

Suejung B. Huh  
Dimitris N. Metaxas

## A collision resolution algorithm for clump-free fast moving cloth

---

Published online: 24 May 2006  
© Springer-Verlag 2006

---

S.B. Huh (✉) · D.N. Metaxas  
Division of Computer and Information  
Sciences, Rutgers The State University of  
New Jersey, 110 Frelinghuysen Road,  
Piscataway, NJ 08854-8019, USA  
{suejung,dnm}@cs.rutgers.edu

**Abstract** Cloth animation is an important area of computer graphics due to its numerous applications. However, so far a fast moving cloth with multiple wrinkles has been difficult to animate because of the cloth clump problem. Cloth clumps are the frozen areas where cloth pieces are clustered unnaturally — an obstacle in making a realistic cloth animation. Hence we present a novel cloth collision resolution algorithm that prevents clump formation during fast cloth motions. The goal of our resolution algorithm is to make cloth move swiftly without having any unnatural frozen cloth clumps, while preventing any cloth-

cloth and rigid-cloth penetrations at any moment of a simulation. The non-penetration status of cloth is maintained without the formation of cloth clumps regardless of the speed of cloth motion. Our algorithm is based on a particular order that we found in the resolution of cloth collisions, and can be used with any structural modeling approaches such as spring-masses or finite elements. This paper includes several realistic simulation examples involving fast motions that are clump-free.

**Keywords** Cloth animation · Collision resolution · Collision detection

---

### 1 Introduction

Cloth clumps are areas of clustered cloth pieces. In dynamic cloth simulations, cloth clumps are often formed and hamper the process of making a realistic animation. Due to recent developments in cloth animation systems' stability [2, 8, 12], simulating fast moving cloth without having excessive damping has become possible. However, because of the clumping problem, making fast moving cloth with interesting cloth interactions has been extremely difficult. Hence in this paper, we present a new approach of clump-free cloth collision resolutions that allows realistic animations of fast complex moving cloth.

Cloth clumps in fast motion are very noticeable and look unnatural when they exist. Such cloth clumps are formed when the velocities of colliding cloth nodes are averaged. Since the node velocities are averaged, these

nodes move similarly from then on, and appear as a clump. Although one may attempt to divide a clump into cloth pieces using repulsion forces, or implementing a cloth thickness, it is not always possible to break up a cloth clump into pieces in time. Although the proximity limit (thickness) constraint or the repulsion force may help cloth from penetrating, this method alone does not prevent cloth from forming clumps, nor guarantee the penetration free condition of cloth. When cloth is in fast motion, it may penetrate itself despite having repulsion (penalty) forces for proximity violations, unless penetrations are strictly forbidden. If the initial cloth pieces given to the penalty handling module already have self-penetrations, even a rigorous cloth thickness enforcement would not help to resolve self-penetrations. To minimize the chance of self-penetrations, the technique of averaging the velocities of colliding cloth nodes (handling colliding cloth nodes as one rigid entity temporarily) has been widely



**Fig. 1.** A cloth in fast motion

used. However, this technique tends to leave the colliding cloth pieces in their vicinities after the collision resolution.

Once such cloth clumps are formed, it is very difficult to correct them, and having such clumps is an obstacle to creating an interesting and fast moving cloth animation. When numerous cloth pieces are involved in simultaneous collisions, like the cases shown in Fig. 13(a,d), correcting self-penetrations and resolving cloth clumps are simply not tractable.

Cloth collision resolutions can be categorized by how they handle multiple cloth collisions comprehensively and simultaneously. Volino et al. [16] used barycentric node relations to solve multiple collisions, while Provot [13] and Bridson et al. [6, 7] employed the impact zone (IZ) method that treats colliding cloth pieces as a rigid entity. Since the method proposed by Volino et al. maintains barycentric relations of the positions and the velocities of colliding nodes, it effectively finds the non-colliding positions of multiple nodes. However, it is unknown how the subsequent collisions resulting from collision resolutions should be resolved. When Bridson et al. introduced the idea of maintaining an exact cloth proximity limit — a reasonable idea to represent cloth thickness — they adopted the IZ approach, originally proposed by Provot [13]. This IZ approach handles simultaneous multiple collisions as a group, and treats the participating colliding cloth pieces as one rigid entity by averaging the velocities of colliding cloth nodes. Physically this means that the colliding cloth pieces behave like a rigid entity after the collision resolution. If the initial cloth pieces were in fast motion, the magnitude of averaged node velocities will be overwhelmingly greater than the magnitudes of the repelling velocities from the proximity forces. Hence clumps will be formed.

### 1.1 Why are cloth clumps formed?

Cloth collisions are highly inelastic collisions from a mechanical point of view, where the participating cloth pieces lose a significant amount of kinetic energy. By this token, averaging the velocities of colliding cloth nodes appears not to pose any problem. However, this is somewhat misleading. A set of discretized cloth pieces actually represents a continuous cloth surface. When cloth pieces are in collisions, only small parts of the cloth surfaces represented by these cloth pieces may actually be in collision. Averaging the node velocities of all the colliding cloth pieces means that we assume all the cloth surfaces represented by cloth pieces are in collisions and complete contact with each other. This is an unrealistic assumption. The consequence of assuming such contact in cloth collisions is the loss of an unnecessary amount of kinetic energy in collision resolutions. This is the physical reason why cloth clumps are formed after such self collision resolutions.

### 1.2 Contributions

Cloth collisions are of two particular types: cloth-cloth (self) and rigid-cloth. Since these two types of cloth collisions represent physically different situations, it is reasonable to resolve them separately. However, the resolution of either type of cloth collisions may create subsequent new collisions. Without having a method to handle such subsequent collisions, one cannot achieve comprehensive complete collision resolutions.

The first contribution of this paper is to identify a possible order in cloth collision responses, and to build an algorithm based on that order. Based on observations made in Sect. 2, we give reasons for the particular choice of order in cloth collision responses. Adhering to this order ensures the success of collision responses and guarantees maintaining the penetration-free status of the cloth at all times.

To avoid cloth clumping, the authors have employed the simultaneous self collision resolution method previously proposed in [10], while it was originally inspired by the rigid body simultaneous collision resolution method presented in [1]. Our self collision resolution method uses colliding clusters (CC), and has been incorporated with rigid-cloth collision resolutions. The second contribution of this paper is to present an algorithm that incorporates resolutions of two types of cloth collisions comprehensively, while cloth clumps are avoided.

This paper is organized as follows. In the next section, we identify a possible order in the resolution process of cloth collisions. Based on this discussion, in Sect. 3, we build a cloth collision response algorithm. From Sects. 3.1 to 3.5, each phase of the algorithm is presented. Section 4 reports the modeling choices of our simulation examples along with the simulation results. Finally Sect. 5 presents our conclusions.

## 2 Fundamental issues in cloth collisions

In this section, we discuss several fundamental issues in cloth collisions before we develop our algorithm. The interrelated issues among two types of cloth collisions (self and rigid-cloth) and proximity violations are investigated.

### 2.1 Self collisions

Before we discuss the order of cloth collision resolutions, i.e. which type of collisions should be resolved first, we investigate why subsequent collisions are an important issue of cloth collision resolutions.

In Fig. 2, we show an example where a self collision inadvertently resolves/creates a rigid-cloth collision. We have an initial condition, as shown in Fig. 2(a), where we have two cloth pieces and a shaded rigid body in a cross sectional view. Assuming two cloth pieces collide as shown in Fig. 2(b), we could have two different collision resolution results as shown in Figs. 2(c,d), depending on the resolution scheme that we choose to use. If we move  $\beta$  with respect to  $\alpha$  to avoid collisions, we will have a result similar to the one shown in Fig. 2(c). On the other hand, we will have a situation similar to Fig. 2(d), if we move  $\alpha$  with respect to  $\beta$ . (Our actual self collision resolution method is different from these, which are presented only to explain subsequent collisions.) In (c), we unwittingly resolved the collisions between  $\beta$  and the rigid body, while, on the contrary, new collisions between  $\alpha$  and the rigid body were created in (d). These newly created collisions are subsequent collisions that resulted from a collision resolution, which exemplifies why one should be ready to handle subsequent collisions, when it resolves any cloth collisions, to maintain a coherent and penetration-free cloth system.

Another point to note in cloth collisions is that any subsequent self collisions caused by rigid-cloth collision

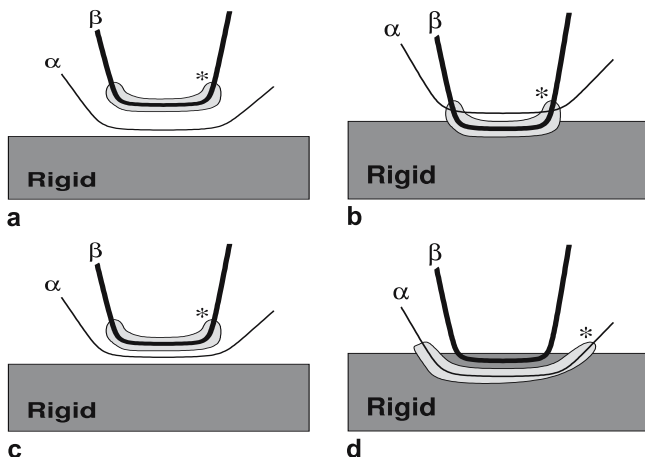


Fig. 2. An example of two self collision responses

responses are inherently rigid-cloth collisions, assuming that we ignore the impact of cloth to rigid objects. For example, if several layers of cloth are pushed by a rigid body, the positions and the velocities of all pushed cloth pieces are determined by the pushing rigid surface. Figure 3 shows a situation where we can expect subsequent self collisions if the rigid object moves leftward. Assume that we have rigid-cloth collisions between the cloth area (\*\*\*) and the rigid body. If we resolve these collisions by moving the colliding cloth with respect to the rigid surface, we may have subsequent collisions between the cloth in (\*) and the cloth in (\*\*). Given this situation, we know the cloth node positions in (\*) are to be computed with respect to the cloth pieces in (\*\*), whose positions are determined by the colliding rigid surface. This means that, although these subsequent collisions appear as cloth-cloth collisions, they are inherent in rigid-cloth collisions. From this observation, we concluded that any subsequent self collisions caused by rigid-cloth collision resolutions should be handled in conjunction with rigid-cloth collision resolutions. For this reason, in our algorithm, the rigid-cloth resolution process is designed to remain in the same phase until all subsequent collisions are handled. On the contrary, in self collision responses, if any subsequent collisions are rigid-cloth collisions, the rigid-cloth collision resolution module has to be invoked. Hence in our algorithm the self collisions are resolved first, before we start rigid-cloth collision resolutions. The procedures of rigid-cloth and self collision resolutions are shown in Figs. 12 and 8, respectively, and in Sect. 3 the whole cloth collision response algorithm is presented.

### 2.2 Proximity violation constraints

PV constraints (implementing thickness) have been widely used to avoid actual penetrations. Preventing cloth pieces from being too close to each other is also known to be effective to reduce false collision detections coming from numerical precision errors.

However, when PVs are cluttered with penetrations, handling of such PVs might aggravate the current penetra-

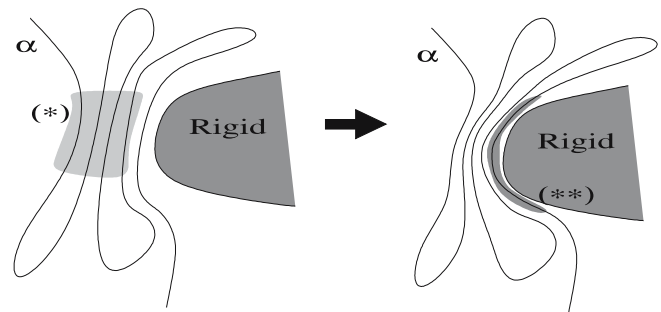


Fig. 3. Collisions between layers of cloth and a rigid body

tions. When a set of cloth pieces are given to a proximity violation handling module, the cloth pieces are expected to be free from penetrations. In other words, the cloth given to a proximity violation handling module has to be penetration free. Otherwise, PVs with penetrations cannot be handled properly. Figure 4 shows two penetrating faces while their nodes are in proximity violations. If PVs are handled in this situation, the nodes  $N_1$ ,  $N_2$  and  $N_3$  will move away from the surface where the other triangle lies. Apparently this will only aggravate the prior penetration. To avoid such situations, our algorithm resolves all the penetrations before we apply proximity violation constraints.

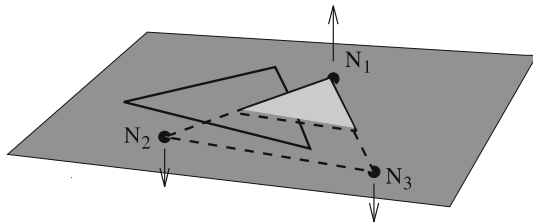


Fig. 4. A penetration case

### 3 Algorithm

Because of the issues discussed so far, we have concluded the following.

1. Self collisions are to be resolved before rigid-cloth collisions.
2. Penetrations are to be cleared before PVs.

Hence the particular collision resolution order that we choose is

```
self=>self PV=>rigid-cloth
=>rigid-cloth PV.
```

Using this order, the algorithm shown in Fig. 5 is built, where the initial cloth status is required to be penetration-free. As long as the initial cloth is penetration-free, the final result of our algorithm is guaranteed to be penetration-free. In each module, by not letting any penetrations pass unresolved to the next time step, we maintain the validity of our system.

Note that the detection of collisions and PVs are performed only once right after the dynamic integration. However, each resolution phase thereafter may create conditions that do not exist at the time of detection. Hence all the modified nodes from each phase have to be checked and resolved for the collisions before they go to the next module, for the soundness of the system. In the following sections, each phase of our algorithm is explained in detail.

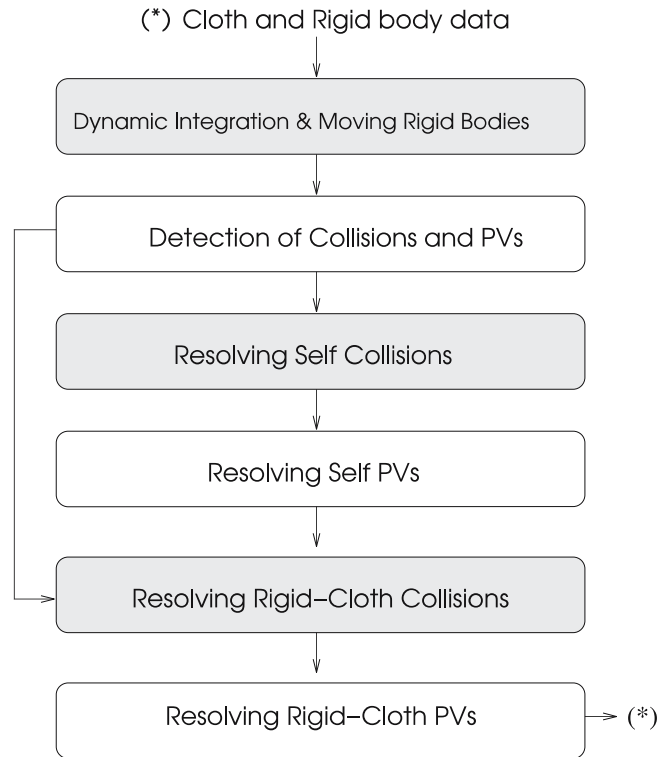


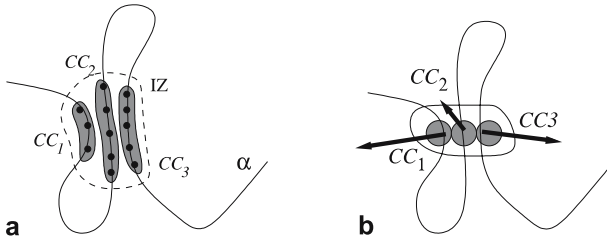
Fig. 5. Cloth collision response algorithm

#### 3.1 Detection of collisions and PVs

In this section, we discuss the first phase of our algorithm — the detection of collisions. Collision detections are performed for all the possible edge-edge pairs and face-node pairs while all the edges, nodes, and faces are registered in axis-aligned voxels. At any instantaneous moment of a simulation, an element may be contained in several voxels simultaneously [17]. Each voxel is a cube where the length of one voxel side is slightly longer than the maximum edge length. All element pairs in a voxel are tested for possible collisions or violations. In the case of collision detections, the elements that occupied the voxel at the previous time step and the elements that have passed the voxel from the previous time step to the current time step are also tested for collisions.

Instead of just reporting colliding pairs, the collision data is stored for the purpose of our simultaneous collision resolution. In the following, the data structure needed is explained.

After we find all the colliding pairs, we group all the inter-colliding entities (faces, nodes, and edges) into one IZ. An IZ is a data structure to represent a group of cloth pieces in multiple collisions simultaneously. In Fig. 6(a), the dotted area, IZ, represents a cloth clump that has three cloth pieces: CC1, CC2 and CC3. If an edge is a member of an IZ, then its two ending nodes are also members of the



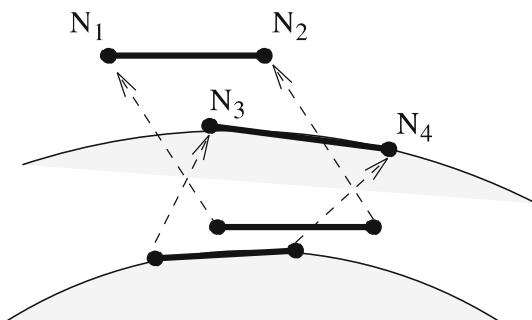
**Fig. 6a,b.** Saving collision and PV data. **a** Identifying CCs in an IZ. **b** Units of collision responses

same IZ, and the same rule applies to the nodes of a member face. Within an IZ, nodes are grouped based on the edge connectivity. Such edge-connected nodes are called a colliding cluster (CC). A more rigorous definition of CC and IZ can be found in [10]. Figure 6(a) shows a cross sectional view of a cloth piece, where the dotted circle represents an IZ and the shaded areas are CCs. These CCs will be handled as units of collision responses as shown in Fig. 9. Treating these CCs as collision units, instead of responding to an individual collision, enables us to resolve collisions in a simultaneous manner with less worries of cloth clumps.

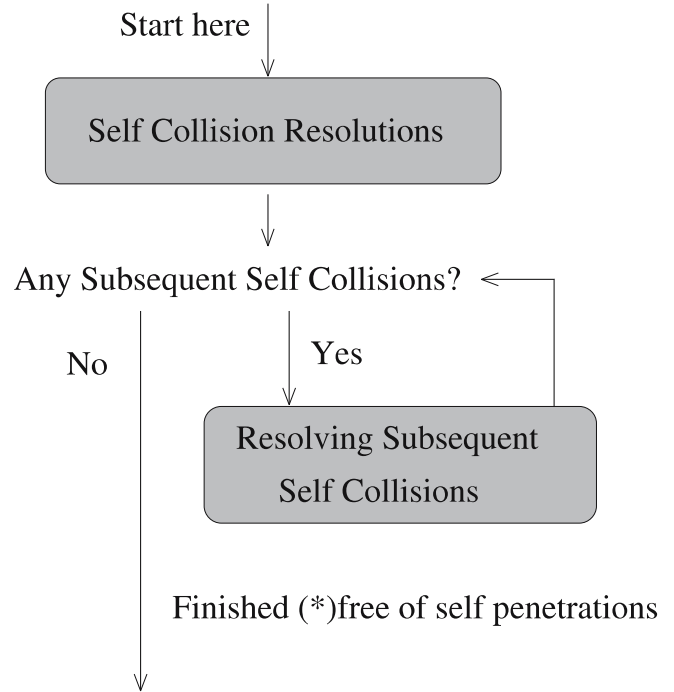
### 3.1.1 False collision detections

To detect collisions happening within a time step, we use the trajectories of node positions from the previous time step to the current time step, while using the collision detection method proposed in [13]. The trajectory of each node is assumed linear, and the velocity of the node to be constant. However, this detection scheme may give us false collision reports at times.

For example, Fig. 7 shows a case where the cloth nodes  $N_1$  and  $N_2$  and the rigid body nodes  $N_3$  and  $N_4$  are moving in the dotted directions. Apparently, this is not a collision case. However, it can be reported as a collision, since we assume linear trajectories with constant velocities. There could be a time  $t$  within a time step, when all the nodes are coplanar and edges made from these nodes are cross-



**Fig. 7.** False collision detection



**Fig. 8.** A self collision resolution module

ing. In that case, the collision detection module reports a collision. Similarly, we can have false collision detections for a face-node case. When exact collision detections are required, false collisions can be eliminated by checking whether the node orientation is flipped with respect to the normal of colliding pieces after collisions. For the example presented, checking the relative orientations will filter the false collision detection. In our test simulations, checking of flipped relative orientations is used for rigid-cloth collision detections.

### 3.1.2 Detection of PVs

Proximity violations are tested only with the node positions of a current time step. PVs are tested with the pairs of node-faces and edge-edges to see if they violate the proximity constraint (2 mm in our animations). Upon detecting a PV, the detection module stores the pairs of the PV entities (node-faces, edge-edges) to each node. These lists of PV violation information stored in each node PV are used later at the PV resolution modules, explained in Sects. 3.3 and 3.5.

### 3.2 Self collision responses

Given detected collision information, we resolve self collisions first. In this section, we explain how we resolve self collisions including multiple self collisions using a simultaneous approach. This section also covers how the CC

method is fundamentally different from the IZ method, and shows why it works for the fast moving cloth.

### 3.2.1 Resolutions of multiple self collisions

Resolving multiple self collisions has been reported to be difficult by many research groups. Several methods have been suggested for this problem including the IZ method by [6, 13], and the CC method by [10]. The IZ resolution method averages the velocities of all the colliding particles [13], and further adjusts the velocity of each particle with the angular velocity computed from the initial condition to preserve the rotation information [6].

Figure 9 shows multiple self collisions where  $A$ ,  $B$  and  $C$  represent CCs. The initial velocities of CCs are  $v_A$ ,  $v_B$ , and  $v_C$ , respectively, where  $v_B = 0$ . When  $v_{A,x}$  and  $v_{A,y}$  represent the  $x$ -axis and the  $y$ -axis components of  $v_A$ , let  $v_{A,y} = -v_{C,y}$ ,  $v_{A,x} = -v_{C,x}$ .

Given this situation, the IZ method gives the result shown in Fig. 9(b), where all the balls have zero velocities. Since the balls  $A$  and  $B$  have lost all the momentum in the collision, they will stay in the vicinity of each other. Hence the balls will appear to be clinging to each other, which is not desirable.

Figure 9(c) shows the result of our CC method. The CC method uses a simultaneous collision response approach. Each ball (CC) is handled as if it were a separate entity. Hence, in our example,  $A$ ,  $B$  and  $C$  represent all different CCs. The CC method treats collisions as elastic collisions [4] to divide them quickly apart. Therefore, the resolution result is identical to the case of three elastic balls, where the result conserves the initial momentum, and loses less kinetic energy than the case not using CCs. Consequently, the cloth nodes will not have any undesirable sticking effect.

However, because of the temporal and spatial coherence, when multiple cloth nodes collide, nodes in their vicinity tend to collide also within a small time period. This means that in the CC method the velocities of colliding regions could keep exchanging their values. Our algorithm improved this problem by using IZ crusts, as discussed in Sect. 3.3. Our approach to solve this problem has been possible since our algorithm handles self PVs and self collisions separately and differently.

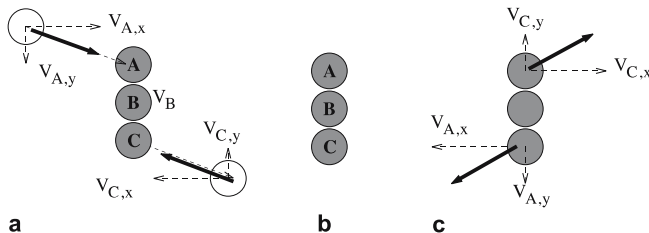


Fig. 9. Three colliding CCs

### 3.2.2 Applying a simultaneous approach

In this section, we discuss how the actual resolution of self collisions is performed. Resolving self collisions is finding proper velocities and positions of colliding nodes. A proper position of a node is a position where the node itself and the faces and the edges connected to this node do not collide with any objects in the environment. Finding such positions for a set of nodes could be a daunting task. However, our algorithm takes advantage of the fact that at the end of the previous time step the positions of these nodes were collision-free. With a time step size generally used (10–0.1 msec), nodes move only a tiny distance within a time step, even in the case where the nodes are in fast motion. For this reason, we use the node positions of the previous time step for the current positions of colliding nodes. When the cloth nodes are in self collisions, we simply restore the node positions of the previous time step as their current positions.

To find velocities of all CCs in one IZ, we first have to identify the colliding direction of CCs. In our method, the colliding direction is the average of all CC normals, where a CC normal is the average of all the nodal normals in the CC. The velocity adjustment is performed only for the components of nodal velocities parallel to this colliding direction. A colliding node keeps the tangential components of its velocity after a self-collision.

Let  $n$  be the number of CCs,  $v_i$  be the velocity of the  $i$ -th CC,  $CC_i$ , before collisions, and  $v'_i$  be the velocity after a collision response of  $CC_i$ . Given  $v_i$ , we have to find  $v'_i$  for all CCs. From Newtonian physics, we have that

$$(v'_j - v'_i) = k_e(v_i - v_j) \quad (1)$$

for all the colliding pairs  $CC_i$  and  $CC_j$ , where  $k_e$  denotes the elastic coefficient. Based on the momentum conservation law, we have that  $\sum_i m_i v_i = \sum_i m_i v'_i$ , where  $m_i$  is the mass of  $CC_i$ . Notably, since we assume that all the cloth nodes have an identical mass,  $m_i$  is simply the number of nodes in  $CC_i$ . In the case of elastic collisions, we have

$$v'_i = total_{2v} - v_i, \quad (2)$$

where  $total_{2v} = (\sum_i 2m_i v_i) / (\sum_i m_i)$ .

A cyclic collision case is the case where we find a loop in collisions. For example, when there are collisions between  $A$  and  $B$  and also  $B$  and  $C$ , if  $C$  and  $A$  also collide, then this is called a cyclic collision case. Cyclic collisions can also be solved by Eq. 2, since Eq. 1 still holds for all colliding pairs in the case of cyclic collisions and single self collisions as well.

Although none of these restored nodes have any collisions within themselves, since the positions of nodes in the vicinity have been changed, we have to check for subsequent collisions. Upon subsequent collisions, we can restore the positions of the previous time step of relevant nodes. An interesting point to consider is the case where

we meet another IZ in subsequent collisions. However, since we are just restoring the positions of the previous time step, the current resolution will not affect any resolution process of the other IZs. Furthermore, since all the node positions of previous time steps are given free of collisions, the number of cloth nodes is invariant from the previous time step, the self collision resolution process is computationally well-bounded and guaranteed to work.

### 3.3 Self PV resolutions

The goal of the self PV resolution module is to reduce as many PVs as possible. Self PV resolutions do not create any subsequent self collisions, since PV resolutions only incorporate collision impacts on node velocities.

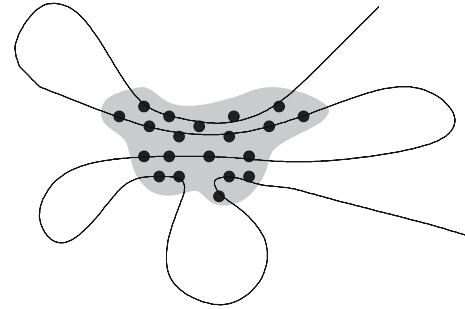
Proximity constraints have been employed by many researchers by putting small spring forces between cloth-cloth and cloth-rigid pieces to prevent penetrations [2, 5, 14]. Although the proximity constraint is a viable technique, since it is easy to understand and simple to use, it permits penetrations at times. Bridson et al. [6] suggested more rigorous proximity constraints by simulating cloth thickness. Their method demands cloth pieces to maintain an exact cloth thickness with the benefit of reducing actual collisions. However, with their method, the cloth nodes tend to lose a substantial amount of momentum in the PV resolution process, since the velocities of cloth nodes are adjusted to reach the proximity limit at the end of integration time steps.

Another important point of having proximity constraints is that one can avoid the chance of numerical errors in detecting collisions by not allowing two entities to be too close or to penetrate. Cloth penetrations have been repaired using the majority orientation method by several research groups [3, 15]. However, in a multiple collisions case, such majority orientation data is not always reliable. Moreover, having penetrations while handling PVs may aggravate the penetrations, as discussed in Sect. 2.2. Hence, in our method, the penetration-free condition is assured before we handle PVs, to ensure the integrity of a system.

Figure 10 shows a cross sectional view of a cloth piece, where the shaded area represents cloth nodes in multiple PVs. To resolve PVs, we first identify the closest face or edge with respect to a node among the list of faces and edges in PV violations. The distance from a node to the closest entity is used to compute the proximity constraint force. Once we identify the closest entity, we apply proximity forces to the node in the direction that alleviates the worst PV violation.

### 3.4 Rigid-cloth collision responses

In rigid-cloth collision responses, the resolution is performed only on the cloth nodes to find non-colliding

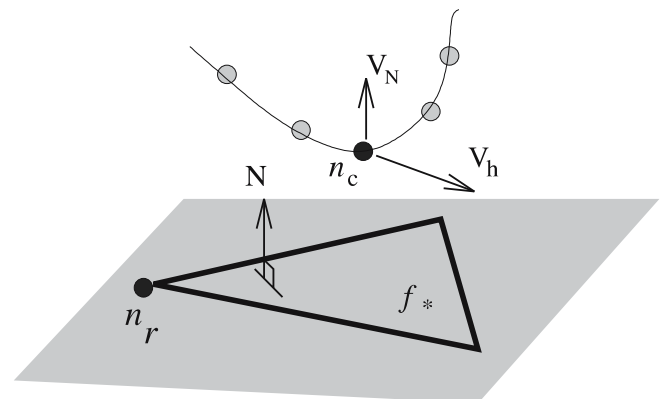


**Fig. 10.** A set of nodes in multiple proximity violations (PVs)

positions and proper velocities with respect to the rigid body, while the colliding rigid nodes are intact. The input given to the rigid-cloth collision response module is free of self collisions, and the result of rigid-cloth resolutions should be free from rigid-cloth and self collisions. The detected rigid-cloth collisions are given in the form of a set of IZs. Then the resolution resolves each IZ separately.

The rigid-cloth resolution starts by identifying the relevant rigid faces of a colliding cloth node. The relevant rigid faces of a cloth node is a set of faces connected to the colliding rigid entities. For example, when a cloth  $n_c$  is one of the nodes in a face  $f_c$  that collides with a rigid node  $n_r$ , the adjacent faces of the rigid node  $n_r$  are included in the relevant rigid faces of the cloth node  $n_c$ . After identifying such relevant rigid faces, we identify the closest rigid face  $f_*$  from the cloth node  $n_c$ . This rigid face  $f_*$  plays a key role in the rigid-cloth resolution of the node  $n_c$ . For a resolution, the relative position of the cloth node  $n_c$  with respect to the position of the rigid face  $f_*$  of the previous time step is restored, given the current position of the rigid face  $f_*$ .

The velocity of  $n_c$  is resolved based on the face normal  $N$  and the velocity  $v^r$  of  $f_*$ . As shown in Fig. 11, when  $v$  is the velocity of  $n_c$  before the collision and  $v'$  is the one



**Fig. 11.** A collision of a cloth node  $n_c$  and a rigid face  $f_*$

after,  $v_N$  is the normal component of  $v$  and  $v_h$  the tangential component of  $v$ . After a rigid-cloth collision, only the normal component of the velocity is affected. The normal component of  $v'_N$  is computed as

$$v'_N = \begin{cases} k_v v_N & \text{if } v_N - v_N^r \geq 0 \\ v_N^r & \text{otherwise} \end{cases}, \quad (3)$$

where  $k_v$  is the impact coefficient, and  $0 \leq k_v \leq 1$ . When the cloth node  $n_c$  moves away from a rigid surface  $f_*$ , i.e.  $v_N - v_N^r \geq 0$ , the normal velocity of  $n_c$  is intact. Otherwise the cloth node will inherit the normal velocity of the rigid face. This resolution is based on the intuition that the rigid-cloth collision will affect the velocity of the colliding cloth node at the moment of the collision.

The frictional force acting on the tangential component of  $v'_h$  is determined as follows

$$f_t = k_f |v'_N - v_N| (v_h - v_h^r), \quad (4)$$

where  $k_f$  is the friction coefficient. Again, when the cloth node  $n_c$  moves away from a rigid surface, it is  $|v'_N - v_N| = 0$ . Hence the node has no friction in such cases.

Once the node positions are resolved, we check whether they create any new collisions, as shown in Fig. 12. We have to check not only the resolved nodes but the edges and faces connected these nodes, since these elements are also affected by the position changes of the nodes. The resolved cloth pieces may be found to create other self-collisions. If a resolved cloth piece intersects another cloth piece, the new colliding cloth piece is in a rigid-cloth collision situation. The resolved positions of the nodes in the new cloth piece are computed the same way. This iteration is repeatedly done until we find no newly created self collisions. Upon failures of such iterative rigid-cloth resolutions, although it rarely happens, we use barycentric node relations with respect to the rigid surface. The new cloth node positions are computed to restore the relative positions of cloth nodes of the previous time step with respect to the rigid body.

### 3.5 Rigid-cloth PV resolutions

The rigid-cloth PV resolution is done similarly to the self PV resolution, except that the velocity of cloth nodes will be adjusted similarly to the case of self PV response.

## 4 Experiments

For the examples presented in this paper, particle-based systems are used with triangulated cloth surfaces. For

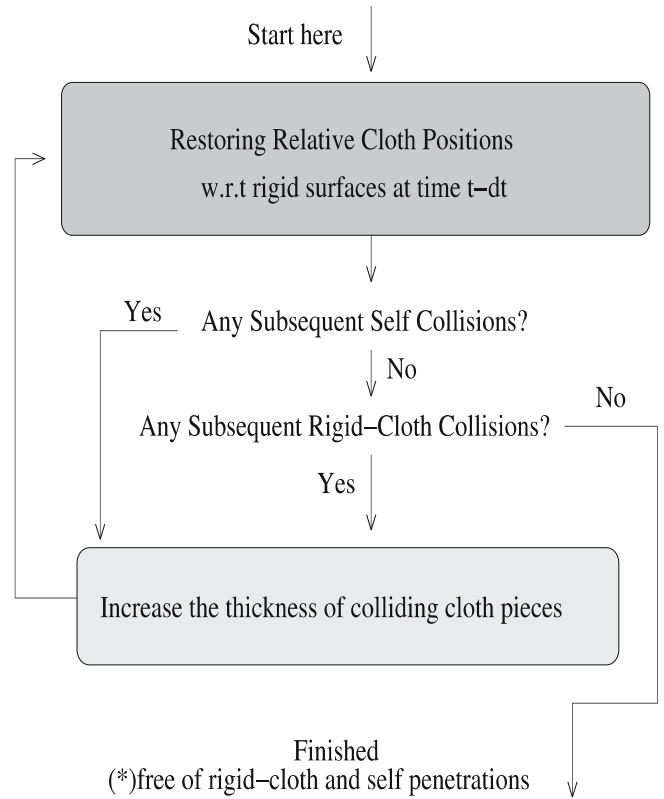


Fig. 12. A rigid-cloth collision resolution module

bending forces, a crossing edge is added for each non-boundary edge as shown in Fig. 14. The integration is done by the implicit conjugate gradient method proposed by [2], while the buckling handling proposed by [8] is adopted. However, our algorithm is independent of any specific definition of a cloth structure and the choice of the dynamic simulation method.

In our examples, the cloth surface was subdivided twice by the loop-subdivision method [9, 11], using the collision avoidance method proposed by Bridson et. al [6]. The environment in our examples had about 0.1 million faces after subdivision. The typical frame time was 1-4 minutes, depending on the number of collisions involved at the moment, using a 3GHz Pentium IV CPU on a Linux machine.

The pictures shown in Fig. 13 are the snapshots from the simulations. The cloths in Fig. 13(a,c) show situations of multiple cloth collisions. The circled areas are where the previous methods are likely to fail because of extensive self and rigid-cloth interactions. Figures 13(b,d) show the views from below of the frames in Figs. 13(a,c), respectively, with the cylinder in the center rendered invisible. The images of dancers shown in Fig. 13(f,e) are



Fig. 13. Fast moving cloth examples

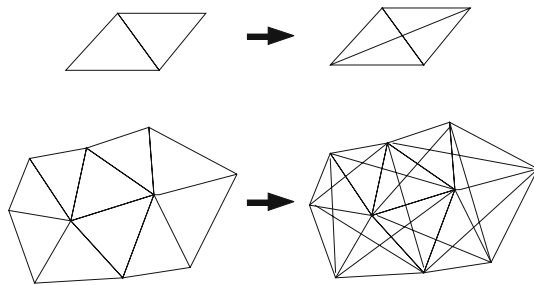


Fig. 14. Adding edges to triangulated meshes

snapshots from animations using our collision resolution module. The cloth in (e) was stiffer than the one in (f), and simulated at twice the dancing speed of the one in (f).

## 5 Discussion

The major contribution of this paper is the identification of a particular order in handling cloth collisions and building a cloth collision response algorithm based on this order. Our algorithm takes two types of cloth collisions into account comprehensively, in conjunction with adequate treatments of iterative subsequent collision issues, which is essential to provide a reliable cloth collision response algorithm. Our algorithm has allowed us to animate difficult and fast moving cloth motions that are clump-free.

**Acknowledgement** The authors were funded by an NSF ITR 0205671. The animations in Figs. 1 and 13(a,b,c,d) were rendered by Paul Kanyuk. We appreciate his work. Authors also appreciate the support of AUTODESK for Maya which had been used for designing clothes.

## References

1. Baraff, D.: Analytical methods for dynamic simulation of non-penetrating rigid bodies. In Proceedings of SIGGRAPH 23(3), Computer Graphics Proceedings, Annual Conference Series, pp. 223–232. ACM Press/ACM SIGGRAPH (1989)
2. Baraff, D., Witkin, A.: Large steps in cloth simulation. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 43–53. ACM Press/ACM SIGGRAPH (1998)
3. Baraff, D., Witkin, A., Kass, M.: Untangling cloth. In Proceedings of SIGGRAPH, Vol. 22(3) of Computer Graphics Proceedings, Annual Conference Series, pp. 862–870. ACM Press/ACM SIGGRAPH (2003)
4. Beer, F.P., Johnston-Jr., E.R.: Vector Mechanics for Engineers – Dynamics. WCB/McGraw-Hill
5. Breen, D.E., House, D.H., Wozny, M.J.: Predicting the drape of woven cloth using interacting particles. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 365–372. ACM Press/ACM SIGGRAPH (1994)
6. Bridson, R., Fedkiw, R., Anderson, J.: Robust treatment of collisions, contact and friction for cloth animation. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 594–603. ACM Press/ACM SIGGRAPH (2002)
7. Bridson, R., Marino, S., Fedkiw, R.: Simulation of clothing with folds and wrinkles. In Symposium on Computer Animation, pp. 28–36. Eurographics/ACM Siggraph, Eurographics Association (2003)
8. Choi, K.-J., Ko, H.-S.: Stable but responsive cloth. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 604–611. ACM Press/ACM SIGGRAPH (2002)
9. deRose, T., Kass, M., Truong, T.: Subdivision surfaces in character animation. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 85–94. ACM Press/ACM SIGGRAPH (1998)
10. Huh, S., Metaxas, D.N., Badler, N.I.: Collision resolutions in cloth simulation. In IEEE Proceedings of Computer Animation, Seoul, South Korea (2001)
11. Loop, C.: Triangle mesh subdivision with bounded curvature and the convex hull property. Technical Report MSR-TR-2001-24. Microsoft Research (1995)
12. Provot, X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In Graphics Interface, pp. 147–155. Graphics Interface (1995)
13. Provot, X.: Collision and self-collision handling in cloth model dedicated to design garments. In Proc. of Graphics Interface, pp. 177–189. Graphics Interface (1997)
14. Volino, P., Courchesne, M., Thalmann, N.M.: Versatile and efficient techniques for simulating cloth and other deformable objects. In Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 137–144. ACM Press/ACM SIGGRAPH (1995)
15. Volino, P., Magnenat-Thalmann, N.: Collision and self-collision detection: efficient and robust solutions for highly deformable surfaces. In Terzopoulos, D., Thalmann, D. (eds.) Computer Animation and Simulation '95, pp. 55–65. Springer, Berlin Heidelberg New York (1995)
16. Volino, P., Magnenat-Thalmann, N.: Accurate collision response on polygonal meshes. In Proceedings of the 2000 Conference on Computer Animation, pp. 154–163. IEEE Computer Society (2000)
17. Zhang, D., Yuen, M.: Collision detection for clothed human animation. In Proceedings of the 8th Pacific Graphics Conference on Computer Graphics and Application, pp. 328–337. IEEE Computer Society (2000)



SUEJUNG B. HUH, PHD has been a research associate in Computer and Information Sciences at Rutgers, The State University of New Jersey, USA since 2003. Dr. Huh was a post-doc in 2002 at University of Pennsylvania, USA, where she received her PhD in Computer Science. Her research interests are in deformable object modeling and human modeling.



DIMITRIS N. METAXAS, PHD has been a Professor in the Division of Computer and Information Sciences and Professor in the Department of Biomedical Engineering at Rutgers University since 2001. He is directing the Center for Computational Biomedicine, Imaging and Modeling (CBIM). He received a PhD in Computer Science from the University of Toronto, Ontario, Canada in 1992. Dr. Metaxas has been conducting research in computer vision, computer graphics and medical imaging. Dr. Metaxas has published over 200 research articles and a book in these areas. He is a member of the editorial board of Medical Imaging, Associate Editor of GMOD, and Editor of CAD. Dr. Metaxas received two best paper awards for his work on fluid modeling and a best paper award and a patent on his heart modeling work. He was awarded a Fulbright Fellowship in 1986, is a recipient of an NSF Research Initiation and Career awards, an ONR YIP, and is a Fellow of the American Institute of Medical and Biological Engineers. He is also the Program Chair of ICCV 2007 and the General Chair of MICCAI 2008 .