Tight Normal Cone Merging for Efficient Collision Detection of Thin Deformable Objects

Dong-Hoon Han, Chang-Jin Lee, Sangbin Lee, and Hyeong-Seok Ko

Seoul National University, Korea

Abstract

When simulating thin deformable objects such as clothes, collision detection alone takes a lot of computation. One way of reducing the computation is culling false-positives as much as possible. In the context of bounding volume hierarchy, Provot proposed a culling method that is based on hierarchical merging of normal enclosing cones. In this work, we investigate Provot's merging algorithm and show that there is some room for improvement. We propose a new merging algorithm, in the context of discrete collision detection, which always produces an equal or tighter mergence than Provot's merging. We extend the above algorithm so that it can be used in the context of continuous collision detection. Experiments show that the proposed method makes about 25% reduction in the number of triangle pairs for which vertex-triangle or edge-edge collision test has to be performed, and 18% reduction in time for collision detection.

CCS Concepts

• ***Computing methodologies*** → ***Collision detection;***

1. Introduction

When simulating the movement of clothing represented with triangular meshes, handling collisions occurring between cloth surfaces is an important problem. It calls for the checking of triangle pair collisions (**triangle pair test**), which eventually calls for finding the root of the cubic equations which are derived from the vertex-triangle and edge-edge coplanarity. Since each triangle pair test takes a non-negligible amount of computation, several methods for culling unnecessary checks (i.e., **false-positives**) have been proposed. This paper attempts an improvement to existing methods in false-positives culling.

In normal situations, the collar does not collide with the pants or legs. Culling of that kind can be achieved by employing the bounding volume hierarchy (BVH) [Hub93, KHM*98, Ber97]. On the other hand, *within* the same piece of cloth, Volino and Thalmann [VT94] pointed out that, if the triangle normals point in similar directions, there cannot occur self-collisions. More systematically, Provot [Pro97] proposed the use of the normal cone hierarchy (NCH) such that, if all the triangle normals of the sub-tree can be contained within a half side of the unit sphere, we can conclude there is no self-collision. The above was to cull false-positives at an arbitrary instant, i.e., in the context of discrete collision detection (DCD). Since most simulators adopt continuous collision detection (CCD), Tang et al. [TCYM09] extended the above to cull false-positives over the simulation time step duration Δt. Wang et al. [WLT*17] shortened the time taken for the contour test of [VT94] by employing an unprojected normal cone. Heo et al. [HSK*10] proposed a dual cone method that replaced the contour test by using a binormal cone in addition to the normal cone, and Wang et al. [WTWT18] improved the dual cone method to cover unhandled cases.

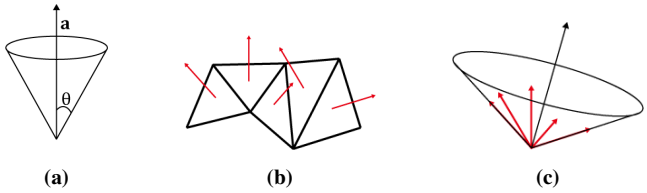


Figure 1: *Figures for defining notations/terminologies. (a) $B(\boldsymbol{a},\theta)$, (b) $S=\{N_1,\ldots,N_5\}$, (c) $\mathscr{C}(S)$.*

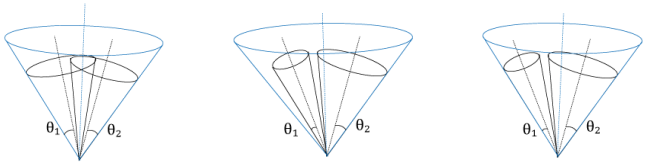
[VT94, Pro97, TCYM09] have been used as the industry standard over a decade. Although those methods have been culling a significant amount of false-positives, through this work, we find that there is still some room for improvement in culling binary tree-based BVH. We will show that, by applying simple modifications to the existing methods, more false-positives can be culled.

2. Preliminary

2.1. Notations and Terminologies

We will use the capital letters N,T to refer to a unit normal, triangle, respectively. In the context of CCD, although time t ranges

DOI: 10.2312/egs.20211021



(a) (b) (c)

Figure 2: *Results of merging two cones can vary depending on the employed merging algorithm. (a) By $\mathscr{M}_{Provot}$ when $\theta_1 \approx \theta_2$, (b) By $\mathscr{M}_{Provot}$ when $\theta_1 \ll \theta_2$, (c) By $\mathscr{M}_{Proposed}$*

$[t, t+\Delta t]$ during a time step, we will notate it as t ranges the interval $[0,1]$. $N(t)$ means the normal of triangle T at time t.

As shown in Figure 1 (a), we will use B to denote a normal cone subtended in the unit sphere whose apex is at the origin. We will use the notation $B(\mathbf{a},\theta)$ to refer the normal cone whose **axis** is unit vector $\mathbf{a}$ and **spread angle** from the axis is θ. The degenerate case when $\theta = 0$ will be called a **skinny cone**.

We will define two operations related to normal cones: **merging** $\mathscr{M}$ and **cone-enclosing** $\mathscr{C}$. $\mathscr{M}(B_1,B_2)$ means the construction of a new cone that encloses the given two cones B_1 and B_2 as shown in Figure 2. We may use subscripts such as $\mathscr{M}_{Provot}(B_1,B_2)$ and $\mathscr{M}_{Proposed}(B_1,B_2)$ to denote whose merging algorithm is used. When a set of unit normals $S = \{N_1, N_2, \ldots\}$ are given (Figure 1 (b)), the cone-enclosing $\mathscr{C}(S)$ denotes the construction of a cone that encloses all the elements of S as demonstrated in Figure 1 (c).

2.2. Normal Cone Hierarchy Construction

For each leaf node of the NCH, we take the skinny cone in the DCD context but the non-skinny cone in the CCD context [TCYM09]. Then, for each non-leaf node of NCH, we take the mergence $\mathscr{M}(B_L,B_R)$ as its normal cone, where B_L and B_R are the normal cones of its left and right children, respectively. This merging is continued bottom-up until arriving at the root node.

3. Compact Geometrical Merging of Two Normal Cones

3.1. Previous Merging Algorithm

Let's suppose that two cones $B_1(\mathbf{a}_1,\theta_1)$ and $B_2(\mathbf{a}_2,\theta_2)$ need to be merged. Provot [Pro97] proposed the merging algorithm

$$\mathscr{M}_{Provot}(B_1(\mathbf{a}_1,\theta_1), B_2(\mathbf{a}_2,\theta_2)) = B(\mathbf{a}_{Provot}, \theta_{Provot}), \quad (1)$$

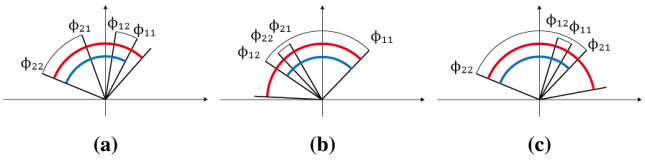
where

$$\mathbf{a}_{Provot} = unitize(\mathbf{a}_1 + \mathbf{a}_2) \quad (2)$$

$$\theta_{Provot} = \frac{1}{2} cos^{-1}(\mathbf{a}_1 \cdot \mathbf{a}_2) + max(\theta_1, \theta_2). \quad (3)$$

The above merging constructs a new cone B that does enclose the given two cones. However, as exemplified in Figure 2 (b), the resultant cone may not be as tight as one may expect. In this section, as demonstrated in Figure 2 (c), we propose a new merging algorithm that guarantees tight mergence in any case.

The above two cones B_1 and B_2 (if their axes are brought to the origin) will intersect the unit sphere, which will produce two circles Ω_1 and Ω_2, respectively, in 3D. Then, the problem of finding the mergence of the two cones is reduced to finding a circle Ω on the surface of the unit sphere that encloses Ω_1 and Ω_2. By restricting the above to the plane formed by the origin and the centers of Ω_1 and Ω_2, the problem can be converted to a cross-sectional 2D problem. Referring to the cross-sectional diagram Figure 3, let the **angular spans** (ϕ_{11},ϕ_{12}) and (ϕ_{21},ϕ_{22}) represent the cones $B_1(\mathbf{a}_1,\theta_1)$ and $B_2(\mathbf{a}_2,\theta_2)$, respectively, with $\phi_{11} < \phi_{12}$, $\phi_{21} < \phi_{22}$, and $\phi_{11} + \phi_{12} \leq \phi_{21} + \phi_{22}$, where all of ϕ_{11}, ϕ_{12}, ϕ_{21}, and ϕ_{22} are within the range $[-\pi, \pi)$. Then, Provot's merging is reduced to



(a) (b) (c)

Figure 3: *Cross-sectional drawing of two cones. (a) non-inclusive, (b) B_1 includes B_2, (c) B_2 includes B_1. Red and blue arches show the results of $\mathscr{M}_{Provot}$ and $\mathscr{M}_{Proposed}$, respectively.*

$$\alpha_{Provot} = \frac{1}{4}(\phi_{11} + \phi_{12} + \phi_{21} + \phi_{22}), \quad (4)$$

$$\beta = \frac{1}{2}(\phi_{21} + \phi_{22} - \phi_{11} - \phi_{12}), \quad (5)$$

$$\theta_{Provot} = \frac{1}{2}\beta + max\left(\frac{\phi_{12} - \phi_{11}}{2}, \frac{\phi_{22} - \phi_{21}}{2}\right) \quad (6)$$

where α denotes the cone axis direction in the cross-section.

3.2. Proposed Merging Algorithm

The new merging algorithm we propose in this paper is stated as

$$\alpha_{Proposed} = \frac{1}{2}\{max(\phi_{22},\phi_{12}) + min(\phi_{21},\phi_{11})\}, \quad (7)$$

$$\theta_{Proposed} = \frac{1}{2}\{max(\phi_{22},\phi_{12}) - min(\phi_{21},\phi_{11})\}. \quad (8)$$

The resultant span $(\alpha_{Proposed} - \theta_{Proposed}, \alpha_{Proposed} + \theta_{Proposed})$ includes (ϕ_{11},ϕ_{12}) and (ϕ_{21},ϕ_{22}). We prove that the proposed merging algorithm $\mathscr{M}_{Proposed}$ always produces an equal or tighter resultant cone than $\mathscr{M}_{Provot}$.

Theorem 1 In the above, $\theta_{Provot} \geq \theta_{Proposed}$.

Proof Let $D = \theta_{Provot} - \theta_{Proposed}$. Referring to Figure 3, if $\phi_{22} \leq \phi_{12}$, B_2 is included in B_1, and if $\phi_{22} > \phi_{12}$ and $\phi_{11} \geq \phi_{21}$, B_1 is included in B_2. We will call above two cases as **inclusive** cases. Then,

$$D = \begin{cases} \frac{1}{4}\{(\phi_{21}+\phi_{22}) - (\phi_{11}+\phi_{12})\} & \text{when inclusive,} \\ \frac{1}{4}\{-(\phi_{12}-\phi_{11}) - (\phi_{22}-\phi_{21}) & \\ \quad +2max(\phi_{12}-\phi_{11}, \phi_{22}-\phi_{21})\} & \text{otherwise,} \end{cases} \quad (9)$$

both of which are non-negative. □

The above implies that D will be large (1) when the two axes $\mathbf{a}_1$ and $\mathbf{a}_2$ are distant for the inclusive case, and (2) when two angular spreads θ_1 and θ_2 are very different for the non-inclusive case.

3.3. Extension to CCD

In merging cones in the CCD context, Tang et al. [TCYM09] addressed the issue of how to accommodate time-varying triangle normals at the leaf nodes to the NCH. They found that the normal $N(t)$ for all $t \in [0,1]$ comes within the cone that encloses

$$H = \{N(0), unitize(N(0)+N(1)-\delta), N(1)\}, \quad (10)$$

where $\delta = (\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)$ and $\mathbf{v}_a$, $\mathbf{v}_b$, $\mathbf{v}_c$ are the displacements of three vertices of the triangle. They obtained the cone enclosing using $\mathscr{M}_{Provot}$, i.e.,

$$\mathscr{C}_{Tang}(H) = \mathscr{M}_{Provot}(\mathscr{M}_{Provot}(B_1(N(0),0), B_2(N(1),0)), B_3(unitize(N(0)+N(1)-\delta),0)). \quad (11)$$

We can expect that, if we use $\mathscr{M}_{Proposed}$ instead of $\mathscr{M}_{Provot}$ in Equation 11 (let's call it $\mathscr{C}_{Proposed}(H)$), $\mathscr{C}_{Proposed}(H)$ can be tighter than $\mathscr{C}_{Tang}(H)$. According to the discussion given in Section 3, the spread angle difference between $\mathscr{C}_{Tang}(H)$ and $\mathscr{C}_{Proposed}(H)$ is zero when $N(0)+N(1)-\delta$ points the mid-point of $N(0)$ and $N(1)$, and the difference increases as $N(0)+N(1)-\delta$ points away from the mid-point.

3.4. Implementation

This section explains how the above cross-sectional analysis is applied to the real situation in 3D, i.e., if we denote the proposed merging in 3D as $\mathscr{M}_{Proposed}$, we explain how to find $\mathbf{a}_{Proposed}$ and $\theta_{Proposed}$ in

$$\mathscr{M}_{Proposed}(B_1(\mathbf{a}_1,\theta_1), B_2(\mathbf{a}_2,\theta_2)) = B(\mathbf{a}_{Proposed}, \theta_{Proposed}).$$

Firstly, we perform the cross-sectional calculation. For that, we imagine the cross-section in a polar coordinate system with $\mathbf{a}_1$ as the polar axis. Then, we can draw the other axis $\mathbf{a}_2$ by calculating $\cos^{-1}(\mathbf{a}_1 \cdot \mathbf{a}_2)$. Then, the spread angles θ_1 and θ_2 give ϕ_{11}, ϕ_{12}, ϕ_{21}, and ϕ_{22}. We can use Equations 7 and 8 to calculate $\alpha_{Proposed}$ and $\theta_{Proposed}$, respectively. Now, only the calculation of $\mathbf{a}_{Proposed}$ remains, which is nothing but the spherical linear interpolation (Slerp) [Sho85] of $\mathbf{a}_1$ and $\mathbf{a}_2$ by the angular amount $\alpha_{Proposed}$.

Note that, for the inclusive cases (Figure 3 (b), (c)), we set the outer cone as the result without performing the Slerp. Also, if any of $\theta_{Proposed}$, θ_1, and θ_2 is greater than or equal to $\frac{\pi}{2}$, then the spread angle of the parent nodes (as well as the current node) is set to $\frac{\pi}{2}$ without further calculation.

4. Experiments

To see how much reduction the proposed method brings in the spread angle, triangle pair test, and time, we implemented the proposed method and collected various statistics as we ran the clothing simulation with two ensembles and two onepiece dresses, which are shown in Figure 4. The computer we used for the experiments was Intel Core i9-9900 3.60GHz CPU.

There are several other techniques that can affect culling performance. In this work, we considered whether to include the culling techniques NP filter [TMT10a], BVTT-Front [TMT10b], R-Triangles [CTM08] to the simulator. For all simulations in this work, we included the NP filter and R-Triangles, but we excluded BVTT-Front since it may hinder fair comparison.

(a) *Ensemble 1* **(b)** *Ensemble 2* **(c)** *Onepiece 1* **(d)** *Onepiece 2*

Figure 4: *Experimented cases*

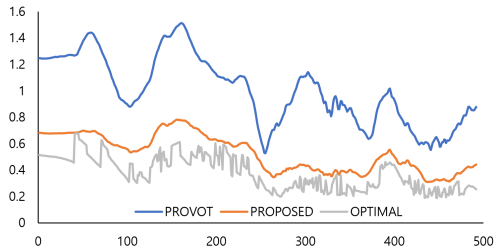


Figure 5: *Spread angle at a non-leaf node over simulated time steps*

4.1. Reduction in the Spread Angle of BVH Cones

In the following, PROPOSED means the simulations with ($\mathscr{M}_{Proposed}$, $\mathscr{C}_{Proposed}$) and PROVOT means the simulations with ($\mathscr{M}_{Provot}$, $\mathscr{C}_{Tang}$). In the draping, for all cases, PROPOSED and PROVOT produced the same results. To see how much spread angle the PROPOSED reduces compared to PROVOT, we selected a node somewhere in the middle of the BVH while simulating Ensemble 2. For that node, as the PROPOSED and PROVOT simulations progress, we measured the spread angle. As Figure 5 shows, the difference is conspicuous.

Barequet and Elber [BE05] proposed an algorithm that can produce the *optimal* bounding cone that includes the given normal set, where the number of normals in the set can be arbitrary. In order to construct the normal cone of a NCH node with this method, the normals of *all* the descendent triangles of the node must be given, and it is an $O(n \log n)$-time algorithm, where n is the number of given normals. Therefore the method is inadequate for NCH construction. However, the method can be used to see how the spread angles of PROPOSED and PROVOT compare with the optimal spread angle. For that purpose, we calculated the optimal normal cone for a particular node using Barequet and Elber's method, which is plotted in grey (named as OPTIMAL) in Figure 5.

4.2. Reduction in the Number of Triangle Pair Test

Table 1 compares the number of triangle pair tests. PROPOSED takes clearly fewer tests than PROVOT. The reduction rate was on

	# of Triangles	PROVOT	PROPOSED
Ensemble 1	18,920	39,997.78	30,795.03
Ensemble 2	34,877	87,607.41	60,140.21
Onepiece 1	20,849	50,518.85	39,499.42
Onepiece 2	61,949	165,412.73	122,977.72

Table 1: *Average number of triangle pair test*

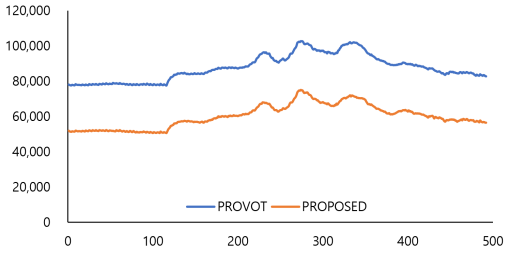


Figure 6: *Number of triangle pair tests over simulated time steps*

average 25.45%. Figure 6 plots the number of triangle pair tests in Ensemble 2 as the simulation progresses up to the 500th frame. It shows that the reduction is quite consistent across the frames. (In the other cases, we obtained similar results.)

4.3. Reduction in Time

Table 2 summarizes the time comparison. We observed that, for PROPOSED, construction of BVH takes slightly more time, but the remaining part of collision detection (including the BVH traversal) takes significantly less time. In total, the reduction of PROPOSED with respect to PROVOT in time was on average 18.24%. (We note that OPTIMAL is computationally very expensive compared to PROPOSED. Optimal takes on average about 4,000 ms to construct BVH.)

5. Conclusion

When simulating thin deformable objects such as clothes, collision detection alone takes a lot of computation. One way of reducing the computation is culling false-positives as much as possible. In the context of BVH, Provot [Pro97] proposed a culling method that is based on hierarchical merging of normal enclosing cones. When two cones are given, the merging produces a single cone that encloses the two given cones. For culling efficiency, it is desirable the merging should be done as tightly as possible.

In this work, we investigated Provot's merging algorithm $\mathscr{M}_{Provot}$ and found that there is some room for improvement in the case of binary merging. We proposed a new merging algorithm $\mathscr{M}_{Proposed}$ which always produces an equal or tighter mergence than $\mathscr{M}_{Provot}$. Inspired by $\mathscr{C}_{Tang}$, by employing $\mathscr{M}_{Proposed}$, we proposed a new cone enclosing algorithm $\mathscr{C}_{Proposed}$ targeted for CCD. The experiment showed that ($\mathscr{M}_{Proposed}$, $\mathscr{C}_{Proposed}$) out-performs ($\mathscr{M}_{Provot}$, $\mathscr{C}_{Tang}$) in the spread angle, triangle pair tests, and time.

Acknowledgements

This work was supported in part by the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (MEST) (NRF-2018M3E3A1057302) and ASRI (Automation and Systems Research Institute at Seoul National University).

	BVH Construction		Collision Detection		Total	
	PROVOT	PROPOSED	PROVOT	PROPOSED	PROVOT	PROPOSED
Ensemble 1	3.03	3.67	17.17	12.72	20.12	16.39
Ensemble 2	8.52	11.86	37.77	22.10	46.29	33.97
Onepiece 1	3.96	4.61	18.01	14.29	21.97	18.90
Onepiece 2	11.93	14.06	62.69	48.33	74.62	62.40

Table 2: *Time taken for BVH construction and CCD (ms)*

References

[BE05] BAREQUET G., ELBER G.: Optimal bounding cones of vectors in three dimensions. *Information Processing Letters 93*, 2 (2005), 83–89. 3

[Ber97] BERGEN G. V. D.: Efficient collision detection of complex deformable models using aabb trees. *Journal of graphics tools 2*, 4 (1997), 1–13. 1

[CTM08] CURTIS S., TAMSTORF R., MANOCHA D.: Fast collision detection for deformable models using representative-triangles. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games* (2008), pp. 61–69. 3

[HSK*10] HEO J.-P., SEONG J.-K., KIM D., OTADUY M. A., HONG J.-M., TANG M., YOON S.-E.: Fastcd: Fracturing-aware stable collision detection. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2010), SCA '10, Eurographics Association, p. 149–158. 1

[Hub93] HUBBARD P. M.: Interactive collision detection. In *Proceedings of 1993 IEEE Research Properties in Virtual Reality Symposium* (1993), IEEE, pp. 24–31. 1

[KHM*98] KLOSOWSKI J. T., HELD M., MITCHELL J. S., SOWIZRAL H., ZIKAN K.: Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE transactions on Visualization and Computer Graphics 4*, 1 (1998), 21–36. 1

[Pro97] PROVOT X.: Collision and self-collision handling in cloth model dedicated to design garments. In *Computer Animation and Simulation'97*. Springer, 1997, pp. 177–189. 1, 2, 4

[Sho85] SHOEMAKE K.: Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1985), SIGGRAPH '85, Association for Computing Machinery, p. 245–254. 3

[TCYM09] TANG M., CURTIS S., YOON S.-E., MANOCHA D.: Iccd: Interactive continuous collision detection between deformable models using connectivity-based culling. *IEEE Transactions on Visualization and Computer Graphics 15*, 4 (2009), 544–557. 1, 2, 3

[TMT10a] TANG M., MANOCHA D., TONG R.: Fast continuous collision detection using deforming non-penetration filters. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games* (2010), pp. 7–13. 3

[TMT10b] TANG M., MANOCHA D., TONG R.: Mccd: Multi-core collision detection between deformable models using front-based decomposition. *Graphical Models 72*, 2 (2010), 7–23. 3

[VT94] VOLINO P., THALMANN N. M.: Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. *Computer Graphics Forum 13*, 3 (1994), 155–166. 1

[WLT*17] WANG T., LIU Z., TANG M., TONG R., MANOCHA D.: Efficient and reliable self-collision culling using unprojected normal cones. *Computer Graphics Forum 36*, 8 (2017), 487–498. 1

[WTWT18] WANG T., TANG M., WANG Z., TONG R.: Accurate self-collision detection using enhanced dual-cone method. *Computers & Graphics 73* (2018), 70–79. 1