Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2009)
H. Prautzsch, A. Schmitt, J. Bender, M. Teschner (Editors)

Friction Handling for Penalty-Based Methods

O. Lazarevych, J. Spillmann, C. Renner, G. Szekely, M. Harders

Computer Vision Laboratory, ETH Zurich, Switzerland

Abstract

In order to handle collisions between interacting deformable objects, the penalty method is widely employed since it is simple to implement and computationally inexpensive. In this paper a novel penalty method for handling collisions with the focus on the simulation of resting states is proposed. In detail, a novel time-coherent formulation for the static friction forces is presented that reproduces both the resting states and the transitions between sliding and sticking in a physically realistic way. The method is tested on a range of challenging real-time and off-line scenarios, underpinning the conceptual benefits of the approach.

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

1. Introduction

The contact handling of interacting solid bodies is a well-investigated research area in animation or haptics. The goal of contact handling is to compute forces that prevent the interpenetrations of the simulated solids. Moreover, the dissipative effects due to frictional contacts must be considered.

In the past, many different approaches have been proposed to handle contacts. Most of them can be split into two classes, notably *constraint methods* and *penalty methods*. Constraint methods [Bar89, KEP05, OTSG09] usually impose geometric non-penetrations constraints, based on the relative positions and orientations of the involved bodies. The resulting system of equations can then be solved with a variety of methods, in particular, iterative approaches. Constraint methods are the favorable choice for simulations that require high mechanical accuracy. However, they are often computationally expensive and complex in terms of implementation.

In contrast, penalty methods compute spring forces that are based on a geometric interpenetration measure [HTK*04, KMH*04, FBAF08, HVS*09]. These springs generate external forces on the colliding objects and drive them apart. It has to be emphasized, that despite their lack of physical plausibility, penalty methods are still widely used in practice. This is because, on the one hand, they are simple in terms of implementation, which makes them appealing in the context of rapid prototyping and education. On the other hand, penalty methods commonly compute the spring forces directly, requiring only a small constant number of operations per contact point. Therefore, they are also frequently used in applications requiring fast update rates, *e.g.* in computer games or haptic rendering.

However, penalty methods suffer from the drawback that collisions are never resolved entirely. This is particularly problematic in the context of stacking simulations that rely on the accurate computation of the tangential friction forces, because the tangent spaces cannot be obtained unambiguously if the objects interpenetrate. Consequently, it has been difficult to simulate resting states of complex stacking problems in a physically plausible way with these methods.

Contribution. In this paper, we propose a novel penalty method for handling collisions of deformable objects. The starting point is the predictor-corrector scheme proposed in [SBT07]. We extend the method with a novel time-coherent formulation for determining the static friction forces, which is especially useful for simulating stacked deformable objects. The strength of the approach stems from the consideration of the spatial and temporal discretization of the simulation. Although our method is not physically-based, it is able to reproduce both the resting states and the transitions between sliding and sticking in a physically plausible way. We test our method on a variety of different examples, ranging from real-time scenarios to difficult stack-

DOI: 10.2312/PE/vriphys/vriphys09/135-144

ing problems with hundreds and thousands of interacting objects. Our technique also performs well when tested against cases with analytical solutions.

2. Related work

Handling contacts and friction for solid objects is a common problem for computer graphics as well as computational mechanics applications. Even if having different objectives, these two fields of computer modeling share a large variety of methods.

Computational mechanics applications require highly precise and physically realistic simulations. Therefore, mathematical programming methods are very popular in this field. Physical accuracy has much higher priority than computational speed, wherefore most applications use highly accurate physical approaches, *e.g.* finite element models, and rely on off-line computations. The theoretical basis of the mechanics is thoroughly discussed in the works of Stronge [Str90], Persson [Per98] and others. Wriggers and Panagiotopoulos [WP99] present a variety of methods that are employed in computational mechanics, and particularly in contact problems. Many methods use a formulation of the contact in the form of a Linear Complementarity Problem (LCP). This approach has successfully been applied for instance in simulations of rigid [ST96, AP97] as well as deformable [EE06, EE07] interacting objects.

Constraint methods are also widely used for rigid body simulations in computer graphics due to their physical correctness. A fundamental and thorough study of modeling rigid body dynamics is available in classical papers of Baraff [Bar89, Bar91, Bar94]. Based on the concepts of rigid body dynamics, a variety of different methods of contact handling were proposed later on. Kaufman et al. [KEP05] employed a contact model that uses information about mass, location, and velocity of all contacts at the moment of maximum compression to compute rigid body velocity. They developed their friction model in the configuration space of the rigid body based on the principle of maximal dissipation. Guendelman et al. [GBF03] proposed to merge collision and contact handling algorithms tightly with the time integration scheme. This allowed them to cleanly separate collisions and contacts without using *ad hoc* threshold velocity, and to model difficult frictional effects. Twigg et al. [TJ08] considered a hard problem of time-reversed integration of rigid body motion and proposed methods to allow time-stepping of body systems backward in time.

Application of the constraint methods for deformable body collision response may be found in recent papers of Otaduy et al. [OTSG09], Kaufman et al. [KSJP08], Duriez et al. [DDKA06], Pauly et al. [PPG04], etc. Though constraint based methods are physically accurate, they are computationally expensive.

Impulse-level methods demonstrate good results for dynamic collisions. Several such procedures have been proposed by Mirtich et al. [MC95], Bridson et al. [BFA02], Cirak and West [CW05] and Galoppo et al. [GOM*06]. Although these approaches are working well for dynamic collisions, they are not suitable for static scenes. Resting states cannot be modeled correctly by them, wherefore extensions are needed, like the introduction of a threshold velocity to distinguish static cases for separate treatment.

Predictor-corrector methods for interactive objects modeling were proposed in several papers, like Bender and Schmitt (for rigid bodies) [BS06], Weinstein and Teran [WT06] or Müller et al. [MHHR07]. The scheme proposed by Spillmann et al. [SBT07] combines predictor-corrector methods with penalty approaches giving promising results. Unlike other penalty methods this approach avoids overshooting. However, the friction handling is not accurate. We take this model as a basis of our development and extend it by introducing improved friction handling, including force post-correction to ensure momentum preservation and static friction.

In the following we briefly describe the simulation scheme and give a short outline of the algorithm.

3. Algorithm Overview

3.1. Simulation Environment

For the simulation of dynamic scenes with objects discretized into tetrahedral elements we use a predictor-corrector scheme, in which each simulation step consists of several successive phases.

The predictor phase starts with the computation of the internal forces for all objects in the scene. They are combined with the external forces, e.g. gravity or other potential forces, and numerically integrated, resulting in unconstrained values of the positions and velocities of the simulated mass points.

A collision detection algorithm based on optimized spatial hashing [THM*03] provides a set of collisions that have to be resolved. The collision response procedure calculates contact forces aiming at preventing the interpenetrations. The computed contact forces are added to the external and elastic forces and re-integrated to obtain constrained positions and velocities.

3.2. Contact Handling Algorithm

Our contact handling is an extension of the penalty-based method proposed by Spillmann et al. [SBT07]. We employ a correction algorithm to ensure the conservation of the momentum for the forces evolved during the contact. Thereafter, for the robust treatment of the static friction, we distinguish among dynamic and resting state configurations. The latter cases are further processed by our algorithm,

which provides numerically stable simulation devoid of non-physical effects, e.g. unjustified slipping.

The following steps are performed during the response force calculation for each collision:

1. The penetration depth vector of the contact point is computed [HTK*04].
2. The contact force is calculated based on the value of the corresponding depth vector according to the model of Spillmann et al. [SBT07].
3. Static and dynamic cases are distinguished and treated separately. The friction force is computed and added to the contact force resulting in an aggregate response for the contact point.
4. The response force is distributed among the mass points involved in the collision according to the force post-correction algorithm.

3.3. Basic Contact Model

Here we briefly recapitulate the contact model of Spillmann et al. [SBT07] which is the basis for the further collision response method. In the following discussion we consider a collision (i, T_i) – a pair of basic elements of the surfaces colliding at the current simulation time step: a contact point i and a contact surface triangle $T_i = \{q, r, s\}$. Note that each vertex of the triangle T_i may in turn be a contact point of a different collision. All colliding vertices at a given time step constitute the contact surface.

The relative position of the contact with respect to the triangle is described by the barycentric coordinates. Since the simulation gives object positions only at discrete time steps Δt, the precise position of the collision between the point and triangle cannot be found, thus we reasonably assume the contact position to be an orthogonal projection of the point i onto the surface of the triangle T_i. The error introduced by this assumption is tolerable in the context of this approach, although it may cause visible inaccuracy in the case of oblique collision. Barycentric coordinates $\_{ij}$ of all points of the contact in consideration are collected in the mapping matrix $H = (h_{ij}) \in \mathbb{R}^{n \times n}$, where n is the number of simulated points.

$$h_{ij} = \begin{cases} \_{ij}, & j \in T_i \\ 0, & \text{else} \end{cases} \tag{1}$$

For each collision a penetration depth vector $\mathbf{d}_i(t + \Delta t)$ is calculated. Each pair (i, T_i) induces a set of contact forces: $\mathbf{f}_i$ applied to the contact point and $\{\mathbf{f}_q, \mathbf{f}_r, \mathbf{f}_s\}$ applied to the corresponding vertices of the contact triangle. Since each of these points may participate in other collisions, in general several forces act on one point. This coupling makes it difficult to find a consistent set of contact forces for the whole contact region.

Spillmann et al. [SBT07] make the assumption that the contact space of the collision (i, T_i) can be considered to be spanned by the penetration depth vector $\mathbf{d}_i(t + \Delta t)$, and that the penetration depths of the vertices of T_i conform with it.

$$\mathbf{d}_q(t + \Delta t) = \mathbf{d}_r(t + \Delta t) = \mathbf{d}_s(t + \Delta t) = -\mathbf{d}_i(t + \Delta t) \tag{2}$$

In other words, it is assumed that the contacting surfaces are almost flat.

Accordingly, the resulting contact force $\mathbf{F}_i$ on the contact point i, which is the sum of all local forces $\mathbf{f}_i$ induced by the collisions acting on this point, is collinear with the depth penetration vector and can be calculated as

$$\mathbf{F}_i = \frac{m_i}{\Delta t^2} \; \_i \mathbf{d}_i(t + \Delta t) \tag{3}$$

where m_i is the mass of the point i, and the dimensionless coefficient $\_i$ ensures that the resulting position of the point and its corresponding triangle is collision free.

$$\_i = \frac{\sum_{j=1}^{n} c_j h_{ij} m_j}{c_i m_i + \sum_{j=1}^{n} c_j h_{ij} m_j} \tag{4}$$

The normalization factor c_i accounts for the spatial discretization of the surfaces and can be computed as

$$c_i = \frac{1}{1 + \sum_{j=1}^{n} h_{ji}} \tag{5}$$

The coefficients $\_i$ and c_i are derived from the statements of local and global conservation of momentum, respectively. For details refer to [SBT07].

The assumptions of the model allow to decouple the contact force computation. For each contact point the forces can be computed separately without solving a system of equations. Moreover, unlike usual penalty-based methods, this approach inhibits overshooting, because it directly considers the time step of the numerical integration in the calculations. A collision-free state is maintained during the whole simulation. However, note that this is exact only under the assumption of flat contact surfaces. Another advantage of the method is its high computational efficiency.

It should be emphasized that the basic model makes strong assumptions about the collinearity of the penetration depth vectors $\mathbf{d}_i$ for adjacent collisions and uniform mass distribution between the mass points. This means that the model holds only for the limit case of locally flat surfaces, which corresponds to finely discretized meshes. However, since our approach is tailored for interactive and real-time applications, we have to consider the errors induced by more sparse spatial discretizations.

4. Extension of the Contact Model

In order to address the aforementioned problems of the basic model we introduce an additional step in the contact force computation – a force post-correction, which ensures momentum preservation taking into account the spatial and temporal discretization.

4.1. Force Post-Correction

To briefly summarize the assumptions of the discussed model, it has been assumed that (a) the contact space is one-dimensional for each collision, which is equivalent to flatness of the contact surface, and (b) contact points of each contact triangle in turn collide themselves with other triangles.

However, due to the spatial and temporal discretization of the simulation model, the assumption (b) obviously does not hold for points on the boundary of the contact surface, where a point of a contact triangle in turn does not collide with another triangle. Because of this, collisions are not entirely resolved on the boundaries. Moreover, for highly curved or concave objects the assumption (a) is also violated.

Inconsistencies between the mentioned assumptions and the actual simulation model result in slight interpenetrations during collisions and violation of the momentum conservation. The first effect in most cases may be tolerable, because during the time of contact elastic bodies deform, especially around the contact surface, and therefore hide the interpenetration regions. However, the violation of the momentum conservation leads to unnatural drifting of objects, which is visible and adversely affects the physical realism of the simulation. For example, in the scene shown in the Fig. 3(a) the clamped box commences to turn counterclockwise which leads to an unstable position. After several simulation time steps the box finally slips out from between the bars to one side.

To address this problem we derived a force post-correction operator $\mathcal{COR}(\mathbf{F}_j)$, which modifies the force $\mathbf{F}_j$ taking into account all contact forces applied to points that participate in contacts with the point j. Formally the post-correction operator may be defined as following (see the Appendix for a detailed derivation).

$$\mathcal{COR}(\mathbf{F}_j) \rightarrow \mathbf{F}_j^{new} \stackrel{\text{def}}{=} \frac{1}{2} \cdot \mathbf{F}_j - \frac{1}{2} \cdot \sum_{i=1}^{n} h_{ij}\mathbf{F}_i \qquad (6)$$

Practically, this post-correction operator distributes a part of the contact force acting on contact points to the vertices of corresponding contact triangles (Fig. 1). Each term of the right sum in (6) corresponds to a particular contact pair (i, T_i), in which point j is a vertex of T_i, and accounts for a part of the force $\mathbf{F}_i$ distributed to the vertex j. The summation is done over all contact pairs with triangles containing point j as a vertex (Fig. 2).

The proposed force post-correction allows to loosen the assumption about the local flatness of the contact surface. It accounts for possible non-collinearity of the resulting contact forces of the points involved in particular collisions. Its formulation directly considers the spatial and temporal discretization of the simulation and therefore overcomes the aforementioned problem. However, it does not guarantee a complete collision free state at the end of a time step.

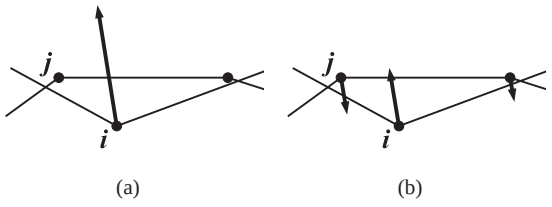


Figure 1: *The post-correction distributes the contact force of contact point i (1(a)) to other points involved in the collision (1(b)). For simplicity, distribution of only one force component ($\mathbf{F}_i$) is shown, i.e. the unmodified contact forces of the other points are not visible.*

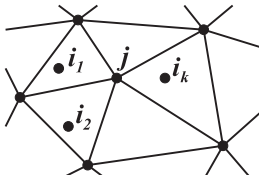


Figure 2: *The summation is done over all contact pairs containing the point j as a vertex of a contact triangle*

It is important to notice that the friction force should not be treated directly by the post-correction operator. The first term in (6) scales the contact forces to avoid overshooting at the next time step. But if the friction force is similarly scaled as a component of the contact force it does no longer correspond to the value of the friction coefficient. In order to avoid this we separately distribute the friction force between the points of the contact triangle without scaling.

However, the post-correction still influences the value of the friction force. It generally modifies the tangential component of the contact force, because the $\mathbf{F}_j$, $j \in T_i$ are generally not parallel. This tangential component could be considered as one of the constituents of the friction force. Although, the friction force only slightly changes, it may seriously affect static friction handling, because even small errors in the friction force value would lead to visible slip of an object, which should theoretically remain in a resting position.

This problem of precise friction handling is addressed in the next section.

4.2. Static Friction Handling

According to Coulomb's friction model [MC95], the friction force is the tangential component $\mathbf{f}$ of the contact force and it depends on the magnitude of its normal component $\mathbf{f}^n$. In the dynamic case, when the point is moving on the surface with the relative tangential velocity $\mathbf{u}$, the friction force is

defined by

$$\mathbf{f}^{\max} = -\mu|\mathbf{f}^n| \cdot \frac{\mathbf{u}}{|\mathbf{u}|} \tag{7}$$

In the static case the law gives

$$|\mathbf{f}| \leq \mu|\mathbf{f}^n| \tag{8}$$

while the actual magnitude and direction of the friction force is not specified but can be found as one to maintain the static position on the surface.

We adapt the continuous Coulomb friction model to the discrete case. The relative velocity of a given contact pair is defined as the difference between tangential components of the contact point velocity and the triangle velocity. The latter is the sum of the triangle vertex velocities weighted by the corresponding barycentric coordinates of the contact point.

The value of the relative velocity without any friction force is given by $\tilde{\mathbf{u}}$. A non-zero friction force $\mathbf{f}$ results in the modified relative velocity $\mathbf{u}$. Thus, for a resting contact, after application of friction forces, we should have $\mathbf{u} = 0$ (note, that in the static case generally $\tilde{\mathbf{u}} \neq 0$. Thus, the direction of the friction forces can easily be computed). In cases when $\tilde{\mathbf{u}} = 0$ (within acceptable error range), the direction does not need to be determined, since the friction force has to be zero, too, to maintain the resting state. For the following discussion we assume that $\tilde{\mathbf{u}} \neq 0$ and operate with scalar values of corresponding quantities — *i.e.* with their projections on the direction $\frac{\tilde{\mathbf{u}}}{|\tilde{\mathbf{u}}|}$, e.g. $v = \mathbf{Proj}(\mathbf{v}) = \mathbf{v} \cdot \frac{\tilde{\mathbf{u}}}{|\tilde{\mathbf{u}}|}$.

In order to distinguish between dynamic and static friction we first assume the dynamic case and then compare the values of the relative velocity of the contact pair before and after applying the friction force.

The contact point velocity with applied dynamic friction force is computed by the integration algorithm as

$$v_i = \tilde{v}_i + f^{\max} \cdot \frac{\Delta t}{m_i} \tag{9}$$

where $\tilde{v}_i$ is the velocity obtained disregarding any friction.

Similarly, the velocity of the contact triangle is defined as

$$v_{T_i} = \tilde{v}_{T_i} - f^{\max} \cdot \quad (h_{ij}) \tag{10}$$

where (h_{ij}) defines how the friction force is distributed between the vertices of the contact triangle and does not depend on the value of the friction force, but only on the relative position of actual contact between the point and the triangle (*i.e.* the barycentric coordinates), as well as the masses of involved points and the simulation time step.

This way, the relative velocity u^* assuming dynamic friction is found as

$$u^* = \tilde{u} + f^{\max} \cdot \frac{\Delta t}{m_i} + f^{\max} \cdot \quad (h_{ij}) \tag{11}$$

where $\tilde{u} = \tilde{v}_i - \tilde{v}_{T_i}$.

If u^* and $\tilde{u}$ have the same sign (*i.e.* positive), then the friction is not strong enough to stop the sliding during the time step Δt. In this case we have dynamic friction. Otherwise, the friction force stops the sliding during the current simulation time step and we have static friction.

In the static case the value of the dynamic friction force has to be scaled such that the relative velocity u becomes zero. If we use a scaling coefficient : $f = \cdot f^{\max}$, $\in [0,1]$, then the value of the relative velocity u is found similarly to (11).

$$u = \tilde{u} + \frac{f}{} \cdot \frac{\Delta t}{m_0} + \frac{f}{} \cdot \quad (h_{ij}) \tag{12}$$

Now if we assume $u = 0$ and take into account (11) we get

$$= \frac{\tilde{u}}{\tilde{u} - u^*} \tag{13}$$

As discussed above, if the force post-correction operator is applied after the calculation of friction force, the forces are changed such that a static position is no longer held. In order to account for this post-correction artifact in advance, we propose to additionally scale the friction force

$$f = \cdot f \tag{14}$$

Now our aim is to find such that the relative velocity for this static friction case equals zero at time $t + \Delta t$, *i.e.* at the beginning of the next time step.

The scaling factor defines the contact force which is then modified by the post-correction algorithm and integrated, resulting in the relative velocity at the next simulation time step. Thus, the effect of the post-correction algorithm can be seen as a mapping of the set of scaling factors $\{ _1, _2, \ldots, _N\}$ to the set of relative velocities $\{u_1, u_2, \ldots, u_N\}$, where N is the number of contact points involved in static friction. In order to compute static friction forces that ensure a static position of the system, we need to find a solution of a system of non-linear equations.

$$\begin{aligned} u_1(\ _1, \ _2, \ldots, \ _N) &= 0 \\ u_2(\ _1, \ _2, \ldots, \ _N) &= 0 \\ &\ldots \\ u_N(\ _1, \ _2, \ldots, \ _N) &= 0 \end{aligned} \tag{15}$$

For finding a solution of this system we employ iterative methods. Note that the relative velocity in the static friction case should remain zero for successive simulation time steps (obviously, until some external force or change of the geometrical configuration causes a violation of the static state condition). Therefore, the relative velocities in this special case depend only on the introduced variables $_i$ and are independent of any other simulation parameters. Therefore, it is sufficient to perform one iteration of this static friction correction per one time step of the dynamical simulation. Note that this is only possibly in cases of static friction. Moreover, the iterative process should converge faster than any changes

of local geometry of the contact surface or of the external forces. Experiments showed that an acceptable number of iterations, depending on the scene, varies from $30-40$ to ~ 100 simulation steps.

Let the current iteration time step, corresponding to the time point t, be denoted as the k-th. In the following discussion we omit the arguments of the functions u_i. Taking into account the specifics of the implementation, at the beginning of the time step the following values are available or can be computed:

- u_i^k – the relative velocity in the current time step;
- $\;_i^k$ – the scaling factor used in the previous step, which defines the relative velocity in the current step.

Using a substitution

$$\;_i = \;_i - \;_i \cdot u_i \tag{16}$$

where $\;_i$ is a constant that affects the convergence rate and can be defined experimentally, we come to the system of equations $\;_i = \;_i, i = 1 \ldots N$ which is equivalent to $u_i = 0, i = 1 \ldots N$, and can be solved by a fixed-point iteration [BF00].

$$\;_i^{k+1} = \;_i^k = \;_i^k - \;_i \cdot u_i^k \tag{17}$$

If we use one common value $\;$ for all $\;_i$ we obtain a relaxation method with one parameter. In the implementation of the contact handling method the post-correction algorithm distributes the contact force of the point i as soon as it is computed, i.e. before processing the next contact point $i+1$. Therefore, the relative velocity u_i^k is calculated using the updated values of the arguments $\;_j^{k+1}$, $j < i$, leading to an iterative procedure, which is analogous to Seidel's method [BF00].

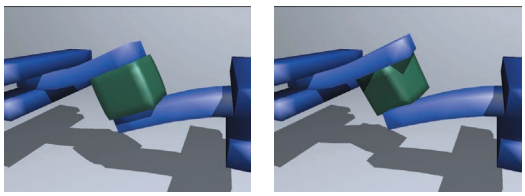
$$\begin{aligned}
\;_1^{k+1} &= \;_1^k - \;\cdot u_1(\;_1^k, \;_2^k, \;_3^k, \ldots, \;_N^k)\\
\;_2^{k+1} &= \;_2^k - \;\cdot u_2(\;_1^{k+1}, \;_2^k, \;_3^k, \ldots, \;_N^k)\\
\;_3^{k+1} &= \;_3^k - \;\cdot u_3(\;_1^{k+1}, \;_2^{k+2}, \;_3^k, \ldots, \;_N^k)\\
&\ldots\\
\;_N^{k+1} &= \;_N^k - \;\cdot u_N(\;_1^{k+1}, \;_2^{k+2}, \;_3^{k+1}, \ldots, \;_N^k)
\end{aligned} \tag{18}$$

This method converges if the Jacobian matrix $\left(\frac{\partial u_i}{\partial \;_j}\right)_{ij}$ satisfies the condition of diagonal prevalence [BF00]. However, since the explicit form of the functions u_i is unknown due to the non-linearity of the post correction operator, it is impossible to verify this condition. Nevertheless, convergence and stability of the solution could be verified experimentally. It should be also noticed that the convergence is dependent on the constant $\;$ which usually needs to be experimentally chosen to guarantee acceptable convergence.

5. Results

The new approach tested in different scenes ranging from simple setups, *e.g.* a box on an inclined plane or a box pressed between stiff bars (Fig. 3), to complex scenes with hundreds of non-convex objects in contact. During the simulations two parameters were observed for characterizing the performance: the kinetic energy of the moving object and the linear relative velocity of several selected contact points of the objects in contact.

Good results were obtained for $\;_i = \;, i = 1, \ldots, N$ in the range $\; \in [-1, 1]$. Best convergence rate was consistently reached for $\; = -0.1$, which has been used in all presented examples. For example, in the simple scene with a box falling on an inclined plane, a solution in an acceptable error range is found after 8 – 16 time steps. The relative velocity drops from $\sim 10^{-3}$ at the beginning of the static friction phase, to an acceptable value (within error range) of $\sim 10^{-9}$.



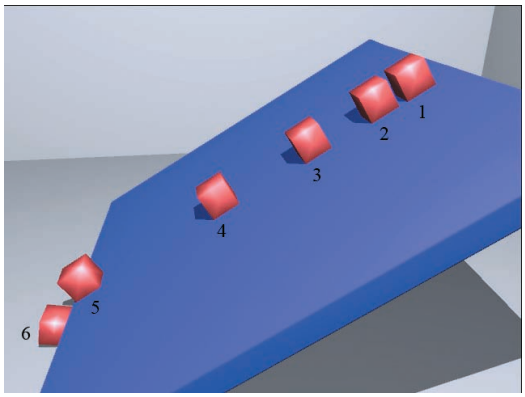
(a) (b)

Figure 3: *A box pressed between stiff bars: stable static position simulated with the new method 3(a) and unrealistic sliding in the simulation with the old method 3(b)*

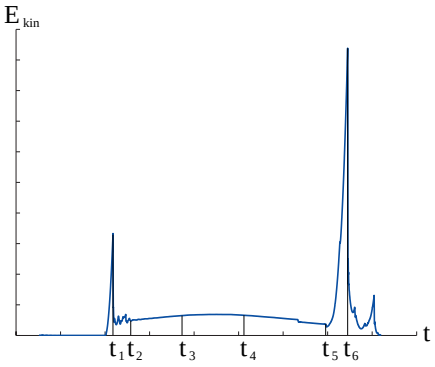
A box on an inclined plane is one of the simplest setups to test the correctness of the friction handling algorithm. A box should slide down if the inclination is beyond a critical angle, and stick to the plane if the inclination is small. The critical angle $\;$ is related to the friction coefficient μ by $\tan \; = \mu$. If the inclination of the plane is smaller than critical, the sliding box is coming to a rest position, if larger then it accelerates downwards. If the inclination is critical then the box continues to move with its initial speed (or stays in rest if the initial velocity is zero).

Fig. 4(a) displays the process of friction handling. The box is sliding on the membrane having a positive initial velocity (t_1). The inclination of the membrane is critical, but since it is deformed under its own weight and the weight of the box, the effective angle of inclination is slightly larger before the box passes the middle of the membrane, and smaller afterwards. As a consequence, the box is first accelerating (t_2, t_3), and then slowing down (t_4), which is hardly visible in the simulation but clearly seen in the plot of the kinetic energy vs. simulation time (Fig. 4(b)). The middle part between the spikes (t_1, t_6) corresponded to falling on and off the plane indicating the initial speeding-up followed by the deceleration. The numbers of the box positions in Fig. 4(a) correspond to the simulation time points in Fig. 4(b).

In experiments with a rigid plane, where the inclination is smaller than critical, the objects decelerate, stop sliding,

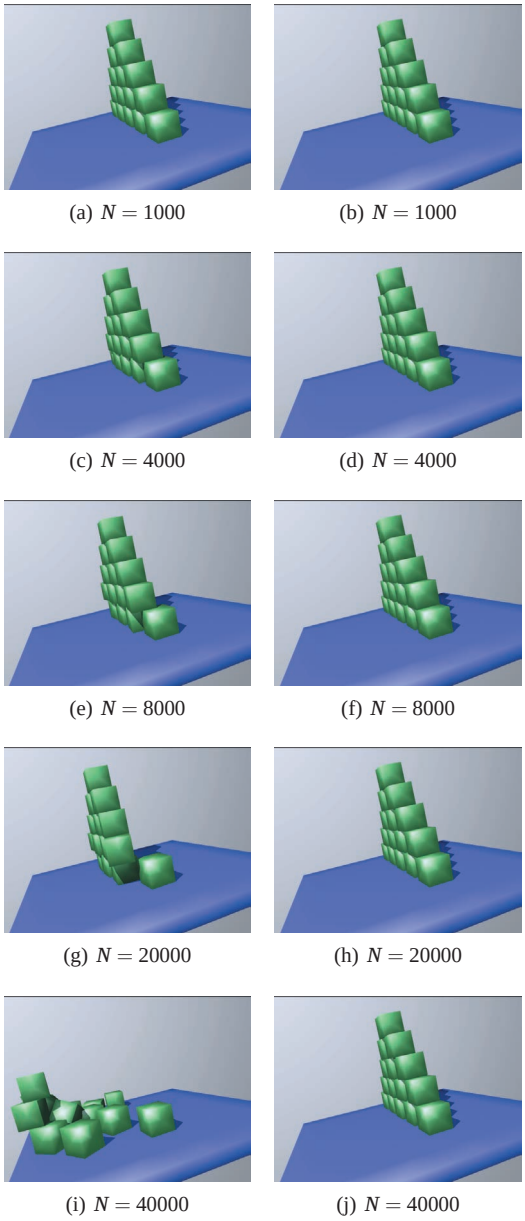


(a)



(b)

Figure 4: *A box on the inclined membrane 4(a): $\mu = 0.6$, $= 30.96^{\circ}$. The middle part of the kinetic energy vs. time plot 4(b) reflects acceleration and deceleration of the box*



(a) $N = 1000$ (b) $N = 1000$

(c) $N = 4000$ (d) $N = 4000$

(e) $N = 8000$ (f) $N = 8000$

(g) $N = 20000$ (h) $N = 20000$

(i) $N = 40000$ (j) $N = 40000$

Figure 5: *Simulation of the pyramid on the inclined plane ($\mu = 0.3, \tan = 0.2$): without friction stabilization (left) and with stabilization (right). N denotes the simulation step*

and then remain in a stable resting position. This demonstrates that our method also provides appropriate transition from dynamic to static friction.

Static friction on inclined plane. The importance of correct and stable static friction handling is demonstrated in the following example. A pyramid of boxes is placed on an inclined plane with an inclination angle smaller than critical. Due to uncompensated influence of the post-correction the boxes in the left column of Fig. 5 start to slide, causing deformations and breaking of the pyramid, while using our method the pyramid remains stable as shown in the right column.

Stacking objects. The stability of the contact handling may be well tested in a simulation of a large stack of non-convex objects. In the setup shown in Fig. 6 objects are falling into a basket. The whole system should have zero kinetic and constant potential energy after all oscillating effects associated with deformations diminish.

The plot in Fig. 6(d) illustrates the development of the kinetic energy over simulation time. It compares the results obtained by using our new method and by the original one.

From the beginning of the simulation to the time step t_a the plot of the kinetic energy has spikes and oscillates for both methods due to generation of new objects falling into the basket. Complex contact interactions between the objects cause them to move locally, finally arriving at stable rest-

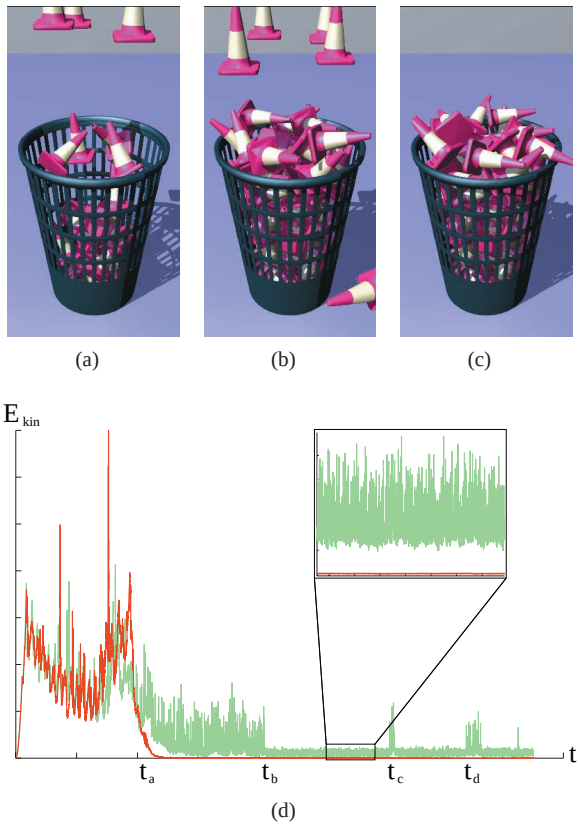


Figure 6: *Time dependencies of the kinetic energy for a system of stacking objects (6(a) – 6(c)) are compared for the original (green line) and new (red line) methods 6(d)*

ing positions. Due to the complex shapes of the bodies their packing in the basket is not dense. Therefore, the stability of the object stack heavily relies on correct static friction handling.

The fast drop of the kinetic energy for the new friction handling method shows the fast stabilization of the stack, which remains stable until the end of the simulation. The kinetic energy of the system oscillates in an acceptable error range and therefore the objects do not visually move.

Due to inaccurate static friction computations in the old method, a considerable period of time (t_a to t_b) is passing before a stabilization of the stack. After the objects come to stable positions the kinetic energy of the system still oscillates and objects continue to slide at the contacts. Although the sliding is not largely visible, it nevertheless may lead to a configuration where a particular object comes out of the static state and moves to a new position. This movement causes spikes like the one at time t_c. Sometimes a movement of one object also causes neighboring objects to translate. In these cases several successive spikes of the kinetic energy are generated (t_d).

In the phase of the resting configuration the potential energy oscillates for both methods instead of being zero. However, the amplitude of oscillations as well as its time average value is much lower for the new method (see Fig. 6(d)). The average ratio between the mean values resulting from the two simulations is around 0.01. This demonstrates that the new method is significantly more stable and therefore giving more realistic results.

A pyramid of balls. The correctness and stability of the friction handling for curved objects is nicely demonstrated in the simulation of a pyramid of four balls as shown in Fig. 7(a).

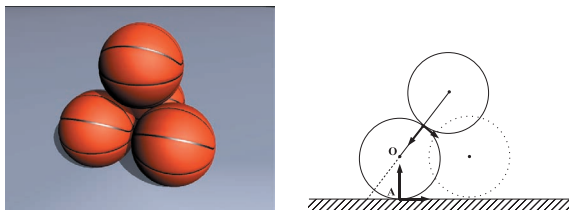


Figure 7: *A pyramid of four balls. In Fig. 7(b) only forces applied to the lower ball are shown*

Examination of the static problem of stacked rigid balls (Fig. 7(b)) provides an estimation of the value of the friction coefficient required for a stable pyramid. Theoretical analysis of torques relatively to point O gives the critical friction coefficient between the lower ball and the ground: $\mu_g \approx 0.14$. Moreover, consideration of torques relatively to point A gives $\mu_b \approx 0.22$ for friction between balls.

In the simulations the same friction coefficient is used for all contacts. Therefore, theoretically the pyramid should be stable if the friction coefficient is larger than μ_b. If it is lower the pyramid should break. Moreover, if in the latter case the friction coefficient is larger than μ_g then during the breaking of the pyramid the balls slide at the contacts between each other, but the lower balls do not slide on the ground surface.

The experiments showed that the pyramid is stable for $\mu = 0.24$ and breaks for $\mu = 0.20$, which corresponds well with the theoretical values. Further, in the latter case the lower balls indeed do not slide on the ground as predicted. Higher precision of the experimental results could not be achieved, which is likely due to the balls not being ideally spherical but having a triangulated surface. Moreover, the balls are slightly deformed in the simulation. However, the results show that the friction is also handled appropriately for non-flat cases.

6. Discussion and Conclusions

We have proposed a novel, penalty-based method for handling collisions of deformable objects and implemented it

for the predictor-corrector simulation scheme proposed earlier in [SBT07]. Our time-coherent formulation for calculating the static friction forces directly considers the spatial and temporal discretization of the simulation. We apply it to simulate objects discretized into terahedral elements, since this is one of the most popular and simple ways to represent volumetric objects. However, our approach is not limited to this case.

The implementation of the presented numerical method is simple and efficient. The fixed-point iteration needs only one additional variable per contact pair to be stored and the computation complexity is still remaining linear with the number of collisions, as is the case for the original method [SBT07]. Successive processing of the collisions preserves the initial underlying predictor-corrector scheme. Nevertheless, the method requires the definition of the constants $_i$, which are dependent on the unknown shape of the post-correction function and can only be found experimentally. However, other iteration methods which do not have this drawback could also be considered. We for instance also implemented and tested the secant method [BF00], which does not require any experimentally defined parameters. The iterative step in this case has the form

$$ {}^{k+1}_{i} = {}^{k}_{i} - \frac{{}^{k}_{i} - {}^{k-1}_{i}}{u_i^k - u_i^{k-1}} \cdot u_i^k \qquad (19) $$

However, the implementation requires storing three additional values per contact pair (${}^{k-1}_{i}$, ${}^{k}_{i}$ and u_i^{k-1}) instead of one (${}^{k}_{i}$) for the fixed-point approach. The performance of the secant method is comparable with the one of the fixed-point iteration. The difference between two methods may only be identified in the analysis of the velocity plots, but is not visible in the simulations.

The presented friction handling approach is independent of the time-integration scheme and therefore can be used with other integration schemes, *e.g.* Runge-Kutta. However, the latter require several force computations per simulation time step which slow down the performance.

The high computational efficiency of the proposed algorithm allows interactive simulation of complex scenes with large numbers of interacting objects. The presented examples demonstrate realistic transitions between static and dynamic friction providing superior results as compared to the original approach, without significant increase in the necessary computational resources.

7. Acknowledgements

This work has been performed within the frame of the NCCR Co-Me supported by the SNSF, the EU project PASSPORT ICT-223894, and the BBT project Arthroscopy Simulation funded by CTI.

References

[AP97] ANITESCU M., POTRA F.: Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics 14*, 3 (November 1997), 231–247.

[Bar89] BARAFF D.: Analytical methods for dynamic simulation of nonpenetrating rigid bodies. In *Computer Graphics (Proc. Siggraph)* (1989), vol. 23, pp. 223–232.

[Bar91] BARAFF D.: Coping with friction for non-penetrating rigid body simulation. In *Proc. Computer Graphics and Interactive Techniques* (1991), pp. 31–41.

[Bar94] BARAFF D.: Fast contact force computation for nonpenetrating rigid bodies. In *Proc. Computer Graphics and Interactive Techniques* (1994), pp. 23–34.

[BF00] BURDEN R. L., FAIRES D. J.: *Numerical Analysis*, 7 ed. Brooks Cole, December 2000.

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

[BS06] BENDER J., SCHMITT A.: Constraint-based collision and contact handling using impulses. In *In Proceedings of the 19th international conference on computer animation and social agents (Geneva, Switzerland)* (2006), pp. 3–11.

[CW05] CIRAK F., WEST M.: M.: Decomposition contact response (dcr) for explicit finite element dynamics. *International Journal for Numerical Methods in Engineering 64* (2005).

[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.

[EE06] EBRAHIMI S., EBERHARD P.: A linear complementarity formulation on position level for frictionless impact of planar deformable bodies. *ZAMM 86*, 10 (2006), 807–817.

[EE07] EBRAHIMI S., EBERHARD P.: Frictional impact of planar deformable bodies. 2007, pp. 23–32.

[FBAF08] FAURE F., BARBIER S., ALLARD J., FALIPOU F.: Image-based collision detection and response between arbitrary volume objects. In *2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (July 2008), pp. 155–162.

[GBF03] GUENDELMAN E., BRIDSON R., FEDKIW R.: Nonconvex rigid bodies with stacking. *ACM Transaction on Graphics 22*, 3 (2003), 871–878.

[GOM*06] GALOPPO N., OTADUY M., MECKLENBURG P., GROSS M., LIN M.: Fast simulation of deformable models in contact using dynamic deformation textures. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2006), Eurographics Association, pp. 73–82.

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

[HVS*09] HARMON D., VOUGA E., SMITH B., TAMSTORF R., GRINSPUN E.: Asynchronous contact mechanics. *ACM Trans. Graph. 28*, 3 (2009), 1–12.

[KEP05] KAUFMAN D., EDMUNDS T., PAI D.: Fast frictional dynamics for rigid bodies. *ACM Trans. Graph. 24*, 3 (July 2005), 946–956.

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

[KSJP08] KAUFMAN D., SUEDA S., JAMES D., PAI D.: Staggered projections for frictional contact in multibody systems. In *SIGGRAPH Asia '08: ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), ACM, pp. 1–11.

[MC95] MIRTICH B., CANNY J.: Impulse-based dynamic simulation. In *WAFR: Proceedings of the workshop on Algorithmic foundations of robotics* (Natick, MA, USA, 1995), A. K. Peters, Ltd., pp. 407–418.

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *J. Vis. Comun. Image Represent. 18*, 2 (2007), 109–118.

[OTSG09] OTADUY M., TAMSTORF R., STEINEMANN D., GROSS M.: Implicit contact handling for deformable objects. *Computer Graphics Forum (Proc. of Eurographics) 28*, 2 (apr 2009).

[Per98] PERSSON B.: *Sliding Friction: Physical Principles and Applications*. Springer, 1998.

[PPG04] PAULY M., PAI D., GUIBAS L.: Quasi-rigid objects in contact. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2004), Eurographics Association, pp. 109–119.

[SBT07] SPILLMANN J., BECKER M., TESCHNER M.: Non-iterative computation of contact forces for deformable objects. *Journal of WSCG 15*, 1-3 (2007), 33–40.

[ST96] STEWART D., TRINKLE J.: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering 39*, 15 (1996), 2673–2691.

[Str90] STRONGE W.: Rigid body collisions with friction. *Proceedings: Mathematical and Physical Sciences 431*, 1881 (1990), 169–181.

[THM*03] TESCHNER M., HEIDELBERGER B., MÜLLER M., POMERANTES D., GROSS M. H.: Optimized spatial hashing for collision detection of deformable objects. In *Proc. Vision, Modeling, Visualization* (2003), pp. 47–54.

[TJ08] TWIGG C., JAMES D.: Backward steps in rigid body simulation. *ACM Trans. Graph. 27*, 3 (2008), 1–10.

[WP99] WRIGGERS P., PANATIOTOPOULOS P. (Eds.): *New Developments in Contact Problems*. SpringerWienNewYork, 1999.

[WT06] WEINSTEIN R., TERAN J.: Dynamic simulation of articulated rigid bodies with contact and collision. *IEEE Transactions on Visualization and Computer Graphics 12*, 3 (2006), 365–374. Member-Fedkiw„ Ron.

Appendix A: Force post-correction

Here we briefly present a proof of the validity of the post-correction operator.

Its general form may be written as:

$$\mathcal{COR}(\mathbf{F}_j) \stackrel{\text{def}}{=} a \cdot \mathbf{F}_j - b \cdot \sum_{i=1}^{n} h_{ij}\mathbf{F}_i \tag{20}$$

where the scalars $a \in (0,1)$ and $b \in (0,1)$ are constant.

The relation between coefficients a and b is found from the condition of momentum conservation. The sum of the corrected contact forces over all contact points should be zero, which leads to:

$$\begin{gathered}\sum_{j=1}^{n}\left(a \cdot \mathbf{F}_j - b \cdot \sum_{i=1}^{n} h_{ij}\mathbf{F}_i\right) = \\ a \cdot \sum_{j=1}^{n} \mathbf{F}_j - b \cdot \sum_{i=1}^{n}\left(\sum_{j=1}^{n} h_{ij}\right)\mathbf{F}_i = 0\end{gathered} \tag{21}$$

Taking into account that $\sum_{j=1}^{n} h_{ij} = 1$ and $\sum_{j=1}^{n} \mathbf{F}_j \equiv \sum_{i=1}^{n} \mathbf{F}_i$, we get that $a = b$ to satisfy the momentum conservation.

Although the force post-correction accounts for the non-collinearity of the contact forces, it still has to be consistent with the basic model discussed previously. This means that if the assumption of a flat contact surface is satisfied then the post-correction operator should not modify the force calculated by (3).

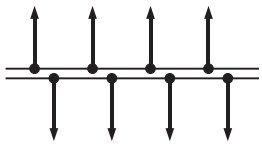


Figure 8: *Force distribution in the ideal case of the flat contact surface*

If we consider the ideal case of an infinite flat contact surface of uniformly discretized objects (Fig.8), it is intuitively clear that $\mathbf{F}_i = -\mathbf{F}_j$, where $j \in T_i$, and $\sum_{i=1}^{n} h_{ij} = \sum_{i=1}^{n} h_{ji} = 1$. Since the post-correction operator should not change the value of the computed force it leads to

$$\mathbf{F}_j = a \cdot \mathbf{F}_j - a \cdot \sum_{i=1}^{n} h_{ij}\mathbf{F}_i \tag{22}$$

Taking into account that $\mathbf{F}_i = -\mathbf{F}_j$ and $\sum_{i=1}^{n} h_{ij} = 1$, we obviously get $a = \frac{1}{2}$. Hence, the resulting form of the force post-correction is

$$\mathcal{COR}(\mathbf{F}_j) = \frac{1}{2} \cdot \left(\mathbf{F}_j - \sum_{i=1}^{n} h_{ij}\mathbf{F}_i\right) \tag{23}$$