Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

The Intersection Contour Minimization Method for Untangling Oriented Deformable Surfaces

Juntao Ye and Jing Zhao †

Institute of Automation, Chinese Academy of Sciences

Abstract

The Intersection Contour Minimization (ICM) method [VM06] has been proven to be an effective history-free algorithm for resolving collisions between non-oriented deformable surfaces. In many circumstances, however, surface orientation information are often implied in the context. Being completely blind to such information in the ICM method often leads to unexpected result: either failure or slow convergence in certain intersection configurations. By introducing the concept of "repulsive normal" into ICM, many of those once-failure configurations can be resolved successfully. Even for those once-successful configurations, repulsive normals usually speed-up the convergence. Moreover, the ICM method that was originally designed for polygonal meshes can actually be adapted to resolve collisions between a polygon mesh and an analytical surface. This paper presents one such extension – collisions between a polygon mesh and a capsule.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism —Animation

1. Introduction

Cloth simulation relies on proper mechanical models that can reproduce the behaviors of cloth materials, and suitable numerical methods that can integrate the equations of motion in reasonable computation time. Over the past two decades, the mass-spring model has become the dominant cloth dynamics models because it allows fast simulation [Pro95, BW98], although we recently see a trend that the more accurate but slower continuum model [VMF09] is again gaining more attentions due to the hardware computing power boost. Implicit integration methods have been widely used ever since the adoption of the semi-implicit backward Euler [BW98] by Baraff and Witkin. Other variants include backward differential formula (BDF2) [CK02], etc. These progresses have made the cloth simulation succeed in several industrial fields, such as movies and video games. However in many practical applications, collision handling plays an even more important role, and is often the key to a successful simulation system. The state-of-the-art technique is still a process plagued with problems, especially for complex scenarios.

An active research direction is algorithms that try to prevent intersections based on continuous collision detection (CCD). In a context where deformable surfaces are represented as triangular meshes, the objective of CCD is to find out, for a time interval, the profile of all pairwise triangle intersections, including the first contact location and the corresponding contact time instant. Generally fifteen cubic equations are to be solved for each triangle-triangle pair test [Pro97]. The motivation behind these methods is an anticipation that if the last timestep was intersection-free then the next timestep is also intersection-free. The most comprehensive way of preventing intersections to occur is presented by Bridson et al. [BFA02]. Most recently, Harmon et al. [HPSZ11] proposed another history-based collision detection and response method, for the domain of geometric modeling. By requiring the surfaces to be intersection-free at the start, the trajectory of the penetrating regions sweep out a so-called space-time interference volume (STIV). The interference is resolved by reducing the magnitude of STIV to zero through a minimization process. Unfortunately, whatever complexity these methods are, a truly intersection-free state is hard to maintain, due to round-off errors or the constraints enforced by the modeling or simulation context. If the simulation is subject to external constraints such as col-

† email:{juntao.ye | jing.zhao}@ia.ac.cn

DOI: 10.2312/SCA/SCA12/311-316

lisions with solid objects whose behaviors are physically incorrect, these constraints might force the cloth into an illegal state even with the presence of collision handling. One such example, given by Baraff et al. [BWK03], was the cloth pinched by two interpenetrating geometric objects. In these situations if some part of the cloth ends up on the wrong side, it will be "remembered" as not, and the simulator will work hard to keep it on the wrong side forever.

Since all methods relying on a legal prior state fail if such a state is not available, people resort to history-free methods. Unlike the history-based techniques which try to prevent collision from happening, history-free methods allow penetrations to occur but try to detect and "repair" them.

Baraff et al. [BWK03] were probably the first to put forward a history-free collision resolution algorithm. It worked by a Global Intersection Analysis (GIA) of the current state of penetration, and proposed actions to untangle the cloth. This method tracks intersections by reconstructing the intersection paths of the colliding regions. Once a closed contour has been identified, the penetration region is flood-filled on the mesh and the collision response mechanism would bring this region back to its non-colliding relative positions, together with their "flypapering" method. However, this method is effective only if the intersection regions are well-defined by a closed path, usually without the intervention of the surface boundaries. When boundaries are involved, not all possible intersections results in definite regions being clearly on the wrong side, thus flood-filling fails. Wicke et al. [WLG06] further investigated this problem. They analyzed all possible intersections and presented a classification of all possible intersection paths considering boundaries. For paths that define an inside region, this region is minimized as did in [BWK03]. For paths that do not define an inside region, the path is pushed to the cloth boundary for a separation.

Volino and Magnenat-Thanlmann [VM06] proposed an Intersection Contour Minimization (ICM) method that does not suffer from the limitations of open paths. Their method resolves intersections between two surfaces by inducing relative displacements which minimize the length of the intersection contour. There is no need to identify colliding surface regions as done by Baraff et al. [BWK03]. Aiming at a broader application field, no classification of the intersection paths is necessary and all types of contours are treated uniformly. While their algorithm are capable of handling some very complex situations, for some other situations it either fails or is reluctant to converge. Interestingly, ICM is pretty good at the cases of boundary involved paths, but for cases of closed paths the iterations usually oscillate before converging. In some sense, the ICM is a complementary to Baraff et al.'s method.

Designing a purely history-free and orientation-free collision resolution algorithm is often an ambiguous and ill-conditioned problem. In many circumstances, however, surface orientation information are often implied in the context. For example, when simulating a dressed avatar, the body mesh defines an inside volume that the cloth geometry should not go into. In this case, the body mesh has a clear orientation. For simulating a complex set of layered garments on a virtual character, a natural requisite is that the outercoat layer does not get into the undercoat layer, thus these two meshes are actually oriented relatively to each other. Being completely blind to such information sometimes just makes our task unnecessarily complicated.

Given the already powerful ICM algorithm, we propose to introduce "repulsive normals" into the ICM, so that many of those once-failure configurations can be resolved successfully. Even for those once-successful configurations, repulsive normals usually speed-up the convergence. We demonstrate the efficiency of our modified ICM through several special examples, in which the direction of repulsive normals can be designated by user a prior. How to automatically and dynamically determine the proper repulsive normal directions is still a pending question (see discussions in § 6). Another contribution of this paper is the extension of the original ICM method, designed for polygonal mesh collisions, to resolve collisions between a polygon mesh and an analytical surface. We used the capsule as a test case and hope the formulation applies to other types of analytical surfaces as well.

2. Revisit of the ICM Method

2.1. Description of ICM

When using the Intersection Contour Minimization method for detecting collisions between triangle meshes, surface interferences are typically detected as intersections between edges and triangles. The triangle-triangle intersection contour is actually described as two lines, drawn identically on the two faces. On the polygonal meshes, this contour is indeed a polygonal line, and the edge-triangle intersections define the vertices supporting it. The core of ICM was to define a collision response scheme that induced a relative displacement between intersecting edges and triangles so as to reduce the length of the intersection contour, ultimately leading to the disappearance of the surface intersection.

Figure 1, borrowed from [VM06], shows a geometric configuration of two intersection polygons. A directional vector **G**, along which the edge **E** should be moved relatively to polygon **A**, is to be determined so as to get the best reduction of the intersection contour length. Since an edge is typically shared by two triangles (excluding the boundary edges), the finalized vector **G** combines contributions from them:

$$\mathbf{G} = \sum_{i=1}^{2} (\mathbf{R}_i - \frac{\mathbf{E}\cdot\mathbf{R}_i}{\mathbf{E}\cdot\mathbf{N}}\mathbf{N}). \quad (1)$$

Note this equation is slightly different from Equ 4 of [VM06]. They mistakenly assumed the total amount of

length reduction from two intersecting faces was $\mathbf{l}_i + 2\mathbf{k}_i$, instead of $2(\mathbf{l}_i + \mathbf{k}_i)$. Their assumption contradicts the fact that the intersection contours on two colliding surfaces coincide and are identical. Yet this small mistake has very limited negative effect on the result.

Vector $\mathbf{G}$ is a response vector that can be properly scaled to act as a repulsion force, or a direct velocity/position change to the cloth particles. Either each $\mathbf{G}$ is applied independently to the corresponding cloth nodes, or all the $\mathbf{G}$s belonging to the same contour are combined and averaged to a single vector, and then applied to cloth uniformly over all edge-polygon intersections involved in the contour.

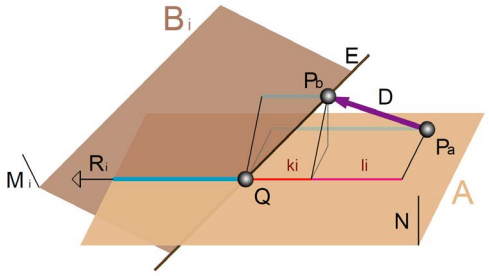

Figure 1: *A geometric configuration of two intersecting polygons $\mathbf{B}_i$ and $\mathbf{A}$, $\mathbf{M}_i$ and $\mathbf{N}$ being the plane normals. Edge $\mathbf{E}$ is penetrating polygon $\mathbf{A}$. The unit vector $\mathbf{R}_i$ is along the intersection segment between two polygons, pointing from edge $\mathbf{E}$ to the inside of $\mathbf{B}_i$. A small displacement $\mathbf{D}$ of $\mathbf{E}$ relatively to $\mathbf{A}$ moves the intersection point $\mathbf{P}_a$ to $\mathbf{P}_b$ and changes the length of the intersection segment. The magnitude of $\mathbf{l}_i + \mathbf{k}_i$ illustrates the length reduction. Image courtesy of [VM06].*

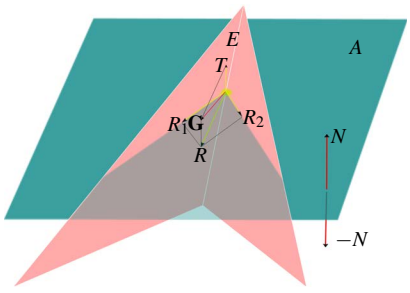

Figure 2: *A geometric configuration of two adjacent polygons intersecting the third one.*

2.2. Analysis of the original ICM

Although several directional vectors ($\mathbf{R}_i, \mathbf{E}$ and $\mathbf{N}$) participate in the calculation of $\mathbf{G}$, the ICM is in fact an orientation-free technique. On choosing the direction for the vectors in Equ 1, the only restriction is that $\mathbf{R}_1$ and $\mathbf{R}_2$ pointing inside their triangles, respectively. For $\mathbf{E}$ and $\mathbf{N}$, negating any one or both does not change the second term of the right-hand-side of Equ 1, nor $\mathbf{G}$. Orientation-free is really an appealing feature, as long as it does not cause ambiguity. However, the reality is often unlike what we expect. Missing orientation information causes problems to the ICM algorithm. Now let us take a close look at what happens.

Equ 1 can be re-written into

$$\mathbf{G} = \mathbf{R} + \mathbf{T}, \quad \textit{where} \quad \mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2, \quad \mathbf{T} = -\frac{\mathbf{E}\cdot\mathbf{R}}{\mathbf{E}\cdot\mathbf{N}}\mathbf{N}. \quad (2)$$

The first term, $\mathbf{R}$, is a vector within the plane of polygon A and points to the inside of the hinge formed by the triangle pair. (If two triangles are coplanar, $\mathbf{R}$ is null, and so is $\mathbf{G}$). It has the tendency to shift edge $\mathbf{E}$ towards the boundary of polygon $\mathbf{A}$. The second term $\mathbf{T}$ is a vector orthogonal to the plane, and it also points to the inside of the hinge. (If $\mathbf{E}$ is collinear with $\mathbf{N}$ then $\mathbf{T}$ vanishes). It has the tendency to move the edge above or below the plane. Whether $\mathbf{T}$ is pointing above or below the plane depends on the configuration of the hinge. As the case shown in Figure 2, $\mathbf{T}$ is pointing above. If we fix edge $\mathbf{E}$ as a rotation axis and rotate the two adjacent triangles away from the viewer so that $\mathbf{R}$ begins to point away into the screen, then $\mathbf{T}$ begins to point down.

If, as shown in Figure 2, polygon $\mathbf{A}$ happens to be a face on a closed surface with the up direction denoting the inside, moving $\mathbf{E}$ upwards means moving it into closed region. We illustrate the situation by an example in Figure 3. The green mesh is a cut-off from a teapot model with the outside faces are rendered green and inside faces gray. The golden mesh is a teacup intersecting the teapot. Both meshes are treated as solid models that do not deform. Each step of collision detection is followed by a step of position-based repulsion as collision resolution.

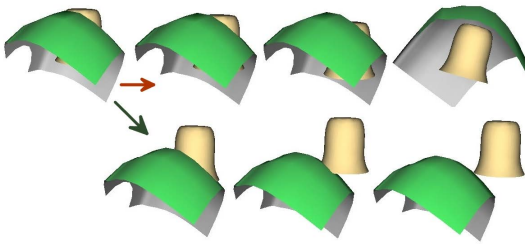

Figure 3: *An example of ambiguous collision resolution result by using the ICM. Top row: the original ICM method takes 53 oscillating steps of repulsions to separate the two objects. Bottom row: our modified ICM takes only 8 steps of repulsions.*

3. The ICM with Repulsive Normals

We make a simple improvement to the ICM so that it handles oriented surface collisions better. With our improved ICM,

the direction of vector **N** in Equ 2 is no longer arbitrarily chosen. Instead, **N**, now called "repulsive normal", always points to the outside of the surface. Since sometimes it is hard to tell the "inside" and "outside" for non-closed surfaces, an alternative way to understand this concept is that **N** always has a tendency to push back any "invading" objects to where they are from. Vector **G** should not contain any component along the direction of $-\mathbf{N}$, which otherwise will push the colliding geometries further in the wrong direction. To achieve this, the direction of **T** needs to be checked on the fly. If **T** is opposite to **N**, we force $\mathbf{T} = \mathbf{0}$. Then the new equation for computing **G** becomes

$$\mathbf{G} = \begin{cases} \sum \mathbf{R}_i & \text{if } \frac{\mathbf{E}\cdot\sum\mathbf{R}_i}{\mathbf{E}\cdot\mathbf{N}} > 0 \\ \sum \mathbf{R}_i - \frac{\mathbf{E}\cdot\sum\mathbf{R}_i}{\mathbf{E}\cdot\mathbf{N}}\mathbf{N} & \text{otherwise} \end{cases} \tag{3}$$

Based on this modification, the result of **G** matters if the direction of **N** is flipped. It is desired that the direction is determined on the fly, yet this is a non-trivial work. Since our goal is to show how the repulsive normal affects the intersection contour minimization process, all the test examples are set up by designating the direction of repulsive normal ahead of time. Figure 4 illustrates four possible configurations of repulsive normals for two intersecting surfaces. Among them only case (a) has the repulsive normals properly set up so that the two surfaces intend to repel each other to reach the collision-free state (e). In case (b) and (c), one surface intends to repel but the other intends to attract, thus the combined force/impulse direction is unpredictable, which is the main reason for oscillating. In case (d), both surfaces intends to attract each other, making the intersection get worse.

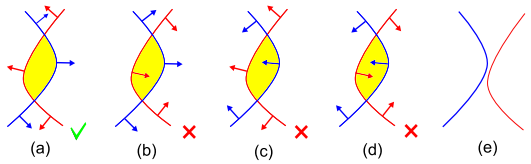

Figure 4: *The 2D analogue for four possible configurations of the repulsive normals for two intersecting surfaces, and the desired after-collision state (e). The yellow region denotes the volume enclosed by the penetration surfaces. (a): Both surfaces intend to repulse each other; (b): the blue surface intends to repulse but the red intends to attract; (c): the red surface intends to repulse but the blue intends to attract; (d): Both surfaces intend to attract.*

4. The ICM for Analytical-Polygonal Surface Collisions

The ICM was initially designed to resolve polygon-polygon collisions, yet we found it can be extended to handle collisions between a polygonal mesh and an analytical surface. As long as the intersection between a line segment and the curved surface can be detected and the surface normal at the intersecting point are easily available, the ICM works in a straightforward manner. We notice a fact that compared to polygonal meshes, generic algebraic curved surfaces are less frequently used in computer graphics, due to much difficulties in tessellation, parametrization and intersection tests, etc. Yet certain geometric shapes (such as spheres, ellipsoids and capsules) are widely used, often as bounding volumes to more complicated polygonal meshes for collision detection. Particularly, capsules are often the replacement for body segments of a virtual character in real-time systems. In this section, we use the capsule as one example of analytical surfaces and extend the ICM to handle mesh-capsule collisions.

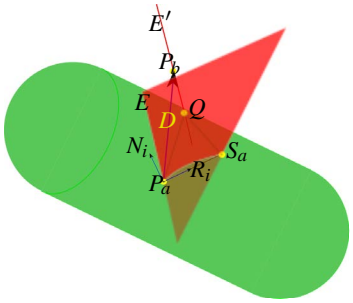

Figure 5: *Triangle-capsule intersection: $\widetilde{P_aS_a}$ is the intersection contour. Edge E is moved to E' by a displacement* **D** *and intersects the capsule at a new point Q.*

Figure 5 shows a configuration that two edges intersect a capsule, creating two intersection points P_a, S_a and form an intersection contour $\widetilde{P_aS_a}$. Since the contour is a curve on a surface, its corresponding chord is a line segment $\overline{P_aS_a}$ inside the capsule. Our objective is to reduce the length of the contour by displacing the intersecting edges. An equivalent problem is to reduce the length of the corresponding chord.

We then need to find the vector **G** at each intersection point, along which the edges are to be displaced so that the chord length is best reduced. To displace E, we temporarily fix the other intersecting edge and only E is allowed to move. We suppose edge E moves to E' by a small displacement **D** relatively to the capsule, and point P_a moves to P_b accordingly. Edge E' intersects the capsule at a new point Q, so the intersection contour becomes $\widetilde{QS_a}$, the corresponding chord being $\overline{QS_a}$. Since the displacement is assumed to be very small, the capsule normal at point P_a is approximately orthogonal to the vector $\overline{P_aQ}$:

$$\mathbf{N}_i \cdot \overline{P_aQ} = \mathbf{N}_i \cdot (\mathbf{D} + \overline{P_bQ}) = 0,$$

where $\mathbf{N}_i$ is the capsule normal at P_a. Thus there is $\mathbf{N}_i \cdot \overline{P_bQ} = -\mathbf{N}_i \cdot \mathbf{D}$. Recall that $\overline{P_bQ}$ is parallel to E, we have an equation similar to Equ 1 of [VM06]:

$$(\mathbf{N_i} \cdot \mathbf{E})\overline{P_bQ} = (\mathbf{N_i} \cdot \overline{P_bQ})\mathbf{E} = -(\mathbf{N_i} \cdot \mathbf{D})\mathbf{E},$$

which yields $\overline{P_bQ} = -\frac{\mathbf{N_i}\cdot\mathbf{D}}{\mathbf{N_i}\cdot\mathbf{E}}\mathbf{E}$. Similar to the polygon-polygon case, we define a unit vector $\mathbf{R}_i$ along the inter-

section chord $\overline{P_aS_a}$ and pointing to the inside of the triangle. Therefore the length reduction k_i of the capsule chord is computed as

$$k_i = \overline{P_aQ} \cdot \mathbf{R}_i = (\mathbf{D} - \frac{\mathbf{N_i} \cdot \mathbf{D}}{\mathbf{N_i} \cdot \mathbf{E}}\mathbf{E}) \cdot \mathbf{R}_i = (\mathbf{R}_i - \frac{\mathbf{E} \cdot \mathbf{R}_i}{\mathbf{E} \cdot \mathbf{N_i}}\mathbf{N_i}) \cdot \mathbf{D} \quad (4)$$

Since each edge is usually adjacent to two triangles, we can now sum up the contributions of length reduction from them, and express it with the relative displacement **D** and a vector **G**, as follows:

$$\sum_{i=1}^{2} k_i = \mathbf{G} \cdot \mathbf{D}, \quad where \quad \mathbf{G} = \sum \mathbf{R}_i - \frac{\mathbf{E} \cdot \sum \mathbf{R}_i}{\mathbf{E} \cdot \mathbf{N_i}}\mathbf{N_i} \quad (5)$$

Taking into account the repulsive normals, the final formula for computing **G** is of the same form as Equ 3.

A triangle-capsule intersection could create two, four or six intersection points. Please note that since the collision detection is based on edge-capsule intersection test, it fails when a triangle intersects a capsule but none of the intersection points is on the edges. Yet this case means either the triangle is too large relatively to the capsule or the triangle is merely superficially into one of the hemispheres of the capsule. As a capsule is typically used as a bounding volume, it not only much larger than any individual triangle faces, but also slightly larger than the whole mesh being enclosed. Then these two issues should not be a concern.

5. Implementation and Results

Our modified ICM method has been integrated in a cloth simulator based on particle systems as described by Choi and Ko [CK02]. Numerical integration is performed using the Implicit Euler, as described by Baraff and Witkin [BW98] and Hauth et al. [HES03]. Broad-phase collision detection is performed using Bounding Volume Hierarchies with k-DOP from [TCYM09]. Surface intersections are detected through edge-polygon intersections. In contrast with Volino's implementation [VM06], we have implemented a very robust intersection contour reconstruction algorithm, particularly for the capsule and polygonal meshes intersections. Collision response is performed by enforcing the collision constraints through geometric corrections of position distributed on mesh vertices. We illustrate the effectiveness of our method through a set of testing examples. For all surfaces of these examples the direction of repulsive normals have been predefined.

In our test examples Figure 6 to 8, each side of the cloth is rendered either in color (red, green or gold) or in gray. The colors are set in a way that the user-defined repulsive normals points from the gray side to the colored side. Figure 6 shows configurations that can not be solved by the original ICM method due to the ambiguity of the correct non-intersecting states, yet they are solved with our modified ICM with the help of the orientation. For the first configuration, the top-hanging cloth was pushed both upward and

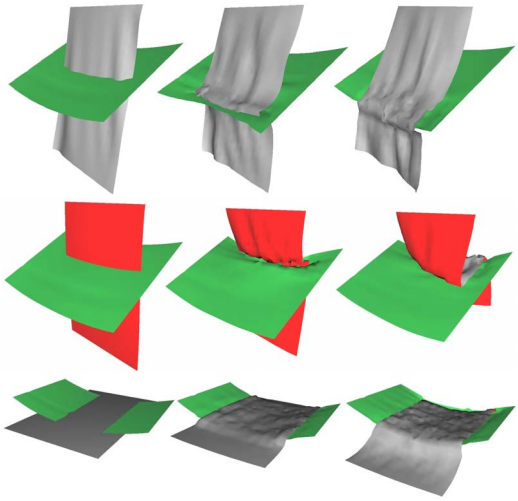

Figure 6: *Collision configurations that can be solved by our modified ICM method but not the original ICM method. The first two rows show a different repulsive normal set-up for the same scene, thus the after-collision states are different. Both cloth meshes have* 20×30 *particles.*

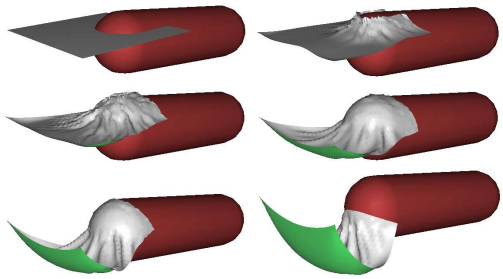

Figure 7: *Resolving a mesh-capsule collision using our method. The cloth mesh has* 41×41 *particles with two corners being pinned.*

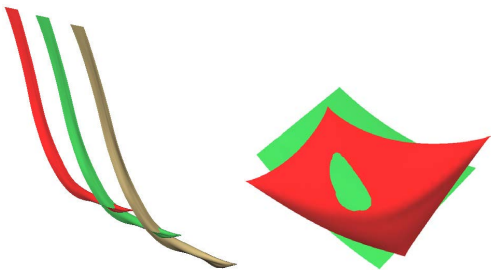

Figure 8: *Left: Collisions among three ribbons. Repulsive normal directions should be specified dynamically for different colliding regions. Right: The penetrating regions from two meshes form a closed surface. The repulsive normals defined for the vicinity of this closed surface should point to the outside.*

aside, yet it reached the boundary of the green sheet before it is lifted above. Figure 7 shows the process of resolving the collision between a cloth mesh and a capsule using our modified ICM method.

6. Discussions and Conclusion

We have proposed a modified ICM method for resolving deformable surface intersections. In the above examples, we have demonstrated the power of the modified ICM method. The repulsive normal is the major contributor for the improvement, yet the necessity of predefining its direction also imposes a restriction.

When setting up the repulsive normal direction, the user is only possible to specify all face normals to be uniformly pointing to one side of a mesh or the other. If a mesh has multiple colliding regions (with other meshes or with itself), the repulsive normal for each region should ideally be set independently, or on a per object-pair basis. For a situation as shown in Figure 8, three ribbons hang on the ceiling and the green ribbon intersects with both the red and the gold one, resulting in two colliding regions in it. In order for our modified ICM to function properly, the two regions in the green ribbon are desired to have their repulsive normals pointing to opposite sides of the mesh. Since it is difficult for a user to foresee the behaviors of the moving surfaces, properly setting repulsive normal is out of the question.

To overcome the fore-mentioned limitation, an ideal solution is to adopt the "dynamic repulsive normals" and try to determine the normal direction for each region on-the-run. However, this is also a non-trivial work, so heuristics must be used. Similarly in Baraff et al.'s work [BWK03], an assumption was made in order to decide which side of an intersection path is the interior: smaller regions are considered the interpenetrated ones. This assumption is usually safe due to the fact that intersections that arise during simulation are generally tiny compared to the size of the meshes. This assumption is also helpful for us to determine the dynamic repulsive normals in certain circumstances. For the collision configuration as shown in Figure 8(b), the colliding regions from the two meshes can be firstly identified – according to Baraff's flood-fill suggestion – and these two patches form a watertight volume. Then the dynamic repulsive normal should be defined to point out of the closed surface. Figure 4(a) consists with this proposition. In addition to the closed contours, in some cases where an open contour partitions a mesh into two regions (e.g. in Figure 8(a)), dynamic repulsive normals should also be determined based on the paradigm "small is illegal". However, sometimes an intersection contour does not partition a mesh thus no smaller region is identified. On the other hand these cases usually can be resolved by the original ICM method without any modification. As the surface-surface intersection can be of various status, the technique of ICM with dynamic repulsive normals definitely needs more investigation in the future.

We have used the capsule to show the power of the ICM for tangling a polygonal mesh out of an analytical mesh. An interesting question is: since projecting an inside vertex to the surface of the analytical object is straightforward, is the ICM more efficient? We believe the answer is "yes". The major cost of the two methods is actually in the stage of edge-surface intersection computation. Once the intersection point is found, the vector $\mathbf{G}$ is almost handy for the ICM, no need to do extra work of projection. Moreover, the projection method usually moves a vertex via the shortest out-of-surface path, yet this may not be a desired path. For the case in Figure 5, the projection method may move the inside vertex unexpectedly downward, away from its opposite edge.

7. Acknowledgements

This work was supported by NSF of China under the grants #61070105, #90820303, #60903145 and NSF of Beijing under the grant #4102062.

References

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. *ACM Transactions on Graphics 21*, 3 (2002), 594–603. 1

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of SIGGRAPH* (1998), pp. 43–54. 1, 5

[BWK03] BARAFF D., WITKIN A., KASS M.: Untangling cloth. *ACM Transactions on Graphics 22*, 3 (2003), 862–870. 2, 6

[CK02] CHOI K.-J., KO H.-S.: Stable but responsive cloth. *ACM Transactions on Graphics 21*, 3 (2002), 604–611. 1, 5

[HES03] HAUTH M., ETZMUSS O., STRASSER W.: Analysis of numerical methods for the simulation of deformable models. *The Visual Computer 19*, 7–8 (2003), 581–600. 5

[HPSZ11] HARMON D., PANOZZO D., SORKINE O., ZORIN D.: Interference aware geometric modeling. *ACM Transactions on Graphics 30*, 6 (2011), 137:1–137:10. 1

[Pro95] PROVOT X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface* (1995), pp. 147–154. 1

[Pro97] PROVOT X.: Collision and self-collision handling in cloth model dedicated to design garments. In *Eurographics Workshop on Computer Animation and Simulation* (1997), pp. 177–189. 1

[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* (2009), 544–557. 5

[VM06] VOLINO P., MAGNENAT-THALMANN N.: Resolving surface collisions through intersection contour minimization. *ACM Trans. Graph. 25*, 3 (2006), 1154–1159. 1, 2, 3, 4, 5

[VMF09] VOLINO P., MAGNENAT-THALMANN N., FAURE F.: A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28*, 4 (2009), 1–16. 1

[WLG06] WICKE M., LANKER H., GROSS M.: Untangling cloth with boundaries. In *Proceedings of Vision, Modeling, and Visualization (VMV)* (2006), pp. 349–356. 2