

Line-Triangle Test for Collision Detection and Response in Soft Bodies

Jaruwan Mesit
jmesit@cs.ucf.edu

Ratan K. Guha
guha@cs.ucf.edu

School of Electrical Engineering and Computer Science
University of Central Florida, Orlando, Florida 32826

KEYWORDS

soft body collision, animation, spatial subdivision,
spatial hashing

ABSTRACT

Soft-body models are common in games to simulate cloth or elastic objects. To realistically simulate soft-body objects, collision detection and response is required. In addition, soft-body models must re-arrange their internal structure to react to the collision. This paper presents a new collision detection and response algorithm which can simulate a variety of soft-body material behaviors ranging from stiff to elastic. In this approach, a line-triangle intersection test is used for collision detection and force propagation is used for collision response. Implementation and experiments using the algorithm show that complex deformable objects composed of thousands vertices can be animated at interactive speeds.

1. INTRODUCTION

Soft-body, or deformable, objects in games and simulations include flags, banners, cloaks, rubber or elastic walls, trampolines or bounce pads, and semi-solid surfaces such as slime. Soft-body objects impose additional complexity to collision computation over solid-body objects due to: (1) changing collision detection boundaries, and (2) collision response that requires change of the solid body internal structure.

Deformable models have had some recent research attention [6, 30, 26, 32], many of which utilize a mass-spring system. Mass-spring systems can be classified as either finite-element methods (FEM) or long-element methods (LEM). Bounding volume hierarchies have been developed to speed up intersection tests of close bodies. For example: bounding spheres [14, 15], axis-aligned bounding boxes (AABBs) [2, 31], oriented bounding boxes (OBBs) [16], quantized orientation slabs with primary orientations (QuOSPOs) [11], and discrete-

oriented polytopes (K-DOPs) [19], octtree [25], BSP tree [27], brep-indices [3], k-d tree [13], bucket tree [8], hybrid tree [20], BVIT [28], and uniform space subdivision [31]. There are also collision detection algorithms for large environments such as I-COLLIDE [4] and CULLIDE [9].

In order to simulate realistic collision response [12, 25, 17] in deformable objects, a basic idea is to use discrete-time simulations. Penalty forces are generated to eventually separate deformable objects that are colliding. The function of penetration depth represents the distance and the direction of vertices in deformable objects. Additionally, response force is considered as a function of the relative velocity of colliding structures and their penetration depth. Many penetration depth approaches focus on specific problems in large penetration. However, these methods are not suitable to our algorithm.

The method proposed in this paper extends existing deformable modeling techniques by incorporating efficient ways to compute the contact surfaces in colliding objects. For computational efficiency a mass-spring system is used. As a result large environments of at least 10 thousand faces of deforming primitives can be simulated at interactive speeds.

1.1 SOFT BODIES

This section explains the modeling of soft-body behavior with a pressure force. The model in this paper is based on the classic soft body method presented in [22]. Pressure force is represented as a vector. Matyka and Ollila [22] computed pressure force vector by applying pressure at particular point to the normal vector of that surface, since pressure force is always acting in a direction normal to the surface. The expression for pressure at a specific point is:

$$\vec{P} = P \cdot \hat{n} \left[\frac{N}{m^2} \right]$$

where P is a pressure value, \hat{n} is a normal vector to surface where the pressure force is acting, m^2 is a

surface area, and N is a force dimension. Pressure forces can be evaluated with \vec{P} multiplied by $A[m^2]$ which is the area of the surface. The equation becomes:

$$\vec{F}_p = \vec{P} \cdot A[m^2]$$

Note that a larger force might generate a smaller pressure if it is distributed in a wider surface area. In contrast, a smaller force can create a larger pressure if the area is smaller. The value of P is from thermodynamic approximation known as ideal gas approximation. In the Clausius Clapeyron equation:

$$PV = nRT$$

where V is volume of the body, n is gas mol number, R is ideal gas constant, and T is a temperature. Then, the pressure can be generated by the temperature and volume of the soft-body as:

$$P = V^{-1}nRT$$

It is assumed that the temperature does not change while volume of a soft body changes.

1.2 EULER INTEGRATION

The following notation will be used throughout the paper.

Geometry

$v_0^i \dots v_n^i$ are the positions of vertices for object i .
 m^i is the mass of object i .

Direction

${}_n v_k^i$ is the velocity of v_k^i at time n .
 ${}_n n_k^i$ is the normal vector of v_k^i at time n .
 ${}_n f_k^i$ is the force of v_k^i at time n .

Note: k is vertex number of object i

For forces that are applied to the objects:

1) Rigid body has one external force.

${}_n f_g^i$ is the external force (gravity)

2) Flexible body has one external force and one internal force.

${}_n f_g^i$ is the external force (gravity)

${}_n f_s^i$ is the internal force (mass spring)

${}_n f_k^i = {}_n f_g^i + {}_n f_s^i + \text{Pressure force } (\vec{P})$

Euler 1st order integration is used to simulate the soft bodies for ease of implementation. Each time step in the animation is computed by the interval

from frame to frame. From Euler 1st integration we have:

$$\begin{aligned} d v_k^i / dt &= f_k^i / m^i \\ d v_k^i &= f_k^i dt / m^i \end{aligned}$$

Next,

$${}_{n+1} v_k^i = {}_n v_k^i + ({}_n f_k^i / m^i) dt$$

Then,

$${}_{n+1} V_k^i = {}_n V_k^i + {}_{n+1} v_k^i dt$$

Using integration, the position of each vertex in the flexible object is computed. Then collision detection and collision response will be applied.

2. ALGORITHM OVERVIEW

For collision detection and collision response of dynamically deforming bodies in a large environment, four steps of collision detection and collision response algorithm proceed as follows:

(1) *Multi-level Subdivided Bounding Box* (Multi-level SB): All soft bodies are surrounded by a bounding box (AABB) for tracking. Two points, a minimum and a maximum, define the bounding box. Then, the bounding box is subdivided to n -levels of subdivision. A 3-level subdivision is used for this simulation.

(2) *Box Hash Function (BHF)*: The box hash function is applied to each point that we use to create the subdivided bounding box. One subdivided bounding box has 8 points. Each point is hashed and given hash index by box hash function. A list of subdivided bounding boxes is put in hash table related to hash index. Then, the contact surface is computed from vertices that belong to subdivided bounding box in the lists of hash table. Multi-level SB Collide and the box-hash function which are further detailed in [24].

(3) *Collision for Flexible Models (CF)*: A line-triangle intersection test detects collision. To reduce high computation in triangle-triangle intersection, we use multi-passes of line-intersection. Using the line-triangle intersection, there are two possibilities: 1) one line from one triangle interferes in another triangle 2) two lines from one triangle intersect with another triangle. A triangle intersects with another triangle if and only if there is one or two lines in the triangle intersecting with another triangle.

(4) *Collision response (CR)*: For soft bodies, the objects change shape dynamically in response to its typical environment. Response is handled by using

normal vector at the surface of colliding object and response is provided accordingly.

2.1 MULTI-LEVEL SUBDIVIDED BOUNDING BOXES (Multi-level SB)

Step one of the algorithm is tracking with multi-level subdivided bounding boxes. Each object is surrounded by an axis aligned bounding box (AABB). A minimum point and maximum point are calculated to bound the soft body. The AABB is subdivided in to 2^n regions, where n is level of subdivision (see detail in [24]).

2.2 BOX HASH FUNCTION (BHF)

We use a **Box Hash Function (BHF)**. The idea is to use **BHF** and hash to 8 points that we use to create the subdivided bounding box. To facilitate hashing, initially a hash table is created which is based on grid size. The formula is:

$$\begin{aligned} \text{Index} = & (\text{floor}(\text{Grid.min.x} / \text{length}) * \text{xprime} + \\ & \text{floor}(\text{Grid.min.y} / \text{height}) * \text{yprime} + \\ & \text{floor}(\text{Grid.min.z} / \text{width}) * \text{zprime}) \\ & \% \text{bucketsize}; \end{aligned}$$

where length is grid length, height is grid height, width is grid width, grid.min.x, grid.min.y, and grid.min.z are minimum points of each grid, xprime, yprime, and zprime are any prime number for x, y, and z. Next, the **BHF** is applied to 8 points of the subdivided bounding box. Since there are 8 points, [0..7], in one subdivided bounding box, the **BHF** is:

$$\begin{aligned} \text{HashIndex} = & (\text{floor}(\text{point}[0..7].x / \text{length}) * \\ & \text{xprime} + \\ & \text{floor}(\text{point}[0..7].y / \text{height}) * \\ & \text{yprime} + \\ & \text{floor}(\text{point}[0..7].z / \text{width}) * \\ & \text{zprime}) \% \text{bucketsize} \end{aligned}$$

where xprime, yprime, and zprime are any prime number for x, y, and z, length, height, and width are length, height, and width of grid cell.

2.3 MULTI-PASS LINE-TRIANGLE INTERSECTION

All particles in the subdivided bounding boxes which are in the same hash index will be passed to the next step to find if the objects are colliding or not. Three passes of the line intersection test are as follows:

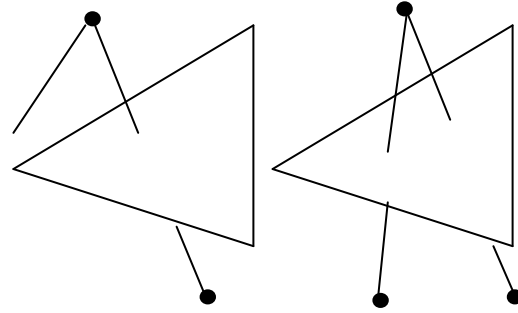


Figure 1: Two examples of line intersection; one line is intersecting with a triangle (left) and two lines are intersecting with a triangle (right) (black dots represent the particle points)

First pass: The algorithm takes first line of the triangle in 3D space. Then, it finds if the first line is penetrating the triangle or not. If so, the collision occurs and then finds the intersection point. The algorithm continues to the second pass

Second pass: the algorithm takes second line of the triangle. It checks again if the second line is interfering with the triangle or not. If so, the collision happens and calculates the intersection point. In case the first line and second line are interesting with triangle, the algorithm stops. On another hand, algorithm continues to next pass if either first or second line is intersecting.

Third pass: The algorithm take the third line and check for intersection test as describe in first pass and second pass

To test for line and triangle intersection, first we compute the point of intersection between line and plane. Next, we check if the point is in the triangle or not by using Barycentric coordinates. For a 3D plane, we drop the component in which the plane normal has the biggest absolute value. If the point is in the triangle, the point of intersection is returned. If not, the line is not intersecting with the triangle. section. Next the intersection point is passed to the to next step, which is collision response.

2.4 COLLISION RESPONSE

Collision response for soft bodies in most analytical methods is based on penalty forces. The methods calculate response forces based on penetration depths to resolve colliding objects. However, those methods are not appropriate for our soft body representation. To simplify the algorithm, we use a method for response based on contact

surface and the intersection point. There are two cases.

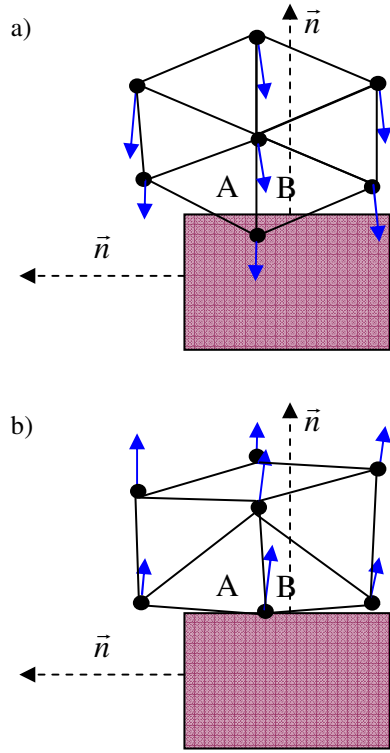


Figure 2: a) collision detection algorithm detects the collision between soft bodies and another object surface (dark vectors represent velocities of the particles and dashed vector represent normal vector of the surface) b) collision response pulls back the position of the particle to object surface.

In first case (figure 2), there is only one intersecting surface. The position of particle is changed to be at the intersection point of line. If there are more than one intersection lines for that particle, we take the smallest distance between particle and intersection point. Then, we move the position of the particle to the intersection point. The velocity of the particle will be calculated form normal vector at that surface.

In the second case (figure 3), there are two intersecting surfaces. We keep track of the first surface and second surface. Then the position of the particle will be pulled back to the corner between those two surfaces. The velocity of the particle will be computed by the average of normal vector from those two surfaces.

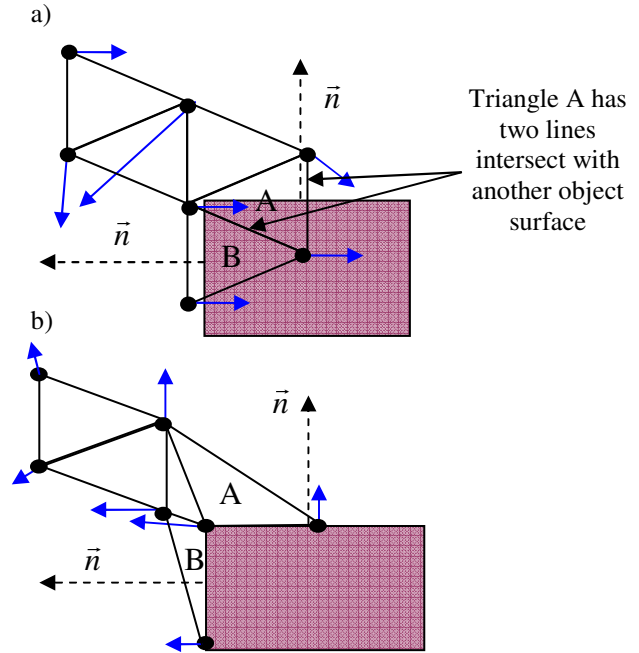


Figure 3: a) our collision detection algorithm detects the interference between two objects: (again dark vectors represent velocities of particles, dot vectors represent normal vectors at the particular surfaces) b) our collision response pulls back the velocities of the particles by the normal vector of the surfaces.

3. EXPERIMENTS

The simulation was implemented in Visual C++ with OpenGL. It ran collision detection in soft bodies up to 10,000 vertices at more than 30 frames per second on Pentium4 3.2 GHz Laptop with Nvidia GeForce Go 6800 GPU. Clearly this is more vertices than required for most games, but the simulation was done to test the limits of the algorithm. Figure 4 shows the collision detection between two dynamically deformed soft bodies. The gray spheres represent the colliding points between those two objects.

From the results, 2480 faces of soft bodies can detect collision at 120 frames per second. Figure 5 is a comparison of using our collision response and without collision response. The soft body bouncing with the ground is shown in figure 5a. Figures 5b and 5c illustrate that (b) some particles of the soft body are passing through the ground, then in (c) the particles colliding with ground are pulled up above the ground when collision is detected. For this simulation, frame rate is captured at 80 frames per second. Figure 6 shows soft bodies that are falling

down from the top to the bottom and colliding with another sphere at 45 frames per second.

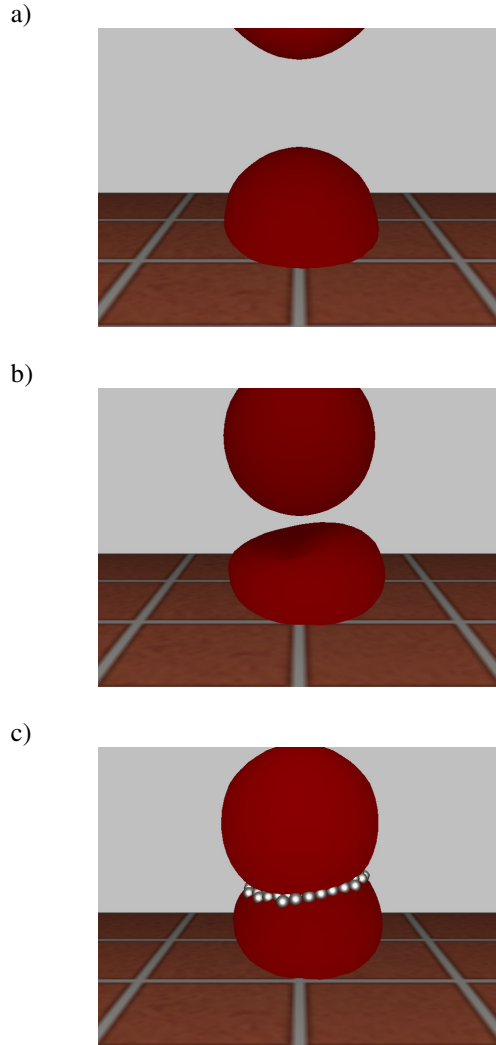


Figure 4: Two deformable objects are colliding; the gray sphere presents the colliding surfaces that have been captured by our collision detection algorithm; the simulation is captured at frame 100, 200, and 300 shown in a), b), and c) respectively

Figure 7 and 8 present the comparison between our multi-level SB collide with collision response and spatial hashing proposed presented in [31]. 10k faces of soft bodies are simulated and animated from frame 1 to frame 2000 measured in frame per second (FPS) with the same number of collision. The performance in figure 7 is decreasing when the frame number is increasing while the performance in figure 8 is slightly decreasing. Both show that our proposed method is more efficient than normal spatial hashing method.

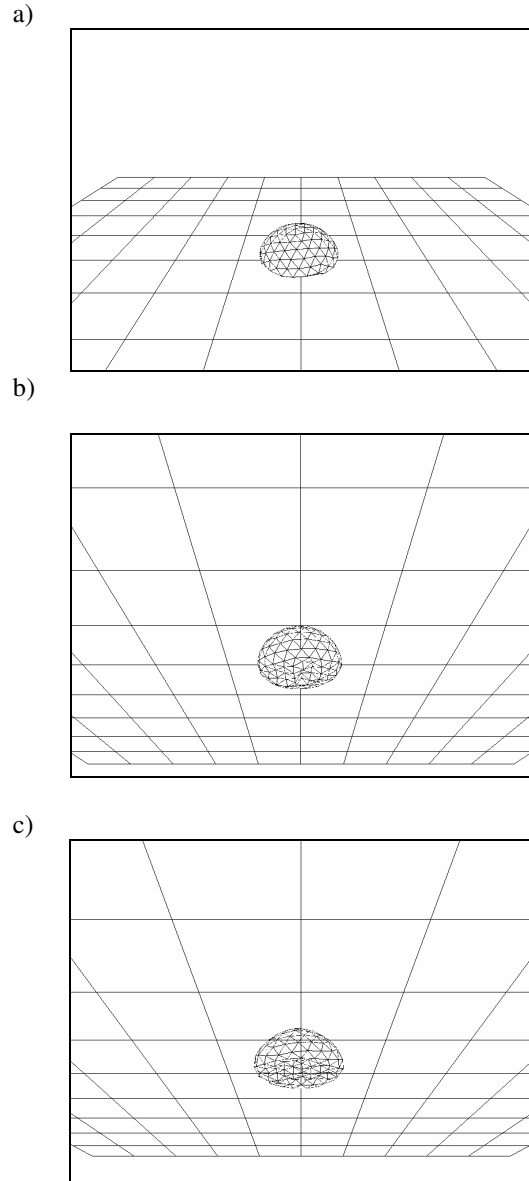


Figure 5: a) a soft body is colliding with the ground (eye position is set to look above the object and the direction is set to look down to the object and ground, b) some particles colliding with the ground are passing through the ground without collision response algorithm, c) some particle colliding with the ground are not passing through the ground with our collision response (eye position is set to look below the ground and direction is set to look up to the ground in b and c)

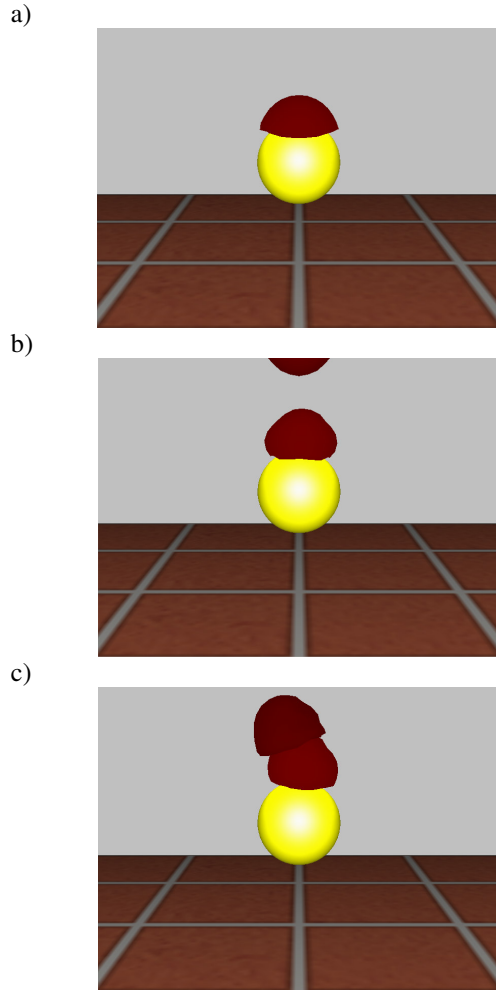


Figure 6: our deformable objects are falling down from the top to the bottom and colliding with one sphere; again the deformable objects are modeling by our modeling algorithm and detecting the collision by our collision detection algorithm shown in a), b), and c) respectively

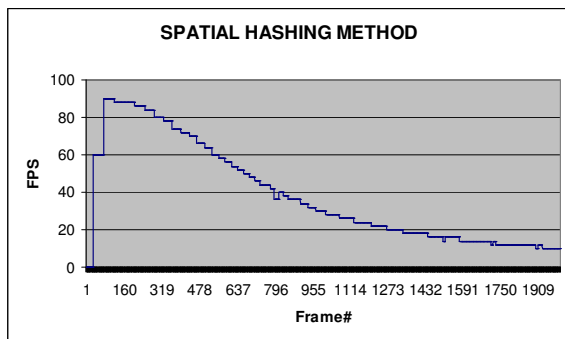


Figure 7: Performance of the collision detection by spatial hashing

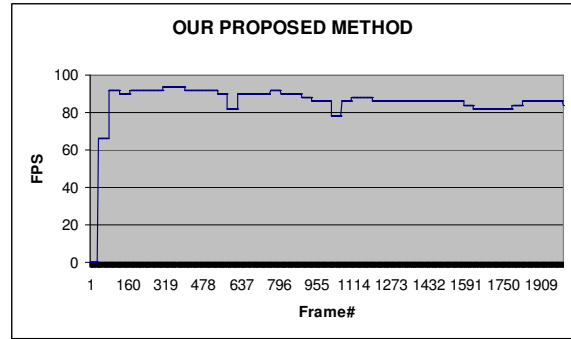


Figure 8: Performance of the collision detection by our proposed method

4. CONCLUSION AND FUTURE WORKS

In 3D games and simulations collision detection and collision response of soft bodies is a significant problem. Since soft bodies are composed of possibly thousands of moving particles, it becomes time consuming and adds additional complexity over regular, solid-body collision detection. To speed up collision detection and collision response, we propose a line-triangle intersection test to extend Multi-Level SB Collide. Several passes of line-triangle intersection give accurate results for collision detection and collision response in soft bodies. There are four main steps in the algorithm: *Multi-level Subdivided Bounding Box*, *Box Hash Function*, *Collision for Flexible Models* and *Collision response*. Experimental results show that our algorithm is efficient method for real-time collision detection and collision response in soft bodies for real time gaming.

There are several ways to extend this research. The first is to simplify the operation process to optimize the algorithm. The second possibility is to use a tree structure to determine the area of collision in the object body and perform the collision detection in the overlapped area to reduce time complexity. And finally to create additional, suitable test cases to compare with other algorithms.

REFERENCES:

- [1] Balaniuk, R. and Salisbury, K. 2002. "Dynamic simulation of deformable objects using the Long Elements Method," *Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 58–65.
- [2] Bergen, G.V.D. 1997. "Efficient Collision Detection of Complex Deformable Models Using AABB Trees." *Journal of Graphics Tools* 1997, vol. 2, Issue 4 (Apr.): 1-13.
- [3] Bouma, W. and Vanecek, G. Jr. 1991 "Collision Detection and Analysis in a Physical Based Simulation." *Eurographics Workshop on Animation and Simulation 1991* (Vienna) 191-203.

- [4] Cohen, J.D., Lin, M.C., Manocha, D., and Ponamgi, M. 1995. "I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scale Environments." *Proceedings of the 1995 symposium on Interactive 3D graphics 1995* (Monterey, CA, United States) 189-196.
- [5] Cotin, S., Delingette, H., and Ayache, N. 1999. "Real-time elastic deformations of soft tissues for surgery simulation," *Visualization and Computer Graphics, IEEE Transactions on*, Vol. 5 Iss. 1, pages 62-73.
- [6] Debunney, G., Desbrunz, M., Caniy, M.-P., and Barrx, A. 2000, "Adaptive simulation of soft bodies in real-time," *Comp. Anim.*, pages 133-144, May 2000.
- [7] Erickson, J., Guibas, L.J., Stolfi, J., and Zhang, L. 1999 "Separation-Sensitive Collision Detection for Convex Objects." *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms 1999*, Baltimore, Maryland, 327 - 336.
- [8] Ganovelli, F., Dingliana, J., and O'Sullivan, C. 2000. "Buckettree: Improving Collision Detection between Deformable Objects." *In Spring Conference in Computer Graphics SCCG 2000*, Bratislava, 156-163.
- [9] Govindaraju, N.K., Redon, S., Lin, M.C., and Manocha, D. 2003. "CULLIDE: Interactive Collision Detection Between Complex Models in Large Environments using Graphics Hardware." *Siggraph Eurographics Graphics Hardware 2003* (San Diego, CA, Jul. 26-27).
- [10] Gottschalk, S., Lin, M.C., and Manocha, D. 1996. "OBB Tree: A Hierarchical Structure for Rapid Interference Detection." *Proceeding of ACM Siggraph 1996* (New Orleans, Louisiana, Aug. 4-6), 171-180.
- [11] He, T. 1999. "Fast Collision Detection Using QuOSPO Trees." *Proceedings of the 1999 symposium on Interactive 3D graphics*, (Atlanta, Georgia, Apr. 26-29), ACM 1999 55-62.
- [12] Heidelberger, B., Teschner, M., Keiser, R., Müller, M., Gross, M. 2004. "Consistent Penetration Depth Estimation for Deformable Collision Response" *VMV 2004* Stanford, USA, November 16-18.
- [13] Held, M., Klosowski, J.T., Mitchell, J.S.B. 1995. "Evaluation of Collision Detection Methods for Virtual Reality Fly-Throughs." *Proceedings Seventh Canadian Conference on Computational Geometry 1995*, 205-210.
- [14] Hubbard, P.M. 1995. "Collision Detection for Interactive Graphics Applications." *IEEE Transactions on Visualization and Computer Graphics 1995*, 1(3):218-230.
- [15] James, D. L. and Pai, D.K. 2003. "Multiresolution green's function methods for interactive simulation of large-scale elastostatic objects," *Transactions on Graphics (TOG)*, Vol. 22 Iss. 1.
- [16] James, D.L. and Pai, D.K. 2004. "BD-tree: Output-Sensitive Collision Detection for Reduced Deformable Models." in *Proceedings of ACM SIGGRAPH 2004*, (Los Angeles, CA .Aug 8-12).
- [17] Kim, Y.J., Lin, M.C., and Manocha, D. 2002. "Incremental Penetration Depth Estimation Between Convex", *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*
- [18] Kim, Y.J., Lin, M.C., and Manocha, D. 2004. "Incremental Penetration Depth Estimation between Convex Polytopes Using Dual-Space Expansion", *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 2.
- [19] Klosowski, J.T., Held, M., Mitchell, J., Sowizral, H., and Zikan, K. 1998. "Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs." *IEEE Transactions on Visualization and Computer Graphics*, vol4 issue 1(Jan.): 21-36.
- [20] Larsson, T. and Akenine-Möller, T. 2001. "Collision Detection for Continuously Deforming Bodies" *In Eurographics 2001*, Manchester, UK, 325-333.
- [21] Matthew Moore, M. and Wilhelms, J. 1988. "Collision Detection and Response for Computer Animation", *Computer Graphics*, Volume 22, Number 4.
- [22] Matyka, M. and Ollila, M. 2003. "Pressure Model of Soft Body Simulation," *SIGRAD2003, November 20-21*.
- [23] Mesit, J.; Guha, R.;Hastings, E., 2004 "Optimized Collision Detection For Flexible Objects". International Conference on Computer Games: Artificial Intelligence, Design, and Education CGAIDE2004.
- [24] Mesit, J.; Guha, R.;Hastings, E., 2006 "Multi-level SB collide: collision and self-collision in soft bodies". International Conference on Computer Games: CGAMES2006.
- [25] Moore, M. and Wilhelms, J. 1988. "Collision Detection and Response for Computer Animation" *In proceedings Computer Graphics SIGGRAPH 1988*, 22(4):289-298.
- [26] Nixon, D., and Lobb, R. 2002. "A fluid-based soft-object model," *Comp. Graph. and App., IEEE*, Vol. 22 Iss. 4, pages 68-75, July-Aug. 2002.
- [27] Naylor, B., Amatodes, J.A., Thibault, W. 1990. "Merging BSP Trees Yields Polyhedral Set Operations." *In proceedings Computer Graphics SIGGRAPH 1990*, 24(4):115-124, 1990.
- [28] Otaduy, M.A. and Lin, M.C. 2003. "CLODs: Dual Hierarchies for Multiresolution Collision Detection." *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing 2003* (Aachen, Germany, Jul. 27-31), 94-101.
- [29] Rabaetje, R. 2003. "Real-time simulation of deformable objects for assembly simulations," *Proceedings of the Fourth Australian user interface conference on User interfaces*, Vol. 18.
- [30] Teran, J., Blemker, S., Ng Thow Hing, V., and Fedkiw, R. 2003. "Cloth & deformable bodies: Finite volume methods for the simulation of skeletal muscle," *Euro. Symp. on Comp. Anim. (SIGGRAPH Proc.)*, pages 68-74.
- [31] Teschner, M., Heidelberger, B., Müller, M., Pomeranets, D., and Gross, M. 2003. "Optimized Spatial Hashing for Collision Detection of Deformable Objects." *Vision, Modeling, and Visualization 2003*. (Munich, Germany, Nov. 19-21).
- [32] Witkin, A. and Baraff, D. 1993. "An Introduction to Physically Based Modeling," *SIGGRAPH Course Notes*.

Line-Triangle Test for Collision Detection and Response in Soft Bodies

Jaruwan Mesit
jmesit@cs.ucf.edu

Ratan K. Guha
guha@cs.ucf.edu

School of Electrical Engineering and Computer Science
University of Central Florida, Orlando, Florida 32826

KEYWORDS

soft body collision, animation, spatial subdivision, spatial hashing

ABSTRACT

Soft-body models are common in games to simulate cloth or elastic objects. To realistically simulate soft-body objects, collision detection and response is required. In addition, soft-body models must re-arrange their internal structure to react to the collision. This paper presents a new collision detection and response algorithm which can simulate a variety of soft-body material behaviors ranging from stiff to elastic. In this approach, a line-triangle intersection test is used for collision detection and force propagation is used for collision response. Implementation and experiments using the algorithm show that complex deformable objects composed of thousands of vertices can be animated at interactive speeds.

BIOGRAPHY



Ms. Mesit has been a Ph.D. student at University of Central Florida since 2002. She got her M.Sc. in computer science and B.Sc. in computer science from NIDA (National of Development Administration) and Rajabhat Institute Phetchaburi Thailand, respectively. She is working as a graduated research assistant with Dr.Ratan Guha. Her research area is computer graphics in collision detection for flexible objects such as cloths, soft body, and water.



Dr. Ratan Guha is a professor at the department of Computer Science, University of Central Florida, with primary interests in Computer Networks, Distributed Computing, Distributed Simulation, and Computer Graphics. He received his Ph.D. from the University of Texas at Austin, and his M.Sc. in Applied Mathematics, and B.Sc. with honors in Mathematics from the Calcutta University, India.