An Implicit Finite Element Method for Elastic Solids in Contact

Gentaro Hirota, Susan Fisher, Andrei State, Chris Lee*, Henry Fuchs
Department of Computer Science, University of North Carolina at Chapel Hill
{hirota|sfisher|andrei|fuchs}@cs.unc.edu
*University of Colorado Health Sciences Center
chris@chs.uchsc.edu

Abstract

This work focuses on the simulation of mechanical contact between nonlinearly elastic objects such as the components of the human body. The computation of the reaction forces that act on the contact surfaces (contact forces) is the key for designing a reliable contact handling algorithm. In traditional methods, contact forces are often defined as discontinuous functions of deformation, which leads to poor convergence characteristics. This problem becomes especially serious in areas with complicated self-contact such as skin folds.

We introduce a novel penalty finite element formulation based on the concept of material depth, *the distance between a particle inside an object and the object's boundary. By linearly interpolating pre-computed material depths at node points, contact forces can be* analytically *integrated over contact surfaces without raising computational cost. The* continuity *achieved by this formulation supports an efficient and reliable solution of the nonlinear system.*

This algorithm is implemented as part of our implicit finite element program for static, quasistatic *and* dynamic analysis *of nonlinear viscoelastic solids. We demonstrate its effectiveness on an animation showing realistic effects such as folding skin and sliding contacts of tissues involved in knee flexion. The finite element model of the leg and its internal structures was derived from the Visible Human dataset.*

1. Introduction

When animal and human bodies move, they deform due to mechanical contact between components such as skin, muscles or bones. As body posture changes, organs push and slide against each other, changing the shape of the body. As a joint bends, the skin surface around it may stretch and fold, creating complicated geometry.

Simulation of such phenomena would provide automatic methods to generate the deformation. In animation, this capability could help create believable-looking details of organic bodies, eliminating time-consuming manual intervention [7]. It could also benefit training physicians and applications such as medical image registration, for example between pre-operatively acquired CT or MRI datasets and intra-operative ultrasound or X-ray imagery. Surgical simulation often requires procedure-specific postures, which can be derived by deforming generic models such as the "Visible Human" dataset [36].

A short version of this paper was presented as [18].

2. Previous work

When simulating deformations of elastic objects, one of the major challenges is avoiding penetration of deformable structures. In the following, we describe two major approaches in dealing with this problem.

2.1. Kinematic approaches

Most deformation techniques employed in computer animation use kinematic approaches. Their major advantage is interactive performance due to the relatively small computational cost. Two examples are free-form deformation (FFD) [34] and "skinning" or skeleton subspace deformation (SSD) [23], which is the smooth blending of multiple rigid transformations. FFD and SSD belong to a group of algorithms that employ "space deformation" [5], which can be viewed as a 3D transformation. One can also deform objects by directly moving the control points of surfaces [11].

In these methods, the impenetrability constraint is satisfied by heuristic techniques, often requiring extensive user interaction to produce the desired effects.

Space deformation can be applied to the human body without causing penetration between the organs [7], but sliding effects between organs cannot be obtained with this method.

Many commercial software packages allow animators to embed formulae to express specific needs for deformations [1]. These “deformers” can be written in such a way that penetration between objects is minimized, but only for specific, limited scenarios.

By using pose space deformation [23], one can partially automate the process. For example, a user can “teach” the system to avoid penetration of the skin around an elbow by directly adjusting the skin geometry. Henceforth, the system automatically reduces penetration. The drawback of the method is that it is highly inflexible since the user must instruct the system how to handle every new contact scenario.

2.2. Simulation of physical laws

To overcome the drawbacks of kinematic methods, many techniques employ the notion of force and energy [14, 21]. Wilhelms et. al. simulate a sliding skin layer by relaxation of a spring mesh [39]. The relaxation scheme does not account for buckling, hence realistic folding does not occur.

Accurate physical simulation, which has been studied in computational mechanics, can provide a powerful tool for automatically generating realistic deformations. They have been traditionally very expensive in terms of computation time but are becoming increasingly affordable with the continually growing performance of computer hardware. The basic approach for contact problems is to detect penetration between objects and compute an appropriate response that eliminates, minimizes or reduces penetration. (This is also characteristic of our approach.)

Graphics researchers have demonstrated animations of elastic bodies in contact [2,29,37,40]. As mentioned, the contact problem (i.e., avoiding penetration) has been extensively studied in the engineering community [19]. Solution methods for the contact problem can be categorized by their approach to satisfying the impenetrability constraint. We now discuss the two major approaches.

2.2.1. Hard constraint. Most algorithms utilizing hard constraints use the concept of *slave nodes* and *master surfaces* to define constraints. Consider two colliding objects. Nodal points on the surface mesh of one object are designated as slave nodes, whereas master surfaces are defined from the surface mesh of the second object. If a slave node penetrates a master surface, a constraint to keep the node on the surface is created. A set of constraints is formed by all of the penetrating slave nodes. With each iteration, new surface geometry is obtained by solving this constrained optimization problem, and the set of constraints is updated accordingly. The process repeats until the constraint set becomes stable.

This method suffers from two major drawbacks, the first being that it requires frequent constraint updates. Frequent constraint updates occur when highly tessellated surfaces are in sliding contact. As soon as a slave node travels across one master surface to another, the existing constraint becomes invalid, and a new one must be created. Thus the lifetime of each constraint in the optimization process is very short.

The second drawback is known as the locking problem: the surface becomes artificially stiff due to an excessive number of constraints. Hallquist et al. handle this problem by partitioning the surface of each object into master and slave regions [15]. However, in the presence of evolving self-contact, it becomes extremely difficult to maintain the distinction between these two types of surface elements.

Another method uses heuristic rules to find master-slave pairs based on the history of movements, and computes the exact time of collision, [8,16]. The two colliding objects are treated symmetrically. In this method, even if the collision times are computed, some penetrations are tolerated for several time steps before they are eliminated. To completely prevent penetrations, the exact birth and death times of impenetrability constraints for slave-master pairs must be computed at every step. Such computation is prohibitively expensive.

Belytschko et al. [6] proposed the “splitting pinball” method, which uses repulsive forces between hierarchical bounding spheres around individual surface elements. Although interference checks between spheres are very efficient, the algorithm’s applicability to complex contact is not clear. The fine details of a surface such as folding skin would require that the bounding spheres be repeatedly subdivided, resulting in excessive computational cost.

Ideally, if two objects are in contact, they share a single contact surface. However, because it is virtually impossible for two independently discretized surfaces to have common surface geometry, the methods discussed above cannot prevent small penetrations or gaps. The frequency of constraint updates is also directly related to the resolution of surface discretization, and can therefore lead to very high computational cost.

2.2.2. Penalty methods. By definition, penalty methods allow small amounts of penetration to occur. To resolve penetration, penalty forces proportional to the depth of penetration are calculated [3,9,30]. Unlike constraint methods, a slave node is allowed to travel without keeping track of all master surfaces on its path. Thus, the effect of the resolution of surface discretization on computational cost is not as dramatic as with constraint methods.

2.2.3. Gap function computation. Most conventional methods (including constraint methods) seek projections to evaluate the *gap function* (i.e. the negative depth of penetration) and its derivative [12]. Because of the geometric complexity of a projection search, these methods are not appropriate for handling complicated boundary surfaces.

Most methods (except for the pinball method) examine penetration at slave nodes only. Because of their "point sampling" nature, the contact force applied on a slave node becomes a discontinuous function of deformation (i.e. of node movements), which often causes a convergence problem.

In section 5, we introduce a robust method to compute a continuous gap function for even very complex surface geometry. We also show that, based on our gap function, contact penalty forces can be analytically integrated as continuous functions.

3. Static analysis

Due to their low mass density and fairly low viscosity, biological tissues tend to rapidly converge to a final state when subjected to external forces. For example, it is hard to flex a finger or change facial expression quickly enough to be able to observe viscosity or inertia effects such as creep and oscillations. In fact, for many applications (including animation), the user is only interested in the (static) equilibrium shapes of flexible tissues for a given posture or other specified constraints. In animation applications, dynamic postures and constraints can be used to steer the animation. Static analysis is adequate for these applications, and, hence methods optimized for the static problem must be developed.

In static analysis, the geometry of an elastic object depends solely on the forces applied to the object. The relationship between geometry and forces is described by a differential equation defined on the continuous domain of the elastic object.

We assume the resting (undeformed) shape of the object is known, and elect it as a *reference configuration.* The current (deformed) shape is referred to as the *current configuration.*

At the equilibrium state, an elastic object in contact satisfies the following equation:

$$\int_{\Omega} (\operatorname{div} \boldsymbol{T} + \boldsymbol{f}) \cdot \delta\boldsymbol{u}\, dV + \int_{\Gamma} p\boldsymbol{n} \cdot \delta\boldsymbol{u}\, da = 0 \quad \forall \delta\boldsymbol{u} \qquad (1)$$

In this equation, Ω is the interior of the object, Γ is its boundary, $\boldsymbol{T}$ is the first Piola-Kirchhoff stress tensor, div is the divergence operator, $\boldsymbol{f}$ is the density of body forces (such as gravity), $\boldsymbol{u}$ is the displacement of particles, $\delta\boldsymbol{u}$ is an arbitrary variation of $\boldsymbol{u}$, dV is the differential volume in the reference configuration, p is the pressure on the contact surface, $\boldsymbol{n}$ is the normal of the contact surface, da is the differential area in the current configuration, and $p\boldsymbol{n}$ is the surface traction force on the contact. Since we assume frictionless contact, the traction force is normal to the surface. $p\boldsymbol{n}$ is integrated over the boundary. (Actually, its value is non-zero only on the contact area.)

3.1. Discretization

Because of its complex boundary conditions and nonlinearity, Eqn. (1) cannot be solved analytically. Instead, it is discretized using a finite element method, and an approximate solution is sought [22].

We use tetrahedral elements for the interior and triangular elements for the boundary of objects. The triangular elements are chosen to be a subset of the sides of the tetrahedral elements.

The displacements of *particles* (internal material points) are obtained by linearly interpolating displacements at nodes. (The interpolation functions are called *shape functions*.) Elastic forces at nodes are computed by substituting the virtual displacement $\delta\boldsymbol{u}$ with the corresponding shape functions.

The derivatives for all forces must also be computed to construct a stiffness matrix, which is crucial for Newton iteration (described in subsection 3.2).

The details of the contact force computation are explained in section 5.

As a result of discretization, we obtain a nonlinear equation of the form:

$$\mathrm{R}(\bar{\boldsymbol{u}}) = \boldsymbol{0} \qquad (2)$$

Here, $\bar{\boldsymbol{u}}$ represents the displacement vector $(u_{1,1},\ u_{1,2},\ u_{1,3}, \ldots, u_{n,1},\ u_{n,2}, u_{n,3})$, where $(u_{i,1}, u_{i,2}, u_{i,3})$ denotes the displacement of the ith node and n is the number of nodes. We can impose boundary conditions for displacement by assigning fixed values to the components of $\bar{\boldsymbol{u}}$.

3.2. Solution of the nonlinear system

By solving Eqn. (2), we can obtain a new shape of the elastic object as the displacement vector $\bar{\boldsymbol{u}}$. There are three factors that contribute to the nonlinearity of this system:

- **Finite deformation:** To handle finite deformation (as opposed to infinitesimal deformation), the force-displacement relationship must be described by nonlinear equations (geometric nonlinearity).
- **Nonlinear material:** Realistic elastic materials are all nonlinear. The choice of materials is explained in subsection 3.3.
- **Collision and contact of objects:** Collision and contact are events that introduce additional nonlinearity into the system.

On top of the nonlinearity, the large size of the system increases the complexity; a typical finite element analysis of our interest produces a system with tens of thousands of variables. We have developed a robust and efficient algorithm to solve large nonlinear systems.

3.2.1. Selecting a search direction. Nearly all solution methods start with an initial guess and proceed in a given search direction in a step-by-step manner. Finding the proper direction in the high-dimensional search space is critical for the algorithm's efficiency. Several different methods exist to determine the best search direction.

The maximum gradient descent method chooses the direction of the forces (i.e. the negative of the residual $\mathrm{R}(\bar{\boldsymbol{u}})$) for each step. This method turns out to be very slow because it takes many steps for a local force to propagate through the entire mesh. Furthermore it is impossible for this method to predict rapid force changes caused by the deformation of objects.

On the other hand, the Newton method uses the derivative of the forces (i.e. the stiffness matrix), which provides information about how the forces vary as a function of deformation. Each Newton step consists of computation of the residual, computation of the stiffness matrix, and solution of a linear system. The process continues until the residual drops below a given tolerance. We chose this method due to both its speed and reliability.

3.2.2. Avoiding illegal steps. Two questions that remain to be answered are: how to obtain the initial guess and how far to proceed in a selected search direction.

If there are no displacement boundary conditions, the object is deformed only by external forces. In this case, an obvious choice of the initial guess is the initial shape, i.e. $\bar{\boldsymbol{u}} = \boldsymbol{0}$.

But if displacement boundary conditions are specified, some components of $\bar{\boldsymbol{u}}$ are externally given. In such cases $\bar{\boldsymbol{u}}$ initially contains zeros for free components and final values for constrained components. The corresponding mesh may contain a tetrahedral element whose orientation is reversed, resulting in an illegal configuration. For this reason, constrained components must be gradually incremented, and as soon as an illegal configuration is detected, the step size must be reduced (adaptive incremental loading). Each incremental step performs a Newton iteration and the solution is cascaded into the next step. After the second step, the initial guess is computed by extrapolating the previous two solutions (two-point predictor).

Given an initial guess, we must determine how far to proceed in a given search direction. If $\mathrm{R}(\bar{\boldsymbol{u}})$ is smooth, and the initial guess is close to the solution, a full step towards the solution of the linear system can safely be taken. However, the nonlinearity of the system often causes a full step to lead to divergence. A full Newton step can also bring the mesh to an illegal configuration. Our method checks if the full Newton step is legal and if the residual decreases. If both of these are true, the step is taken. If not, the step size is halved until the two conditions are satisfied. The scaling value of the step size is called a *damping* factor. This line search strategy greatly improves the robustness of our method.

The resulting algorithm can be summarized as three nested loops:

```
LOOP1: Adaptive Incremental Loading
    2-Point Prediction
    LOOP2: Newton Iteration
        Linear System Construction
        Linear System Solution
        LOOP3: Line Search
```

3.2.3. Linear system solution. Newton iteration depends on the stable solution of linear systems. The accuracy of the solution is traded for speed. Since the degree of nonlinearity of the equation is high, the residual is not greatly reduced by a single Newton step. Consequently, computing an exact solution for each linear system does not improve the rate of convergence. For this reason, an iterative method is better suited for the linear system solution than are direct methods.

The linear systems constructed in our finite element method are symmetric, sparse, and usually positive definite. Around a bifurcation point, however, the matrix of the system is sometimes not only indefinite but also nearly singular. The (bi)conjugate gradient method tends to behave in an erratic manner. We found that an implementation of the Generalized Minimum Residual method (GMRES) with diagonal preconditioning [33] is both efficient and stable. The iteration is terminated when either the residual reduces to one tenth or a predetermined iteration limit is reached. This strategy keeps the computation time for solving the linear systems relatively low without slowing down the convergence of the Newton iteration.

We use another technique to prevent failure in the linear system solver. Each node is assigned a "viscosity" proportional to the sizes of the surrounding elements. A well-chosen value for viscosity gives the algorithm the tendency to pick an energy-minimizing solution at a bifurcation point, i.e. to prefer a stable equilibrium point. Also, this method allows starting with an obviously ill-conditioned initial configuration such as an object placed in mid-air, which will fall until it hits the ground.

3.3. Material models

The property of an elastic material is defined as the relationship between stress and strain (Constitutive Law). In Eqn. (1), the relationship provides a formula that associates the stress tensor $\boldsymbol{T}$ with the displacement $\boldsymbol{u}$. A few quantities must be introduced to describe the relationship: the deformation gradient $\boldsymbol{F} = \nabla \boldsymbol{u} + \boldsymbol{Id}$ ($\boldsymbol{Id}$ denotes the identity matrix), the right Cauchy-Green strain tensor $\boldsymbol{C} = \boldsymbol{F}^T \boldsymbol{F}$, the *invariants* of $\boldsymbol{C}$ defined as $I_1 = tr(\boldsymbol{C})$, $I_2 = \frac{1}{2}\{tr(\boldsymbol{C})^2 - tr(\boldsymbol{C}^2)\}$, $I_3 = \det(\boldsymbol{C})$, and finally the stored energy function $\psi(\boldsymbol{F})$. The stress tensor $\boldsymbol{T}$ is given as $\boldsymbol{T} = \partial\psi(\boldsymbol{F})/\partial\boldsymbol{F}$, thus $\psi(\boldsymbol{F})$ actually determines the material property.

The properties of organic tissues are being actively studied in biomechanics and several models have been proposed based on stress-strain data obtained from in vivo and in vitro experiments. However, due to the limitations of measurement technology, those models have not been rigorously validated [27].

We have implemented compressible variations of three different material models. The first model is the Mooney-Rivlin material [10,28]

$$\psi(\boldsymbol{F}) = C_1(I_1 - 3) + C_2(I_2 - 3) + a(I_3 - 1) - (C_1 + 2C_2 + a)\ln I_3.$$

C_1 and C_2 are constants to control stiffness. a determines compressibility. This model exhibits a relatively linear strain-stress curve.

Biological tissues are often characterized by higher nonlinearity; the stiffness (modulus) dramatically increases as they are stretched. The second model we use, the Veronda material, expresses this nonlinearity with an exponential function [31,38]

$$\psi(\boldsymbol{F}) = 2\alpha \exp(I_1 - 3) + \alpha\beta(I_2 - 3) + aI_3 - a\ln I_3,$$

where α controls overall stiffness and β governs the rate of stiffness increase.

In addition to the nonlinearity, many tissues show anisotropy due to their microscopic fiber structures [17,20]. We use the fiber-reinforcement model

$$\lambda_f{}^2 = \boldsymbol{f}^T \boldsymbol{C}\, \boldsymbol{f} \quad \text{and}$$

$$\psi(\boldsymbol{F}) = \alpha \exp(\beta(\lambda_f{}^2 - 1)) - \alpha\beta\lambda_f{}^2,$$

where λ_f is the fiber stretch along the fiber direction $\boldsymbol{f}$, and α and β are material constants similar to the ones in the Veronda model. We superimpose this model onto the Veronda or Mooney-Rivlin materials for reinforcement rather than using it alone.

In all the models we use, the energy tends to infinity as the volume compression ratio I_3 tends to zero. This is an essential property for preventing the element reversal phenomenon. Nearly incompressible versions of the above models will also be implemented.

4. Dynamic and quasistatic analysis

We extended our method to quasistatic (i.e. inertia ignored) and dynamic analysis.

The equilibrium equation now includes inertial and viscous forces. Time derivatives are discretized by the implicit Euler scheme [3,37]. Each time step becomes a Newton iteration (see `LOOP2` in section 3.3), which solves for velocities of node points. The velocities are then used to update node positions.

We also implemented an explicit method. The explicit time integration is more appropriate than implicit one in high-speed applications such as crash/impact problems, for which time steps must be chosen so small that stress does not propagate far in a step [24,29].

5. Contact problem

5.1. Gap function

The gap function plays a crucial role in describing the contact relationship between objects. Fig. 1 illustrates an object at the reference or undeformed configuration (Fig. 1, left) and the current or deformed configurations (Fig. 1, center and right). The rightmost configuration exhibits

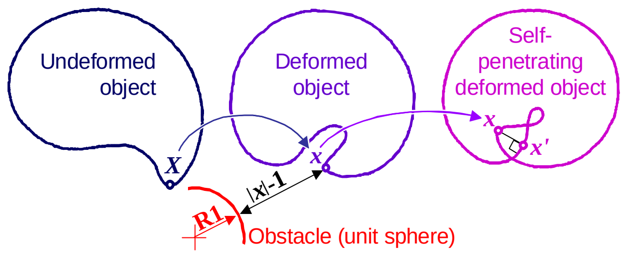

Fig.1. Gap function.

self-penetration. The gap function $g(\boldsymbol{X},\boldsymbol{x})$ is the distance between a particle $\boldsymbol{X}$ (coordinate at the reference configuration, used to label the particle) on a boundary of an object at $\boldsymbol{x}$ (coordinate at the current configuration) and an obstacle. For example, if the obstacle is a unit sphere located at the origin (as shown for the center configuration in Fig. 1), then $g(\boldsymbol{X},\boldsymbol{x}) = |\boldsymbol{x}| - 1$.

The sign of $g(\boldsymbol{X},\boldsymbol{x})$ is positive if $\boldsymbol{x}$ is outside the obstacle. In this simple example, $g(\boldsymbol{X},\boldsymbol{x})$ depends solely on $\boldsymbol{x}$. We consider penetration between deformable objects; therefore multiple particles may share the same location $\boldsymbol{x}$. The parameter $\boldsymbol{X}$ identifies the particle under examination.

Self penetration may occur (Fig. 1, right configuration), i.e. the obstacle may be the object itself; hence, $g(\boldsymbol{X},\boldsymbol{x})$ cannot be defined as the minimum distance, which is always zero for particles on the boundary. Instead of the minimum distance, conventional methods use the distance from $\boldsymbol{x}$ to its projections on object boundaries. A projection of $\boldsymbol{x}$ on a surface is a point $\boldsymbol{x}'$ such that $\boldsymbol{x}-\boldsymbol{x}'$ is normal to the surface (Fig. 1, right configuration). By the definition of projection, the search algorithm has to rely on the normals of the surfaces. This leads to three major problems:

1) Numerous candidates
2) Plurality of projections
3) Discontinuity with respect to deformation

First, if a surface has high curvature, the algorithm has to check many candidate projections (Fig. 2). This case often occurs when buckling creates wrinkles on a surface. It is not even obvious that one of the candidates can be selected (e.g. the closest one) because a particle $\boldsymbol{x}$ may intrude into multiple objects. When multi-object intrusion occurs, multiple projections should be used (Fig. 3). Thus, to find projections, a global and exhaustive search algorithm is required. Furthermore, as a surface deforms (i.e. as node displacements change), a projection can emerge or disappear abruptly (Fig. 4), in which case the gap function is not a continuous function of the node displacements. This discontinuity undermines the convergence of the Newton iteration, which requires the first and second derivatives of the gap function.

These three problems become more significant as the penetration depth increases. Solution steps mentioned in section 3 may bring objects to such a configuration. Furthermore, it is convenient for modeling if the initial geometry is allowed to have deep penetration. Therefore, the ability to recover from deep penetration greatly improves the robustness and versatility of an algorithm.

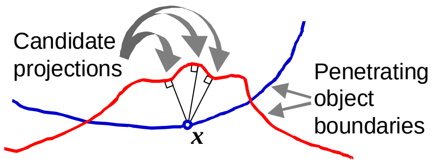

Fig. 2. Multiple projection candidates.

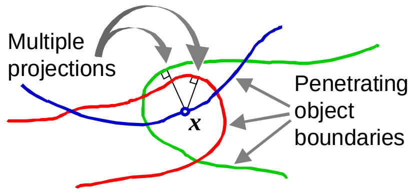

Fig. 3. Projections on multiple objects.

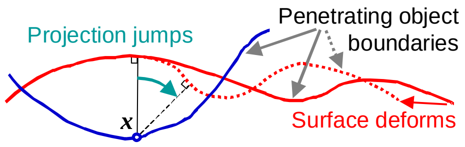

Fig. 4. Abruptly emerging and disappearing projections.

5.2. Material depth

We introduce *material depth* as a new way to compute the gap function. Fig. 5 explains this notion. The deformation maps particle $\boldsymbol{X}$ on a boundary to the current position $\boldsymbol{x}$. As a result, $\boldsymbol{X}$ collides with particle $\boldsymbol{Y}$. The material depth is the distance from $\boldsymbol{Y}$ to the boundary at the reference configuration. It maintains a constant intrinsic value for a particle (or material point), hence the term "material depth." Since there is no self-penetration at the reference configuration, the material depth can be computed regardless of self-penetration at the current configuration. Also, unlike the distance to a projection, material depth never changes abruptly due to deformation.

Material depth is an approximation of the distance fields at the current configuration. As an object is

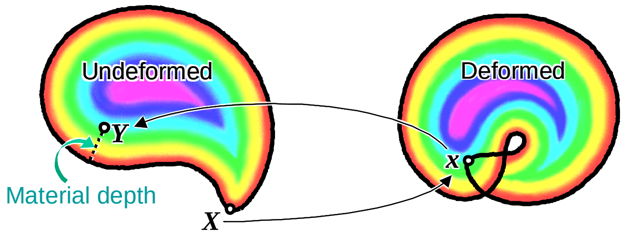

Fig. 5. Material depth (encoded as color values).

stretched and compressed, the value deviates from the actual distance. The inaccuracy does not cause a significant problem since in practice a large penalty factor can be used without causing numerical instability [9].

5.3. Integrating penalty forces

As mentioned above, the material depth is substituted for the gap function $\mathrm{g}(\boldsymbol{X},\boldsymbol{x})$. The contact force $p\boldsymbol{n}$ in Eqn. (1) is approximated by the penalty force:

$$p\boldsymbol{n} = \varepsilon \mathrm{g}(\boldsymbol{X},\boldsymbol{x})\boldsymbol{n}.$$

ε is the penalty factor. $\boldsymbol{n}$ is a normal at $\boldsymbol{x}$ and is the gradient of g. Therefore,

$$p\boldsymbol{n} = \varepsilon \mathrm{g}(\boldsymbol{X},\boldsymbol{x})\frac{\partial \mathrm{g}}{\partial \mathrm{x}}(\boldsymbol{X},\boldsymbol{x}).$$

To use the finite element method, $p\boldsymbol{n}$ must be integrated over a contact surface. Our method does not compute the exact value of the material depth at every point on a penetrating boundary. Instead, it uses a linear interpolation of the material depths at nodes, which can be pre-computed. The resulting method requires only collision detection between triangles and tetrahedra and uses analytical integration. The algorithm performs the following steps:

1) Compute the material depth for each node at the reference configuration.
2) Find the collision between a boundary element (triangle) and a volumetric element (tetrahedron).
3) Compute the intersection of the triangle and the tetrahedron.
4) Triangulate the intersection polygon.
5) For each triangle, integrate the penalty force and its derivatives over the area of the triangle and add their contributions to the global residual and the stiffness matrix.

The first 2 steps are briefly discussed in subsections 5.4 and 5.5. Steps 3-5 are explained in greater detail in the following. As shown in Fig. 6 (left), a tetrahedral element with nodes $\boldsymbol{P}_1 \sim \boldsymbol{P}_4$ and a triangular element $\boldsymbol{Q}_1 \sim \boldsymbol{Q}_4$ are intersecting. The intersection is a convex polygon with up to 7 vertices. To facilitate integration, the intersecting region is divided into triangles (Fig. 6, right). There are up to 5 triangles to be considered and they are all individually processed. Let $\boldsymbol{R}_1 \sim \boldsymbol{R}_3$ denote the vertices of one such triangle. We define a penalty potential energy W for the triangle:

$$W = \int_{triangle} \frac{\varepsilon}{2} g^2 \mathrm{d}a.$$

Using a parameterization s and t on the triangle, we obtain:

$$\begin{aligned} W &= 2A\int_0^1\int_0^{1-t} \frac{\varepsilon}{2} g^2 \mathrm{d}s\mathrm{d}t \\ &= \varepsilon A\int_0^1\int_0^{1-t} (G_1+(G_2-G_1)s+(G_3-G_1)t)^2 \mathrm{d}s\mathrm{d}t, \qquad (3) \end{aligned}$$

where

$$A = \left|(\boldsymbol{R}_2-\boldsymbol{R}_1)\times(\boldsymbol{R}_3-\boldsymbol{R}_1)\right|/2$$

is the area of the triangle and $G_1 \sim G_3$ are the material depths at the triangle's vertices.

$G_1 \sim G_3$ are functions of $\boldsymbol{P}_1 \sim \boldsymbol{P}_4$ and $\boldsymbol{R}_1 \sim \boldsymbol{R}_3$, and $g_1 \sim g_4$ are material depths at $\boldsymbol{P}_1 \sim \boldsymbol{P}_4$. For example, G_1 is obtained by solving the linear system

$$\begin{aligned} G_1 &= l_1 g_1 + l_2 g_2 + l_3 g_3 + l_4 g_4 \\ \boldsymbol{R}_1 &= l_1 \boldsymbol{P}_1 + l_2 \boldsymbol{P}_2 + l_3 \boldsymbol{P}_3 + l_4 \boldsymbol{P}_4 \\ l_1 + l_2 &+ l_3 + l_4 = 1, \end{aligned}$$

where $l_1 \sim l_4$ are the barycentric coordinates of $\boldsymbol{R}_1$ inside the tetrahedron.

The integration (3) is performed analytically.

$\boldsymbol{P}_1 \sim \boldsymbol{P}_4$ and $\boldsymbol{Q}_1 \sim \boldsymbol{Q}_4$ are node positions and thus directly related to the displacement $\bar{\boldsymbol{u}}$. The penalty force and its derivatives are $\partial W/\partial \bar{\boldsymbol{u}}$ and $\partial^2 W/\partial \bar{\boldsymbol{u}}^2$ respectively. These derivatives are computed by numerically applying the chain rule. The dependency of variables and constants is shown in Fig. 7.

The area of the intersection

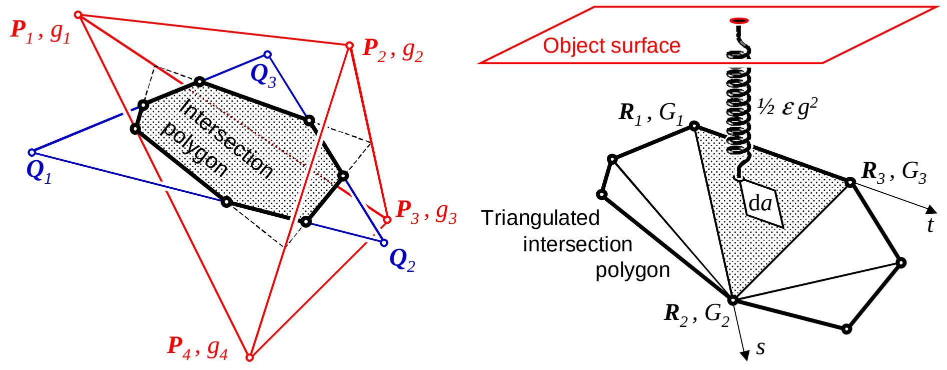

Fig. 6. Intersecting elements.

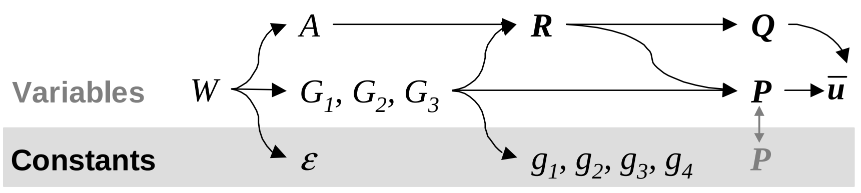

Fig. 7. Dependency graph.

varies continuously. The material depth is also continuous. Thus the penalty force is mostly a smooth function (C^1) of $\bar{\boldsymbol{u}}$. The only exception is the case when an edge of a triangle and a side of the tetrahedron are coplanar, in which case the penalty force is no longer smooth. It is still continuous (C^0) however, which is crucial for equation solving, since a solution may not even exist without continuous penalty forces. This is the major advantage of our method over traditional methods.

5.4. Initial depth computation

For an object with few triangles, an exhaustive search algorithm is sufficient for pre-computation of material depths at node points. For more complex objects, we use the fast marching level set method, which quickly computes distance values [35].

By utilizing the tetrahedral mesh, we could use the finite element version of the fast marching method [4], which can compute distances even for self-intersecting objects. This approach is left for future work.

5.5. Collision detection

To accelerate collision detection between tetrahedral and triangular elements, a bounding volume tree is constructed for the tetrahedral mesh. A node of the tree represents an X, Y, or Z coordinate interval that bounds the intervals of all descendant nodes. The interval of a leaf node contains a tetrahedral element. The overlaps between the intervals of each triangle element and the intervals of tree nodes are examined and possible collisions are quickly determined.

The tree structure is built by top-down partitioning of elements in the direction of their greatest extent. We divide a simulation into smaller simulation runs. Since we set up our simulation such that the mesh does not deform much in a single run, the tree structure is built only once at the beginning of each run. The interval values are efficiently updated in a bottom-up manner at every solution step. As a result, the collision detection occupies only a minor part of the total computation time (see next section).

6. Results

We simulated flexion of a human knee joint using a finite element model of a right human leg (Fig. 8). To build the model, we first generated boundary polygons of all the organs from a manually segmented mask image volume of the Visible Human Male. The Visualization Tool Kit [32] and Maya [1] were used to extract, smooth, decimate, and assemble the polygons. Then a tetrahedral mesh was generated from the polygonal boundaries using SolidMesh [26]. The mesh contains about 10,000 nodes, 10,000 triangular elements, and 40,000 tetrahedral elements. It consists of a femur (thigh bone), a patella (knee cap), a tibia (shin bone), a quadriceps (a collection of four major anterior thigh muscles), a patella ligament, tendons that connect the patella and the quadriceps, and a monolithic skin-fat layer (Fig. 9). Our model is not anatomically complete since it lacks other major muscles and many important ligaments at the knee joint. They are included either in the skin-fat layer or are part of a hollow space around the knee joint. All the boundaries are treated as frictionless interfaces except for the inner part of the tibia, which is attached to the skin-fat layer. Various material parameters are assigned to tetrahedral elements in order to approximate the mechanical properties of different parts. The Mooney-Rivlin and Veronda models were both applied, but images shown in this paper were obtained by using the Mooney-Rivlin model only.

The femur is fixed in space. The cross section of the thigh is constrained so that it can only move on the cutting plane. The tibia is rotated around an axis in the knee joint. These positional constraints constitute a displacement boundary condition. The tibia's total 150-degree rotation was divided into 50 three-degree intervals and the algorithm was applied to each interval to generate deformations.

The complete simulation took 376 minutes on a single 300MHz R12000 CPU of an SGI Onyx system. Most of the time (63%) was consumed by the force and stiffness matrix computations. 22% were spent on collision detection, out of which the bounding volume tree construction took less than 1%. The rest, 15%, were spent on the linear system solution.

Figures 10 through 13 show more images of our results, including dynamic analysis (Fig. 13). Additional examples and animations can be found at `http://www.cs.unc.edu/~us/fem/`.

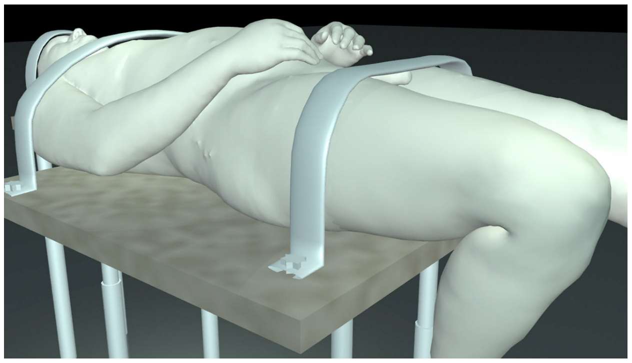

Fig. 8. Visible Human dataset with flexed knee

7. Conclusions

We have addressed the frictionless contact problem for elastic objects. Our main contribution is a novel penalty finite element method that uses material depth for evaluating gap functions and their derivatives. Unlike projection-based gap functions used in traditional methods, our gap function varies continuously as objects deform. The field of material depth is approximated by a linear interpolation of depth values at finite element nodes. This simplification enables efficient analytical integration of contact penalty forces over the contact area and thus results in penalty forces that are continuous functions of deformation. The achieved continuity reduces the oscillation and divergence problems often present in traditional approaches.

Contact problems demand the solution of a large-scale highly nonlinear system. We developed a reliably converging solver that integrates various numerical techniques such as Newton iteration, adaptive incremental loading, two-point predictor, line search (or variable damping factor), and quasi-viscosity.

We have demonstrated the performance of our method by simulating very large deformations on part of a human anatomical model. To our knowledge, this is the first demonstrated simulation of large-scale motion of a complex model derived from the widely used Visible Human dataset and encompassing multiple tissue types including bone, muscle, tendons, and skin.

8. Future work

This work is limited to frictionless contact. This limitation is justified because the friction between organs inside bodies is known to be small [25]. This is not the case for friction between (non-lubricated) skin surfaces, an area that should be investigated. The residual stress (such as skin tension) and atmospheric pressure should also be considered in order to improve the accuracy of the simulation.

Chemical and biophysical phenomena contribute to internal stresses. Muscle contraction is the most dramatic example for this. Such stresses should be included as part of the external body forces ($\boldsymbol{f}$ in Eqn. 1) to simulate “active” aspects of biological tissues.

To handle more complex anatomical models, a further performance improvement is desirable. Since most computation is local to each finite element, parallelization techniques should enable significant acceleration of our algorithm [8].

We hope that methods such as the ones described here will lower the cost of deforming complex anatomical models, making possible a wide variety of applications for which currently available techniques have been prohibitively expensive.

9. References

[1] Anderson, P., et al, Using Maya. Alias|Wavefront, 1999.

[2] Baraff, D., and Witkin, A., Dynamic simulation of non-penetrating flexible bodies. Proceedings of SIGGRAPH 92, Computer Graphics, 26, 2 (July 1992), pp. 303-308.

[3] Baraff, D., and Witkin, A., Large steps in cloth simulation. Proceedings of SIGGRAPH 98, Computer Graphics Proceedings, Annual Conference Series, 1998, pp. 43-54.

[4] Barth, T.J., and Sethian, J.A., Numerical schemes for the Hamilton-Jacobi and level set equations on triangulated domains. J. Computational Physics 145(1), pp. 1-40, September 1998.

[5] Bechmann, D., Space deformation models survey. Computer & Graphics, 18(4), pp. 571-586, 1994.

[6] Belytschko, T., Yeh I.-S., The splitting pinball method for general contact. In R.Glowinksi (ed.), Computing Methods in Applied Science and Engineering. Nova Science Publishers, New York, NY, 1992.

[7] Bregler, C. (chair), Hollow Men and Invisible People - Layers of a digital actor. SIGGRAPH 2000 Technical Sketch.

[8] Brown, K., Attaway, S., Plimpton, S., Hendrickson, B., Parallel strategies for crash and impact simulations. Computer methods in applied mechanics and engineering 184, pp. 375-390, 2000.

[9] Carstensen, C., Scherf, O., and Wriggers, P., Adaptive finite elements for elastic bodies in contact. SIAM J. Scientific Computing 20(5), pp. 1605-1626, 1999.

[10] Ciarlet., P.G., Mathematical Elasticity. North-Holland, 1988.

[11] DeRose, T., Kass, M., Truong, T., Subdivision surfaces in character animation. Proceedings of SIGGRAPH 98, Computer Graphics Proceedings, Annual Conference Series, 1998, pp. 85-94.

[12] Donzelli, P.S., A mixed-penalty contact finite element formulation for biphasic soft tissues, PhD Thesis, Dept. of Mech. Eng., Aeronautical Eng. and Mechanics, RPI, Troy, NY, 1995.

[13] Fung, Y.C., Biomechanics. Springer-Verlag, 1993.

[14] Gourret, J.-P., Thalmann, N.M., Thalmann, D., Simulation of object and human skin deformations in a grasping task. Proc. of SIGGRAPH 89, Computer Graphics, 23, 4 (Aug. 1989), pp. 21-30.

[15] Hallquist, J.O., Goudreau, G.L., and Benson, D.J., Sliding interface with contact-impact in large scale Lagrangian computation, 1987.

[16] Heinstein, M.W., Attaway, S.W., Swegle, J.W., Mello, F.J., A general contact detection algorithm for finite element analysis. In Aliabadi, M. H., Brebbia, C. A. (eds.), Contact Mechanics. Computational Mechanics Publications, Southampton, UK, 1993.

[17] Hirokawa, S., Tsuruno, R., Three-dimensional deformation and stress distribution in an analytical / computational model of the anterior cruciate ligament. Journal of Biomechanics 33 (2000), 1069-1077.

[18] Hirota, G., Fisher, S., State, A., Fuchs, H., Lee, C., Simulation of Deforming Elastic Solids in Contact. SIGGRAPH 2001 Technical Sketch. In SIGGRAPH 2001 Conference Abstracts and Applications, p. 259.

[19] Kikuchi, N., Oden, J.T., Contact Problems in Elasticity: A study of variational inequalities and finite element methods. SIAM Studies in Applied and Numerical Methematics, 1988.

[20] Klisch, S.M., Lotz, J.C., Application of a fiber-reinforced continuum theory to multiple deformations of the annulus fibrosus. Journal of Biomechanics 32 (1999) 1027-1036.

[21] Koch, R. M., Gross, M. H., Carls, F. R., von Büren, D.F., Fankhauser, G., Parish, Y.I.H., Simulating facial surgery using finite element methods. proceedings of SIGGRAPH '96, Computer Graphics Proceedings, Annual Conference Series, 1996, pp. 421-428.

[22] Le Tallec, P., Numerical methods for solids. In Ciarlet, P.G., Lions, J. L. (eds.), Handbook of Numerical Analysis. North-Holland, 1994.

[23] Lewis, J.P., Cordner, M., Fong, N., Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. Proceedings of SIGGRAPH 2000, Computer Graphics Proceedings, Annual Conference Series, 2000, pp. 165-172.

[24] Lin J.I., DYNA3D: A nonlinear, explicit, three-dimensional finite element code for solid and structural mechanics, User manual, Methods Development Group, Lawrence Livermore National Laboratory, December 1998.

[25] Malcolm, L.L., An experimental investigation of the frictional and deformational response of articular cartilage interfaces to statistic and dynamic loading. PhD thesis, Univ. of California San Diego, 1976.

[26] Marcum, D.L., and Weatherill, N.P., unstructured grid generation using iterative point insertion and local reconnection. AIAA Journal, 33(9), September 1995, pp. 1619-1625.

[27] Miller, K., Chinzei, K., Orssengo, G., Bednarz, P., Mechanical properties of brain tissue in-vivo: experiment and computer simulation. Journal of Biomechanics 33 (2000), pp. 1369-1376.

[28] Mooney, M., A theory of large elastic deformation. Journal of Applied Physics. 11 (1940), pp. 582-592.

[29] O'Brien J.F., Hodgins J.K. Graphical modeling and animation of brittle fracture. Proceedings of SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, 1999.

[30] Papadopoulos, P., and Taylor, R.L., A simple algorithm for three-dimensional finite element analysis of contact problems. Computers & Structures 46(6), pp. 1107-1118, 1993.

[31] Pioletti, D.P., Rakotomanana, L.R., Benvenuti, J.F., Leyvraz P.F., Viscoelastic constitutive law in large deformations: application to human knee ligaments and tendons. Journal of Biomechanics 31 (1998), pp. 753-757.

[32] Schroeder, W., Martin, K., Lorensen, B., The visualization toolkit: an object-oriented approach to 3-d graphics. Prentice-Hall, 1997.

[33] Seager, M., A SLAP for the masses. Lawrence Livermore Nat. Laboratory Technical Report, UCRL-100267, December 1988.

[34] Sederberg, T.W., and Parry, S.R., Free-form deformation of solid geometric models. Proceedings of SIGGRAPH 86, Computer Graphics, 20, 4 (August 1986), pp. 151-160.

[35] Sethian, J.A., Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge Univ. Press, 1999.

[36] Spitzer, V., Ackerman, M.J., Scherzinger, A.L., Whitlock, D., The visible human male: a technical report. J. Am. Med. Inform. Assoc. 3(2), Mar-Apr 1996, pp. 118-130.

[37] Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K., Elastically deformable models. Proc. of SIGGRAPH'87, Computer Graphics, 21, 4 (Aug. 1987), pp. 205-214.

[38] Veronda, D.R., Westmann, R.A., Mechanical characterization of skin-finite deformation. Journal of Biomechanics 3 (1970), pp. 111-124.

[39] Wilhelms, J., and Van Gelder, A., Anatomically based modeling. Proceedings of SIGGRAPH 97, Computer Graphics Proceedings, Annual Conference Series, 1997, pp. 173-180.

[40] Zhuang, Y., Real-time simulation of physically-realistic global deformations. PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 2000.

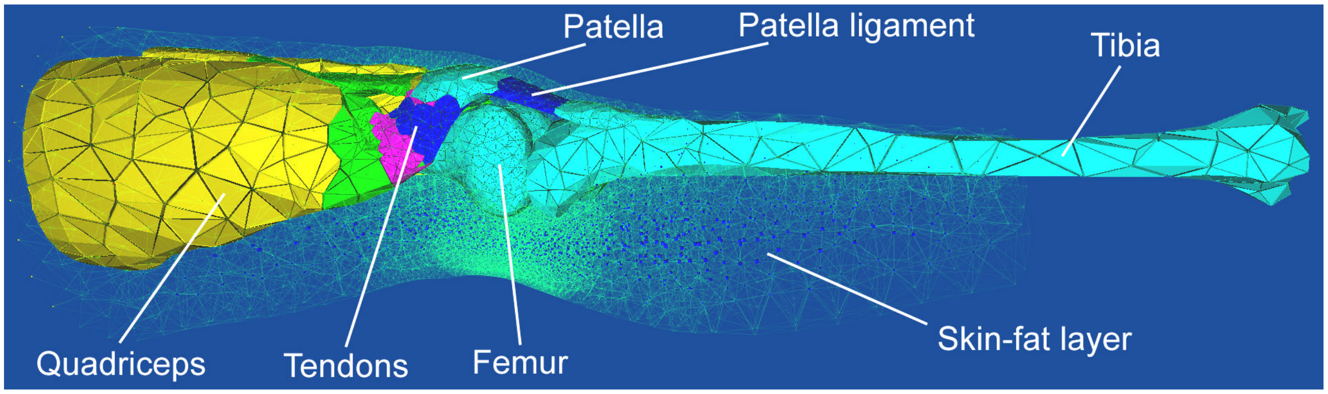

Fig. 9. The constituent parts of the leg model, derived from the Visible Human database

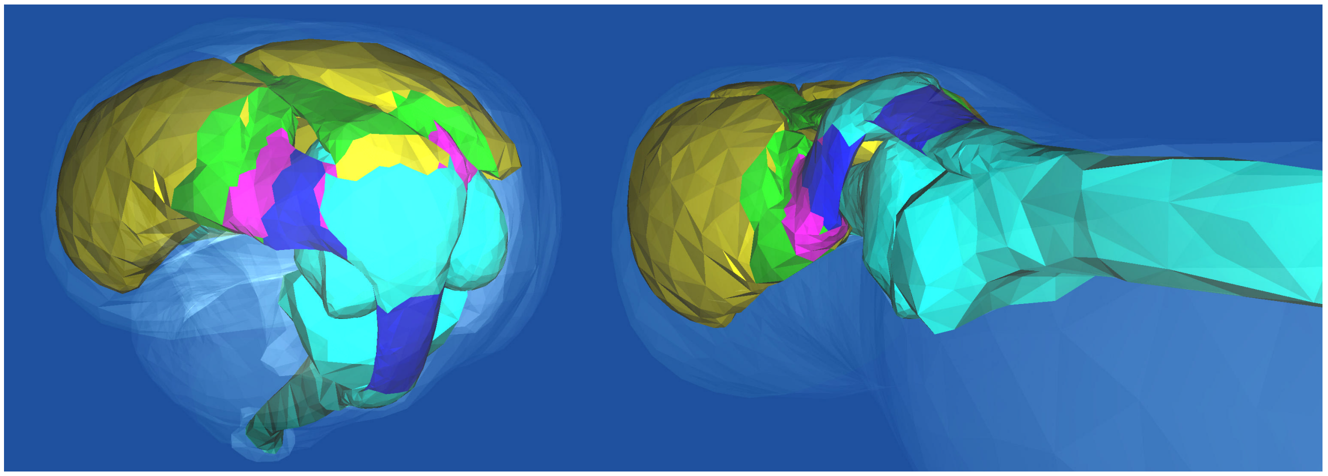

Fig. 10. Bent knee (left) and stretched (initial) position (right). The patella automatically slides over the femur as a result of the simulation

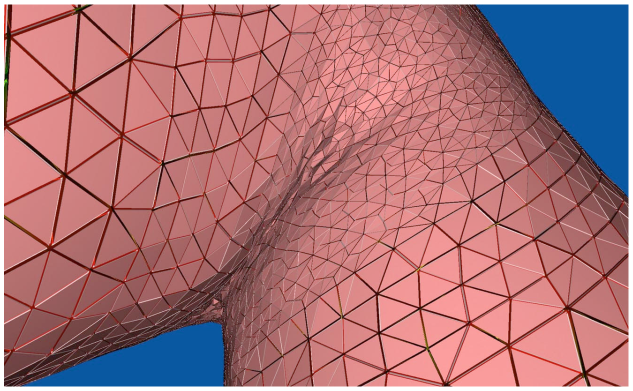

Fig. 12. Close-up of knee, illustrating pattern of skin folding

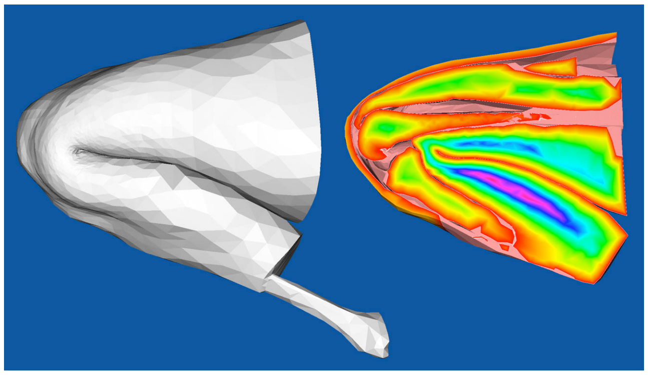

Fig. 11. Skin surface of highly flexed knee (left), cut-away view of the same flexed knee (right). Only parts of the tibia and femur are visible in the cut-away, since they are partly in front of or behind the cutting plane. Note natural-looking sliding contact between skin areas, skin and bones/muscles, patella and femur. The complex self-contact of folding skin was handled without visible penetration. The colors encode the material depth value.

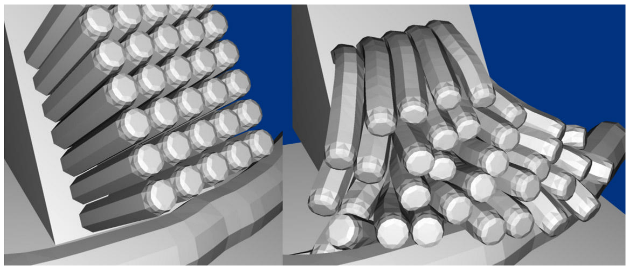

Fig. 13. An example of dynamic analysis: elastic bars deformed by their own weight