Real-time collision detection and response techniques for deformable objects based on hybrid bounding volume hierarchy

Yuanfeng Zhu
Department of Computer Science, University of California Davis, Davis, California, USA, and
Jun Meng
Department of Computer Science and Engineering, Dalian University of Technology, Dalian, China

Abstract
Purpose – The purpose of this paper is to propose a physically plausible solution based on hybrid bounding volume (BV) hierarchy for real-time collision detection (CD) and response between a deformable and a rigid object.

Design/methodology/approach – Hybrid BV can be used to build BV hierarchy for the deformable object. The overlapping tests based on separating axis theorem (SAT) can be used to deal with CD. The physics conception of restitution coefficient and other important forces can be used to more real collision response.

Findings – Many methods focus on a specific application, but none of them gives an approach to physically plausible, real-time simulation of CD and response up to 10,000 of deforming primitives. The paper finds that hybrid BV AABB-Sphere for deformable object could increase the efficiency for CD, and restitution coefficient and other important physical concepts could provide more real collision response.

Research limitations/implications – The paper does not deal with all types of CD, such as CD for two deformable objects.

Originality/value – Using AABB-Sphere hybrid BV to build hierarchical BV tree for deformable object, and OBB-Sphere hybrid BV for rigid object. Owing to the variety of hybrid BV structures, during different phases of CD, different overlapping tests are used to accelerate CD effectively. Using important physics conceptions provides a good solution to simulate more real collision response.

Keywords Collisions, Material-deforming processes, Modelling, Simulation

Paper type Research paper

1. Introduction
In the field of computer graphics, much attention is paid to research on simulation and modeling of deformable objects. Analyzing the previous research, we have discovered that many methods emphasize the importance of precision, but not fit for the real-time virtual environment; however, others emphasize the signification of real-time simulation. Considering the growing demand for interactive deformable modeling especially in electronic games, a method of efficient deformable models with physically plausible dynamic behavior is much more important than the one requiring only
physically correct deformable models while ignoring the real-time simulation. Therefore, here we focus on physically plausible collision detection (CD). It is assumed that a soft body changes its shape during collision with a rigid object throughout the simulation, and that all the objects’ surfaces are constituted by triangles.

First, AABB-Sphere is used to build hybrid bounding volume hierarchy (HBVH) for the deformable object, and OBB-Sphere is used to build HBVH for the rigid object (Zhu and Meng, 2008). Second, the physics conception of restitution coefficient and some important forces is used to deal with more real collision response. Last, extensive experiments with various scenes of CD and response indicate that our method is efficient in terms of real-time CD and response.

2. Previous work
In this section, we discuss some classic CD approaches that meet different requirements of animation and simulation environments with dynamically deforming objects.

2.1 Bounding volume hierarchies
Bounding-volume hierarchies have been proven to be one of the most efficient data structures for CD. A variety of bounding volume (BV) types has been explored, such as sphere (Hubbard, 1996), AABB (Zhang and Kim, 2007), OBB (Gottschalk et al., 1996), and K-DOP (Zachmann, 1998), convex hulls (Ehmann and Lin, 2001). So far, AABB should be preferred to other BV, such as OBB (Teschner et al., 2004).

2.2 Stochastic methods
This idea is motivated by several observations. First, polygonal models are just an approximation of the true geometry. Second, the perceived quality of most interactive 3D applications does not depend on exact simulation, but rather on real-time response to collisions (Uno and Slater, 1997). At the same time, humans cannot distinguish between physically correct and plausible behavior of objects (Barzel et al., 1996). Therefore, it can be tolerated to improve the performance of CD while degrading its precision.

2.3 Distance fields
Distance fields (Bremer et al., 2002; Zhao et al., 2003; Bridson and Fedkiw, 2003) specify the minimum distance to a closed surface for all points in the field. These methods are advantageous because there are no restrictions about topology. Further, the evaluation of distances and normals is extremely fast and independent of the geometric complexity of the object, giving method a wide range of applications.

2.4 Spatial subdivision
Early spatial subdivision approaches have been proposed for neighborhood queries, e.g. in molecular dynamics. There are various approaches that propose spatial subdivision for CD of rigid objects. These algorithms employ uniform grids (Ganovelli et al., 2000), octrees (Bandi and Thalmann, 1995) or BSP trees (Melax, 2000).

2.5 Image-space techniques
Recently, several image-space techniques have been proposed for CD (Baciu and Wong, 2002; Govindaraju et al., 2003; Heidelberger et al., 2003). These approaches commonly process projections of objects to accelerate collision queries. Since they do not require
any pre-processing, they are especially appropriate for environments with dynamically deforming objects. Furthermore, image-space techniques can commonly be implemented using graphics hardware.

In sum, many methods focus on solutions to a specific application, but none of them give an approach to physically plausible, real-time simulation of CD and response up to 10,000 of deforming primitives with elasticity, gravity, damping, friction, and restitution coefficient.

3. Collision detection for HBVH

During pre-processing, we use HBVH based on OBB-Sphere data structure to construct a bounding volume tree (BVT) for rigid object (Zhu and Meng, 2008), and present the algorithm of HBVH based on AABB-Sphere data structure to construct a BVT for deformable object in a top-down manner. We build an AABB with the edges represented as $a$, $b$, and $c$ to approximate a tree node. Then, we get the circumscribed sphere of AABB, whose radius $R$ is calculated as in:

$$R = \sqrt{\frac{a^2 + b^2 + c^2}{2}}$$

(1)

The BVT of deformable object falls into three successive layers, Layers X, Y and Z. Layer X uses sphere as its BV; Layer Y uses AABB-Sphere; Layer Z uses AABB. The BV overlapping tests are named Sphere-Sphere, Sphere-AABB and Sphere-OBB, AABB-OBB. In fact, Sphere-AABB test based on separating axis theorem (SAT) could be implemented by Sphere-OBB test. The AABB-OBB test based on SAT could be implemented by OBB-OBB test.

We, respectively, define $a_i$ and $b_i$ as the nodes on the BVTs of deformable object A and rigid object B. Our kernel algorithm CDTraverse() is shown as Figure 1, which shows how HBVH proceed intersection tests between node $a_i$ and $b_i$. CDFlag is the flag to judge overlapping between BVs.

3.1 Function CDSphere_Sphere()

When searching the nodes between Layer X or Y of object A and Layer X or Y of object B, we use the test of sphere vs sphere (sphere-sphere), defining Function CDSphere_Sphere(). Since the tightness of sphere BV in Layer X or Y is not good enough, so we can use sphere-sphere to speed up intersection tests and quickly exclude the possibility of collision between two objects far away.

3.2 Function CDSphere_OBB()

When searching the nodes between Layer X of object A and Layer Z of object B, we use the test of sphere vs OBB (Sphere-OBB) to increase the performance of CD, defining Function CDSphere_OBB(). If using sphere-sphere at this time, we would search more nodes although it is quite fast for each pair of nodes. We may also fail to determine overlapping accurately even searching the bottom nodes of BVT of object B. We also have another choice to use OBB-OBB, but it works not well. First, it may take a long time to detect collision, because in the worst situation, using OBB-OBB based on SAT will require 15 tests, but sphere-OBB based on SAT only need three tests. Second, the BV of Layer X of object A is weakly tightened, so OBB-OBB have to require 15 tests to
Figure 1. Flow chart of algorithm CDTraverse
judge overlapping when both of the objects are in close proximity, which will cost much time on each intersection test.

3.3 Function CDOBB_AABB ()
When searching the nodes between Layer Y or Z of object A and Layer Z of object B, we use the test of OBB vs AABB (OBB-AABB) to increase the accurateness of CD, defining Function CDOBB_AABB (). Since the tightness of BV in Layer Y is better and the tightness of BV in Layer Z is much nicer after the contraction of Layer Y. If using Sphere-OBB again at this time, we would search more nodes although it is faster than OBB-AABB for each pair of nodes; furthermore, we would fail to judge overlap accurately even if the searching nodes of object B were at the bottom of the BV tree. Although each OBB-AABB test may require more time, the accuracy of determination is more important than time.

3.4 Function PrimTest ()
When searching the leaf nodes, we use primitive test called PrimTest() for overlapping test between triangles.

4. Collision response for deformable object
We assume that the deformable object is a mass-spring system, in which particles are connected by structural springs without mass to simulate the relationship of inner parts of the object. Structural spring is used to avoid the unduly stretching of soft body. $F_{spring}$ is the elastic force of structural springs. In order to simulate the physically plausible collision response for deformable object at real-time speed, our method updates the status of deforming primitives with elasticity, gravity, damping force, friction, and restitution coefficient.

4.1 Dynamic analyses on mass-spring
During the given time step $\Delta t$, any particle’s displacement from the position $\bar{P}(t_0)$ at moment $t_0$ to the position $\bar{P}(t_0 + \Delta t)$ at moment $(t_0 + \Delta t)$ can be supposed as uniformly accelerated motion as in:

$$\bar{P}(t_0 + \Delta t) = \bar{P}(t_0) + \bar{v}(t_0)\Delta t + \frac{1}{2} \ddot{a}(t_0)\Delta t^2.$$  \hspace{1cm} (2)

$\bar{v}(t_0)$ is initial speed at moment $t_0$, $\ddot{a}(t_0)$ is acceleration of the particle.

We choose Verlet algorithm which is one of the most popular explicit integrators. The velocity at each time step is then not calculated until the next time step:

$$\ddot{x}(t_0 + \Delta t) = 2\ddot{x}(t_0) - \ddot{x}(t_0 - \Delta t) + \ddot{a}(t_0)\Delta t^2 + O(\Delta t^4) \hspace{1cm} (3)$$

$$\ddot{v}(t_0) = \frac{\ddot{x}(t_0 + \Delta t) - \ddot{x}(t_0 - \Delta t)}{2 \Delta t}. \hspace{1cm} (4)$$

Therefore, $\bar{P}(t_0 + \Delta t)$ could be calculated as in:

$$\bar{P}(t_0 + \Delta t) = 2\bar{P}(t_0) - \bar{P}(t_0 - \Delta t) + \ddot{a}(t_0)\Delta t^2. \hspace{1cm} (5)$$

The motion of particle can be expressed by the second Newton law as in:
The force brought to particles is the composition of forces $\mathbf{F}_{\text{inside}}(t_0)$ from springs and other forces $\mathbf{F}_{\text{outside}}(t_0)$ outside of the mass-spring system. The forces of spring include elasticity of spring and damping. Other forces include gravity, resistance and so on.

### 4.2 Computation of $\mathbf{F}_{\text{inside}}(t_0)$

#### (1) Elasticity

In mass-spring system, elasticity comes from springs. This paper calculates elasticity $\mathbf{F}_{\text{spring}}^i$ of any particle $i$ by using Hooke law as in:

$$
\mathbf{F}_{\text{spring}}^i = \sum_j a_{ij} (\|\mathbf{l}_{ij}\| - l_{ij}).
$$

$a_{ij}$ is elasticity coefficient of structural spring; $\|\mathbf{l}_{ij}\|$ is the length between particle $i$ and $j$ after being stretched, $l_{ij}$ is the original length between $i$ and $j$.

#### (2) Damping force

Mass-spring system can be considered as a vibroseis with the spring and particle. Since the particle speed is not fast, the particle damping is proportional to speed, as in:

$$
\mathbf{F}_{\text{damping}} = -\gamma \mathbf{\dot{v}}(t_0).
$$

$\gamma$ is damping coefficient which can avoid that deformable object is overly stretched. From equation (8), damping force of particle $i$ is calculated as in:

$$
\mathbf{F}_{\text{damping}}^i = -\gamma \sum_j (\mathbf{\dot{v}}_i(t_0) - \mathbf{\dot{v}}_j(t_0)).
$$

$\mathbf{\dot{v}}_i(t_0)$ and $\mathbf{\dot{v}}_j(t_0)$ is accordingly the speed of particle $i$ and $j$ which is the neighbor particle to particle $i$ at moment $t_0$.

### 4.3 Computation of $\mathbf{F}_{\text{outside}}(t_0)$

#### (1) Gravity

The gravity of a particle can be computed as in:

$$
\mathbf{F}_{\text{gravity}} = m \mathbf{g}.
$$

$m$ is the particle mass, $\mathbf{g}$ is gravity acceleration.

#### (2) Friction force

When deformable object moves, friction exists between it and other surrounding objects. Since there are usually few differences between maximal coefficient of static friction $\mu'$ and coefficient of sliding friction $\mu$, this paper uses sliding friction to describe the friction $\mathbf{F}_{\text{friction}}$ between objects as in:

$$
\mathbf{F}_{\text{friction}} = -\frac{\mathbf{\dot{v}}_\parallel}{\|\mathbf{\dot{v}}_\parallel\|} \mathbf{\hat{N}} \mu.
$$

$\mathbf{\hat{N}}$ is the pressure vertical to the plane; $\mu$ is frictional coefficient; $\mathbf{\dot{v}}_\parallel/\|\mathbf{\dot{v}}_\parallel\|$ determines the movement direction of object along the contact surface.

### 4.4 Computation of $\mathbf{\bar{P}}(t_0 + \Delta t)$

During simulation of deformable objects, objects cannot penetrate themselves or penetrate other objects in virtual environment. When collision happens, the method for
collision response should be implemented. When triangles of objects collide with each other, they experience extrusion transformation and elastic recovery phases. In order to describe recovery extent of elastic material, we use coefficient of restitution which is a physical conception defined as in:

\[
e = \frac{\vec{v}'_\perp - \vec{v}'_{\perp}}{\vec{v}_\perp - \vec{u}_\perp}.
\]  

\(\vec{v}_\perp\) and \(\vec{u}_\perp\) is an accordingly normal component on collision plane of the speed \(\vec{v}\), \(\vec{u}\) before collision; \(\vec{v}'_\perp\) and \(\vec{u}'_\perp\) is an accordingly normal component on collision plane of the speed \(\vec{v}'\), \(\vec{u}'\) after collision. This paper takes \(e\) as punishment coefficient. It is not only supported by practical physical significance and data, but also can get more vivid simulation effect. Collision is so short that assumed a time step \(\Delta t\), based on impulse theorem we can conclude that impulse created between collision triangles can even be ignored. We compute the speed of triangle after collision by theorem of momentum as in:

\[
m_i\vec{v}_\perp + m_j\vec{u}_\perp = m_i\vec{v}'_\perp + m_j\vec{u}'_\perp \quad (13)
\]

\(m_i, m_j\) is the mass of two collision triangles correspondingly.

Based on the fact that after collision between a soft body and a rigid one, the soft body, such as little animals, could not effectively influence the movement of rigid body such as a car. Therefore, the speed of rigid body is invariable after collision, namely \(\vec{u}'_\perp = \vec{u}_\perp\). In the reference frame of rigid body, \(\vec{u}_\perp = 0\). Therefore, \(e = (-\vec{v}'_\perp / \vec{v}_\perp)\), namely as in:

\[
\vec{v}'_\perp = -e\vec{v}_\perp. \quad (14)
\]

Analyzing the normal component of collision plane with impulse theorem, we have the equation as in:

\[
((\vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}})_{\perp} + \vec{N})_\Delta t = m(\vec{v}'_\perp - \vec{v}_\perp) = -m(1 + e)\vec{v}_\perp \quad (15)
\]

From equation (15) we get the pressure as in:

\[
\vec{N} = -\frac{m(1 + e)\vec{v}_\perp}{\Delta t} = (\vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}})_{\perp} \quad (16)
\]

From equations (11) and (16), we get the friction as in:

\[
\vec{F}_{\text{friction}} = \frac{\vec{v}_\parallel}{\|\vec{v}_\parallel\} \left\{ \frac{\mu m(1 + e)\vec{v}_\perp}{\Delta t} + \mu (\vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}})_{\perp} \right\} \quad (17)
\]

Then analyzing the tangent component of collision plane with impulse theorem, we have the equation as in:

\[
((\vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}})_{\parallel} + \vec{F}_{\text{friction}}) \Delta t = m(\vec{v}'_\parallel - \vec{v}_\parallel). \quad (18)
\]

From equation (18), we get the tangent component \(\vec{v}'_\parallel\) of \(\vec{v}'\) as in:
\[ \vec{v}' = \vec{v} + \vec{F}_{\text{friction}} \cdot \frac{\Delta t}{m} + \left( \vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}} \right) \cdot \frac{\Delta t}{m}. \]  \hspace{1cm} (19)

From equations (14) and (19), and the equation of \( \vec{v}' = \vec{v}' + \vec{v}' \), we update the speed \( \vec{v}(t_0) \) of particle after collision as in:

\[ \vec{v}(t_0) = \vec{v} - e\vec{v} + \vec{F}_{\text{friction}} \cdot \frac{\Delta t}{m} + \left( \vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}} \right) \cdot \frac{\Delta t}{m}. \]  \hspace{1cm} (20)

From equation (8) and the result of equation (19), we get \( \vec{F}_{\text{damping}} \), as in:

\[ \vec{F}_{\text{damping}} = -\gamma \sum_j \left( \vec{v}_j - e\vec{v} - \vec{v}(t_0) + \vec{F}_{\text{friction}} \cdot \frac{\Delta t}{m} + \left( \vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}} \right) \cdot \frac{\Delta t}{m} \right) \]  \hspace{1cm} (21)

From equations (6), (7), (10), (16), (17) and (21), we have the result of acceleration as in:

\[ \vec{a}(t_0) = \left( \vec{F}_{\text{spring}} + \vec{F}_{\text{gravity}} + \vec{N} + \vec{F}_{\text{friction}} + \vec{F}_{\text{damping}} \right) / m. \]  \hspace{1cm} (22)

From equations (5) and (22), we have the position \( \vec{P}(t_0 + \Delta t) \) at moment \( (t_0 + \Delta t) \) as in:

\[ \vec{P}(t_0 + \Delta t) = 2\vec{P}(t_0) - \vec{P}(t_0 - \Delta t) + \left( \vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} + \vec{F}_{\text{gravity}} + \vec{F}_{\text{air resistance}} + \vec{F}_{\text{friction}} + \vec{N} \right) \Delta t^2 / m \]  \hspace{1cm} (23)

5. Experiments and results

In this section, we have carried out experiments to test the performance of our algorithm based on HBVH. We build a platform HBVHS to interactively implement the experiments about some applications of CD and response based on our deformable models.

5.1 The design and implementation of experiments

All experiments have been executed on a dual-core AMD Athlon64 X2 4000 + PC processor with 4.0 GB DDR2 800 MHz memory and NVIDIA 7800 GTX. Through HBVHS, we could interactively set some of the parameters such as gravity, structure stiffness, damping coefficient, real-time FPS and so on. The flow chart of HBVHS is shown in Figure 2.

5.2 Information of the models

The experiments mainly use a car for rigid object, and three different animals (frog, bunny, and Chinese dragon as in Figure 3), whose default properties are shown as in Figure 4. The property of the gravity acceleration in our experiments is \(-9.8 \text{ m/s}^2\).

5.3 Melting of deformable models

This experiment proves that HBVHS could simulate the melting progress of deformable models through interactive regulating material’s elasticity coefficient of structural spring. In fact, hard and deformable materials can be represented by
adjusting this coefficient which contributes to preserving the shape of model. Figure 5 shows the melting progress of our deformable model.

5.4 Damping of deformable models
This experiment substantiates that HBVH could truly simulate the progress of reaching static equilibrium of deformable models with different materials’
damping coefficients. In fact, the damping coefficient contributes to attenuate the kinetic energy of the deformable models until the deformable models by themselves reach to the status of static equilibrium. Figure 6 shows that the three frogs fall to the ground at the same height, and then reach static equilibrium one by one.

5.5 Friction of deformable models
This experiment substantiates that our HBVH could effectively simulate the friction between deformable models and rigid object. The three deformable frogs with different friction coefficients fall down to the same slant which make an angle of $15^\circ$ with the plane, and then slide along the slant until stop, as in Figure 7. The images illustrate that friction contributes to attenuate the kinetic energy of the deformable models.

5.6 Restitution coefficient of deformable models
The purpose of this experiment is to prove the influence of restitution coefficient onto deformable model’s velocity after collision. As in Figure 8, the three frogs synchronously with different restitution coefficients fall to the same board, and then...
they jump up to different heights and at last fall down to the ground again. From the images, we can conclude that the higher the restitution coefficient is the higher the frog jumps, which means that the frog could gain more kinetic energy to translate into more potential energy and as a result the frog could jump higher.

5.7 Collisions with a rigid car
Table I includes the rendering speed (FPS) and the speed (km/h) of a car when the car collided with a bunny as Figure 9.

Table II includes the rendering speed (FPS) and the speed (km/h) of a car when the car collided with a frog shown as Figure 10.
Table III includes the rendering speed (FPS) and the speed of a car when the car collided with a Chinese dragon shown as Figure 11.

From these three experiments, we conclude that the rendering speed could be at least 41 FPS when the triangle number in deformable object is less than 8,000, which is a rather satisfactory result. When this number reaches 10,000, such as the Chinese dragon with 5,202 vertexes and 10,000 triangles, rendering speeds could be still more than 29 FPS.

6. Conclusion and future work
In this paper, we have proposed a HBVH algorithm for efficient CD and response among deformable object and rigid one. We built a BVT with AABB-Sphere BVs for a deformable object, whose BVs are selected as Sphere, AABB-Sphere, and AABB in different layers.

Table III includes the rendering speed (FPS) and the speed of a car when the car collided with a Chinese dragon shown as Figure 11.
(Layers X, Y and Z) from top to bottom for the CD using different intersection test (sphere-sphere, sphere-OBB, sphere-AABB and AABB-OBB test). The movement of deformable models is controlled by the forces such as gravity, friction, damping and other parameters such as restitution coefficient as well as the vertex mass of the models.

Our algorithm have been implemented and tested for a variety of models and the design parameters. Our results have shown that our methods could provide a rather satisfactory rendering speed and a physically plausible progress for CD and response between a rigid body and a deformable one with vertex number up to 5,000 as well as primitive number up to 10,000.

Future work will focus on research about the deformation and CD of fluid, and the CD and response between the deformable models.

References


About the authors

Yuanfeng Zhu was born in Dalian, China, in 1982. He received the BS and MS degrees in Computer Science from Dalian University of Technology in 2005 and 2008. He pursues PhD degree in Computer Science at University of California Davis since 2009. His present interests are collision detection and response, physics-based animation. Yuanfeng Zhu is the corresponding author and can be contacted at: yuazhu@ucdavis.edu

Jun Meng was born in Dalian, China, in 1964. She received the BS and MS degrees in Computer Application from Jilin University of Technology in 1986 and 1989. She is an Associated Professor in the Department of Computer Science and Engineering at Dalian University of Technology. Her present interests are data mining, text mining, and virtual reality technology.