4th Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007)
J. Dingliana, F. Ganovelli (Editors)

Virtual adhesive: a way to handle sticky collisions in surgical and biological simulators

Alexandre Carra, Jean-Louis Martiel and Emmanuel Promayon

Laboratoire TIMC-IMAG CNRS UMR 5525, Université J. Fourier,
Institut d'Ingénierie de l'Information de Santé,
Faculté de Médecine 38706 La Tronche Cedex

Abstract

A variety of methods have been proposed to efficiently process collisions between deformable objects. The method presented in this paper allows to model sticky states between deformable objects with triangulated surfaces. In contrast to an often used approach that consists of generating forces which eventually separate colliding objects, our method is based on the creation of an adhesive virtual object (virtual adhesive). This virtual adhesive is composed by "clones" of all particles locally in collision. Particle clones are used to gather forces on the virtual adhesive, which behaves like a rigid body. The resulting displacement of the virtual adhesive is used in turn to constrain the particles displacement: particles stick to their clone. As a result, no further interpenetration is possible and a sticky state is obtained in the considered zone. Moreover, the method correctly resolves contacts without introducing additional energy in the system. Results are presented through several simulations.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling I.3.7 [Three-Dimensional Graphics and Realism]: Animation

Keywords: Physically-based model, sticky contact modeling, collision response, collision detection, deformable objects

1. Introduction

Collision detection and response between deformable bodies have been studied quite extensively [TKH*05]. They are crucial elements of surgical simulators, especially for modeling the interactions between organs and surgical tools, such as lancets and needles. Biological and cellular simulators, also called virtual cell modeling and simulation, are another type of emerging tools in which the collision detection and response is required to model cell deformations and migrations [PMT03]. Although the objectives of surgical simulators are very well understood today, their biological counterpart is of a different nature.

Cell movements and deformations are essential for many important biological and medical processes, including cell division and migration, the formation of viable embryos or the growth of cancerous tumours. A long-standing challenge in modeling complex biological systems is to combine mechanisms of different physical origins (e.g. cell elasticity, cytoskeleton dynamics, chemical reactions) at different times (from the second to the minute) and length scales (from micrometers to tens of millimetres). If we adopt a reductionist approach, the force generation in cells results from the dynamic assembly of individual molecules or filaments (actin, tubulin). These physico-chemical processes are now well understood [PB03]. For example, actin filaments produce piconewton forces, as shown in experiments [KP04] or in physical modeling [BMB*07]. Although these models are necessary to decipher the resulting behavior of complex macromolecular assemblies, they are by far too complex to be useful to account for the dynamics of a cell or a cell population. Therefore, the next step in simplification of approach would use phenomenological models, based on a global description of phenomena, without relying on precise and detailed mechanisms in cells. For example, interactions between a cell rolling on a surface by creating/destroying bonds between the membrane and proteins on the surface can be handled by kinetic models [DTSH88]. These models are based on a phenomenological description of cells rolling over rigid surfaces or over each other and do no necessitate complex mathematical/computer solutions. Depending on the question addressed and on the required degree of accuracy of the representation of experimental data, we can use different approaches, from the simulation of partial differential equations (e.g. to get the elastic response of cells or tissues; to represent the different levels of molecules in cells) to the discrete iterations of an algorithm that finds the problem solution by minimizing a global energy function. However, we can go a step further in the simplification in the mathematical/computer-based modeling. The question "How do cells move on solid substratum or interact with other cells?" can also be analyzed via very simple computer models. In addition, these approaches give valuable insights into the global cellular mechanism that permits cells to deform, slide roll or stall, when they interact with other cells or rigid objects.

In both surgical and biological simulators, contact must be handled efficiently. However a category of contact that are very rarely considered in simulators, due to the complexity of the phenomenon and because there exists no simple theoretical model in physics today is sticky contacts. Deformable organs, tissues or cells are generally surrounded by a membrane or capsule that have adhesive properties. These adhesive properties considerably modify the contact interactions. In such simulators, contact generally occurs for quite a long time and forces between the colliding bodies have to be exchanged. Furthermore, to model adhesive properties a kind of cohesion between the colliding bodies also have to be expressed, which has to be maintained as long as the relative deformation and movement of the bodies and their intrinsic membrane adhesive properties enable it.

This paper presents an alternative solution to modeling sticking deformable objects. Based on a phenomenological approach, it consists of creating a virtual adhesive body in the zone where the bodies are colliding, in order to model the sticky contact. In our approach, a sticking zone is a zone where deformable objects are temporarily glued together i.e. stay in contact whitout intersecting while exchanging forces. Our goal is not to be physically or mechanically correct but instead to model sliding and sticking surfaces so that visual realism is maintained as well as good force transfer within a reasonable computation time.

2. Related work

The handling of contact for rigid and deformable bodies has been widely studied in computer graphics. In this work, we focus on collision between deformable objects. For an extended discussion, the readers can refer to the detailed paper on collision detection of deformable objects [TKH*05].

Collision response schemes for deformable objects can be classified principally into two different categories: the penalty-based method and the constraints method. The penalty-based method is probably the most widely used because of its simplicity to implement. In this approach, a force is exerted on a colliding point whose magnitude is proportional to its penetration depth. Terzopoulos et al. has been one of the first to use it in [TPBF87]. Later, Cani uses isopotential implicit surfaces generated by skeletons to model deformable solids [Can93]. With this implicit formalism, exact contact surfaces are generated between objects. In [ST05], Spillmann and Teschner also focus on the contact surface. They compute an explicit representation of the contact surface between flexible objects and triangulated surfaces. This contact surface exclusively depends on the internal forces; no user-defined parameters are needed. After collision response, the colliding vertices are displaced onto the contact surfaces and a collision-free configuration is obtained. With this method, the authors address discontinuity problems due to discrete time steps and coarse surface representations.
Keiser et al. [KMH*04] propose a combination of contact surface and penalty-based methods. They compute a virtual contact surface for colliding deformable objects and determine the position of the colliding points on the contact surface by applying penalty forces. In [HTK*04], Heidelberger et al. present a method to calculate consistent penetration depth. They consider a set of close surface features to reduce discontinuities of the estimated penetration depth directions for small displacements of the penetrating vertices. The penetration depth and direction are only computed for the colliding points that are close to the penetration surface, and then propagated to the colliding points with larger penetrations. In [DMG05], the authors present a multi-resolution model, based on a deformable octree structure that can handle large displacements. In this case, collision responses are computed using penalty forces.
Recently, Barbic et al. [BJ07] have presented a model that can support haptic rendering of distributed contact between a rigid object and a reduced FEM deformable object (both with non-trivial geometry) by using a rigid distance field and a deformable pointshell. Penalty-based approach is employed to handle contacts: the penalty forces are determined by querying the points of the pointshell object against the signed distance field of the other object. The approach for time-critical evaluation of contact forces is based on several optimizations such as temporal coherence and graceful degradation.

Constraints can also be used to handle collision. The key idea is to define a system of constraints to prevent objects from penetration. This system is formulated as a Linear Complementary Problem (LCP) whose solution gives the contact forces. The contact forces are then computed by solving the system. Duriez et al. [DAK04, DDKA06] use Signorini's model to deal with contact between deformable objects. Then, after solving a LCP, the forces are integrated to find the constrained motion. Based on an explicit finite element method, the approach developed in [DDCB01] locally adapts the sampling at which the simulation is computed to the current local deformation. When a collision is detected, the approach uses non-penetration constraints to move the collided surface points. Moreover, the active mesh nodes that are linked to a concerned surface point are also displaced. In [PPG04], Pauly et al. combine the benefits of rigid body models for dynamic simulation and the advantages of deformable models for solving contacts and producing visible deformations. They set up a LCP that define the contact surface between two objects and solves the tractions that act on this surface. Then they compute the corresponding displacements and the total wrench that acts on the quasi-rigid bodies. In addition, volume preservation is ensured for any pressure distribution acting on the surface. Recently, in [SBT07], the authors combine the constraints method with the penalty approach. They compute the positions of the colliding points and the contact surface by considering the stress on each point. A contact force is then calculated that accelerates the point immediately to its collision-free position. Here, the system of equations is decoupled and can be solved analytically. Moreover, the approach does not require the definition of response constants.

While all these approaches efficiently handle collision, they do not take into account sticky contact between deformable objects. One classic solution to this problem are approaches based on the coulomb friction law. Mirtich uses such a model to handle sticking and sliding contact between rigid bodies [MC94, Mir96] but its solution is computationally expensive. In [CBP05], Clavet et al. present an impulse-based method to make particles stick. They supplement the impulse formulation with an attraction term that accounts for particles that are close to an object. Another solution consists of adding elastic links, e.g. springs between colliding nodes, and removing them when the surfaces move sufficiently appart, as was done in [PMT03]. However these three approaches introduce additional energy into the system.

3. Description of the model

This section first presents a general overview of the model used in this paper. The description of a rigid object is then detailed as it will be needed to model the virtual adhesive bodies. Finally, we explain the principle that allows to link a rigid and an elastic object together. Due to this constraint, an appropriate response is found when such two bodies are in contact.

3.1. General presentation

Our discrete model can handle objects which can be either elastic, rigid, or muscular [MPT06, PMT03]. Each object is composed of a set of particles, which are defined by a position, a mass and a list of its neighbouring particles.

Neighbouring particles are initially defined by using a topological criterion that is invariant during the simulation. Particles are spread to volumetrically describe the object. The external surface of the objects are defined by a triangular mesh based on the surfacic particles which are the most external ones. Like in [BJ07], all the objects used in this paper are only described by their surfacic particles and mesh. Moreover the algorithms presented here work as well when particles are distributed in a volume. In the model, forces are applied on each particle. Two kinds of forces are considered here: force fields (e.g. gravitation force), and internal forces (e.g. elasticity).

The model also takes into account constraints associated to more complex properties such as the incompressibility of an object or the attachment between different bodies. Two kinds of constraints are implemented: first, local constraints can be applied to isolated particles. For instance, the position of a specific particle. Second, global constraints are used to implement incompressibility. For more details on the modeling of the elasticity and the method used to preserve the volume the readers can refer to [MPT06].

The dynamics of the object is classically determined by the integration of the equations of motion. At each time step, boundary conditions are applied on the particles (e.g. null displacements). Next, the forces applied on each particle are computed and summed and the internal forces are redistributed to satisfy the action-reaction principle. The position of each particle is then determined by a standard explicit integration scheme. Finally, the local and global constraints are taken into account by adjusting these positions. The algorithm is detailed below for one iteration (Alg. 1).

Algorithm 1: general algorithm of the model.

```
foreach object of the model do
    foreach particle of the object do
        applyBoundaryConditions()
    end
end
foreach object of the model do
    foreach particle of the object do
        computeForces()
        redistributeInternalForces()
    end
end
foreach object of the model do
    foreach particle of the object do
        sumForces()
        computePosition()
        applyConstraints()
    end
end
```

3.2. Rigid object modeling

In this paragraph, the specific properties of a non deformable body are detailed. By definition, such an object keeps its shape during the simulation which means that the distances between particles do not change.

Let Ω be a rigid object composed by n particles. A local frame is attached to Ω, with $\mathbf{R}$ a 3x3 matrix describing its rotation and $\mathbf{T}$ a vector describing its translation. The origin of the local frame is given by the center of mass $\mathbf{G}$ of Ω. The world position $\mathbf{P_i}$ of particle i in the object can be expressed in terms of its local frame position $\mathbf{P_i^0}$ as $\mathbf{P_i} = \mathbf{R}\mathbf{P_i^0} + \mathbf{T}$. Furthermore, the rigid object has a linear velocity $\mathbf{V}$ and an angular velocity $\mathbf{W}$. The linear acceleration of the rigid object, $\dot{\mathbf{V}}$, is computed as follows:

$$\dot{\mathbf{V}} = \sum_{i=1}^{n} \mathbf{F_i} / \sum_{i=1}^{n} m_i \tag{1}$$

where $\mathbf{F_i}$ is the total of forces applied on the particle i and m_i its mass. Also, the angular acceleration of the rigid object, $\dot{\mathbf{W}}$, is:

$$\dot{\mathbf{W}} = I^{-1}(\tau + (I\mathbf{W}) \times \mathbf{W}) \tag{2}$$

where I is the inertia matrix associated with the global frame and τ the total torque applied on Ω. τ is computed from the following expression:

$$\tau = \sum_{i=1}^{n} \mathbf{GP_i} \times \mathbf{F_i} \tag{3}$$

From $\dot{\mathbf{V}}$ and $\dot{\mathbf{W}}$, the new linear and angular velocities of Ω are computed by an Euler integration scheme in order to update the new position and orientation of Ω.

3.3. Linking rigid and elastic objects

A specific algorithm using direct constraint projection has been developed to simulate elastic or contractile objects fixed to a rigid object, such as a muscle fixed on a bone.

The method constrains the particles located at the body surfaces to keep the same position during the simulation. More precisely, the particles of the interface are **cloned** and the movements of the elastic particles are constrained by those of the rigid particles.
Let Ω_e (resp. Ω_r) be an elastic (resp. a rigid) body. To handle the interaction between Ω_e and Ω_r, we associate to each particle $\mathbf{P_e} \in \Omega_e$ (resp. $\mathbf{P_r} \in \Omega_r$) a clone. During simulation, the original particles and their clone are kept at the same position.
Our collision algorithm will preserve the relative position of $\mathbf{P_e}$ and $\mathbf{P_r}$ so that the attachment between Ω_e and Ω_r are locally and transiently preserved. The different steps of the algorithm are the following:

1. compute the forces $\mathbf{F_e}$ applied on $\mathbf{P_e}$ (the sum of external and internal elastic forces).
2. Calculate the forces $\mathbf{F_r}$ applied on $\mathbf{P_r}$ (the sum of external forces) and add the forces transmitted by $\mathbf{P_e}$ in $\mathbf{F_i}$: $\mathbf{F_i} = \mathbf{F_r} + \mathbf{F_e}$.
3. Using (1), (2) and (3) given in section 3.2, compute the new orientation and position of Ω_r. Update the position of $\mathbf{P_r}$.
4. Constrain $\mathbf{P_e}$ to keep the same position than $\mathbf{P_r}$ (i.e. displacement of $\mathbf{P_e}$ is forced to be the displacement of $\mathbf{P_r}$).

The algorithm of this method is detailed below for the general case (Alg. 2).

Algorithm 2: linking elastic and rigid objects.

```
foreach particle of the elastic object do
    computeForces()
    if (cloned) then
        transmitForces()
    end
end
foreach particle of the rigid object do
    computeForces()
end
computeRigidObjectMotion()
foreach particle of the elastic object do
    if (cloned) then
        constrainPosition()
    else
        computeMotion()
    end
end
```

4. Sticking and adherence

The proposed algorithm to model a sticky state is given in this section. The first paragraph is devoted to the collision detection between deformable objects. The method to provide a sticky state is then detailed in a second paragraph.

4.1. Collision detection

We assume that the object surfaces are represented by triangular meshes. The algorithm proceeds in two stages. First, a coarse detection selects n_c pairs of objects potentially in contact. Spheres are classically used as a rough approximation of the objects shape. Second, the particles of these objects involved in the contact are identified. These particles are labelled as **Colliding Particles** (CPs). The set of actual CPs is determined using a refinement of classical detection algorithm (proximity and interference detection are performed).

4.2. Collision response

Our method is based on the creation of a virtual adhesive object in the zone where the bodies are in contact. We consider the adherence as a quasi static state in which a dynamic set of CPs are temporarily glued together and act as a rigid body linked to the elastic body in contact (see paragraph 3.2). This guarantees no further interpenetration in the considered zone (due to the projection algorithm explained in the paragraph 3.3). However, unlike [Pro97] or [BFA02] where zones of impact provide non-sliding contacts, our method allows several types of contacts such as rolling, sliding and sticking. Moreover, the link between the objects allows an exchange of forces to take place. Indeed, the elastic forces generated by the deformable objects are transmitted to the virtual adhesive. In return, the resultant of these forces and torques are applied on the CPs.

This section is organized as follows: the first paragraph introduces the concept of zone of contact and presents the method to simulate a sticky state. Then, the dynamics of the zones of contact is explained. Finally, the algorithm of the method is detailed.

4.2.1. Zone of contact, clones and virtual adhesive

For each pair of colliding objects, there is one or several zones of contact. A zone of contact is defined by a set of CPs that was detected (Fig. 1a). For each CP of the zone of contact a clone is created. The clone particles are then grouped together. The resulting group, named virtual adhesive (Fig. 1b), is considered as a rigid object which motion provides the collision response.

Clone Each CP belonging to the considered zone of contact is **cloned**. The clone particle has the same position, mass and velocity as the CP and the two particles are linked as defined in paragraph 3.3. Thus, a CP is constrained to be at the same position as its virtual clone whatever the forces applied on it (Fig. 1b).

Virtual adhesive For each zone of contact a virtual adhesive, composed by virtual clones, is created which allows the simulation of a sticky state between colliding objects in this particular zone (Fig. 1b). In order to freeze the distances between the clone particles, regardless of their origin, the virtual adhesive is modeled as a rigid object. Therefore the contact between objects are maintened without penetration. The virtual adhesive motion depends only on the forces acting on both side of the collision.

Rigid motion During the simulation, all forces applied to the CPs are directly transmitted to their clone. These forces are used to update the virtual adhesive force and torque, which in turn are involved in the rigid object motion (paragraph 3.2 and Fig. 1c, 1d and 1e). The colliding objects are temporarily frozen in the considered zone of contact and no further interpenetration is possible (Fig. 1f).

4.2.2. Virtual adhesive population dynamics

As stated before, the interaction between two deformable objects can change the zone of contact and, therefore, a change in the CP population. For example, two colliding objects can part away from one another, decreasing the number of CPs. Alternatively, if the objects agglutinate more and more, the number of CPs will increase. This section describes the method that allows these changes.

When a CP is identified, it has to be assigned to only one virtual adhesive. Three cases can occur. First case, the CP is isolated and a new virtual adhesive has to be created, with the CP being its first element. Second case, the CP has at least one neighbouring particle already in one virtual adhesive. The CP is thus added to this closest virtual adhesive. Third case, there are two or more virtual adhesives close to the CP, i.e. the CP has more than one neighbouring particle in more than one distinct virtual adhesive. All these virtual adhesives are considered to have grown until they became adjacent; they must be merged. In this case, the CP is added to this new larger virtual adhesive.

At each time step, each CP computes the theoretical displacement, $\mathbf{U_e}$, that would have result from elastic internal forces if it were free from any virtual adhesive. This displacement is then compared to the imposed displacement, $\mathbf{U_{VA}}$, resulting from the virtual adhesive motion. If $\mathbf{U_e}$ and $\mathbf{U_{VA}}$ are in opposite directions, the norm of their difference is compared to a given parameter κ_{adh}. When this norm is higher than κ_{adh}, the clone is destroyed. As a result, the CP becomes free to move again. The value of κ_{adh} controls the degree of adherence. If κ_{adh} is small the adherence is small: only a small difference between $\mathbf{U_e}$ and $\mathbf{U_{VA}}$ is required to free the particle from the virtual adhesive.

When a virtual adhesive is composed of less than two CPs, it is no more needed and it has to be destroyed (there are no more collisions). Any virtual adhesive can recruit or loose particles according to the relative motions of the different objects in contact. The value of κ_{adh} controls the virtual adhesive population dynamics allowing rolling, sliding or sticking.

4.2.3. General algorithm

The method proceeds in five consecutive stages (Alg. 3 and Fig. 1):

Stage 1 identifies a set of CPs from each pair of selected colliding objects (paragraph 4.1), defining the zone of contact (Fig. 1a).
Stage 2 clones each CP. From these clone particles, a virtual adhesive is created for each zone of contact (paragraph 4.2.1 and Fig. 1b).
Stage 3 computes the internal forces applied on each particle of the objects. These forces are summed (including elastic forces) and transmitted from the CPs to their clone (Fig. 1c). The position and orientation of the virtual adhesive are then computed by using these transmitted forces as input forces for the rigid body motion algorithm (paragraph 3.2 and Fig. 1d and 1e).

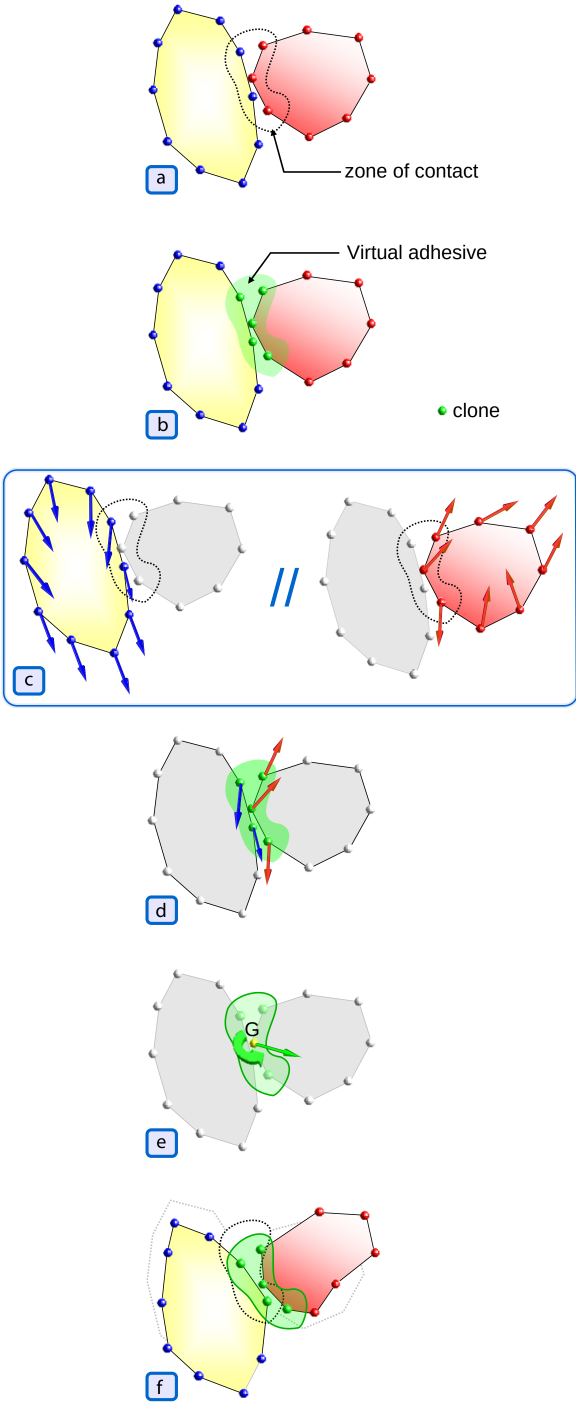

Figure 1: *Simulation of two colliding deformable objects: (a) zone of contact defined by the list of five Colliding Particles (CPs); (b) virtual adhesive corresponding to the contact zone, its virtual adhesive particles in green are clones of the five CPs; (c) separated elastic internal forces computation; (d) elastic forces transmitted from the CPs to their clones are used to compute the rigid body motion of the virtual adhesive; (e) resulting torque and force applied to the virtual adhesive; (f) movement of the deformable objects and the virtual adhesive due to the forces. The CPs are constrained to be at the same position than their clone.*

Stage 4 updates the population of the zones of contact and the corresponding virtual adhesive (paragraph 4.2.2).

Stage 5 computes the motion of the deformable objects taking into account the constrained displacement of the CPs (paragraph 3.3 and Fig. 1f).

Algorithm 3: modeling sticky contact.

```
detectCollision()
createVirtualAdhesive()
foreach object of the model do
    foreach particle of the object do
        computeForces()
        if (cloned) then
            transmitForces()
        end
    end
end
foreach virtual adhesive do
    computeMotion()
    updatePopulation()
end
foreach object of the model do
    foreach particle of the object do
        if (cloned) then
            constrainPosition()
        else
            computeMotion()
        end
    end
end
```

4.2.4. Energy balance

In marked contrast to approaches that consist of adding energy to handle collisions [Mir96, PMT03, CBP05], our method uses the existing forces to resolve contacts. Indeed, the elastic internal forces applied to the CPs are transmitted to their clone. These forces are then used by the virtual adhesive to compute its new position and orientation by using equations (1), (2) and (3) (rigid body motion). The collision response for all CPs are then given by the new position of their clone. As a result, the contacts are resolved without introducing additional energy in the system that is to say simply using the forces in presence.

5. Simulations

All simulations described in this section are performed on an Intel Xeon 5140 2.33 GHz and compiled with gcc v4.1 whithout any particular effort to optimize the code. The method is implemented in C++ and the different models are manipulated in PML [CP04]. All the elastic objects are modeled by particles on their external surfaces, organized in triangular meshes, and are satisfying the volume preservation constraint.

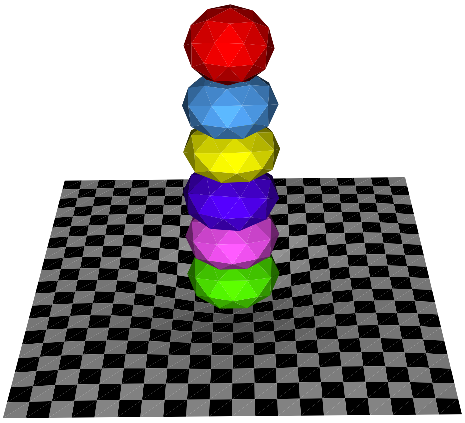

Figure 2: *Incompressible deformable spheres falling onto an elastic plane due to the gravity.*

In a first experiment, six deformable spheres fall onto an elastic plane due to gravity. The bodies deform on impact and end up embedded in the plane. The scene consists of 693 particles and 1280 triangles and the average time for collision detection and response is 549ms. As shown in Fig. 2, our approach provides an interpenetration-free situation between the objects. There are no interpenetrations between the spheres and the bottom sphere does not intersect with the plane. Note that the differences between the shape of the objects are due to elasticity and volume preservation.

An experiment that compares the collision between two types of sphere is shown in Fig. 3. At the top, both spheres are very adhesive while the spheres on the bottom are less sticky. For 84 particles and 160 triangles the average time for collision detection and response is bellow 10ms for the adhesive spheres and bellow 2ms for the others. At the time of impact, the spheres are colliding and slide on one another. Fig. 3 shows the time series of the difference of behavior between the two types of spheres. Note that in the last screenshot the very adhesive objects are still in contact (top right), while the less adhesive bodies are already in a collision-free situation (bottom right). The number of CPs is given for each screenshot. The experiment shows that the method allows the modeling of different sticky states, depending on κ_{adh}.

A deformable ball rolling onto an elastic plane is simulated in Fig. 4 . The scene consists of 217 particles and 368 triangles. The average time for collision detection and response is 33ms. Fig. 4 shows five frames of the dynamic sequence. Note that the deformation of the plane is due to the sticking contact with the ball and the imperfect rolling motion is due to adhesion between the ball and the plane.

In yet another experiment, our approach is tested with more complex objects. The result of the collision between

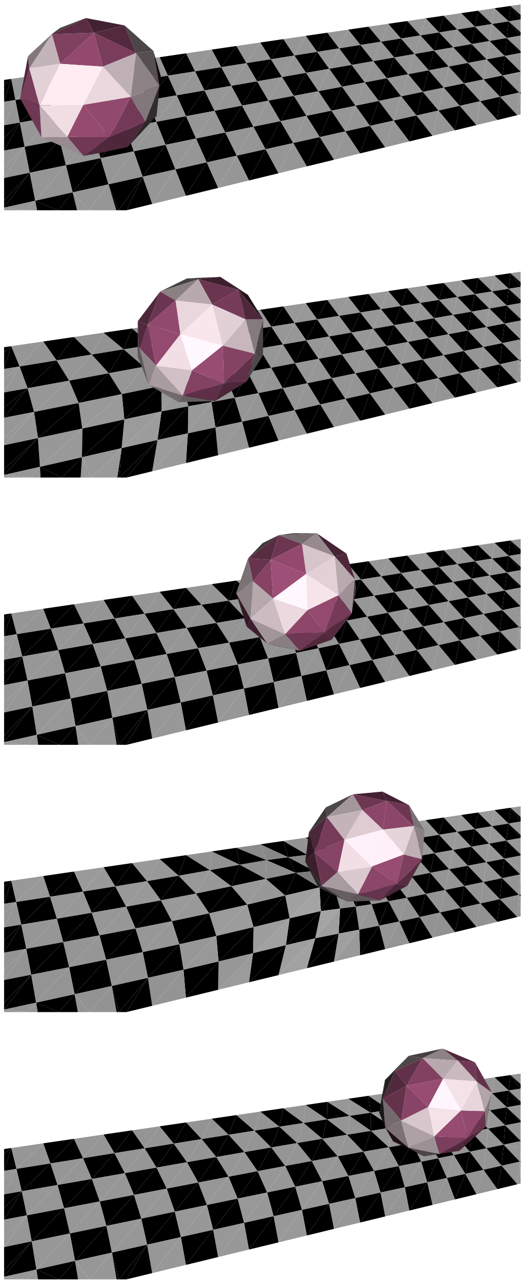

Figure 4: *An elastic ball rolling onto an elastic plane, from top to bottom: t=0s, t=40s, t=80s, t=120s and t=160s. Note the deformation of the plane due to the sticky contact with the ball.*

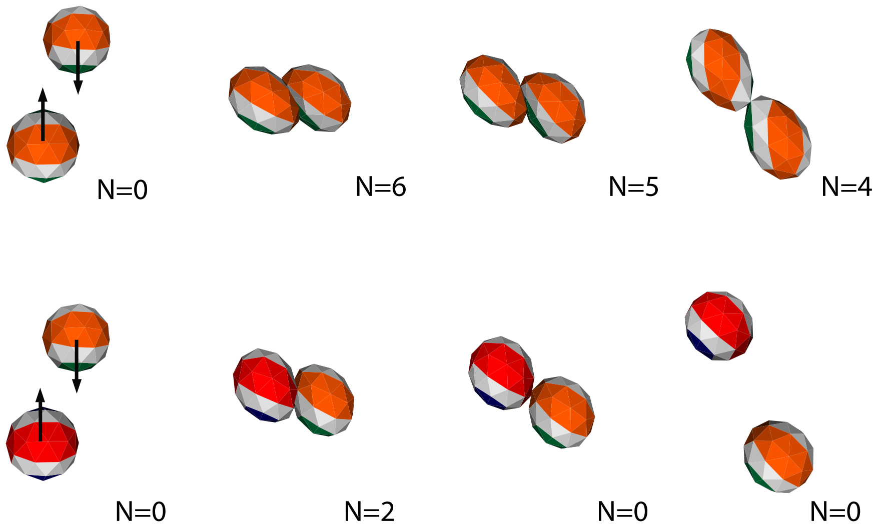

Figure 3: *Collision between two very adhesive objects (top, high κ_{adh}) and two less sticky objects (bottom, low κ_{adh}). On each series, the object of the top is attracted downward, while the object of the bottom is attracted updward, resulting in a collision during the simulation. The figure illustrates the difference of configuration between two values of κ_{adh} at four steps of the simulation. For each screenshot, the number of CPs (N) is specified.*

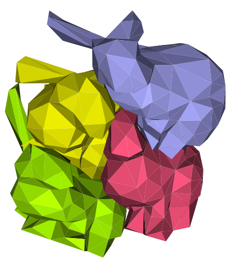

Figure 5: *Collision between four "sticky" bunnies.*

four bunnies is illustrated in Fig. 5. The method still efficiently and correctly resolves all sticky contacts. This test demonstrates that our method can be applied to a wide range of objects and is not limited to spherical bodies. The scene has 567 particles and 1040 triangles, the average time for collision detection and response is 576ms.

In a last test, an experiment is conducted with a complex scene mimicking cell agregation. An illustration of the aggregate behavior of a group of cells is given in Fig. 6. Twenty five incompressible deformable spheres are attracted by the center of the scene, mimicking the source of an attractive chemical signal, little by little colliding and regrouping into a single aggregate. The scene consists of 1050 particles and 2000 triangles. The average time for collision detection and response is 777ms. The percentage of CPs relative to the total number of particles is shown for each step in the bottom slider. The last screenshot in transparency demonstrates that the method still efficiently and correctly resolves contacts: there is no empty space inside the aggregate.

6. Conclusion and future work

We present a specific method for the collision, detection and response between deformable objects in the adhesive situation. The method simulates several types of contacts including rolling, sliding and sticking. The key idea is to detect the zones of contact between colliding object and to create virtual bodies in these zones. Each virtual adhesive is composed by particles locally in contact, using virtual clones.

We illustrated our approach with several experiments showing that the method efficiently and correctly resolves contacts. This approach can be applied to objects with a wide range of shapes, although our collision detection stage is only adapted for spherical shape interaction. Finally, the method allows the modeling of several sticky states, which make it suitable for biological applications as well as surgical simulators.

Currently, we are working on the acceleration of the collision detection step based on an adaptative hierarchical ap-

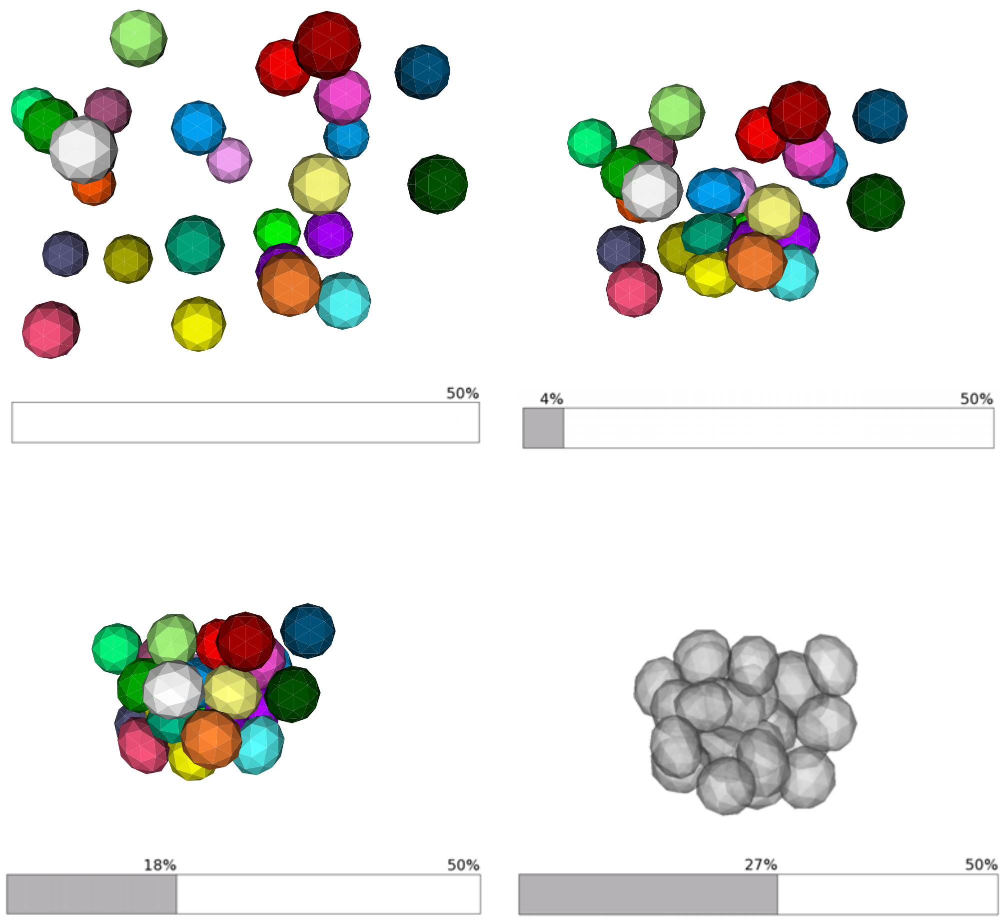

Figure 6: *Simulation of the aggregation of cells: twenty five incompressible deformable spheres move towards an external source of chemical signal. For each screenshot the percentage of CPs according to the total number of particles is shown in the bottom part. From top to bottom and left to right: t=0s, t=100s, t=200s and t=400s. The last frame illustrates in transparency the efficiency of the contact resolution.*

proach. Also our method needs to be optimized to not specifically target spherical shapes. A problem that has not been addressed yet is the self-collision. Future work also includes the detection and response for such situations. Finally, we will use the method by Spillmann and Teschner [ST05] to improve the management of interpenetrations between bodies.

7. Acknowledgment

A.C. acknowledges a fellowship from the French Ministry of Research and the Rhône-Alpes Institute of Complex Systems (IXXI), France.

References

[BFA02] Bridson R., Fedkiw R., Anderson J.: Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 594–603.

[BJ07] Barbič J., James D.: Time-critical distributed contact for 6-dof haptic rendering of adaptively sampled reduced deformable models. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), Eurographics Association, pp. 171–180.

[BMB*07] Berro J., Michelot A., Blanchoin L., Kovar D., Martiel J.: Attachment conditions control actin filament buckling and the production of forces. *Biophys J 92*, 7 (2007), 2546–2558.

[Can93] CANI M.-P.: An implicit formulation for precise contact modeling between flexible solids. In *Computer Graphics (ACM SIGGRAPH)* (1993), ACM, pp. 313–320. Published under the name Marie-Paule Gascuel.

[CBP05] CLAVET S., BEAUDOIN P., POULIN P.: Particle-based viscoelastic fluid simulation. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM Press, pp. 219–228.

[CP04] CHABANAS M., PROMAYON E.: Physical model language: Towards a unified representation for continuous and discrete models. In *Proceedings of International Symposium on Medical Simulation* (2004), pp. 256–266.

[DAK04] DURIEZ C., ANDRIOT C., KHEDDAR A.: Signorini's contact model for deformable objects in haptic simulations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2004), vol. 4, pp. 3232–3237.

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic real-time deformations using space & time adaptative sampling. In *Computer Graphics (ACM SIGGRAPH)* (2001), ACM, pp. 31–36.

[DDKA06] DURIEZ C., DUBOIS F., KHEDDAR A., ANDRIOT C.: Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE Transactions on Visualization and Computer Graphics 12*, 1 (2006), 36–47.

[DMG05] DEQUIDT J., MARCHAL D., GRISONI L.: Time-critical animation of deformable solids. *Computer Animation and Virtual Worlds 16*, 3-4 (2005), 177–187.

[DTSH88] DEMBO M., TORNEY D., SAXMAN K., HAMMER D.: The reaction-limited kinetics of membrane-to-surface adhesion and detachment. *Proceedings of the Royal Society of London. Series B, Biological Sciences 234*, 1274 (1988), 55–83.

[HTK*04] HEIDELBERGER B., TESCHNER M., KEISER R., MÜLLER M., GROSS M.: Consistent penetration depth estimation for deformable collision response. In *Proc. Vision, Modeling, Visualization (VMV)* (2004), pp. 339–346.

[KMH*04] KEISER R., MÜLLER M., HEIDELBERGER B., TESCHNER M., GROSS M.: Contact handling for deformable point-based objects. In *Proc. Vision, Modeling, Visualization (VMV)* (2004), pp. 315–322.

[KP04] KOVAR D., POLLARD T.: Insertional assembly of actin filament barbed ends in association with formins produces piconewton forces. *Proceedings of the National Academy of Science USA 101*, 41 (2004), 14725–14730.

[MC94] MIRTICH B., CANNY J.: *Impulse-Based Dynamic Simulation*. Tech. Rep. CSD-94-815, 1994.

[Mir96] MIRTICH B.: Hybrid simulation: combining constraints and impulses, 1996. Technical Report, Department of Computer Science, University of California, Berkeley.

[MPT06] MARCHAL M., PROMAYON E., TROCCAZ J.: Simulating prostate surgical procedures with a discrete soft tissue model. In *Eurographics Workshop in Virtual Reality Interactions and Physical Simulations, VriPhys06* (2006), I. Navazo C. M., (Ed.), pp. 109–118.

[PB03] POLLARD T., BORISY G.: Cellular motility driven by assembly and disassembly of actin filaments. *Cell 112*, 4 (2003), 453–465.

[PMT03] PROMAYON E., MARTIEL J.-L., TRACQUI P.: Physically-based 3d simulations of cell deformations and migrations. In *Polymer and Cell Dynamics - Multiscale Modeling and Numerical Simulations* (December 2003), Alt W., Chaplain M., Griebel M., Lenz J., (Eds.), Birkhäuser, pp. 125–138.

[PPG04] PAULY M., PAI D. K., GUIBAS L. J.: Quasi-rigid objects in contact. In *Proc. Eurographics/ACM Siggraph Symposium on Computer Animation* (2004), pp. 109–119.

[Pro97] PROVOT X.: Collision and self-collision handling in cloth model dedicated to design garments. *Proceedings of Graphics Interface* (1997), 177–189.

[SBT07] SPILLMANN J., BECKER M., TESCHNER M.: Non-iterative computation of contact forces for deformable objects. *The journal of WSCG 15* (2007).

[ST05] SPILLMANN J., TESCHNER M.: Contact surface computation for coarsely sampled deformable objects. In *Proc. Vision, Modeling, Visualization (VMV)* (2005), pp. 289–296.

[TKH*05] TESCHNER M., KIMMERLE S., HEIDELBERGER B., ZACHMANN G., RAGHUPATHI L., FUHRMANN A., CANI M.-P., FAURE F., MAGNENAT-THALMANN N., STRASSER W., VOLINO P.: Collision detection for deformable objects. *Computer graphics forum 24*, 1 (2005), 61–81.

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. *Computer Graphics (ACM SIGGRAPH) 21*, 4 (1987), 205–214.