Fast and Accurate Collision Detection Based on Enclosed Ellipsoid

Ming-Yi Ju*,, Jing-Sin Liu*, Shen-Po Shiang*, Yuh-Ren Chien*, Kao-Shing Hwang** and Wan-Chi Lee***

***Institute of Information Science 20
Academia Sinica
Nankang, Taipei 115, Taiwan, R.O.C.**

****Department of Electrical Engineering
National Chung Cheng University
Chiayi 160, Taiwan, R.O.C.**

Abstract

A fast and accurate method for detecting the collisions of convex polyhedra in a graphical simulation environment based on a newly developed method of distance estimate is presented. By the use of the enclosing and the enclosed ellipsoids of convex polyhedra, potential collisions can be detected more accurate than those methods using only bounding volume for object representation, and more efficient than the polyhedral methods. An approach for computing the enclosed ellipsoid of a convex polyhedron by compressing, stretching and scaling operations on its best-fit enclosing ellipsoid is introduced. Graphical simulations of moving objects in three-dimensional space and multiple robot arm system applications are conducted to demonstrate the accuracy of the proposed algorithm for collision detection and minimum separating distance computation.

***Keywords*:** distance estimate, collision detection, ellipsoid model, bounding volume representation.

1 Introduction

Methods for computing minimum separation distance and detecting intersections of three-dimensional objects play an important role for many applications in robotics, virtual reality, computer graphics and visualization. There are many research efforts devoted to the collision detection problems [1-7]. In many cases the performance of the overall application is greatly determined by the efficiency of distance computation algorithms. Several methods based on polyhedral model for computing the Euclidean distance between two polyhedra are proposed [11-13]. They interactively find pairs of points, one on each convex polyhedron, so that the distance between the points monotonically converges to the minimum. However, such kind of methods has a computational expense which is nearly linear with the total number of vertices. To reduce the computational complexity for real-time applications, an alternative method, called bounding volume scheme, modeling a polyhedron with a simpler geometrical primitive or a union of primitives to simplify the procedure of minimum distance estimation is adopted widely. Consequently, how to accurately model convex polyhedra is very important for efficient minimum separating distance computation and collision detection.

In bounding volume scheme, it is very common to represent an object using a sphere [2, 3]. The reason for adopting such a simple primitive is to reduce the complexity in representation and collision detection since a sphere is invariant in its orientation. However, e.g. for robotic applications, such kind of representation does not have enough accuracy since the link of a robot arm is usually rectangular. To increase the representational accuracy, a representation approach called dynamic spheres, which are parametric volumes composed of an infinite number of spheres, the positions and radii of which vary linearly over the extent of the object [4]. Hierarchical spherical representations were also proposed for more accurate representations [5, 6, 17]. The collision radii are chosen so that the collision detection time is low and the available free workspace for each object is as large as possible. However, a large number of spheres are needed to represent longer objects with reasonable accuracy, and hence the computation cost for collision detection increases. Besides, when the geometric primitives are near or in collision, such method must check all the pairs of spheres for intersection detection, not efficient in the real-time applications.

Octree is also a widespread hierarchical representation of bounding volume schemes for detecting collision [18, 19]. Octree representation methods recursively decompose objects up to a given resolution level and maintain these boxes using a tree structure. It is easy to detection collision by means of searching the nodes of the tree structure, however, it is not suitable and efficient to represent dynamic objects for separating distance computation and collision detection.

For the applications of robot manipulators, cylinder [21] or the combination of a cylinder with two hemispheres [22, 23] are popular representation to model a robot manipulator due to the geometrical shape of links. Unfortunately, with respect to a Cartesian coordinate system, their mathematical expressions become complicated so that restricts the real-time operation.

Ellipsoids are also used as primitives for collision detection due to their simple geometric features [7, 16]. However, detecting intersection between two ellipsoids is not an easy task and such bounding volume models also suffer from representation accuracy for collision detection.

The aim of this work is to develop an efficient and accurate collision detection method based on a newly developed distance estimate between convex polyhedra to overcome the problems encountered in bounding volume schemes. The proposed method approximates convex polyhedra by enclosing and enclosed ellipsoids, then the collision detection procedure can be performed by computing the distance between two convex polyhedra through their enclosed ellipsoids and their distance estimate errors. The outline of this paper is as follows. Section 2 describes the approach for the construction of enclosing and enclosed ellipsoids of convex polyhedra. Distance estimates of polyhedra based on enclosed ellipsoids are introduced in Section 3. Section 4 presents the proposed method for collision detection and the simulation results are shown in Section 5. Finally, Section 6 concludes the paper.

2 Ellipsoid Models

Complex objects are generally represented as a union of simpler primitives [10]. However, the 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. For the proposed algorithm, ellipsoids are selected to represent objects. 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. As for a non-convex polyhedron, it can be decomposed into a union of convex polyhedra or be paved to form a new convex polyhedron; then the proposed algorithm can be applied to generating enclosing and enclosed ellipsoids. Since the aim of the paper focuses on fast distance estimate, how to convert a non-convex polyhedron into a convex one is not investigated anymore in the remains of this article. An ellipsoid is represented as $\varepsilon^n(\mathbf{y}, \mathbf{Y})$ in this paper, where n is the dimension, **y** is the center, and **Y** is the characteristic matrix.

2.1 Enclosing Ellipsoid

Löwner-John (L-J) ellipsoid, the minimum-volume enclosing ellipsoid of a body, 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 [7] whose solution can be derived by applying the ellipsoid algorithm [8].

2.2 Intersection Check

Since the intersection check plays an important role for enclosed ellipsoid computation, the procedure about how to perform the check is introduced first. To confirm that a convex polyhedron contains an ellipsoid, it is necessary to check whether the ellipsoid intersects with the facets of the convex polyhedron or not. Unfortunately, the distance computation between an ellipsoid and a certain plane is very complex. Therefore some coordinate transformations are brought in to simplify this problem.

The basic idea is to transform an ellipsoid into a unit ball centered at the origin and compute the distance from the ball to the transformed plane. Let $\varepsilon^3(\mathbf{x}_0, \mathbf{X})$ be an enclosed ellipsoid. Since matrix **X** is positive-definite and symmetric, the ellipsoid can be represent as $\mathbf{X}^{-1/2}\mathbf{B_O}+\mathbf{x}_0$, where $\mathbf{B_O}$ represents a unit ball centered at the origin and $\mathbf{X}^{1/2}$ is the square root matrix **X**. 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 represented as

$$\mathbf{x}' = \mathbf{X}^{1/2}(\mathbf{x}\text{-c}). \tag{1}$$

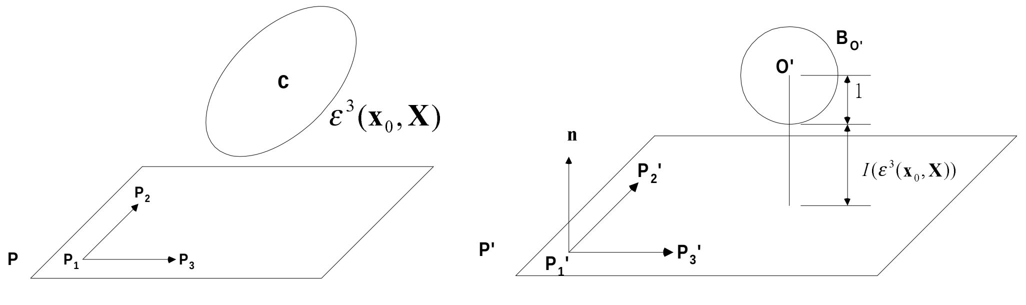
As shown in Fig 1(a), take three points $\mathbf{P_1}$, $\mathbf{P_2}$, and $\mathbf{P_3}$ (for example, three vertices) on the facet **P** into consideration. Through the transformation mentioned in Eq(1), they become $\mathbf{P_1}'$, $\mathbf{P_2}'$, and $\mathbf{P_3}'$, respectively, and form a plane $\mathbf{P}'$ with respect to the new coordinate as shown in Fig 1(b). Then, the plane $\mathbf{P}'$ in three-dimension can be represented as

$$\mathbf{n} \bullet \mathbf{x}' = d, \tag{2}$$

where $\mathbf{n}$ is the normal vector of the transformed plane and is given as $(\mathbf{P_2}' - \mathbf{P_1}')\times(\mathbf{P_3}' - \mathbf{P_1}')$. The d can be derived easily by submitting $\mathbf{P_1}'$, $\mathbf{P_2}'$, or $\mathbf{P_3}'$ into Eq(2). Since the center of the transformed ball is the origin of the new coordinate, therefore, the distance between the transformed ball and the transformed plane can be calculated simply by

$$I(\varepsilon^3(\mathbf{x}_0, \mathbf{X})) = \frac{|d|}{\|\mathbf{n}\|} - 1, \tag{3}$$

where $\|\mathbf{n}\|$ is the norm of normal vector $\mathbf{n}$. As depicted in Fig.1 (b), $I(\varepsilon^3(\mathbf{x}_0, \mathbf{X}))$ is a signed distance and used to be the indicator for checking whether the unit ball intersects with any face of the convex polyhedron or not. If $I(\varepsilon^3(\mathbf{x}_0, \mathbf{X}))$ is small than zero, it means that the case of intersection.


c
$\varepsilon^3(\mathbf{x}_0, \mathbf{X})$
P
P_1
P_2
P_3
$B_{O'}$
O'
1
n
P_2'
$I(\varepsilon^3(\mathbf{x}_0, \mathbf{X}))$
P'
P_1'
P_3'

(a) (b)

Fig. 1. Intersection check procedure for enclosed ellipsoid computation. (a) The original coordinate. (b) The equivalently transformed coordinate.

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. Therefore, the proposed method just focuses on generating 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.

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. Let $\varepsilon^n(\mathbf{y},\mathbf{Y})$ be the minimum volume *n*-ellipsoid containing a convex polyhedron in *n*-dimensional space. Then, the initial enclosed ellipsoid is given as $\varepsilon^n(\mathbf{y},(n+1)^2\mathbf{Y})$, formed by shrinking $\varepsilon^n(\mathbf{y},\mathbf{Y})$ from its center by a factor of $(n+1)$, to guarantee that the polyhedron contains the initial ellipsoid.[10] Therefore, the ellipsoid $\varepsilon^3(\mathbf{y},16\mathbf{Y})$ is selected to be the initial guess for enclosed ellipsoid computation in 3-dimensional case. The regulation of the shrinking factor is based on the bisection methods. 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 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 [9] is applied to expanding the enclosed ellipsoid along a given direction in phase 2. Let $\mathbf{s}$ be the point to adapt to and $\varepsilon^3(\mathbf{c},\mathbf{M})$ 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

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

where β 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

$$\mathbf{M}'^{1/2} = (\mathbf{I} + (\alpha - 1)\mathbf{a}\mathbf{a}^T)\mathbf{M}^{1/2},$$

where

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

It is worth to notice that enlarging an ellipsoid means that its transformation matrix makes the vectors shorter, therefore α is always smaller than 1. In the stretching operation, **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 **c**, the center of enclosed ellipsoid, and $\mathbf{s}_m$ is the mass center of vetices of the farthest facet. In our implementation, β is initialized as 1 and inside the range from 0 to 1. The selection of β is also based on the bisection method. The algorithm terminates while the variation of β is smaller than 0.005.

As mentioned the old border point in the opposite of the stretching direction is still a border point of the new ellipsoid, it implies that perhaps there is free space for the ellipsoid to expand in the opposite side. Therefore, the stretching operation is applied once again for possibly enlarging the ellipsoid. In order to hold the interface point between the facet of a polyhedron and the ellipsoid, the new **s**, which needs to be adapted to, is given as

$$\mathbf{s} = l \cdot (\mathbf{c} - \mathbf{s}_i) / \left\| \mathbf{c} - \mathbf{s}_i \right\|,$$

where $\mathbf{s}_i$ is the interface point.

Phase 3 – One by one enlarging each radius

Let $\varepsilon^3(\mathbf{c}', \mathbf{M}')$ 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 **V**. The relation is expressed as

$$\mathbf{D} = \mathbf{V}^{-1}\mathbf{M}'\mathbf{V}.$$

In fact, matrix **V** is the matrix of eigenvectors of matrix $\mathbf{M}'$ and matrix **D** is the canonical form of $\mathbf{M}'$ — 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 **D** with a scaling matrix **S**, which is also diagonal. Therefore, each new radius of the enclosed ellipsoid can be written as

$$\mathbf{D}' = \mathbf{SD},$$

where **S** is one of $\mathbf{S_x} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{S_y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$, or $\mathbf{S_Z} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s_z \end{bmatrix}$;

and the enlarged enclosed ellipsoid can be represented as

$$\mathbf{M}'' = \mathbf{VD}'\mathbf{V}^{-1}.$$

By the use of scaling operations, the length of each principal axis of the enclosed ellipsoid is extended individually until the enlarging process induces the ellipsoid to intersect with facets of the polyhedron.

Since $\varepsilon^3(\mathbf{c}', \mathbf{M}'')$ is generated by stretching along a specified vector and, then, enlarging some axes of $\varepsilon^3(\mathbf{c}, \mathbf{M})$ generated in phase 1, the following relationship $\varepsilon^3(\mathbf{c}, \mathbf{M}) \subseteq \varepsilon^3(\mathbf{c}', \mathbf{M}'')$ always holds.

The summary of the algorithm for enclosed ellipsoid computation is described in Fig. 2.

```
Phase 1 – Isotropically shrinking all principal axes
    $m \leftarrow 0$;
    $n \leftarrow 16$;
    Repeat{
        $\varepsilon^3(\mathbf{c},\mathbf{M}) = \varepsilon^3(\mathbf{y}, n\mathbf{Y})$, where $\varepsilon^3(\mathbf{y}, n\mathbf{Y})$ is the L-J ellipsoid of the convex polyhedron;
        $t \leftarrow n - m$;
        If ( $I(\varepsilon^3(\mathbf{c},\mathbf{M})) > 0$ for all facets of the convex polyhedron)
            $n \leftarrow (n+m)/2$;
        Else
            $m \leftarrow n$;
            $n \leftarrow m + t/2$;
    } Until ( $t < 0.005$ );

Phase 2 – Stretching
```

```
    $\beta \leftarrow 1$;
    $\beta_L \leftarrow 0$;
    Compute $l$, the maximum distance between the enclosed ellipsoid's center and the farthest facet;
    Compute $\mathbf{s}_m$, the center of the farthest facet;
    $\mathbf{s} \leftarrow l \cdot (\mathbf{s}_m - \mathbf{c}) / \| \mathbf{s}_m - \mathbf{c} \|$;
    $\mathbf{a} = \mathbf{M}^{1/2}(\mathbf{s} - \mathbf{c}) / \| \mathbf{M}^{1/2}(\mathbf{s} - \mathbf{c}) \|$;
    Repeat{
        $\beta_T \leftarrow \beta - \beta_L$;
        $\mathbf{c}'_s \leftarrow \mathbf{c} + \beta(\mathbf{s} - \mathbf{c})$;
        $\alpha \leftarrow 1/(1 + \| \beta \mathbf{M}^{1/2}(\mathbf{s} - \mathbf{c}) \|)$;
        $\mathbf{M}'^{1/2}_s = (\mathbf{I} + (\alpha - 1)\mathbf{a}\mathbf{a}^T)\mathbf{M}^{1/2}$;
        If ( $I(\varepsilon^3(\mathbf{c}'_s, \mathbf{M}'_s)) > 0$ for all facets of the convex polyhedron)
            $\beta_L \leftarrow \beta$;
            $\beta \leftarrow \beta_L + \beta_T / 2$;
        Else
            $\beta \leftarrow (\beta + \beta_L) / 2$;
    } Until ( $\beta_T < 0.005$);
    Compute the interface point $\mathbf{s}_i$ between the facet, which is mentioned above, and the ellipsoid
    $\mathbf{s} \leftarrow l \cdot (\mathbf{c}'_s - \mathbf{s}_i) / \| \mathbf{c}'_s - \mathbf{s}_i \|$;
    $\mathbf{a} = \mathbf{M}^{1/2}(\mathbf{s} - \mathbf{c}'_s) / \| \mathbf{M}'^{1/2}_s(\mathbf{s} - \mathbf{c}'_s) \|$;
    Repeat{
        $\beta_T \leftarrow \beta - \beta_L$;
        $\mathbf{c}' \leftarrow \mathbf{c}'_s + \beta(\mathbf{s} - \mathbf{c}'_s)$;
        $\alpha \leftarrow 1/(1 + \| \beta \mathbf{M}'^{1/2}_s(\mathbf{s} - \mathbf{c}'_s) \|)$;
        $\mathbf{M}'^{1/2} = (\mathbf{I} + (\alpha - 1)\mathbf{a}\mathbf{a}^T)\mathbf{M}'^{1/2}_s$;
        If ( $I(\varepsilon^3(\mathbf{c}', \mathbf{M}')) > 0$ for all facets of the convex polyhedron)
            $\beta_L \leftarrow \beta$;
            $\beta \leftarrow \beta_L + \beta_T / 2$;
        Else
            $\beta \leftarrow (\beta + \beta_L) / 2$;
    } Until ( $\beta_T < 0.005$);

Phase 3 – One by one enlarging each radius
    $\mathbf{V}\mathbf{D}\mathbf{V}^{-1} \leftarrow \mathbf{M}'$;
    For ( axes of the ellipsoid) {
        Repeat{
            Scale one principal axis of the ellipsoid by
                $\varepsilon^3(\mathbf{c}', \mathbf{M}'') \leftarrow \varepsilon^3(\mathbf{c}', \mathbf{V}\mathbf{S}\mathbf{D}\mathbf{V}^{-1})$, where $\mathbf{S}$ is one of $\mathbf{S}_\mathbf{x}$, $\mathbf{S}_\mathbf{y}$ or $\mathbf{S}_\mathbf{z}$;
            If ( $I(\varepsilon^3(\mathbf{c}', \mathbf{M}'')) > 0$ for all facets of the convex polyhedron)
                $\mathbf{D} = \mathbf{S}\mathbf{D}$;
        } Until (The ellipsoid intersects with facets of the polyhedron by scaling the current
        principal axis);
    }
```

Fig. 2. Procedure for enclosed ellipsoid computation.

3 Distance Estimate Based on Enclosed Ellipsoids

As depicted in Fig. 3, the enclosed ellipsoids and the enclosing ellipsoids, i.e. L-J ellipsoids, are used for approximation of convex polyhedra. Since there is a fullblown approach to compute the closest points between two separate ellipsoids; therefore, if the closest points between two enclosed ellipsoids are computed, than a straight line equation can be generated based on these two points. The central idea of the proposed method is to rapidly estimate the closest points between polyhedral objects by means of computing the intersection points of the line equation with the polyhedra or enclosing ellipsoid. Consequently, based on the enclosed ellipsoids, a tight distance estimate between two polyhedra can be derived accurately and efficiently [13].

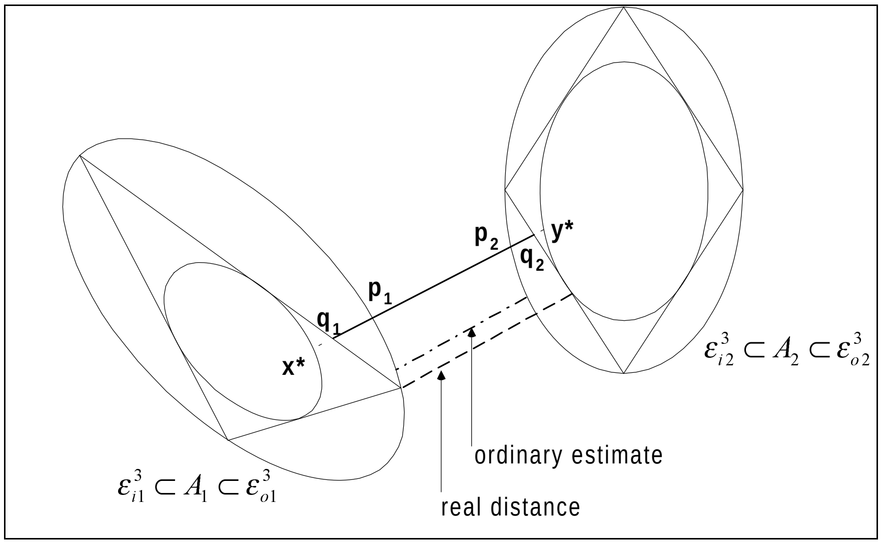

p_2
y^*
q_2
p_1
q_1
x^*
$\varepsilon_{i2}^3 \subset A_2 \subset \varepsilon_{o2}^3$
ordinary estimate
$\varepsilon_{i1}^3 \subset A_1 \subset \varepsilon_{o1}^3$
real distance

Fig. 3. The distance estimates based on enclosed ellipsoids

3.1 Lower Bound

In general, the minimum distance between the bounding volumes, i.e. the enclosing ellipsoids ε_{o1}^3 and ε_{o2}^3, is set as the lower bound of distance estimate and is used for collision detection. However, due to representation error induced from the difference between a real polyhedron and its ellipsoid model, the minimum distance between the enclosing ellipsoids is too conservative to be the lower bound for distance estimate. Therefore, the intersection points $\mathbf{p}_1$ and $\mathbf{p}_2$, at which the shortest path between the enclosed ellipsoids, ε_{i1}^3 and ε_{i1}^3, intersects with the enclosing L-J ellipsoids, are more suitable points for lower bound distance estimate geometrically since $\overline{\mathbf{p}_1\mathbf{p}_2}$ is equal or larger than the minimum distance between the two enclosing ellipsoids and is more close to the real distance between the two polyhedra. The relationship among those estimated distances could be represented as follows:

$$\textit{real distance} > \overline{\mathbf{p}_1\mathbf{p}_2} \geq D_{enclosing},$$

where $D_{enclosing}$ is the minimum distance between the enclosing ellipsoids. The closest points of two enclosed ellipsoids $\varepsilon_{i1}^3(\mathbf{a},\mathbf{A})$ and $\varepsilon_{i2}^3(\mathbf{b},\mathbf{B})$ can be computed as [7]:

$$\mathbf{x}^* = \lambda_{\min}(\mathbf{M})[\lambda_{\min}(\mathbf{M})\mathbf{A} - \mathbf{I}]^{-1}\mathbf{A}\mathbf{a},$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}^{-1} & -\mathbf{I} \\ -\mathbf{A}^{-1/2}\mathbf{a}(\mathbf{A}^{-1/2}\mathbf{a})^T & \mathbf{A}^{-1} \end{bmatrix}.$$

$\lambda_{\min}(\mathbf{M})$ is the eigenvalue with minimal real part and $\mathbf{x}^* \in \varepsilon_{i1}^3(\mathbf{a},\mathbf{A})$. Once $\mathbf{x}^*$ is obtained, $\mathbf{y}^* \in \varepsilon_{i2}^3(\mathbf{b},\mathbf{B})$ can be derived in the same way. Therefore, the intersection points $\mathbf{p}_1$ and $\mathbf{p}_2$ of $L(\overline{\mathbf{x}}^*,\overline{\mathbf{y}}^*)$, the straight line connecting the closest points $\mathbf{x}^*$ and $\mathbf{y}^*$ on the enclosed ellipsoids, with the L-J ellipsoids $\varepsilon^3(\mathbf{a},\mathbf{A})$ and $\varepsilon^3(\mathbf{b},\mathbf{B})$ are computed respectively based on the coordinate transformations

$$\overline{\mathbf{x}}^* = \mathbf{A}^{1/2}(\mathbf{x}^* - \mathbf{a}) \text{ and } \overline{\mathbf{y}}^* = \mathbf{A}^{1/2}(\mathbf{y}^* - \mathbf{a}).$$

Then, the problem becomes how to compute the intersection point of a unit ball $\mathbf{B_O}$ centered at the origin and a line $L(\overline{\mathbf{x}}^*,\overline{\mathbf{y}}^*)$. The points on the line $L(\overline{\mathbf{x}}^*,\overline{\mathbf{y}}^*)$ can be described as vector *v* with a parameter *t*:

$$\mathbf{v}(t) = (\overline{\mathbf{y}}^* - \overline{\mathbf{x}}^*)\cdot t + \overline{\mathbf{x}}^*.$$

The intersection with the unit ball $\mathbf{B_O}$ occurs when

$$|\mathbf{v}(t)| = 1 \text{ or } |\mathbf{a_n}|^2 t^2 + |\overline{\mathbf{x}}^*|^2 + 2(\mathbf{a_n} \bullet \overline{\mathbf{x}}^*)t = 1,$$

where $\mathbf{a_n} = \bar{\mathbf{y}}^* - \bar{\mathbf{x}}^*$, and the solution

$$t' = \frac{\sqrt{(\mathbf{a_n} \bullet \bar{\mathbf{x}}^*)^2 - |\mathbf{a_n}|^2 (|\bar{\mathbf{x}}^*|^2 - 1)} - (\mathbf{a_n} \bullet \bar{\mathbf{x}}^*)}{|\mathbf{a_n}|^2}.$$

The intersection point is found as

$$\mathbf{p}_1 = \mathbf{a} + \mathbf{A}^{-1/2}\mathbf{v}(t').$$

The other intersection point $\mathbf{p}_2$ also can be obtained by the same way. It is worth to notice that $0 < t' < 1$ while the two enclosing L-J ellipsoids are apart. If a L-J ellipsoid intersects with the enclosed ellipsoid of the other polyhedron, t' will be larger than 1.

3.2 Upper Bound

It is intuitive to set the minimum distance between the two enclosed ellipsoids as the upper bound. However, this kind of upper bound still can be improved by taking the polyhedral facets information into consideration. Let the *i*th face of a polyhedron be represented by a plane equation $\mathbf{a_{ni}} \bullet \mathbf{x} = k_i$. Suppose the polyhedron intersects with the line $L(\mathbf{x}^*, \mathbf{y}^*)$, whose points are described by

$$\mathbf{v}(t) = (\mathbf{y}^* - \mathbf{x}^*) \cdot t + \mathbf{x}^*,$$

at the point **q**, the intersection point **q** can be found algebraically by solving the minimum and positive *t*, or, $t_{\min}$ subject to $\mathbf{a_{ni}} \bullet \mathbf{v}(t) = k_i$. The process is given in Fig. 4.

```
for (i = 1 to the number of the facets of the polyhedron)
{
        $t_i \leftarrow \frac{k - \mathbf{a_{ni}} \bullet \mathbf{x}^*}{\mathbf{a_{ni}} \bullet (\mathbf{y}^* - \mathbf{x}^*)}$;    Solutions of $\mathbf{a_{ni}} \bullet \mathbf{v}(t) = k_i$ for the $i_{th}$ facet
        if ($t_i < 0$)
            $t_i \leftarrow 1$;
}
$t_{\min} = \min\{t_i\}$;
$\mathbf{q} = (\mathbf{y}^* - \mathbf{x}^*) \cdot t_{\min} + \mathbf{x}^*$;
```

Fig. 4. Finding the intersection point $\mathbf{q}$ between $L(\mathbf{x}^*, \mathbf{y}^*)$ and a polyhedron.

Thus, the intersection points $\mathbf{q_1}$ and $\mathbf{q_2}$ of both polyhedra can be computed respectively. These two points are close to the closest points of the enclosed ellipsoids between the polyhedra. The upper bound of distance estimate is thus apparently improved by replacing the distance between two enclosed ellipsoids with $\overline{\mathbf{q}_1\mathbf{q}_2}$.

3.3 Distance Estimate Error

For one of the polyhedra and its L-J ellipsoid, the distance estimate error is thus computed by $\overline{\mathbf{p}_i\mathbf{q}_i}$. The error varies with the sizes of the ellipsoids, and the orientations and shapes of the polyhedra. Generally, a slender object will cause a larger estimate error in the direction of its longest axis. Since the distance estimate error plays the role of safe margin in the proposed collision detection algorithm, a larger estimate error will lead to more inaccurate detections.

4 Collision Detection

For collision detection problems, in general, only the bounding volume, e.g. the enclosing ellipsoids, is applied to performing intersection check. However, due to the limit of representation model's precision, such strategies may cause a lot of false alarms. To overcome such kind of problem, more related information about polyhedra are required for accurate detection. To avoid expensive computational expense, a hierarchical strategy based on the proposed distance estimate method is introduced. If all the enclosing ellipsoids are enough apart, the lower bound of the distance estimate between two polyhedra is larger than zero and collision free is guaranteed. The further collision detection needs only be performed when the enclosing ellipsoids intersect with others. In this way, the proposed approach can be used to efficient localize collisions in space.

Based on the relationship of enclosing and enclosed ellipsoids of polyhedra, a heuristic approach is proposed to improve the accuracy of the collision detection algorithms based only on bounding volume representation. According to the geometrical relationship, two types of cases are taken into consideration.

Case 1. Intersection-free between two enclosing ellipsoids

Case 2. Intersection between two enclosing ellipsoids but intersection-free between two enclosed ellipsoids

It is obvious that case 1 is collision free. Therefore, as the lower bound of the distance estimate between two polyhedra is larger than zero, collision-free is guaranteed. Now, only case 2 is indeterminate for collision detection. For further detail, case 2 can be categorized into the following three types:

1. $\varepsilon^3{}_{i1}(\mathbf{c}_{i1},\mathbf{M}_{i1})\cap\varepsilon^3{}_{o2}(\mathbf{c}_{o2},\mathbf{M}_{o2})=\phi$ *and*
   $\varepsilon^3{}_{o1}(\mathbf{c}_{o1},\mathbf{M}_{o1})\cap\varepsilon^3{}_{i2}(\mathbf{c}_{i2},\mathbf{M}_{i2})=\phi$
2. $\varepsilon^3{}_{i1}(\mathbf{c}_{i1},\mathbf{M}_{i1})\cap\varepsilon^3{}_{o2}(\mathbf{c}_{o2},\mathbf{M}_{o2})\neq\phi$ *and*
   $\varepsilon^3{}_{o1}(\mathbf{c}_{o1},\mathbf{M}_{o1})\cap\varepsilon^3{}_{i2}(\mathbf{c}_{i2},\mathbf{M}_{i2})=\phi$, or

   $\varepsilon^3{}_{i1}(\mathbf{c}_{i1},\mathbf{M}_{i1})\cap\varepsilon^3{}_{o2}(\mathbf{c}_{o2},\mathbf{M}_{o2})=\phi$ *and*
   $\varepsilon^3{}_{o1}(\mathbf{c}_{o1},\mathbf{M}_{o1})\cap\varepsilon^3{}_{i2}(\mathbf{c}_{i2},\mathbf{M}_{i2})\neq\phi$
3. $\varepsilon^3{}_{i1}(\mathbf{c}_{i1},\mathbf{M}_{i1})\cap\varepsilon^3{}_{o2}(\mathbf{c}_{o2},\mathbf{M}_{o2})\neq\phi$ *and*
   $\varepsilon^3{}_{o1}(\mathbf{c}_{o1},\mathbf{M}_{o1})\cap\varepsilon^3{}_{i2}(\mathbf{c}_{i2},\mathbf{M}_{i2})\neq\phi$.

Both the geometrical order of the closest points, which locate on the enclosing and the enclosed ellipsoids of the two polyhedra for distance estimate, and the parameters t_i', which are derived for computing the closest points of L-J ellipsoids, are used to tell the type of intersection. A "correct" geometrical order of the set of closet points can be checked easily by using $sign(\overrightarrow{\mathbf{p}_1\mathbf{p}_2}\bullet\overrightarrow{\mathbf{q}_1\mathbf{q}_2})$, the sign of the inner product $\overrightarrow{\mathbf{p}_1\mathbf{p}_2}\bullet\overrightarrow{\mathbf{q}_1\mathbf{q}_2}$. For two enough separate polyhedra, $sign(\cdot)$ always larger than zero. If $sign(\cdot)$ is equal to zero, the two enclosing ellipsoid collide at one point. If $sign(\cdot)$ is small than zero, it implies that the geometrical order is violated and there are intersections among these ellipsoids.

Let's take a view at t_i'. For type 1, both t_i' are in the range of [0, 1]; only one t_i' is larger than 1 indicates the type 2 intersection that the L-J ellipsoid intersects with the enclosed ellipsoid of the other polyhedron; and both t_i' are larger than 1 implies type 3. The type 3 implies that the two polyhedra may be very close and there is a high probability of collision between the polyhedra. Therefore, the situation of type 3 is directly categorized as potential collision. According the different intersection situation, we can understand how emergency about current status of those dynamic polyhedra.

As for collision detection, since the representation of ellipsoid model is not the same as the original polyhedron, a conservative estimate for collision detection is reasonable. The most popular and straightforward way is the use of safe margin. This idea is also adopted here. The distance estimate error, i.e. $\overline{\mathbf{p}_1\mathbf{q}_1}$ or $\overline{\mathbf{p}_2\mathbf{q}_2}$, are set as the safe margin for the proposed algorithm. Since the $\mathbf{q}_1$ and $\mathbf{q}_2$ are derived from the polyhedra and their enclosed ellipsoids, they are not the real closest points between the two polyhedra. In fact, $\overline{\mathbf{q}_1\mathbf{q}_2}$ is larger than the minimal distance between the two polyhedra. Therefore, if the distance, i.e. $\overline{\mathbf{q}_1\mathbf{q}_2}$, between two polyhedra is smaller than $\overline{\mathbf{p}_1\mathbf{q}_1}$ or $\overline{\mathbf{p}_2\mathbf{q}_2}$, the two polyhedra may be in the situation of potential collision. Inspired from the above idea, the criterion for collision detection is therefore given as: if $\overline{\mathbf{q}_1\mathbf{q}_2} > \min(\overline{\mathbf{p}_1\mathbf{q}_1}, \overline{\mathbf{p}_2\mathbf{q}_2})$ is satisfied, it is categorized as collision-free; otherwise, it is judged as potential collision. Since the detection of potential collision is based on the distance estimate error, larger distance estimate error leads to more false warnings for collision detection. It is undesirable for path planning problem in a workspace that is cluttered with obstacles. Hence, an artificial threshold is given to overcome this drawback. The criterion for collision detection is thus replaced by

$$\overline{\mathbf{q}_1\mathbf{q}_2} > \min(\overline{\mathbf{p}_1\mathbf{q}_1}, \overline{\mathbf{p}_2\mathbf{q}_2}, \mathit{threshold}).$$

The flowchart of the whole process for collision detection is exhibited in Fig. 5.

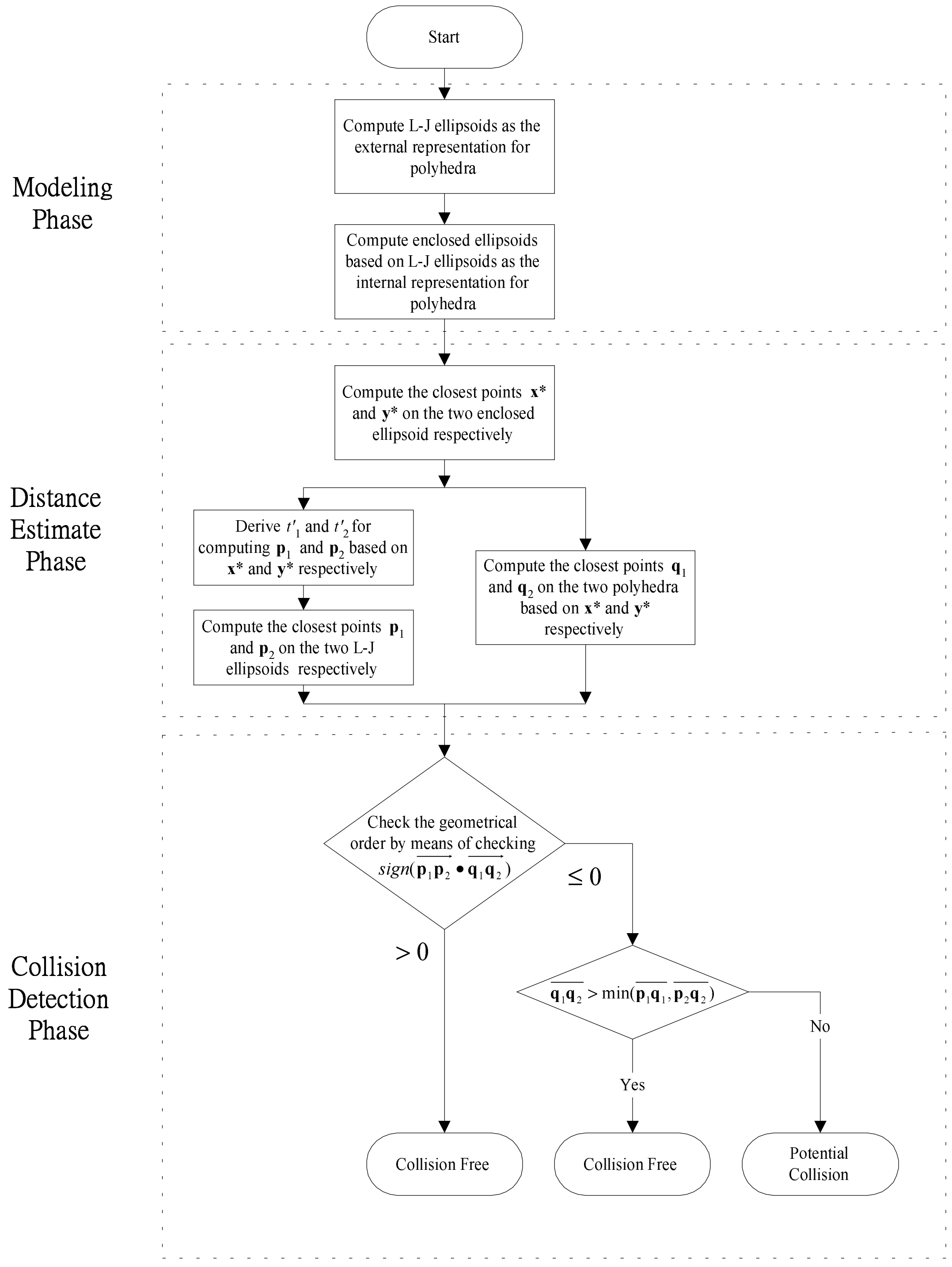

Start
Modeling Phase
Compute L-J ellipsoids as the external representation for polyhedra
Compute enclosed ellipsoids based on L-J ellipsoids as the internal representation for polyhedra
Distance Estimate Phase
Compute the closest points $\mathbf{x}^*$ and $\mathbf{y}^*$ on the two enclosed ellipsoid respectively
Derive t'_1 and t'_2 for computing $\mathbf{p}_1$ and $\mathbf{p}_2$ based on $\mathbf{x}^*$ and $\mathbf{y}^*$ respectively
Compute the closest points $\mathbf{p}_1$ and $\mathbf{p}_2$ on the two L-J ellipsoids respectively
Compute the closest points $\mathbf{q}_1$ and $\mathbf{q}_2$ on the two polyhedra based on $\mathbf{x}^*$ and $\mathbf{y}^*$ respectively
Collision Detection Phase
Check the geometrical order by means of checking $sign(\overrightarrow{\mathbf{p}_1\mathbf{p}_2} \bullet \overrightarrow{\mathbf{q}_1\mathbf{q}_2})$
≤ 0
> 0
$\overline{\mathbf{q}_1\mathbf{q}_2} > \min(\overline{\mathbf{p}_1\mathbf{q}_1}, \overline{\mathbf{p}_2\mathbf{q}_2})$
No
Yes
Collision Free
Collision Free
Potential Collision

Fig. 5. Flowchart of the collision detection procedure.

5 Simulations

Two examples are given in this section to demonstrate the proposed approach for collision detection in various applications.

5.1 Example 1

Three dynamic convex polyhedra, one for rotation and two for translation, are introduced in this simulation. Geometrically, A_1 is a rectangular solid, A_2 is a chock and A_3 is a combination of two pyramids.

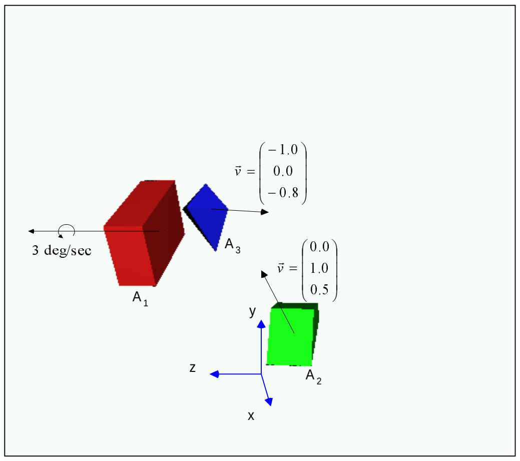

$\vec{v} = \begin{pmatrix} -1.0 \\ 0.0 \\ -0.8 \end{pmatrix}$
3 deg/sec
A_3
$\vec{v} = \begin{pmatrix} 0.0 \\ 1.0 \\ 0.5 \end{pmatrix}$
A_1
y
z
A_2
x

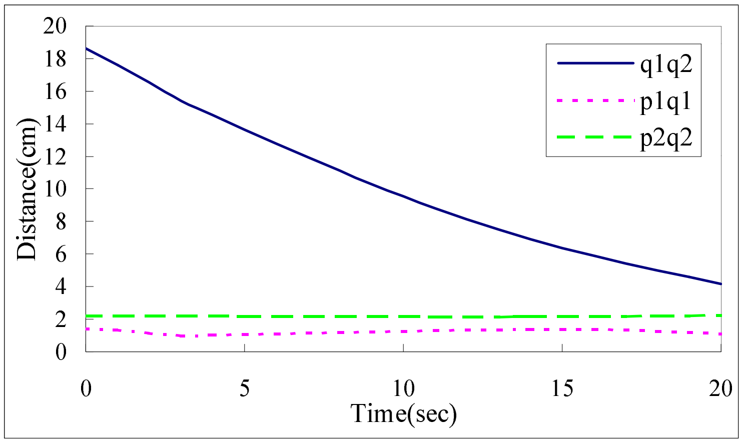
Fig. 6. The moving direction for all dynamic polyhedra.

For L-J ellipsoid computation, the relative accuracy ρ is set as 1.01 in the simulation; in other words, the volume of an approximate enclosing ellipsoid is not 1.01 times larger than the volume of the L-J ellipsoid. The L-J ellipsoids are computed as $\varepsilon^3(\mathbf{c}_{o1}, \mathbf{M}_1)$, $\varepsilon^3(\mathbf{c}_{o2}, \mathbf{M}_2)$ and $\varepsilon^3(\mathbf{c}_{o3}, \mathbf{M}_3)$, respectively. After the L-J ellipsoids containing convex polyhedra are generated, the proposed three-phase approach is then applied to generating the enclosed ellipsoids based on the L-J ellipsoids. The initial value for enclosed ellipsoid computation is given as $\varepsilon^3(c_o, 16\mathbf{M})$ and the enlarging factor for phase 3 is set as 1.01. Through compressing, stretching and scaling phases, these enclosed ellipsoids are computed as $\varepsilon^3(\mathbf{c}_{i1}, \mathbf{N}_1)$, $\varepsilon^3(\mathbf{c}_{i2}, \mathbf{N}_2)$ and $\varepsilon^3(\mathbf{c}_{i3}, \mathbf{N}_3)$.

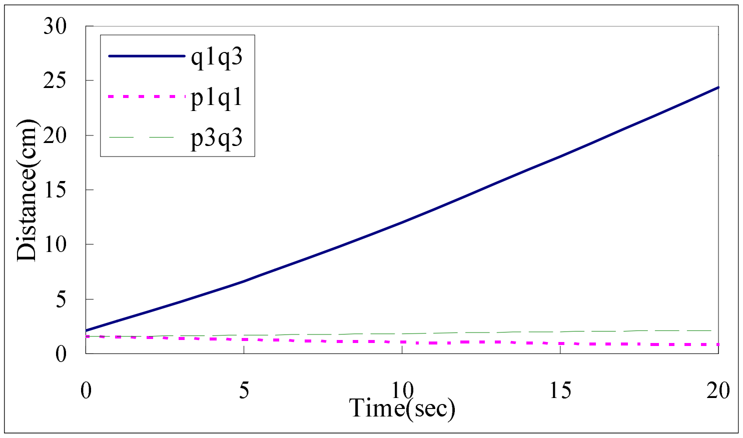
The dynamic polyhedra are described in terms of the motion of their mass centers.

As for the trajectories of the translating polyhedra, the mass center of A_2 and A_3 move from (10.0, 10.0, -4.0) to (10.0, 30.0, 6.0) and from (14.0, 28.0, 10.0) to (-6.0, 28.0, -10.0) in 20 *sec*, respectively. The A_1's mass center locates at (19.0, 25.0, 15.0) and it rotates about *z*-axis by 3 *deg/sec* with respect its mass center from –30 *deg* orientation. The moving directions of polyhedra are shown in Fig. 6.

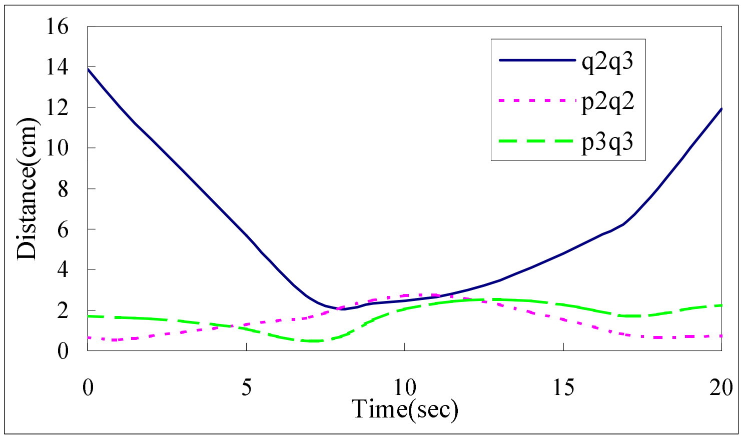
In this simulation, all the above polyhedra move without colliding for all durations. The distances $\overline{\mathbf{q}_i\mathbf{q}_j}$ and $\overline{\mathbf{p}_i\mathbf{q}_i}$ are exhibited in Fig. 7. From the results show in Fig. 7., the proposed collision detection algorithm correctly detects that all the moving polyhedra are collision free. Some snapshots are shown in Fig. 8. As mentioned in Sec. 4, two L-J ellipsoids collide with the other one as $sign(\cdot)$ is equal to –1. The time histories of $sign(\cdot)$ for this simulation are displayed in Fig. 9. As shown in Fig. 9., if only the L-J ellipsoids, i.e. the bounding volumes, are applied for collision detection, some false alarms will be concluded. Therefore, in comparison with those approaches using only bounding ellipsoids, the proposed approach can provide more accurate estimate about the occurrence of collisions than.



(a)

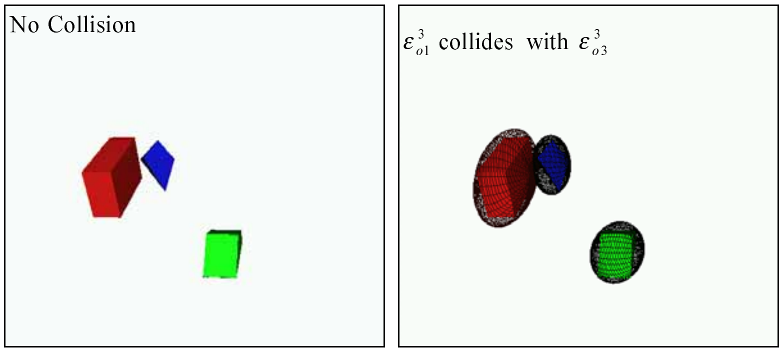


(b)

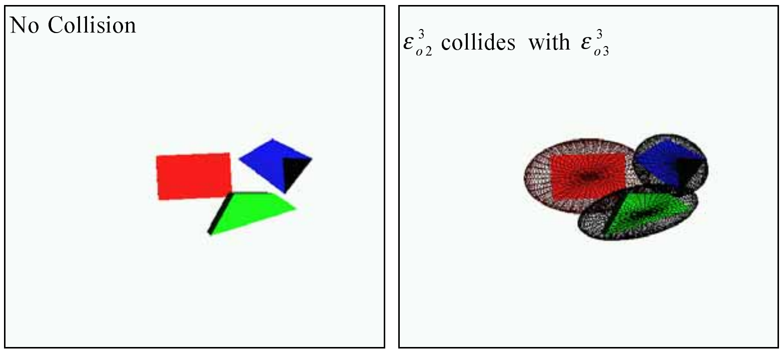


(c)

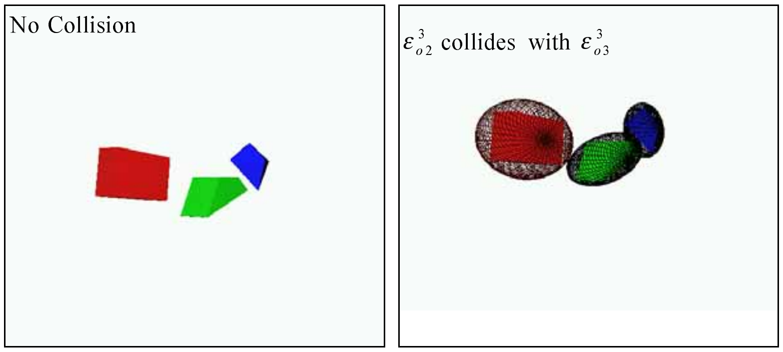
Fig. 7. The time histories of distances, i.e. $\overline{\mathbf{q}_i\mathbf{q}_j}$ and $\overline{\mathbf{p}_i\mathbf{q}_i}$.



(a) Time = 0.0 sec

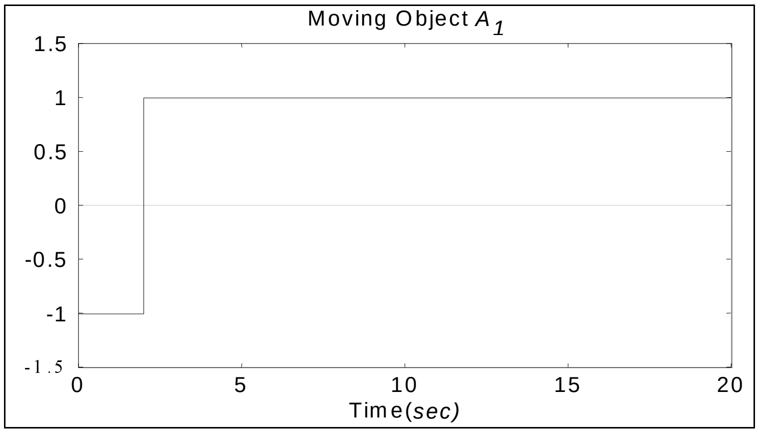


(b) Time = 9.0 sec

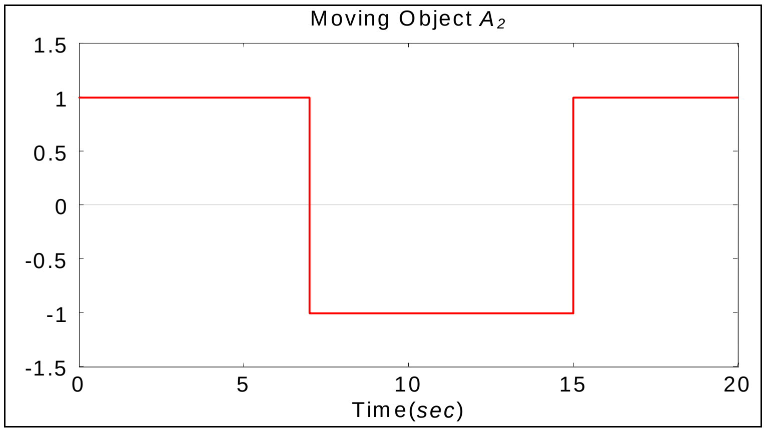


(c) Time = 12.0 sec

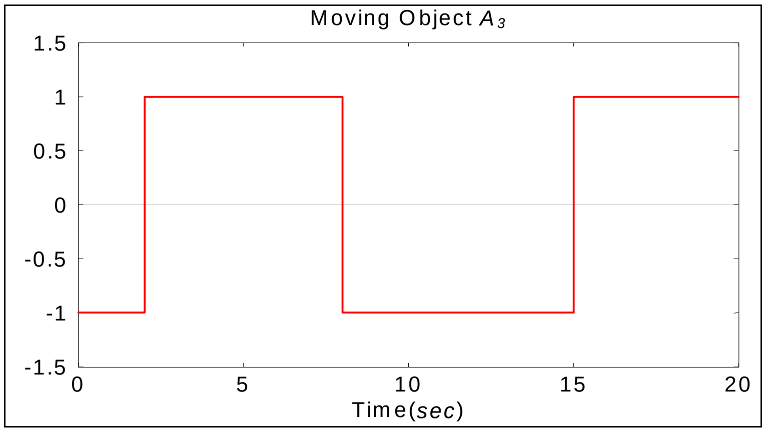
Fig. 8. Status of polyhedra at indicated time steps.



(a)



(b)



(c)

Fig. 9. The time histories of collision detection, i.e. $sign(\cdot)$, based on ellipsoidal bounding volume representation for all moving objects.

5.2 Example 2

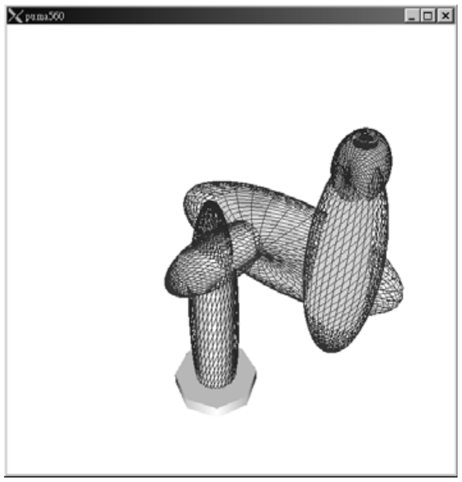
Two PUMA560 robot arms manipulating in a cluttered workspace is introduced in this example. In this simulation, each robot needs to avoid collision with both static obstacles and dynamic obstacles. It is assumed that the two robots stand away from each other so that their link 1 and link 2 may not collide with their counterparts. Therefore, only link 3 to link 6 are involved in collision detection for dynamic obstacles, i.e. the other robot arm. The base coordinates for the two PUMA560 arms with respect to the world coordinate are

$$\mathbf{B}_{robot1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } \mathbf{B}_{robot2} = \begin{bmatrix} -1 & 0 & 0 & 1100mm \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

In this simulation, the link 4 and the link 5 of a PUMA560 are lumped together heuristically for convenience and the saving of computation time since the rotation of joint 5 will not affect the lumped geometrical shape of the link 4 and the link 5. Besides, the L-J ellipsoid computation is based on polyhedral model, therefore, all the curved surfaces of links are first approximated with a union of several facets. The transition from an original PUMA560 manipulator to the polyhedra model of a PUMA560 for ellipsoid computation and the generated enclosing and enclosed ellipsoids for a PUMA560 are shown in Fig. 10. The parameters of those ellipsoids generated for a PUMA560 are given in Reference 15. The operating time for this simulation is 20 *sec*. As shown in Fig. 11, the trajectories for the two robot's tips are given as follows. Robot 1 moves from (-350.0, -650.0, 550.0) to (350.0, -20.0, 550.0) during the first 15 seconds and then moves toward (550.0, 30.0, 550.0) during the rest of time to avoid static obstacle. Robot 2 moves from (650.0, -350.0, 660.4) to (550.0, -150.0, 550.0). All the set points are with respect to the world coordinate.

(a)

(b)


(c)

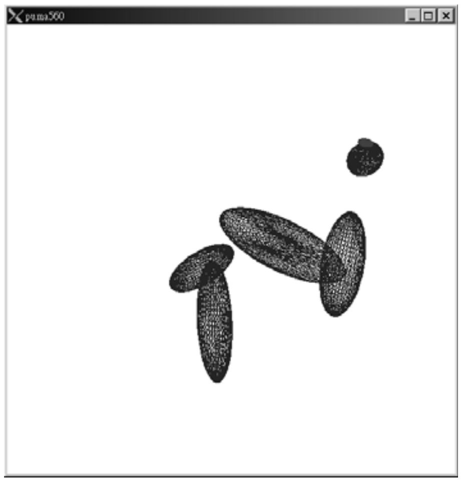

(d)

Fig. 10. (a) PUMA560 manipulator. (b) Polyhedral model of PUMA560. (c) L-J ellipsoids, i.e. the ellipsoidal bounding volume representation, for PUMA560. (d) Enclosed ellipsoids for PUMA560.

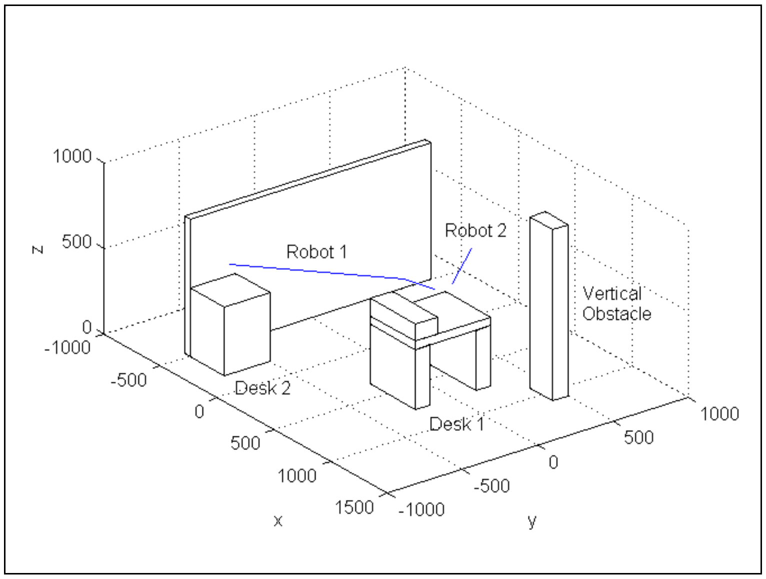
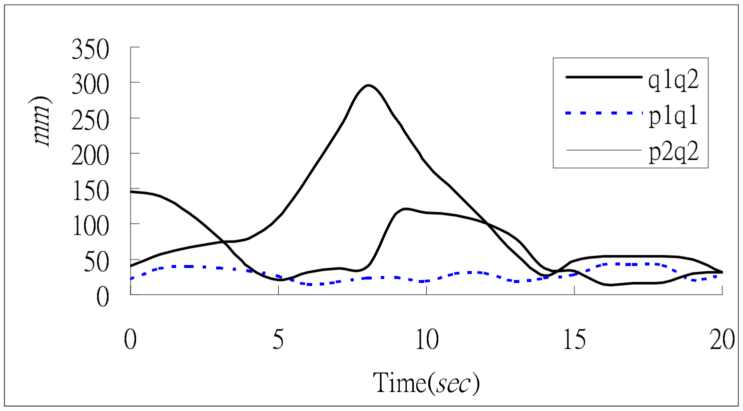
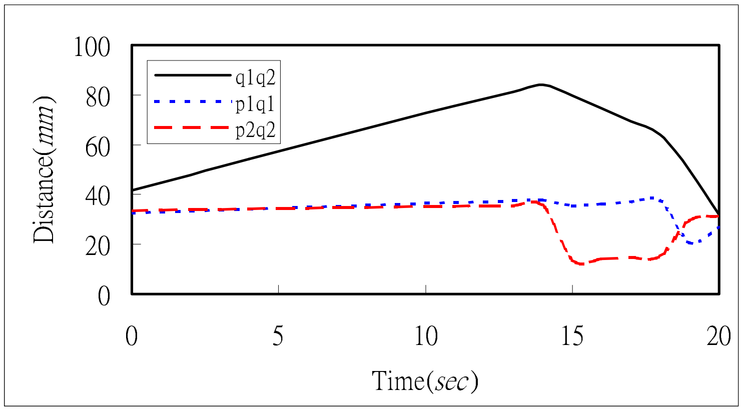


Fig. 11. The tips' trajectories of the two PUMA560 manipulators.

The same as the previous example, during the full operating time, the robot arms don't collide with any obstacle. The minimum distance estimates of $\overline{\mathbf{q}_i\mathbf{q}_j}$ and $\overline{\mathbf{p}_i\mathbf{q}_i}$ for the two robot arms are exhibited in Fig. 12. It shows good results that the proposed method correctly detects collision condition, no false alarms. In contrast, if only the bounding volumes are applied for collision detection, as shown in Fig. 13 and in Table 1, a lot of false alarms will be generated. Several snapshots showing the closest points between robots and obstacles are demonstrated in Fig. 14. The results show that the proposed approach is efficient and accurate for detecting both static and dynamic obstacles; therefore in terms of the number of false alarms the proposed method is superior to the traditional bounding volume scheme in robotic applications.

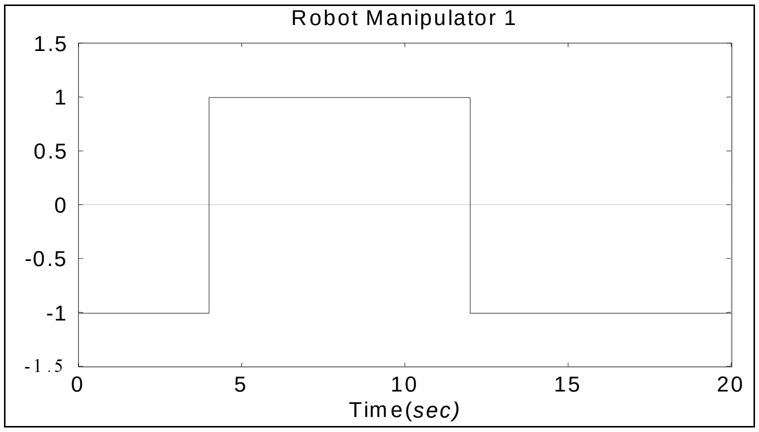


(a) For robot manipulator 1

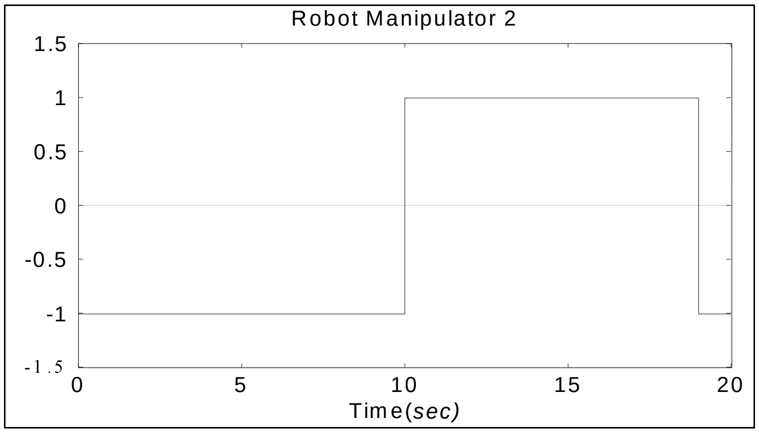


(b) For robot manipulator 2

Fig. 12. The time histories of distances, i.e. $\overline{\mathbf{q}_i\mathbf{q}_j}$ and $\overline{\mathbf{p}_i\mathbf{q}_i}$, for the two robot manipulators.

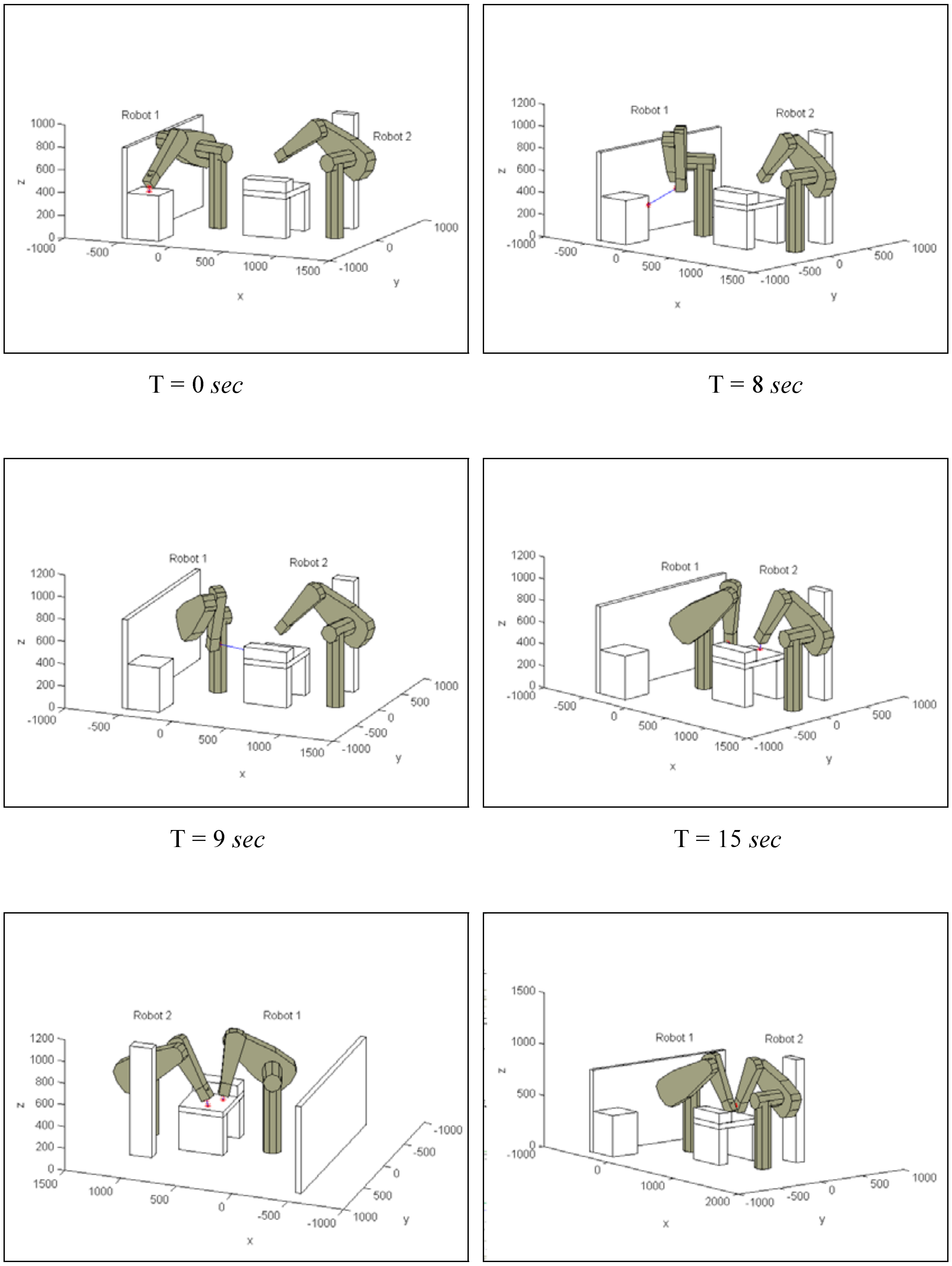


(a)



(b)

Fig. 13. The time histories of collision detection, i.e. $sign(\cdot)$, based on ellipsoidal bounding volume representation.



T = 0 *sec*

T = 8 *sec*

T = 9 *sec*

T = 15 *sec*

T = 17 *sec*

T = 20 *sec*

Fig. 14. Snapshots showing the closest points between robots and obstacles.

Table 1. Collision detection results by different methods.

Time Index	Practical State	Ellipsoidal Bounding Volume	Proposed Method
0	No collision	● Robot1's link4&5 collides with the desk2 ● Robot2's link2 collides with the vertical obstacle	No collision
1	No collision	● Robot1's link4&5 collides with the desk2 ● Robot2's link2 collides with the vertical obstacle	No collision
2	No collision	● Robot1's link4&5 collides with the desk2 ● Robot2's link2 collides with the vertical obstacle	No collision
3	No collision	● Robot1's link4&5 collides with the desk2 ● Robot2's link2 collides with the vertical obstacle	No collision
4	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
5	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
6	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
7	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
8	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
9	No collision	● Robot2's link2 collides with the vertical obstacle	No collision
10	No collision	No collision	No collision
11	No collision	No collision	No collision
12	No collision	● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
13	No collision	● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
14	No collision	● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
15	No collision	● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
16	No collision	● Robot1's link4&5 collides with the desk1 ● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
17	No collision	● Robot1's link4&5 collides with the desk1 ● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
18	No collision	● Robot1's link4&5 collides with the desk1 ● Robot1's link4&5 collides with the obstacle located on the desk1	No collision
19	No collision	● Robot1's link4&5 collides with the desk1 ● Robot1's link4&5 collides with Robot2's link4&5	No collision
20	No collision	● Robot1's link4&5 collides with the desk1 ● Robot2's link4&5 collides with the desk1 ● Robot1's link4&5 collides with Robot2's link4&5	No collision

5.3 Discussion

The proposed method not only provides a more accurate and efficient solution for minimum distance estimate and collision detection, besides, with comparison to the approaches directly computing the minimum distance between two polyhedra, it requires less computation time. For two polyhedra have l and m facets respectively, polyhedral approaches need $l*m$ operations to determine the closest facets for computing the minimum distance. The computational complexity are $O(n^2)$. In contrast, as the closest points of enclosed ellipsoids are determined, the proposed method needs $l+m$ operations to find out the closest facets for distance computation. The computational complexity is therefore concluded as $O(n)$; in other words, the complexity linearly increases as the total number of facets. This advantage makes the proposed method more suitable and practicable for real-time applications than the approaches directly computing the minimum distance.

6 Conclusions

An efficient and accurate collision detection method based on the enclosed ellipsoids of convex polyhedra is presented in this paper. Not only for collision detection, the proposed method also provides fast and accurate distance estimate among convex polyhedra for local path planning methods based on artificial potential fields [20]. In comparison with the traditional bounding volume approaches that model convex polyhedra utilizing only the enveloping ellipsoids or spheres, the proposed method makes use of additional interior representation for minimum distance estimate. It is able to provide more accurate estimate about the occurrence of collision and reduces the number of false alarms. Moreover, due to the use of ellipsoid models to simplify the representation complexity of convex polyhedra, the computational complexity for minimum distance computation is also significantly reduced, as compared with the polyhedral models.

Acknowledgment. This work is supported by National Science Council of R.O.C. under contract NSC 89-2212- E-001-001.

References

[1] P. M. Hubbard, "Collision detection for interactive graphics applications," *IEEE Transactions on Visualization and Computer Graphics* 13, 218-230 (1995)

[2] B. H. Lee and C. S. G. Lee, "Collision-free motion planning of two robots," *IEEE Transactions on System, Man, and Cybernetics* 17(1), 21-32 (1987)

[3] R.A. Basta, R. Mehrotra and M. R. Varanasi, "Collision detection for planning collision-free motion of two robot arms," *Proc.* of the *IEEE Int. Conf. Robotics and Automation*, 638-640 (1988)

[4] J. Tornero, J. Hamlin and R. B. Kelly, "Spherical-object representation and fast distance computation for robotic applications," *Proc.* of the *IEEE Conf. Robotics and Automation*, 1602-1608 (1991)

[5] S. Bonner and R. B. Kelley, "A novel representation for planning 3-D collision free path," *IEEE Transactions on System, Man, and Cybernetics* 20, 1337-1352 (1990)

[6] A. P. Del Pobil, M. A. Serna and J. Llovet, "A new representation for collision avoidance and detection," *Proc.* of the *IEEE Int. Conf. Robotics and Automation*, 246-251 (1992)

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

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

[9] T. Brychcy and M. Kinder, " A neural network inspired architecture for robot motion planning," *Proc.* of the *Int. Conf. Engineering Applications of Artificial Neural Networks* (*EANN'95*), 103-109 (1995)

[10] D. E. Johnson, and E. Cohen, " A framework for efficient minimum distance computations," *Proc.* of the *IEEE Int. Conf. Robotics and Automation*, 3678-3684 (1998)

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

[12] 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 Transactions on Robotics and Automation* 4, 193-203 (1988)

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

[14] 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)

[15] M. Y. Ju, J. S. Liu, and K. S. Hwang, *Ellipsoid modeling for articulated robot manipulators for interactive motion planning*, Technical Report TR-IIS-00-008, (Academia Sinica, 2000)

[16] C. J. Wu, "On the representation and collision detection of robots," *Journal of Intelligent and Robotic Systems* 16, 151-168 (1996)

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

[18] 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*), 93-98 (1989)

[19] D. Jung and K. K. Gupta, "Octree-based hierachical distance maps for collision detection," *Proc.* of the *IEEE Int. Conf. Robotics and Automation*, 454-459 (1996)

[20] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research* 5, 90-98 (1986)

[21] G. Hurteau and N. F. Stewart, "Distance calculation for imminent collision indication in a robot system simulation," *Robotica* 6, 47-51 (1988)

[22] C. Chang, M. J. Chung and Z. Bien, "Collision-free motion planning for two articulated robot arms using minimum distance function," *Robotica* 8, 137-144 (1990)

[23] B. Cao, G. I. Dodds and G. W. Irwin, "An approach to time-optimal, smooth and collision-free path planning in a two robot arm environment," *Robotica* 14, 61-70 (1996)