models of objects, we find that there is a certain point on each object that makes the distance between them minimum. If the point of one object is at the face of the other object, they are regarded as collided with each other. Bounding volume is one of the most important methods in the collision detection area. It can enclose the complex structure of an object by covering it with a simple geometric object such as a sphere. The sphere is a good bounding volume to reduce the computational work because it only requires checking the distance between the centers of spheres. But the tradeoff of sphere bounding volume method is that it contains too much redundant spaces, which fails to make collision detection precise. So there are a lot of bounding volume methods presented to make the detection more precise and less computational works, such as OBB[10], Enclosing Ellipsoid[3] and Enclosed Ellipsoid[7,8]. An ellipsoid is capable of representing a convex polyhedron, such as robot's links, in the direction of its axis. Besides, the main advantage of ellipsoid model is that it is very simple in mathematical representation; therefore it can reduce the complexity of computations to be required. An ellipsoid is represented as $e^n(\mathbf{y},\mathbf{Y})$ in this paper, where n is the dimension, y is the center, and Y is the characteristic matrix. Enclosing Ellipsoid is a better bounding volume compared with sphere or box because it can fit the original object tighter. But there are still unnecessary space bounded by enclosing ellipsoid, especially for the rectangle object which is the most common part in robots. Then enclosd ellipsoid method presented by [7,8] makes a new ellipsoid which is inside the original object from enclosing ellipsoid. By comparing the nearest points between the two enclosed ellipsoids, we can determine whether the objects are collided with each other or not. Although enclosed ellipsoid already reduces much computational work in collision detection, there are still some cases in which to perform full faces checking is needed, and

it costs about 70% of the full collision detection time.(section [5.1]) So we focus on the full faces checking cases and make it a more efficient detection.

2 Previous work

Since collision detection is needed in a wide variety of situations many different methods have been proposed. To decrease the computation complexity and increase the performance of collision detection, objects are generally modeled as a simpler primitive or as a union of simpler primitives[15]. However, the selection of primitives should reflect a good balance between the efficiency of primitive-primitive intersection detections and the number of primitives required to adequately represent the world model. In the proposed algorithm, ellipsoids are selected to represent objects.

2.1 Oriented bounding boxes

Oriented bounding boxes (OBB) perform better than axis-aligned boxes or spheres because they fit the enclosed objects closer. OBB can recursively partition the bounded polygons and bound the resulting groups to build the OBB trees.[10]

2.2 Enclosing Ellipsoid

Löwner-John (L-J) ellipsoid, the minimum-volume enclosing ellipsoid of a convex object, is an intuitively appealing means to lump the detailed geometry into a single quadratic surface. The computation of the L-J ellipsoid is a convex optimization problem [3] whose solution can be derived by applying the ellipsoid algorithm [16].

2.3 Enclosed Ellipsoid

It is a very difficult to generate an enclosed ellipsoid with maximum volume for a convex polyhedron since the procedure depends on the geometrical shape and the selection of the center coordinate of the enclosed ellipsoid.[7] Therefore, the proposed method focuses on applying geometrical deformation operations on the L-J ellipsoid to generate an enclosed ellipsoid which is as large as possible to fit the polyhedron tightly. For our implementation, a 3-phase approach, which shrinks, stretches, and then scales an L-J ellipsoid, is proposed.[8]

Phase.1: Isotropically shrinking all principal axes

An initial enclosed ellipsoid is given by shrinking the L-J ellipsoid along its principal axes isotropically to be contained in the polyhedron in phase 1. To guarantee that the polyhedron contains the initial ellipsoid, the ellipsoid $e^3(\mathbf{y}, 16\mathbf{Y})$ is selected to be the initial guess for enclosed ellipsoid computation in 3-dimensional case [9]. The regulation of the shrinking factor is based on the bisection method. The phase terminates with a user-defined error while the ellipsoid cannot extend further without intersecting with the facets of a polyhedron.

Phase.2: Stretching the enclosed ellipsoid

The phase 1 terminated while the enclosed ellipsoid is very close to one of the polyhedron's facets; however, it still has some free space to enlarge the enclosed ellipsoid. Stretching operation [8] is applied to expanding the enclosed ellipsoid along a given direction in phase 2. Let $\mathbf{s}$ be the point to adapt to and $e^3(\mathbf{c}, \mathbf{M})$ be the enclosed ellipsoid generated in phase 1. The idea is to move the ellipsoid's center towards to the point, i.e. $\mathbf{s}$, and then stretch the ellipsoid along the movement direction such that the old border point in the opposite direction remains a border point. Therefore the new center is represented as

$$\mathbf{c}' = \mathbf{c} + b\,(\mathbf{s} - \mathbf{c}),$$

where b determines how far to move the ellipsoid's center. With the normalized distance vector

$$\mathbf{a} = \mathbf{M}^{1/2}(\mathbf{s}-\mathbf{c}) / \left\| \mathbf{M}^{1/2}(\mathbf{s}-\mathbf{c}) \right\|,$$

the new transformation matrix is given as $\mathbf{M}'^{1/2} = (\mathbf{I} + (a-1)\mathbf{a}\mathbf{a}^T)\mathbf{M}^{1/2}$,

where $a = 1/(1 + \left\| b\mathbf{M}^{1/2}(\mathbf{s}-\mathbf{c}) \right\|)$.

It is worth to notice that enlarging an ellipsoid means that its transformation matrix makes the vectors shorter, therefore a is always smaller than 1. In the stretching operation, $\mathbf{s}$ is given as $l \cdot (\mathbf{s}_m - \mathbf{c}) / \left\| \mathbf{s}_m - \mathbf{c} \right\|$, where l is the distance from the farthest facet of the polyhedron to $\mathbf{c}$, the center of enclosed ellipsoid, and $\mathbf{s}_m$ is the mass center of vetices of the farthest facet. In our implementation, b is initialized as 1 and inside the range from 0 to 1. The selection of b is also based on the bisection method. The algorithm terminates while the variation of b is smaller than a user-defined threshold.

Phase.3: One by one enlarging each radius

Let $e^3(\mathbf{c}', \mathbf{M}')$ be the enclosed ellipsoid generated by means of stretching. Since the matrix $\mathbf{M}'$ is symmetric and positive-definite, it can be diagonalized through a rotational matrix $\mathbf{V}$. The relation is expressed as $\mathbf{D} = \mathbf{V}^{-1}\mathbf{M}'\mathbf{V}$.

In fact, matrix $\mathbf{V}$ is the matrix of eigenvectors of matrix $\mathbf{M}'$ and matrix $\mathbf{D}$ is a diagonal matrix with $\mathbf{M}'$'s eigenvalues on the main diagonal. Since the inverse square roots of matrix $\mathbf{M}'$'s eigenvalues are equivalent to the length of principal axes of the enclosing ellipsoid, the change of ellipsoid's each radius can be performed individually by means of multiplying matrix $\mathbf{D}$ with a scaling matrix $\mathbf{S}$, which is also diagonal. Therefore, each new radius of the enclosed ellipsoid can be written as $\mathbf{D}' = \mathbf{S}\mathbf{D}$, and the enlarged enclosed ellipsoid can be represented as $\mathbf{M}' = \mathbf{V}\mathbf{D}'\mathbf{V}^{-1}$.

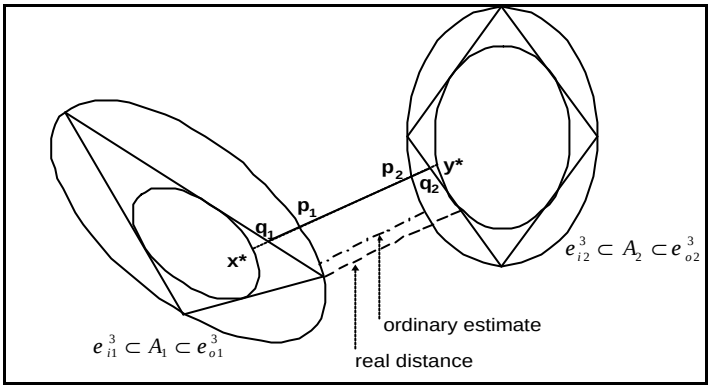


Figure. 0. The distance estimates based on enclosed ellipsoids

Phase.4: Nearest points computing

After the three phases computing, we have both enclosing ellipsoids and enclosed ellipsoids at the same time.(Figure.0) X* and Y* are the nearest points between both enclosed ellipsoids.the line passes through X* and Y* will although intersects original objects as points Q_1 and Q_2, intersects enclosing ellipsoids as points P_1 and P_2.With the criterion $\overline{\mathbf{q}_1\mathbf{q}_2} > \min(\overline{\mathbf{p}_1\mathbf{q}_1}, \overline{\mathbf{p}_2\mathbf{q}_2}, threshold)$, collision detection can be fast computed by the upper bound and lower bound limitations.

3 Facial Area Splitting

3.1 Original computational time

Enclosed ellipsoid method consists of (a)make a LJEllipsoid for the object (b)isotropically shrink the LJEllipsoid to enclosed ellipsoid (c)streteching the enclosed ellipsoid (d)enlarging each radius of ellipsoid (e)computing X*, Y* (f) lower bound and (g) upper bound. Chart.1 shows the time spending in each step . but (a) (b) (c) (d) are the processes which are computed only a single time. They need not to be computed again until objects become deformed or divided. So during most of the time, collision detection only uses the (e) (f) (g) steps. Chart.2 shows that (f) step is the largest proportion of time consuming. That the (f) step takes the largest proportion is due to the full faces checking. It' s used to search the point on each object intersected by the nearest points of both enclosed ellipsoids. this paper focuses on this drawback and provides an improvement on it.

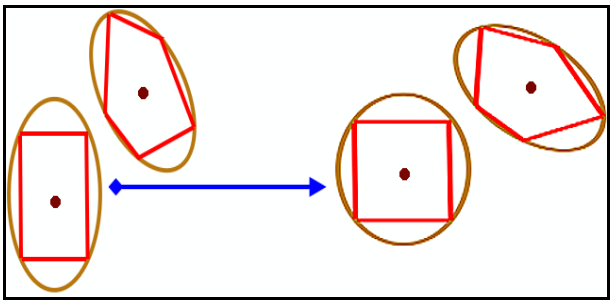


Figure.1 (a)left picture: It' s hard to determine the nearest point between 2 ellipsoids. (b)right picture: it' s much easier to determine if the collision occurs between a sphere and a ellipsoid.

3.2 Transform ellipsoid into sphere

There are two reasons why we need to transform the ellipsoid into the sphere. First, it' s not easy to compute the nearest point between ellipsoids. Second, spheres are simpler for the latter facial area splitting. The general equation of the ellipsoid is

$$\frac{(x-x_0)^2}{a^2}+\frac{(y-y_0)^2}{b^2}+\frac{(z-z_0)^2}{c^2}=1$$

Ellipsoid can also be represented by matrix form.

$$\begin{pmatrix} x & y & z \end{pmatrix} \begin{pmatrix} a & \frac{d}{2} & \frac{f}{2} \\ \frac{d}{2} & b & \frac{e}{2} \\ \frac{f}{2} & \frac{e}{2} & c \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 1 = X\ M\ X^T$$

so we can present an ellipsoid as $e^n(x_0, M) = M^{-1/2}B_0 + x_0$, where B_0 represents a unit ball centered at the origin and $M^{1/2}$ is the square root matrix M . Suppose x is a position vector in world coordinate, while the ellipsoid is transformed into a ball centered at the origin of a coordinate, the position vector x with respect to the new coordinate is presented as $x' = M^{1/2}(x - x_0)$. With this equation, we can transform ellipsoid into spheres. So the distance between two ellipsoids becomes the distance between sphere and ellipsoid which is equal to the distance between a point and ellipsoid but the result needs subtracting the radius of sphere.

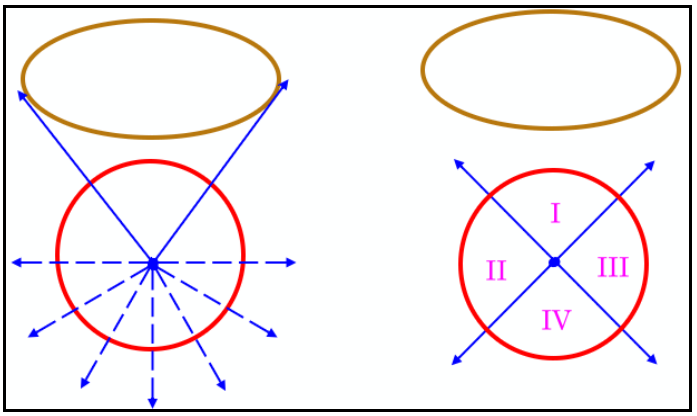


Figure.2 The left picture shows the major portion of surface is unnecessary to check if collided. Right picture shows an facial area splitting example that only part I need to be checked.

3.3 Splitting the Sphere Surface

During the (e) (f) (g) steps of enclosed ellipsoid method, full faces checking is necessary to determine the nearest point at each object . Actually, only a minor portion of surface needs checking, so superfluous work was done but it is just a waste of time. The dash line in figure.2 left picture means unnecessary checking surfaces of sphere. So we can split the sphere surfaces like figure.2 right picture, and use certain way to ignore part II, III and IV because only part I needs to be considered in collision detection. We could split the sphere surface into 8 parts. There is a unique vector in every part which could be used to check if this part could be ignored or not. We assume that P_1 is the center of the ellipsoid, P_2 is the center of the sphere. $\overline{P_1P_2}$ means the vector from P_1 to P_2. The collision is impossible to happen if the dot value is smaller than $-\sqrt{2}/2$ (this value will be proved in Section 4). So we could just do 8 times vector checking and then ignore all the parts which the dot value is smaller than $-\sqrt{2}/2$. All the faces on the ignored part of surface won't be computed during the collision detection, so we could determine the nearest point on each object faster.

4 Facial Area Splitting at the Sphere

4.1 Proposition.1 :

There is at least half of surface of sphere which is unnecessary to be checked if collided with a ellipsoid in enclosed ellipsoid method.

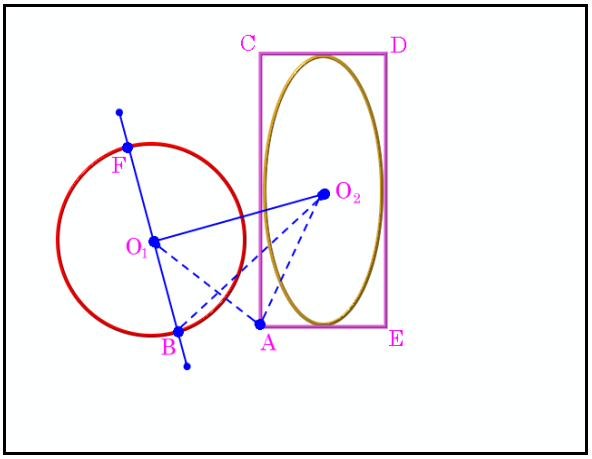


Figure.3 the projection of the Sphere, the Ellipsoid and the box.

Prove : We assume that O_1 is the center of the sphere, O_2 is the center of the ellipsoid. We can find a box which has the same center as the ellipsoid and could just cover the ellipsoid . Project all the three objects to a certain plane, and the box could be projected as a rectangle. $\overline{O_1O_2}$ is the line between sphere and ellipsoid centers. $\overline{FB}$ and $\overline{O_1O_2}$ are perpendicular to each other. So Angle $\angle O_2O_1B = 90°$. If the point A of the rectangle is possible to move to point B, then $\overline{AC}$ will be the line tangent to point B. It will make $\angle O_1AC = 90°$, but $\angle O_1AO_2 \rangle \angle O_1AC$. So that $\angle O_1AO_2 \rangle 90°$ .Combining the angle $\angle O_2O_1B = 90°$ and $\angle O_1AO_2 \rangle 90°$ will find out that triangle ΔO_2O_1B is impossible to exist because of the overflow sum of angles. So A is impossible to move to point B and C is impossible to move to point F for the same reason. Finally, the left half part $\overset{\frown}{BF}$ of surfaceof the sphere is

impossible to be touched by rectangle $AEDC$, and so does the ellipsoid covered by the rectangle. So Proposition.1 is proved.

4.2 Proposition.2:

Suppose P_1 is the center of the ellipsoid and P_2 is the center of the sphere. The collision is impossible to happen if the dot value between $\overline{P_1P_2}$ and the vector of a facial area is smaller than $-\sqrt{2}/2$.

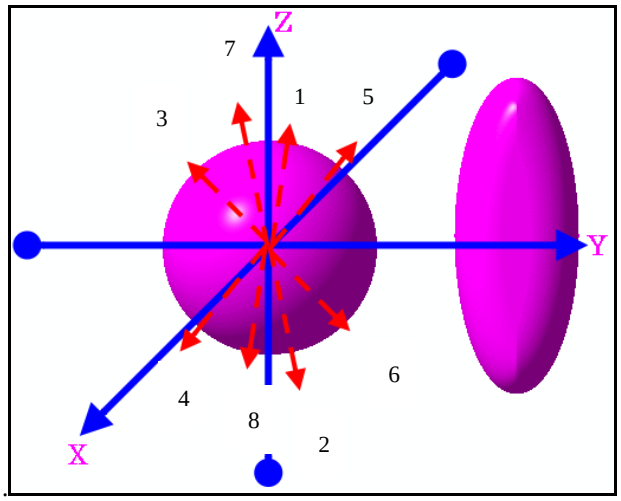


Figure.4 The 8 vectors are the dash lines as well as facial spltting vectors for the Sphere.

A Object	X	Y	Z
Vec.1	+1	+1	+1
Vec.2	+1	+1	-1
Vec.3	+1	-1	+1
Vec.4	+1	-1	-1
Vec.5	-1	+1	+1
Vec.6	-1	+1	-1
Vec.7	-1	-1	+1
Vec.8	-1	-1	-1

Prove: In figure.4, there are 8 vectors of facial areas and the ellipsoid at the right side of the sphere. According to the Proposition.1, we know that at least half of the surface is not necessary to be checked if colliding with the ellipsoid. But if we have already split the surface of sphere into facial areas, then we need to consider again about the dot between centers vector and facial areas vectors because that center vectors are not always the same as one of the facial vectors. Figure.5 is the easier version to explain by projecting figure.4 objects into a plane. Ellipsoid E is at the right side of Sphere S ant the center vector is the same as one of the facial vectors, so Ellipsoid E only needs to take part I and part II into consideration to check if any collision occurs according to *Proposition.1*. But considering another Ellipsoid F, there is no facial area vectors the same as center vector $\overline{FS}$. So Ellopsoid F needs to consider both the facial vectors near himself and 2 answers are obtained—(1)part I and part IV need to check, (2)part III and part IV need to check. So the conclusion from (1) and (2) can only discard the part II. The best condition occurs at the moment when centers vector and facial areas vector are the same that the computational wok is only half of full. The worst condition occurs when the centers vector is at the middle position between 2 facial areas vectors and the included angle between center areas and each nearest facial areas vector is $45°$. So in the possible collision, the lower bound of the dot value of centers vector and facial areas vector would be extended from 0 to $-\sqrt{2}/2$ ($\cos(90°)$ to $\cos(90°+45°)$).That means if the dot value is smaller than , the facial areas are impossible to collide with the ellipsoid. Proposition.2 is proved.

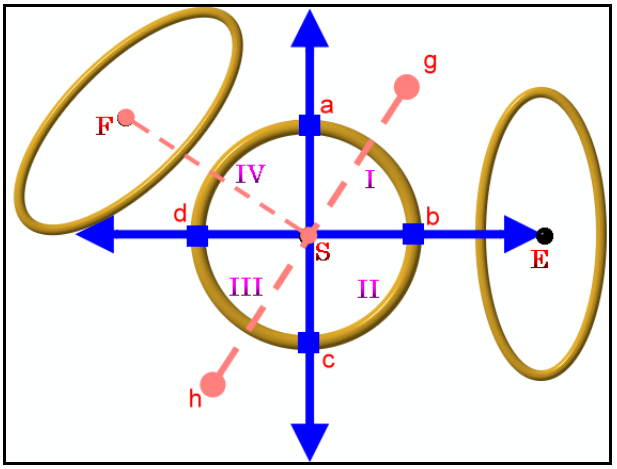
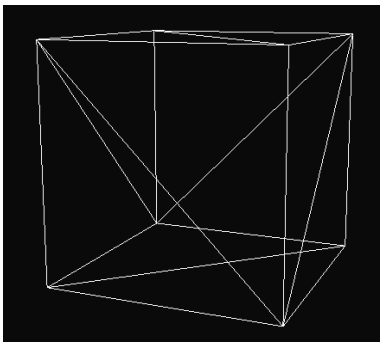


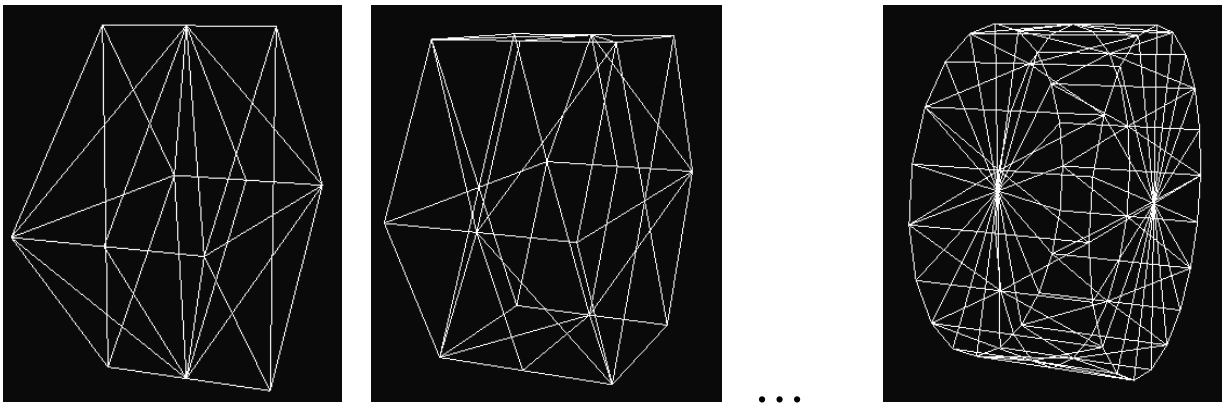
Figure.5 Projection of figure.4 with the additional ellipsoid F.

5 Simulation and Discussion

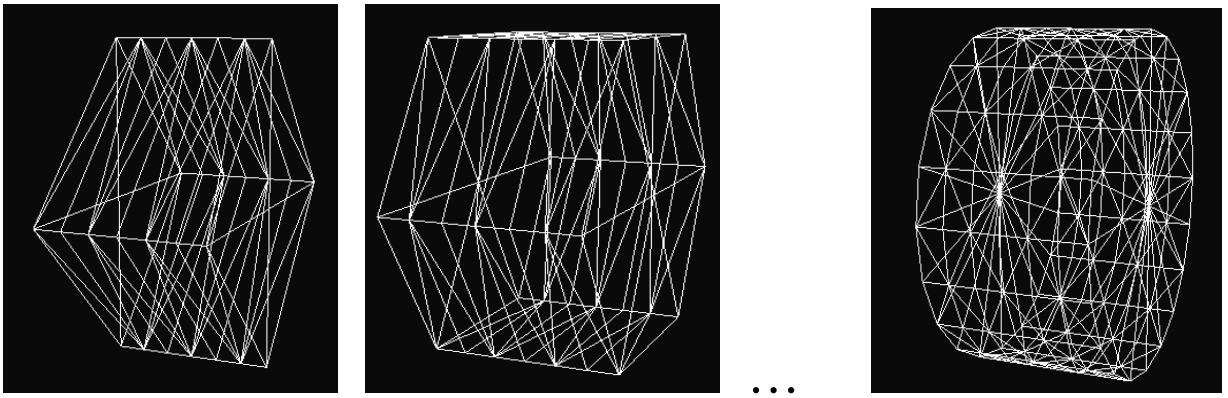
We use the Pentium3-1.1GHz CPU with 384 MB system rams to run the simulations by Visual C++ under Microsoft Windows Xp. Here are the models we use in the simulation.



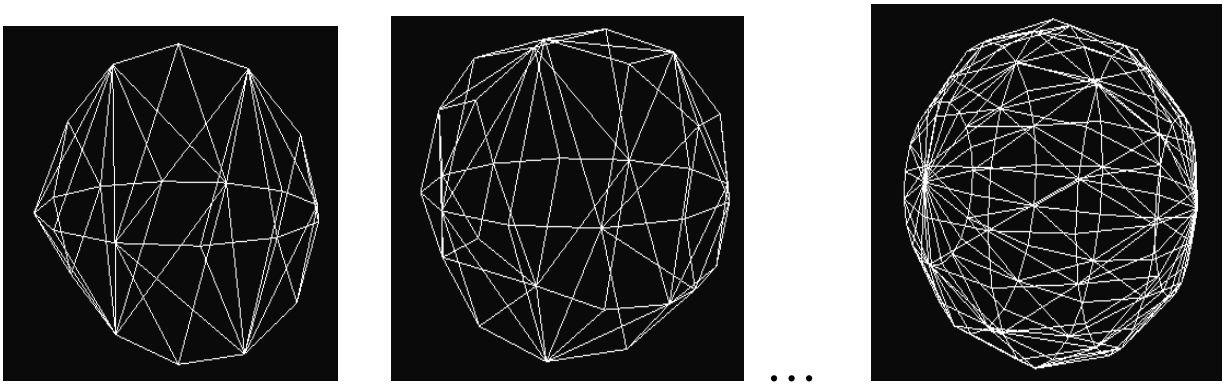
Obj.0: Box(simplest)



Obj.Type1: Cylinders [C02] (10 models)



Obj.Type2: Cylinders(Complex) [C06] (10 models)



Obj.Type3: Spheres (Complex) [S06] (10 models)

	Vertex	Face	Face (after Face splitting)
Box	8	12	36
Cylinder	14 ~ 62	24 ~ 120	59 ~ 212
Sphere	22 ~ 102	40 ~ 200	123 ~ 630
Cylinder(Complex)	30 ~ 142	56 ~ 280	91 ~ 308

Chart 0. Details of the Objects' Models

The Obj.0(Box) is the moving object that the collision detection need to be performed in every time frame with other objects. The reason why we choose box be the moving object is that box is the simplest object. The facial areas splitting method can reduce more computational works if objects are more complex. So it' s better to transform complex object info sphere and splits its surfaces than simple object.

5.1 The computational time of each enclosed ellipsoid step.

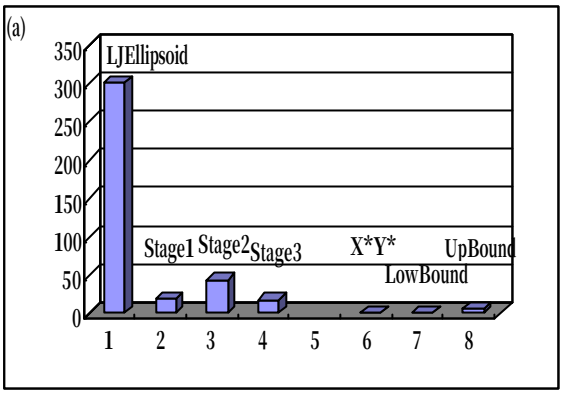


Chart.1: the Unit of Y axis is millisecond

There are 2 kinds of steps of enclosed ellipsoid method. One is offline steps, such as (a)make a LJEllipsoid for the object (b)isotropically shrink the LJEllipsoid to enclosed ellipsoid (c)streteching the enclosed ellipsoid (d)enlarging each radius of ellipsoid. The other kind is online steps, such as (e)computing X*, Y* (f) lower bound and (g) upper bound. This chart shows that the offline steps take much more time than online steps. But offline steps are always computed one time only. Online steps compute faster but the computing work won't stop until the experiment is finished.

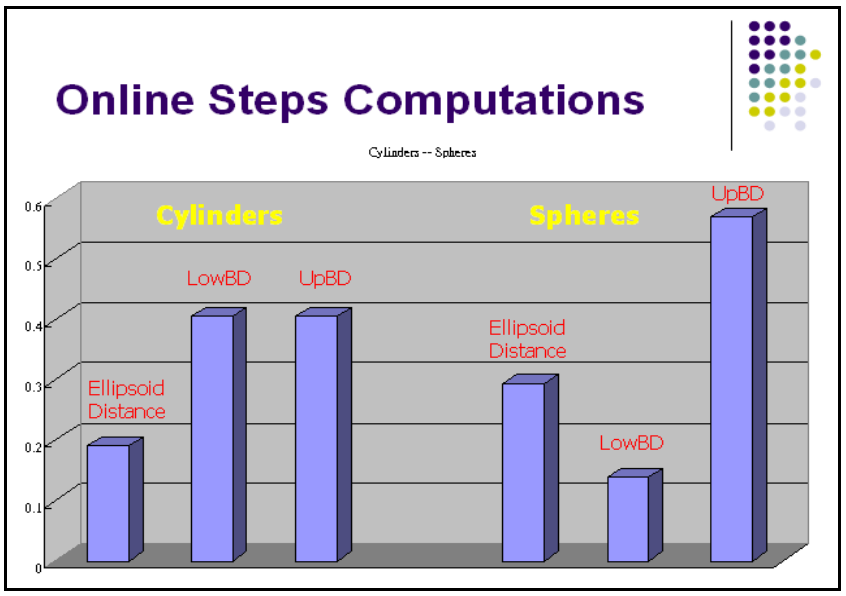


Chart.2(a) the percentages(y axis) of 3 online steps of 2 types models.

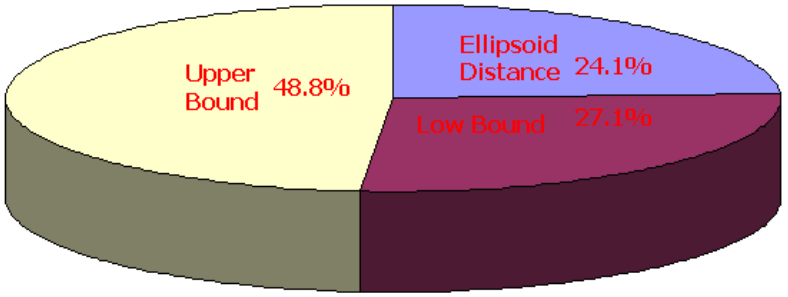


Chart.2(b) the averaged percentages of 3 online steps.

Chart.2 shows that the major part of online steps is the stage of computing upper bound, which is also the only one stage requiring the full faces checking. This paper focuses on this stage and presents a new method to reduce the faces checking computation.

5.2 Facial Areas Splitting vs Original Enclosed Ellipsoid method.

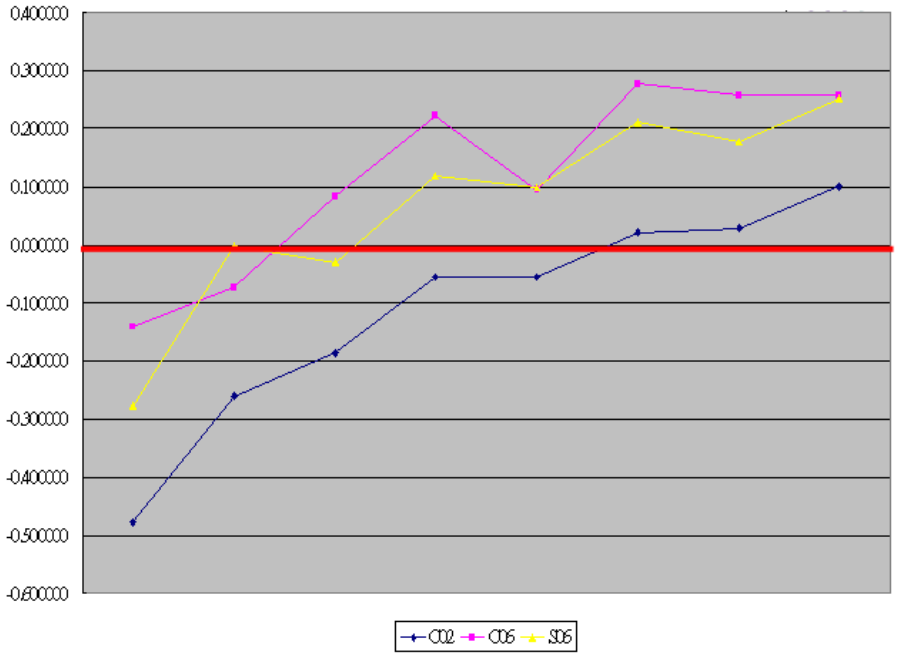


Chart.3(a) the individual time of FAS computing.

the Unit of Y axis is millisecond

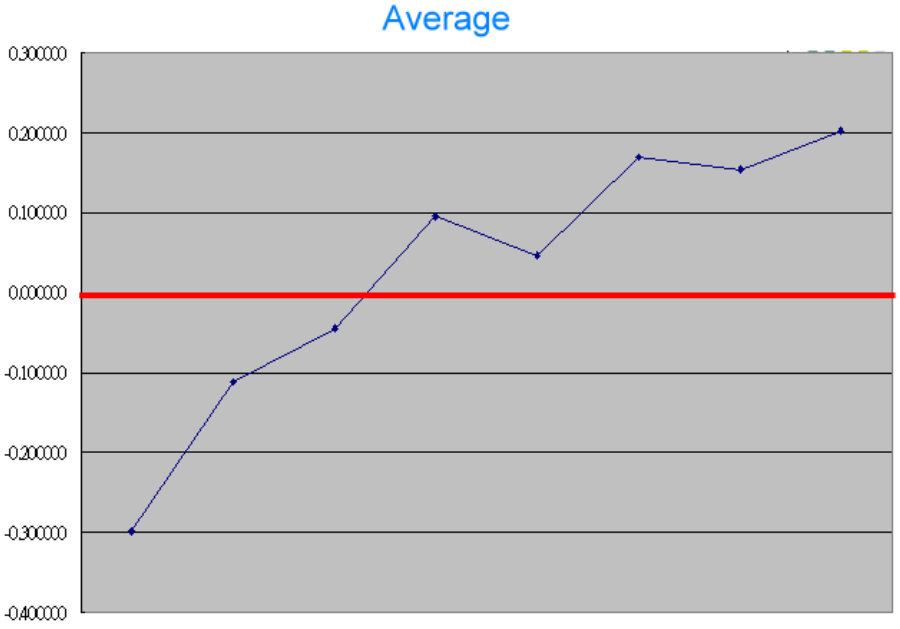


Chart.3(b) the average time of FAS computing.

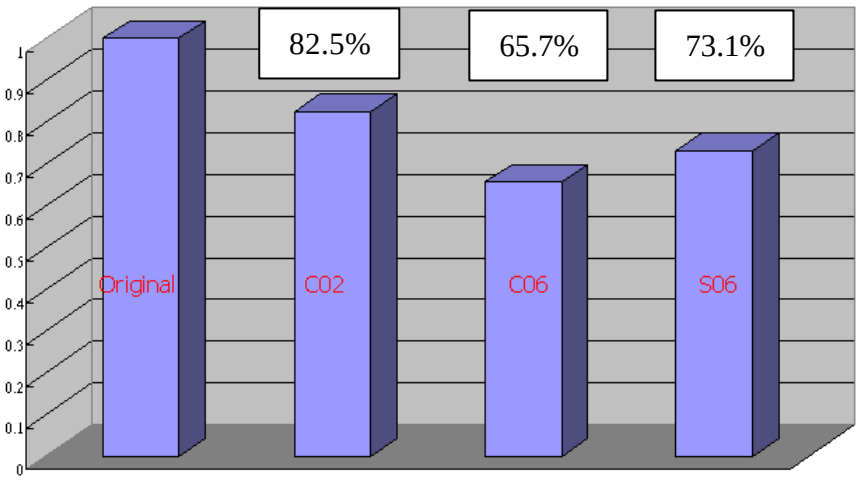


Chart.3(c) the Percentage(y axis) of the faces reduced from the Model.

X axis at chart.3 stands for the Original and 3 types of objects. This chart shows the face reduced by the facial area splitting(righter 3 objects) and original enclosed ellipsoid method(leftest) by all objects. The more face reducing stands for the faster computing ability for collision detection. There are 2 abnormal results that facial areas splitting is slower than original EEM in Chart.3(a). The abnormal results should be due to the simpler object models. This facial areas splitting method for this simutation is splitting surfaces into 8 areas. If the number of faces of original object is less than 8, FAS is impossibly faster than full faces checking because of the checking times is smaller than 8.

Totally, the original EEM takes the average 0.788 seconds while FAS only spends 0.701 second. Consequently, FAS is 14.29% faster than original EEM.

5.3 Total Improvement in Online Steps

We know that Upper bound is the major part of online steps such as (e)computing X*, Y* (f) lower bound and (g) upper bound. Chart.2 at 5.1 also tells us that upper bound take 48% load in online

steps. And section5.2 tells us that FAS could reduce 26.23% faces at online Upper Bound step. So we totally increase the 14.9% performance for online steps of enclosed ellipsoid method.

6 Conclusion

The Proposition.2 shows that the theoretical worst case of facial areas splitting occurs at the moment when dot value is lesser than $-\sqrt{2}/2$. Among all the surfaces of the sphere, there is always 1/8 = 12.5% areas free of collision. The simulation resulting from section 5.2 is about 26.23%. However, the result is worse than ideal average case. That means there is still much space left to improve the facial areas splitting. One way is to split the spheres into more and smaller faces. More and smaller faces can get the porportion of collision free areas bigger as well as reduce more the redundant computational work.

References

[1] M. C. Lin, and D. Manocha, J. Cohen., S. Gottschalk. "Collision Detection:Algorithms and Applications," in Proc. of Algorithms for Robotics Motion and Manipulation, pp. 129-142 , 1996

[2] T. Hudson, M. Lin, J. Cohen, S. Gottschalk and D. Manocha. "V-COLLIDE: Accelerated Collision Detection for VRML," in Proc. of VRML, 1997

[3] E. Rimon and S. P. Boyd, "Obstacle collision detection using best ellipsoid fit," Journal of Intelligent and Robotic System 18, pp. 105–126, 1997

[4] M. Lin and J. Canny, "A fast algorithm for incremental distance calculation," Proc. of the IEEE Int. Conf. Robotics and Automation, pp. 1008-1014, 1991

[5] S. Quinlan, "Efficient distance computation between non-convex objects," Proc. of the IEEE Int. Conf. Robotics and Automation, pp. 3324-3329, 1994

[6] D. P. Dobkin and D. G. Kirkpatrick, "A linear algorithm for determining the separation of convex polyhedra," J. Algorithm 6, pp. 381-392, 1985

[7] S. P. Shiang, J. S. Liu and Y. R. Chien, "Estimate of minimum distance between convex polyhedra based on enclosed ellipsoids," Proc. of the IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 2000

[8] M. Y. Ju, J. S. Liu, S. P. Shiang, Y. R. Chien, K. S. Hwang and W. C. Lee, "A Novel Collision Detection Method Based on Enclosed Ellipsoid," in the 2001 IEEE Int. Conf. Robotics and Automation, 2001

[9] S. Cameron. Enhanceing. "GJK: Computing minimum penetration distances between convex polyhedra," In IEEE InternationalConference on Robotics and Automation, April, 1997

[10] S. Gottschalk, M. C. Lin, and D. Manocha.. "OBBTree: A hierarchical structure for rapid interference detection," In Computer Graphics Proceedings, Annual Conference Series, pages 171–180. ACM SIGGRAPH, 1996

[11] K. E. H. III, T. Culver, J. Keyser, M. Lin, and D. Manocha. "Fast computation of generalized voronoi diagrams using graphics hardware," In Computer Graphics Proceedings, Annual Conference Series, pages 277–285. ACMSIGGRAPH, 1999

[12] G. Garcia and J. F. Le Corre, " A new collision detection algorithm using octree models," Proc. of the IEEE/RSJ Int. Workshop on Intelligent Robots and Systems (IROS'89), pp. 93-98, 1989

[13] J. Cohen, M. Lin, D. Manocha, and M. Ponamgi. "ICOLLIDE:An Interactive and Exact Collision Detection System for Large-Scale Environments," In Proc. of ACM Interactive 3D Graphics Conference, pages 189–196, 1995

[14] M. Held, J.T. Klosowski, and J.S.B. Mitchell. "Evaluation of collision detection methods for virtual reality fly-throughs," In Canadian Conference on Computational Geometry, 1995

[15] Brychcy, T. and Kinder, M., " A Neural Network Inspired Architecture for Robot Motion Planning," Proceedings of the International Conference on Engineering Applications of Artificial Neural Networks (EANN' 95), pp. 103-109, Helsinki, Finland, Aug. 1995

[16] M. Grotschel, L. Lovasz and A. Schrijver, Geometric Algorithms and Combinatorial Optimization 2nd corrected ed., Springer-Verlag, Berlin, 1993