An improved collision detection algorithm based on OBB swept volume for virtual assembly

Wenjing Wang¹, Junhao Wang², Xiaohua Sun³

¹Department of physics and Electronic Engineering, Baoding University, Baoding, 071000, China

²College of Information Science and Technology, Agricultural University of Hebei, Baoding, 071001, China;

³Department of Digital Media, Hebei Software Institute, Baoding, 071000, China

Abstract

The speed and precision of collision detection are the main concerns in a virtual assembly environment. The calculation time and accuracy of conventional collision detection algorithms need to be improved. This paper introduces an improved algorithm called OSV (OBB swept volume) based on OBB (oriented bounding box) for continuous collision detection. A bounding volume boxes tree is built by using the OBB bounding boxes model. The rough collision detection is realized by separating axis testing. Then by the exact intersection test algorithm based on the triangular patches, the objects that might be on a collision course with others are further retested. Experiment results show that the algorithm works accurately and has a higher accuracy rate than other similar methods.

Keywords: Collision detection algorithm, Oriented bounding box, Virtual assembly, Exact intersection test.

1. INTRODUCTION

Virtual assembly is a research area of intense interest in virtual manufacture. The speed and the accuracy of collision detection are the main concerns (Chung, 2007; Valentini 2009; BoehmDavis, 2008; Qin, 2004; Wang, 2010). An efficient and exact collision detection algorithm is needed to improve reality and to enhance immersion of the virtual environment. The geometric objects in the virtual assembly simulation are all rigid ones with complex geometrical shapes. It is worthwhile to optimize and improve the collision detection algorithms to meet the actual requirements of special applications (Jareo and Lori, 2000; Huang, 2012; Mann et al., 2010; Meang et al., 2003; Mini and Ushakumari, 2014). Zhang (2008) proposes an algorithm to detect collisions between the industrial objects and its working environment. Furthermore, Figueiredo and Feenando (2004) proposes a method of superposition testing for oriented bounding boxes (OBB) based on the separation axis theory. Some problems about the constructed test method are discussed. However, due to the large amount of computation, both of the above mentioned methods are difficult to meet the requirement of real-time detection (Terlapu and Raju, 2014; Liang, 2014; Sharma et al., 2010; Li et al., 2015; Zhou, 2015). This paper first introduces the improved algorithm called OSV (OBB swept volume) based on traditional OBB for continuous collision detection in the virtual assembly of complicated machines. It can be used to eliminate most objects that do not collide with one another. Then the objects that might be on a collision course with others are further retested by the exact intersection test algorithm based on triangular patches. The simulation test
shows that compared with the contrast methods of Zhang L.J. and Figueiredo, the new algorithm has better detecting efficiency and accuracy.

2. OSV ALGORITHM

The traditional OBB method is complicated and inefficient. It no longer fulfills the requirements of a large-scale scene in virtual assembly. An improved algorithm called OSV (OBB swept volume) is presented in this paper for collision detection based on the oriented bounding box (OBB) for preliminary collision tests. Between the bounding boxes there is potential collision and interference, and the next step will be preceded by using a triangle patches test to obtain accurate collision detection results.

2.1 Construction of OSV

Swept volume has been investigated due to its importance to mechanic manufacture, collision detection and robot path planning. The construction process is as follows. First, when the OBB is originally established, the three components X, Y, Z of the endpoints are put in the real number container. Then the container is put into a two dimensions container. So each vertex of OBB can be expressed by the corresponding elements of the two dimensions container. The three components of the translation vector expressed as $t$ are judged and taken as the foundation for translating the corresponding element. The detailed process of the algorithm is:

1. read OBB and the translation vector $t$;
2. read the coordinates of the endpoints of OBB, and put them into the two dimensions container. Read the three components $a$, $b$, $c$ of $t$;
3. judge whether $a$, $b$, $c$ are all 0. If satisfied, go back to the container of OBB and go to (7);
4. judge whether there is only one zero in $a$, $b$, $c$. If satisfied, go back to the container of octahedron and go to (7);
5. judge whether there are two zeros in $a$, $b$, $c$. If satisfied, go back to the container of hexahedron and go to (7);
6. go back to the container of dodecahedron;
7. end.

Before going back to the OSV container, the elements in the two dimensions container after translation should be extracted and used to generate the corresponding array geometrical points. This method does not generate complete OSV. OSV is expressed implicitly by translating the corresponding vertex of OBB. The algorithm can reduce the time of the interference examination and improve the efficiency.

2.2 Collision Detection Algorithm Based on OSV
According to the geometric features of OSV, the algorithm for coarse collision detection is proposed in this paper. Supposing that there are $N$ potential collision objects, the initiative scene is expressed as:

$$Q_i (i=1,2,3,...,N) \quad (1)$$

After the motion vector calculation, there is only one object in constant motion. The motion can be expressed as $t$, and the others are all at a quiescent state. After generating the corresponding OBB, the separation axis accelerated algorithm based on OSV is executed. Two guidelines are selected for the accelerated algorithm. The selected separation axis is based on the given parameter $t$, and any vector named $n$ that is perpendicular to $t$ can be taken as the separation axis that appears as the dotted line in Figure 1. Then, $t$ is taken as the second separation axis that appears as the arrow in Figure 1.

The detailed process of the algorithm based on OSV is:

1. read the motion object $O_0$, the motion vector $t$ and the remaining stationary object $O_i (i=1,2,3,...,N)$.
2. calculate the OBB of $O_0$ and name it as $M_0$.
3. calculate the $N$ axis.
4. judge whether $i>N$. If satisfied, go to 8;
5. calculate the OBB of $O_i$ and name it as $M_i$, execute the separation axis algorithm to $M_i$ and $M_0$.
6. judge whether there is any collision between $M_i$ and $M_0$. If not, delete $M_i$;
7. $i=i+1$, go to 4;
8. the supporting point $SP_t$ and $SP_{-t}$ that are along $t$ and $-t$ are obtained by calculating the support function of $M_0$. Translate $SP_t$ along with $t$ to get $SP_t'$.
9. judge whether $i>N$, If satisfied, go to 13;
10. execute the separation axis algorithm to $M_0$ and $P_t, SP_t'$ to axis $t$, the $M_i$ that does not have collision with others is deleted.
11. judge whether there is any collision between $M_i$ and $M_0$. If not, delete $M_i$;
12. $i=i+1$ and go to 9;
13. go back to $O_0$ and the object $O_i$ which needs further detection;
14. end.
The method of representation of extreme points of OBB is used for the vector \( n \). The largest projection interval is obtained by the projection operation to \( M_i \) in the direction of vector \( n \). Because \( n \) is perpendicular to \( t \), when \( M_i \) is moving along the direction of \( t \), the longest swept line can be regarded as the length of the projection interval. So if the separation axis algorithm is executed on vector \( n \), most of the objects that cannot have any collision can be eliminated when \( M_i \) is moving along the direction of \( t \), such as \( M_3 \) and \( M_7 \) in Figure 1.

![Figure 1. Separation axis algorithm](image)

To obtain \( n \), in the 2D situation, suppose that:

\[
t = (t_x, t_y)
\]  

(2)

Based on the property of the vector dot product, if \( n \) is perpendicular to \( t \), it could be derived that:

\[
n \cdot t = 0
\]  

(3)

So it can be expressed as:

\[
n = (-t_y, t_x)
\]  

(4)

And under the 3D situation in which the origin is expressed as \( O \), take an arbitrary vertex \( V_i \) of \( M_i \) and let \( l = V_i - O \), based on the property of the vector cross products, and \( n = t \times l \) can be obtained.

If OBB is expressed by the center radius and the radius of each component is \( \eta \), its support function is:

\[
S_A((x, y, z)^T) = (\text{sgn}(x)\eta_x, \text{sgn}(y)\eta_y, \text{sgn}(z)\eta_z)^T
\]  

(5)

In the function,

\[
\text{sgn}(x) = -1
\]  

(6)

And when \( x < 0 \),
\( \text{sgn}(x) = 1 \)  \hspace{1cm} (7)

Suppose that the only moving OBB in the scene is M0. Before executing the separation axis algorithm to t, the supporting point SP\(t\) of M0 along the direction of t is needed through the support function. Then the supporting point SP-\(t\) of M0 along -t direction needs to be obtained. SP\(t'\) can be obtained after a translational transform along the direction of t to SP\(t\). So the vector \( \overrightarrow{SP_{t'} - SP_t} \) can be obtained. In essence, \( \overrightarrow{SP_{t'} - SP_t} \) is the largest covering line of M0 along the direction of t when t is moving. Project the rest Mi(i=2,3,4,…,N) to the t direction and obtain the interval Ii, which is compared with \( \overrightarrow{SP_{t'} - SP_t} \) to eliminate the objects that would not have collision with others, such as M2 and M5. The separation axis algorithm based on OSV is a coarse detection method which is the foundation of the following precise detection based on a triangle patch.

3. PRECISE COLLISION DETECTION BASED ON TRIANGLE PATCH

The collision detecting method based on ambient-box is a custom method in virtual experiments, but the detecting precision of this method is not exact. An intersection test method based on triangle patch that raises algorithm efficiency of collision detection and strengthens the sense of reality and immersion is presented in this paper. The triangle intersection test consists of the two steps of the calculation of the point of intersection and the position detection, as shown in Figure 2.

![Figure 2. Triangle patch position detection](image)

For the two triangles in space which are named ABC and DEF, respectively, first, the intersection of triangle ABC and the edge DE in triangle DEF should be calculated. Suppose that the normal vector of the triangle is \( N_1 \). The edge DE is projected to \( N_1 \), and the projection is named a. Then, calculate the distance between point D and the plane of the triangle. The result is the projection of AD onto \( N_1 \).

This can be expressed as:

\[
a = \overrightarrow{DE} \cdot \overrightarrow{N_1}
\]

\[
b = \overrightarrow{AD} \cdot \overrightarrow{N_1}
\]
If \( a=0 \) and \( b=0 \), it can be concluded that the line DE is in the plane of the triangle. In this situation, the exact intersection test of the line and the triangle is needed.

If \( a=0 \) and \( b\neq 0 \), it can be concluded that the line DE is parallel to the plane of the triangle.

If \( a\neq 0 \) and \( b=0 \), the value of \( a \) and \( b \) should be recalculated after changing AD to AE.

If \( a\neq 0 \) and \( b\neq 0 \), it can be concluded that there is a converging point of DE and ABC.

The coordinate of the point can be obtained by utilizing the similarity theorem. Suppose that the ratio of the normal projection of AD and DE is:

\[
\mu = \frac{b}{a}
\]  
(10)

If \( \mu>1 \), it can be identified that there is no intersection of DE and ABC.

If \( \mu<0 \), the angle between the direction of DE and normal line is greater than 90 degrees. They will never have an intersection point.

If \( 0<\mu<1 \), there is a common intersection point between DE and ABC.

The coordinates of Q is:

\[
\overrightarrow{OQ} = \overrightarrow{OD} + \mu\overrightarrow{DE}
\]  
(11)

After determining the intersection point between DE and ABC, the relation of the positions between the intersection and each side of the triangle can be judged to determine if the point is in the interior of the triangle. If the point is outside of the triangle, it can be determined that the line is neither parallel nor intersects with the triangle. According to the algorithm, the process of intersection testing method is as shown in Figure 3.
The program is compiled and the collision effect is shown in Figure 4. The color change indicates that there is a collision.

**Figure 4.** The collision effect

**4. THE COLLISION DETECTION PROCESSES**

The flow chart of the collision detection algorithm is shown in Figure 5.
In summary, a new collision detection algorithm that combines an improved OBB and triangle patch is proposed. It can solve the collision detection problems between deformable objects. In the virtual mechanical assembly, for some large-scale fittings with simple shapes, only a rough collision detection is needed. But for the subtle parts, the precise collision detection should be utilized. Figure 6 shows a new collision occurring between the base and the gear during the assembly.

Figure 5. The collision effect processes

Figure 6. A new collision occurring during assembly

When the trainers study the mechanical assembly in the virtual environment, the collision will be detected. Figure 7 shows that when the virtual hand assembles the gears, some collisions occur between the hand and the objects.
Figure 7. Collisions between virtual hand and gears

5. EXPERIMENT RESULTS AND ANALYSES

5.1 Experiment about OSV

Firstly, the experiment is conducted with the OSV algorithm. The environment is CATIA V5R13, and by using the CAA platform, the experiment is performed in VC++6.0. The frequency of the CPU is 4.2GHz and the memory is 8GB. The two different environments are used for testing collision detection. One environment contains three objects and the other contains sixteen components. The two scenes are shown in Figure 8 below.

Figure 8. the two environments for collision detection

Ten experiments are conducted for the specified object in the two environments, respectively. For environment 1, the calculation time comparison between OBB and OSV is shown in Figure 9.
For the environment 2, the comparison between OBB and OSV is shown in Figure 10.

Contrasting the calculation result data, it is determined that the OSV algorithm can significantly increase the collision detection rate. In the case of a relatively small number of objects, such as in environment 1, it seems to show a relatively small difference in the detection rate between the two algorithms. When the environment includes higher numbers of objects, the calculation time of the two algorithms shows a great difference in environment 2. The OSV can simplify the collision model, and thus accelerate the detection process.

5.2 Experiment on precise collision detection

In the experiment on precise collision detection, the test environment remains the same. There are six static objects in the virtual environment. The currently selected object moves at a set speed, and the collision detection time is recorded in periodic intervals. The environment is composed of 6,564 triangular patches, as shown in Figure 11 below.
The methods of Zhang L. J. and Figueiredo are taken as comparison. The result of this experiment is shown in Table 1.

**Table 1. Detection time comparison of the algorithms (ms)**

<table>
<thead>
<tr>
<th>Objects</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhang L J</td>
<td>24.3</td>
<td>23.6</td>
<td>13.7</td>
<td>12.2</td>
<td>9.5</td>
<td>8.7</td>
<td>15.3</td>
</tr>
<tr>
<td>Figueiredo</td>
<td>27.3</td>
<td>22.8</td>
<td>16.0</td>
<td>11.7</td>
<td>10.4</td>
<td>8.5</td>
<td>16.1</td>
</tr>
<tr>
<td>New algorithm</td>
<td>17.5</td>
<td>15.5</td>
<td>8.7</td>
<td>8.1</td>
<td>7.1</td>
<td>4.9</td>
<td>10.3</td>
</tr>
</tbody>
</table>

The experimental data shows that the proposed algorithm combined with the coarse detection and the exact detection can quickly and efficiently improve the efficiency of collision detection. The mean time for detection is only 10.3ms which is significantly less than the algorithms of Zhang L. J. and Figueiredo. It can be proved that the method is quite efficient for virtual mechanical assembly experiments.

**6. CONCLUSION**

This paper introduces a new algorithm for collision detection based on the specific needs of a virtual mechanical assembly experiment. By employing the constructed method of OBB bounding boxes model for reference, a bounding volume boxes tree is built. The rough collision detection is realized by separating axis testing. According to the characteristics of machine parts, the design and implementation of the exact intersection test algorithm of the spatial objects is proposed based on the triangular patches for precise collision detection. Experiment results and theoretical analysis show that the algorithm works accurately and fast for the virtual mechanical assembly experiment. The testing results show that this algorithm is better than previous methods in speed and accuracy, and it is flexible and easy to be realized.

**REFERENCES**


