The Visual Computer

An improved finite-element contact model for anatomical simulations

Gentaro Hirota, Susan Fisher, Andrei State

Department of Computer Science, Campus Box 3175, Sitterson Hall, UNC-Chapel Hill, Chapel Hill, NC 27599-3175 USA

Published online: 3 June 2003

This work focuses on the simulation of mechanical contact between nonlinearly elastic objects, such as the components of the human body. In traditional methods, contact forces are often defined as discontinuous functions of deformations, which leads to poor convergence characteristics and high-frequency noises. We introduce a novel penalty method for finite-element simulation based on the concept of material depth, which is the distance between a particle inside an object and the object's boundary. By linearly interpolating precomputed material depths at node points, contact forces can be analytically integrated over contact surfaces without raising the computational cost. The continuity achieved by this formulation reduces oscillation and artificial acceleration, resulting in a more reliable simulation algorithm.

Key words: Contact problem – Physically based modeling – Deformation – Finite-element method – Human-body simulation

1 Introduction

1.1 The problem

In animal bodies, adjacent components, such as skin, muscles or bones, are not necessarily attached to each other. 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 or fold, creating complicated geometry. Such sliding contact creates challenging problems.

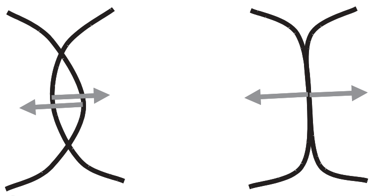
The regions of the boundary surface that are in contact are called contact surfaces. The traction forces acting upon contact surfaces are called contact forces. The contact forces deform the objects. As a result of the deformation, new parts of boundary surfaces may come into contact, or some parts of boundaries previously in contact may be separated. At the beginning of the simulation, the contact surfaces (or contact forces) are unknown. They are usually determined by the iterative recomputation of push-back forces (Fig. 1).

In the final stable configuration, the contact forces are the same as the push-back forces. In conventional methods, the direction of a push-back force, or the normal, is chosen as the direction from a penetrating surface point to the closest projection. Finding a closest projection is quite an expensive operation. Furthermore, in the presence of *self-penetration* and multiple penetrations (three or more bodies occupy the same position), or if a more suitable projection is found farther than the closest one, it is no longer easy to define mathematically the push-back force based on projections alone.

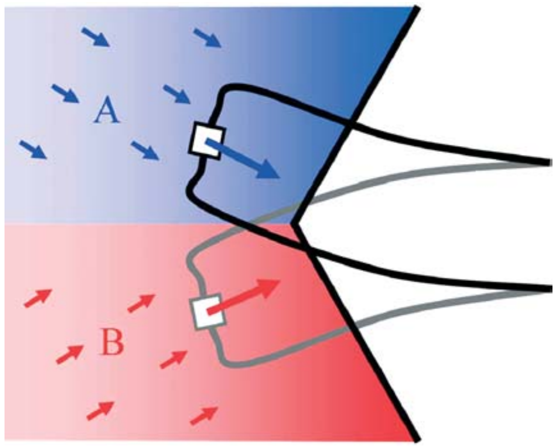
To use the finite-element methods (FEM), the contact forces must be applied to each nodal point of the finite-element mesh. Because of the difficulties in finding the projection, conventional methods evaluate contact forces only at a limited number of points (typically only at nodal points), which leads to a discontinuity problem. This discontinuity does not disappear even if the surface normal is smooth, because the closest projection may jump from one place to another as the object deforms. The discontinuity causes oscillation problems (a phenomenon known as contact chatter [12, 13]) and unnatural, abrupt accelerations in simulations (Fig. 2).

1.2 Proposed solution

We propose an algorithm that significantly reduces the discontinuity of contact forces. The novelty of our algorithm is summarized in two points:



1



2

Fig. 1. Typical steps for solving the contact problem. First, the penetrating regions are found. Then, push-back forces are applied to reduce the amount of penetration. These steps are repeated until a stable configuration is achieved

Fig. 2. An example of contact chatter. A stick (*right*) intrudes into an object (*left*) at a concave location. There is a discontinuity in the push-back direction between upper (*A*) and lower (*B*) regions. The contact force is sampled at a nodal point (*squares*). If the stick swings to the upper region, the sampled force pushes the stick downward. If the stick swings to the lower region, the sampled force pushes the stick upward. As a result, the stick oscillates between the two states

- Approximate normals derived from *material depth*. To avoid the complication of projections, we derive contact force directions (normals) from material depth, the distance from the object boundary in the *material (reference) configuration* (see Fig. 13). In the material configuration, objects are not deformed yet, so there is no self-penetration. Therefore, the material depth can be determined without ambiguity. Furthermore, the material depth is evaluated only at the nodal points of FEM meshes, and then it is linearly interpolated throughout the object.
- Analytical area integration of contact forces. The simplicity of the material-depth computation enables the analytical integration of contact forces over intersecting areas.

The definition of material depth is very simple: it does not have the ambiguity problems that projection has. Area integration provides contact-force continuity, which improves the convergence characteristics of the finite-element analysis.
The algorithm has been applied to anatomical models. The large movement of a musculoskeletal system complete with bones, muscles, and surrounding connective tissues has rarely been simulated. We demonstrate the robustness of our method with a simulation of a large flexion of a human knee.

2 Previous work

This section analyzes various algorithms for contact problems. Since the complexity of a contact problem depends on the type of contact, it is important to examine what kind of problems each algorithm can handle. There are two distinct classes of problems: the *two-body contact problem* and the *self-contact problem*. The two-body contact problem occurs when an object (body) only comes into contact with other objects; that is, the object never comes into contact with itself. The self-contact problem, however, covers all contact scenarios, including two-body contact and contact between two parts of the same object.

2.1 Master-slave approach and contact detection

The concept of *slave nodes* and *master surfaces* (or *master segments* in 2D cases) is used in almost all algorithms. Consider two colliding objects. Nodal points on the surface mesh of one object are designated as slave nodes, whereas master surfaces are selected from the surface mesh of the second object. If a slave node penetrates a master surface, a contact force is applied to push the node back toward the outside of the master surface. Thus the contact force points toward the master surface's normal direction.
To solve two-body contact problems, it is not difficult to assign the master and slave roles to objects' boundary surfaces. Benson and Hallquist propose the *single-surface algorithm*, which is a master-slave

method that can also handle self-contact cases [6]. This algorithm dynamically assigns the master-slave roles based on the result of a neighborhood search. Benson and Hallquist also addressed the asymmetry problem of the master-slave strategy by performing two passes at every time step: the first pass is followed by the second pass with reversed master-slave assignments. This algorithm is used in well-known finite-element software systems (DYNA [17] and NIKE, developed at Lawrence Livermore National Laboratory).

The most difficult task in master-slave algorithms is finding the master-slave pairs. This task is often called *contact detection*. A slave node is almost never located exactly on the master surface. It is impossible to identify master-slave pairs unambiguously. (This problem is discussed in detail in Sect. 4.) Therefore, the pairing has to rely on some estimation method. The single-surface algorithm uses the nearest-node strategy: for each slave node, the closest node is found and the surface element that contains the closest node is considered to be the master surface. This strategy often fails when complex self-contact occurs. The algorithm tries to detect adverse situations by examining surface normals, but it cannot handle all geometrically complex cases.

The *splitting-pinball method* is a unique algorithm proposed by Belytschko and Neal [4, 5]. This method uses repulsive forces between hierarchical bounding spheres around or near individual surface elements. The contact detection is simplified to an interference check between the spheres. Therefore, this method does not suffer from the problem of contact-detection ambiguity found in master-slave approaches. However, 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 high computational cost.

Heinstein et al. proposed a more sophisticated contact-detection method that approximates trajectories of slave nodes and master surfaces by linearly interpolating two successive time steps and then computes the exact time of collision [12, 13]. They use heuristics based on the history of master-slave pairings in previous steps to alleviate problems in complex self-contact cases. Their heuristics also reduce the normal-discontinuity and contact-chatter problems. This algorithm is used at Sandia National Laboratories as a part of a dynamic code (PRONTO) and a quasistatic code (SANTOS). The parallelized version of PRONTO was used to simulate very large scale impact problems [9].

2.2 Contact-force discontinuity and surface smoothness

In master-slave methods, except for rare exceptions, the normal of a master surface is used as the direction of the contact force applied to a slave node. The boundary surfaces are usually triangulated, and the normal is not continuous across the border of triangles. The contact force, seen as a function of nodal point positions, is not continuous. In static analysis, this discontinuity sometimes means that the equilibrium state cannot be achieved. In dynamic analysis, the discontinuity may introduce high-frequency noise into the system. A typical problem is the contact chatter already mentioned.

Padmanabhan and Laursen used a Hermite cubic curve – their method is for 2D problems – for smoothing [23]. Puso and Laursen use Gregory patches for boundary-surface representation, in order to smooth out the normals and thus the contact forces [25]. However, the method seems to be limited to two-body contact problems. Also, surface smoothing has several problems. First of all, there are numerical concerns in maintaining surface continuity. If continuity is maintained as a set of constraints, they introduce an artificial stiffness to the surfaces and increase the discretization error. (In other words, *the surface smoothing decreases the realism of the simulation*, which might sound counterintuitive for graphics researchers.) If automatically continuous surfaces (such as NURBS) are used, the sparsity of the stiffness matrices is compromised, raising computational cost. Secondly, "smooth" surfaces, including Gregory patches and NURBS, do not guarantee smoothness. If buckling occurs in the simulation, normal discontinuity (in the form of cusps) may also emerge. Finally, in many applications, the smoothness of surfaces is not desirable. Surfaces of many objects such as (wrinkled) skin and shock-absorbing bellows in automobiles are not supposed to be smooth.

Smooth-surface normals do not always mean contact-force continuity. A small deformation may change the master-slave pairing and cause the contact force to switch from one direction to the other. The algorithms mentioned above (except for the pinball method) calculate contact forces at slave nodes only.

Because of their point-sampling nature, the contact force applied to a slave node becomes discontinuous again. It is possible to resolve the problems of point sampling by smoothing out contact force discontinuities using intermediate continuous contact surface elements. Papadopoulos and Taylor propose a method that projects contacting-surface elements to an intermediate plane and defines contact elements as intersections of projected elements [24]. However, there is no mention of dynamically generating intermediate planes for high-curvature or self-contacting surfaces. So the idea of intermediate contact elements seems to be applicable to two-body contact between relatively smooth surfaces only.

2.3 Issues in the strict prevention of penetration

In the above discussion, it is assumed that the slave node is often found penetrating other objects. One might wonder why those algorithms do not simply prevent such incidents. It is worth commenting on the possibility of preventing penetration completely.
In Heinstein's method [12, 13], even if the collision times are computed, some penetrations (at nodal points) are tolerated for several time steps before they are eliminated. To completely prevent penetrations, the exact birth and death times of impenetrability constraints for the master-slave pairs must be computed at every simulation step. Such computation is so expensive that there does not seem to be any algorithm that implements this strategy. Enforcing zero penetration seems practically impossible.
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, even if the penetration is completely eliminated, gaps (which are equally inaccurate) emerge. A strict impenetrability constraint does not make sense after the exact shape of boundary is compromised by discretization.

2.4 Techniques used in entertainment applications

Accurate physical simulation, which has been developed in the engineering community, can provide a powerful tool for automatically generating realistic deformations for entertainment applications. Many graphics researchers have demonstrated animations created by techniques based on physical principles [2, 3, 11, 22, 28, 29], but none of them addressed the issue of robustness in contact handling.
Recently, Bridson et al. proposed a robust treatment of contacts for cloth animation [8]. The algorithm applies just enough impulses to nodal points to prevent penetration. The drawback of such a strategy is that they have to assume "inelastic" collisions regardless of the material properties. Furthermore, their impulse method does not always resolve complicated geometrical cases. Whenever the method fails, the algorithm creates rigid impact zones: the deformations of the problematic parts are frozen, and the relative motions between sides in contact are prohibited. Therefore, their method seems to be limited to certain types of high-friction cloth simulation.

3 Simulation of tissue deformation

3.1 Finite-element method

The macroscopic passive behavior of tissues is studied in the field of continuum mechanics. The relationship between the applied forces and the resulting motions are encoded in partial differential equations (PDE). A typical PDE is

$$\operatorname{div} \boldsymbol{\sigma}(\phi) + \boldsymbol{f} = 0 \qquad (1)$$

where $\boldsymbol{\sigma}$ is stress tensor, ϕ is deformation function, and $\boldsymbol{f}$ is the density of external forces.
The simulation of an anatomical structure involves two tasks:

- Defining PDEs for a particular simulation problem
- Solving the PDEs

The first task is called *modeling*. Modeling consists of two parts. One part is determining material properties for various tissue types (hence PDEs) and the other part focuses on defining the geometry of organs (i.e., the boundary of the PDEs' domain). The boundary conditions may also include PDEs that define forces that act on boundary surfaces (such as air pressures).
The second task, solving the PDEs, requires a discretization method such as FEM. After FEM discretization, the object's configuration (shape) is described entirely by a finite number of nodal points.

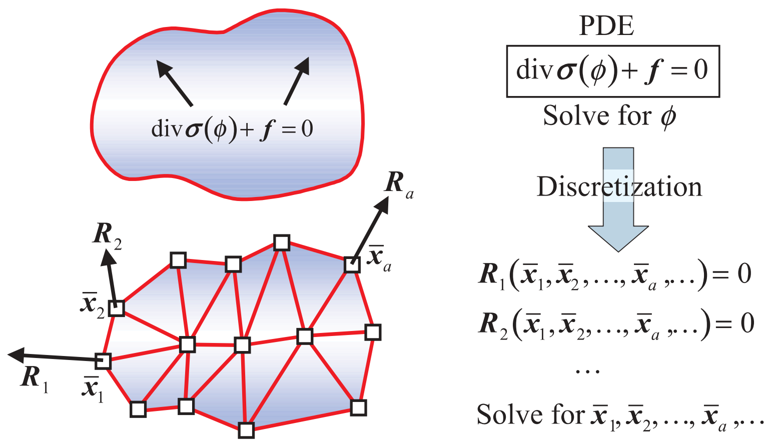

$\mathrm{div}\,\sigma(\phi)+f=0$
PDE
$\mathrm{div}\,\sigma(\phi)+f=0$
Solve for ϕ
Discretization
R_a
R_2
$\bar{x}_a$
$\bar{x}_2$
R_1
$\bar{x}_1$
$R_1(\bar{x}_1,\bar{x}_2,\ldots,\bar{x}_a,\ldots)=0$
$R_2(\bar{x}_1,\bar{x}_2,\ldots,\bar{x}_a,\ldots)=0$
…
Solve for $\bar{x}_1,\bar{x}_2,\ldots,\bar{x}_a,\ldots$

Fig. 3. Finite element method. A continuous object is discretized by the finite element method. The partial differential equation, $\mathrm{div}\,\boldsymbol{\sigma}(\phi)+\boldsymbol{f}=0$ is replaced by algebraic equations in terms of the nodal residual forces $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, . . .

Discretization also converts the PDEs to a system of nonlinear equations. The equations are the forces acting on nodal points, and they are functions of the nodal positions (Fig. 3). In static analyses, simulation is nothing more than solving the system of nonlinear equations to find an equilibrium (zero-force) configuration. In dynamic analyses, PDEs contain derivatives with respect to time, which is usually discretized by finite-difference methods, and the resulting nonlinear system represents, again, an equilibrium state of all participating forces, including inertial ones. Our FEM program performs implicit dynamic analyses: our solver uses the Newton–Raphson method with line search to find a zero-force configuration at each point in time. This method is more robust than the common semi-implicit scheme [3, 28] that lacks convergence check.

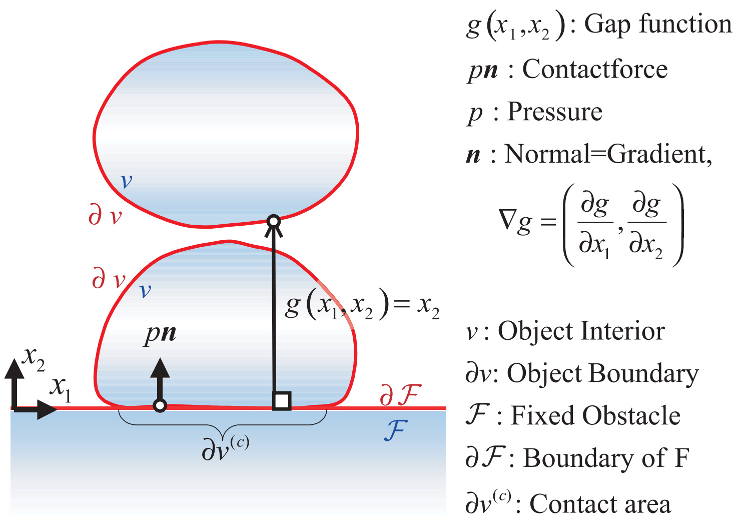
3.2 Material models

The relationship between stress $\boldsymbol{\sigma}$ and deformation ϕ determines the material properties. 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 [20]. Nevertheless, some of the important properties are known about most biological tissues. Those properties include the following: (1) infinite energy with zero volume, (2) highly nonlinear stress-strain relationships, and (3) anisotropy. Every material used in this research has an elastic potential energy that tends to infinity as the volume compression ratio tends to zero. This is not only a realistic assumption but also important in the numerical sense: this property effectively prevents the element-reversal phenomenon in the finite-element solution. To simulate the properties of tissues, we use two isotropic materials, Mooney–Rivlin and Veronda, and a fiber-reinforced anisotropic material [14–16].

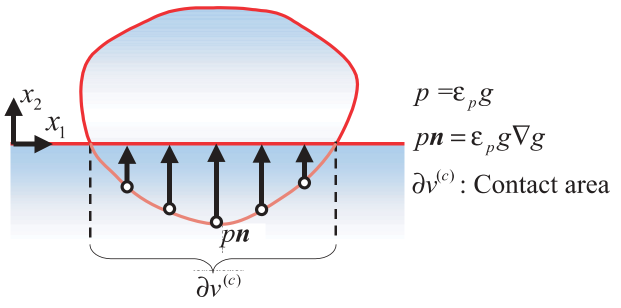
4 The contact problem

4.1 Stationary rigid obstacle: introduction

In many contact situations, one side of the contact is assumed to be a fixed rigid obstacle. In this section, this problem, *Signorini's problem*, is used to introduce a few important concepts, such as gap functions, contact forces, and penalty formulations.
First, it is necessary to define the basic kinematics of contact problems (see Fig. 4). There is a deformable body, whose interior (open and bounded) is υ and whose boundary is $\partial\upsilon$ in the 3D Euclidian space. There is a fixed and rigid obstacle, whose interior


$g(x_1,x_2)$: Gap function
$\boldsymbol{pn}$: Contactforce
p : Pressure
$\boldsymbol{n}$: Normal=Gradient,
$\nabla g = \left(\frac{\partial g}{\partial x_1}, \frac{\partial g}{\partial x_2}\right)$
v : Object Interior
∂v: Object Boundary
$\mathcal{F}$: Fixed Obstacle
$\partial\mathcal{F}$: Boundary of F
$\partial v^{(c)}$: Contact area
v
∂v
$\boldsymbol{pn}$
$g(x_1,x_2)=x_2$
x_2
x_1
$\partial\mathcal{F}$
$\mathcal{F}$
$\partial v^{(c)}$

4


$p = \varepsilon_p g$
$\boldsymbol{pn} = \varepsilon_p g \nabla g$
$\partial v^{(c)}$: Contact area
x_2
x_1
$\boldsymbol{pn}$
$\partial v^{(c)}$

5

Fig. 4. Geometric configuration of a deforming object and fixed obstacle
Fig. 5. Penalty formulation

is $\mathcal{F}$, and its boundary is $\partial\mathcal{F}$. The distance to $\mathcal{F}$ is defined by a scalar function $g(\boldsymbol{x})$, which is called a *gap function* and measures the clearance, or *gap*, for avoiding collision with the obstacle $\mathcal{F}$. The sign of $g(\boldsymbol{x})$ is positive if $\boldsymbol{x}$ is outside of the obstacle. For example, if the obstacle is a ground plane at level zero (as shown in Fig. 4), then $g(\boldsymbol{x}) = x_2$.
The contact dealt with in this work is frictionless; therefore, the contact force is normal to the surface. The normal is the partial derivative of g with respect to spatial coordinates. Thus the contact force is

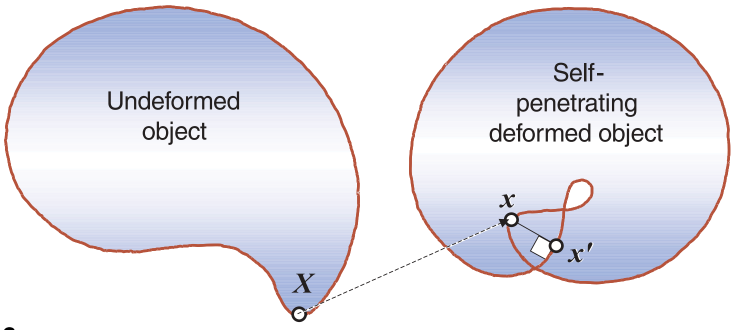
$$\boldsymbol{t} = p\boldsymbol{n} = \frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}} = p\nabla g. \qquad (2)$$

The penalty method replaces the pressure p with the penalty $\varepsilon_p g$ (Fig. 5).

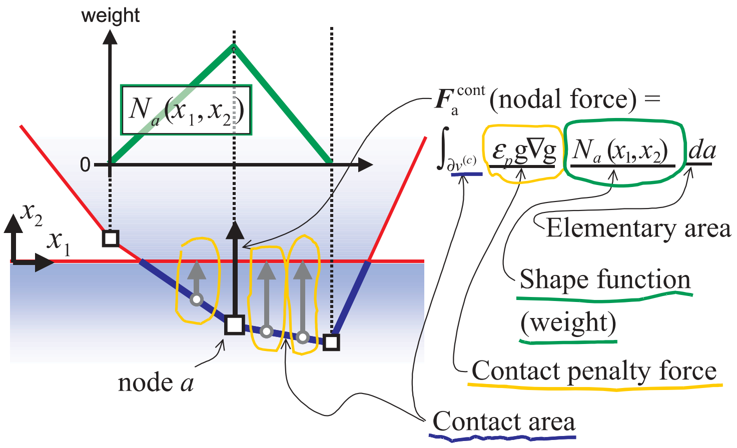
4.2 Self-contact: general contact

4.2.1 Projection

We now examine contact between flexible objects (Fig. 6). If an object can be deformed, it is possible that two parts of the object make contact. Therefore, in general, self-contacting cases should be taken into consideration.
If penetration occurs, multiple particles may share the same location $\boldsymbol{x}$. The location alone is not enough information to determine which particle is examined. To prevent this ambiguity, the gap function must be defined as a function of the coordinates $\boldsymbol{X}$ in the initial configuration, i.e., $g(\boldsymbol{X})$.
If self-contact is considered, the "obstacle" may be the deforming body itself. Unlike in Signorini's



6


weight
$N_a(x_1,x_2)$
0
x_2
x_1
node a
$\boldsymbol{F}_a^{\text{cont}}$ (nodal force) =
$\int_{\partial v^{(c)}} \varepsilon_p g \nabla g \; N_a(x_1,x_2) \; da$
Elementary area
Shape function (weight)
Contact penalty force
Contact area

7

Fig. 6. Geometric configuration of self-penetrating object
Fig. 7. 2D illustration of contact-force discretization: The *upper object* intrudes into the *lower object*. The *upper object's* boundary is discretized as line segments (in 3D, it is a triangle mesh). The *squares* are nodal points. The nodal contact force of the ath node is the weighted average of the contact penalty force ($\varepsilon_p \nabla g$). The force is averaged over the contact area ($\partial v^{(c)}$), and the shape function of the node (N_a) is used for the weight. da is the elementary surface area

problem, the gap function cannot be defined as the distance between the point and boundaries since it is always zero. Instead of the 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. 6). A projection is a local minimizer of distance.

4.2.2 Discretization of the contact force

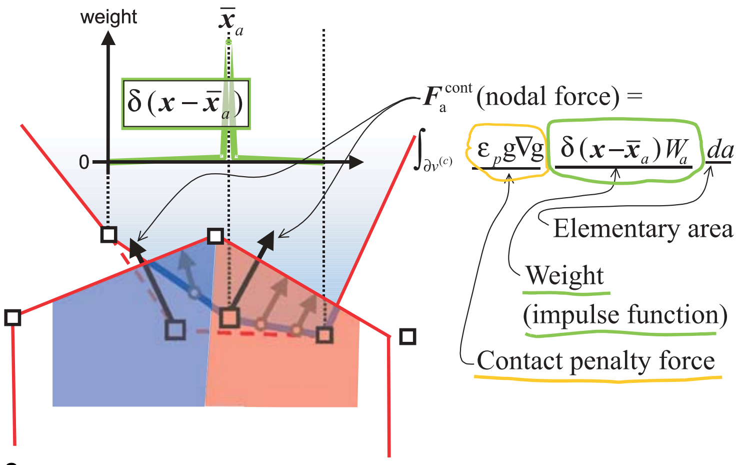
In FEM, all forces must be discretized into nodal equivalent forces. The contact force is no exception. The contact force [Eq. (2)] in the penalty formulation is discretized as follows (see also Fig. 7):

$$\boldsymbol{F}_a^{\text{cont}} = \int_{\partial v^c} \varepsilon_p g \nabla g N_a \text{d}a. \qquad (3)$$

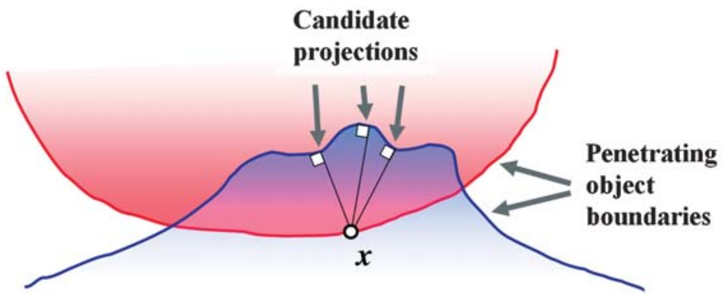
This is the nodal equivalent contact force on node a.

4.2.3 Point collocation

Now the numerical integration of Eq. (3) is considered. Since it is difficult to obtain an analytical expression of g and ∇g, this integration cannot be performed easily. To simplify the computation, most conventional methods evaluate g and ∇g only at nodal points (or a limited number of points): the integration is performed by the point collocation as (see


weight
$\bar{x}_a$
$\delta(x-\bar{x}_a)$
0
F_a^{cont} (nodal force) =
$\int_{\partial v^{(c)}} \varepsilon_p g\nabla g \; \delta(x-\bar{x}_a) W_a \; da$
Elementary area
Weight
(impulse function)
Contact penalty force

8



9

Fig. 8. Point collocation and contact force discontinuity. The weight is an impulse function scaled by a constant node weight W_a. A small node movement causes an abrupt nodal force change

Fig. 9. Derivative discontinuous gap function. A bumpy surface has many projections for a given particle x. In such a case, a small movement of x results in a sudden jump of the closest projection. In this case, the gap function (defined as the closest projection) is continuous with respect to x, but its derivative with respect to x is not continuous

also Fig. 8)

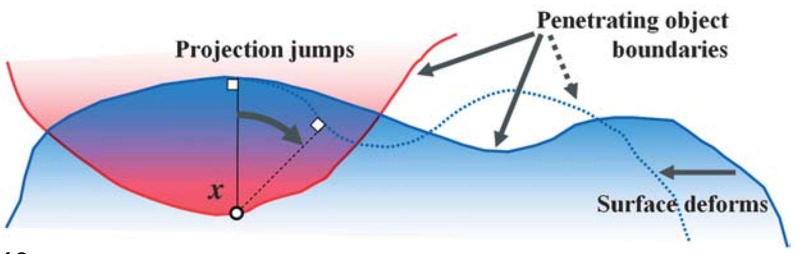
$$\boldsymbol{F}_a^{\text{cont}} = \int_{\partial v^c} \varepsilon_p g \nabla g \delta(\boldsymbol{x} - \bar{\boldsymbol{x}}_a) W_a \mathrm{d}a. \tag{4}$$

With this method, the nodal force depends only on the gap value at the nodal point and becomes discontinuous.

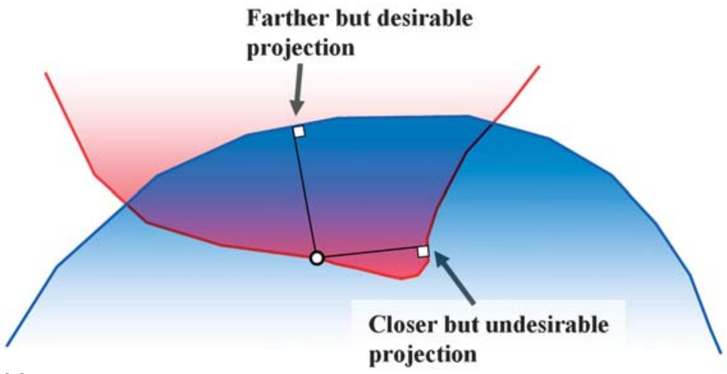
4.2.4 Sensitive projection location and gap discontinuity

This section discusses in more detail the difficulties in determining the gap function. The gap function is evaluated by finding projections. By the definition of projection, the search algorithm has to rely on the surface normals. The problems can be summarized as the following two points:

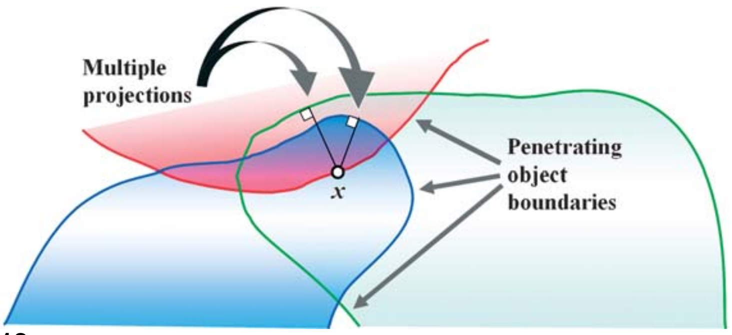
1. If the gap function is defined as the distance to the closest projection from a particle at x, the gap function becomes discontinuous with respect to the configuration because the location of the projection is sensitive to the deformation (see Figs. 9 and 10).
2. The criteria for selecting a projection among many candidates are difficult to define. In fact, choosing the closest projection is not always desirable (see Figs. 11 and 12).



10


Farther but desirable projection
Closer but undesirable projection

11



12

Fig. 10. Discontinuous gap function. A small deformation creates a new projection, which can be closer than the old projection, causing an abrupt jump of the gap function value. This case is worse than Fig. 9 since the gap function itself is discontinuous

Fig. 11. Closest but undesirable projection. The closest projection found in the neighbor is intuitively not desirable. However, it is hard to define criteria that make the algorithm prefer the desirable projection. In the presence of self-contact, both *red* and *blue* regions may belong to a single object. Therefore, it is not possible to mechanically reject projections on red regions

Fig. 12. Multiple valid projections. Since the particle x intrudes into more than one region of nearby objects, it makes more sense to use multiple projections. The *red, blue*, and *green* regions may be just parts of a single object

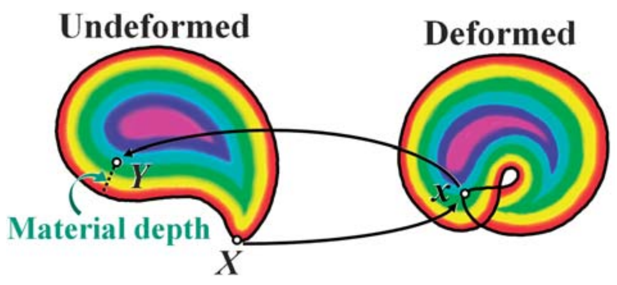


Fig. 13. Material depth

4.2.5 Convergence and noise problem

The point-collocation technique [Eq. (4)] samples the discontinuous function g and ∇g. The resulting penalty force $\boldsymbol{F}_a^{\text{cont}}$ is therefore discontinuous with respect to the mesh configuration. F_a^{cont} is a component of the residual force ($\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots$ in Fig. 3). This discontinuity often manifests itself as the convergence problem (oscillation) in nonlinear system solving, because with the presence of a discontinuity, a nonlinear equation often lacks a solution. This discontinuity also causes high-frequency noises in simulations of sliding contact.

4.3 The new algorithm

The culprit in the convergence problem is the discontinuity of the contact force. The gap function is intrinsically nonsmooth. As long as the point collocation of Eq. (4) is used, discontinuous contact forces are inevitable. A possible remedy to the discontinuity problem is to perform the area integration of Eq. (3). Area integration averages various values of g and ∇g, resulting in a smooth contact force.

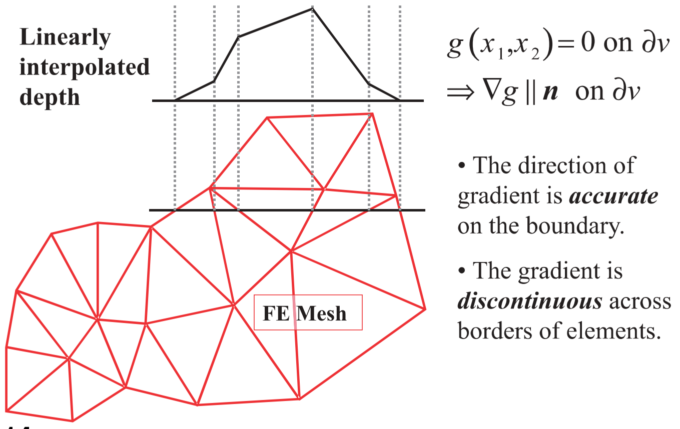
By sampling many points in the area, similar effects can be achieved, but that is expensive computationally. Therefore, a precise analytical integration is desirable. However, if a projection is used to evaluate g and ∇g, the formulation of the analytical integration is not easy. As described in Sect. 4.2.4, it is already difficult to uniquely determine an optimum projection for a point. Analytically integrating a quantity that is related to such values seems impossible. A more stable and simpler substitute for the projection is required. The following sections explain a new algorithm based on the use of *material depth* for evaluating the gap function.

4.3.1 Material depth and gap continuity

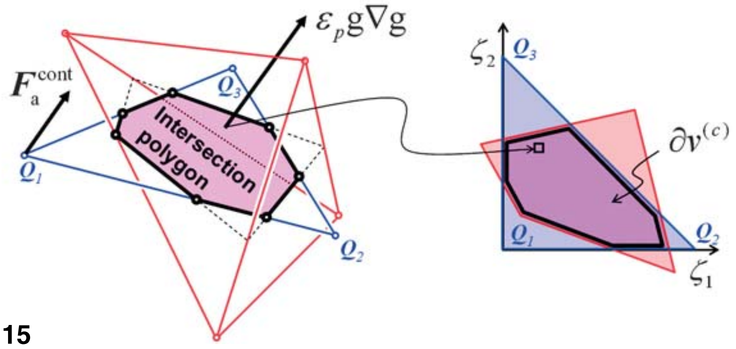
Figure 13 shows the notion of *material depth*. The deformation maps particle $\boldsymbol{X}$ on a boundary to the current position $\boldsymbol{x}$. As a result, $\boldsymbol{X}$ collides with particle $\boldsymbol{Y}$. The material depth is the distance from $\boldsymbol{Y}$ to the boundary in the reference (undeformed) configuration. It maintains a constant intrinsic value for a particle (or material point); hence the term material depth. Since there is no self-penetration in the reference configuration, the material depth can be computed regardless of self-penetration in the current (deformed) configuration. Also, unlike the distance to a projection, material depth never changes abruptly due to deformation.

Using the material depth as the gap function, as described above, is still not simple enough to allow analytical integration. The new algorithm approximates the material depth by the linear interpolation of the material depths between nodal points of the FE mesh (see Fig. 14). The contact force is integrated for each intersecting pair of a boundary triangle element and an interior tetrahedron element. The integration is performed in the parameter space of the triangle element (see Fig. 15). The detailed derivation of analytical integration is mathematically quite involved and is presented elsewhere [16].

Material depth is an approximation of the distance field in the current configuration. As an object is stretched or compressed, the depth value deviates from the actual distance. However, the inaccuracy does not cause a significant problem in terms of the maximum penetration allowed since in practice a large penalty factor can be used without causing numerical instability [10]. Thus the magnitude of the depth value is not a significant issue.


Linearly interpolated depth
$g(x_1,x_2)=0$ on ∂v
$\Rightarrow \nabla g \parallel \boldsymbol{n}$ on ∂v
• The direction of gradient is *accurate* on the boundary.
• The gradient is *discontinuous* across borders of elements.
FE Mesh

14



15

Fig. 14. Linear interpolation of material depth
Fig. 15. Intersecting elements and area integration

A more important aspect of material depth is its gradient, which is used as an approximate surface normal. Since the material depth deviates from the spatial distance, the gradient also deviates from the actual normal. But this is not a serious problem either. The direction of the gradient is in fact the same as the exact normal on the boundary. Thus the gradient of the material depth is indeed a good approximation of the surface normal, and the directional accuracy increases as the algorithm converges to a solution with small penetration.

4.3.2 Algorithm summary

The algorithm performs the following steps:

1. Compute the material depth for each node in the reference configuration. This is an off-line process that can be done by a brute-force method once per model.
2. Find the collision between a boundary triangular element and an interior tetrahedral element. This step is described in Sect. 4.3.4.
3. Compute the intersection of the triangular element and the tetrahedral element.
4. Integrate the penalty force over the area of the intersection polygon, and add their contributions to the global residual and the stiffness matrix.

4.3.3 Area integration and gap gradient continuity

The area of the intersection of the boundary triangles and the interior's tetrahedron varies continuously. The material depth is also continuous. Thus the penalty force is mostly a smooth function (C^1) of the mesh configuration. There are two exceptions. One occurs when an edge of a triangle and a side of the tetrahedron are coplanar, in which case

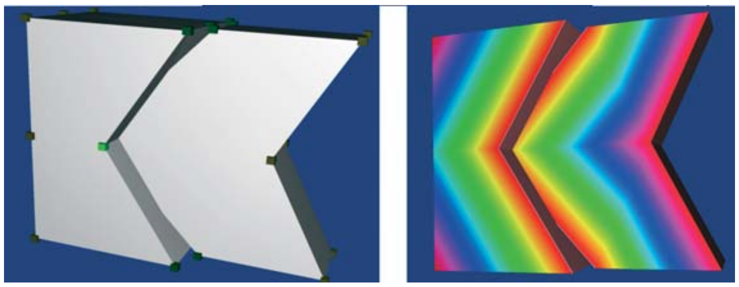


Fig. 16. Two intersecting objects

the penalty force is no longer smooth but it is still continuous (C^0). The other exception occurs when the triangle is coplanar with a side of the tetrahedron. In this case the continuity no longer holds, but for this to occur, three vertices of the triangle must fall into the side of the tetrahedron, a behavior not likely to happen in practice. Furthermore, if the side of tetrahedron is one of the object's boundary triangles, continuity still holds. Thus the penalty forces are continuous in all plausible situations. This is a crucial point with respect to equation solving, since a solution may not even exist without continuous penalty forces. This continuity is the major advantage of the new method over traditional methods. A more rigorous mathematical analysis can be found elsewhere [16].

4.3.4 Collision detection

The algorithm needs to know which boundary elements (triangles) intersect with which interior elements (tetrahedra). In our applications, the number of elements reaches tens to hundreds of thousands. Algorithms that are based on sorting along coordinate axes and overlap lists suffer from a large number of overlapping tetrahedra per triangle in a coordinate and perform very poorly.

We use hierarchical bounding volumes. The frequent deformation of the finite-element mesh demands that the algorithm update the tree quickly. Therefore, a simple bounding volume, an interval in a coordinate, is chosen. The hierarchy is stored as a binary tree. 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 final candidates are collected from tetrahedra in overlapping leaf nodes.

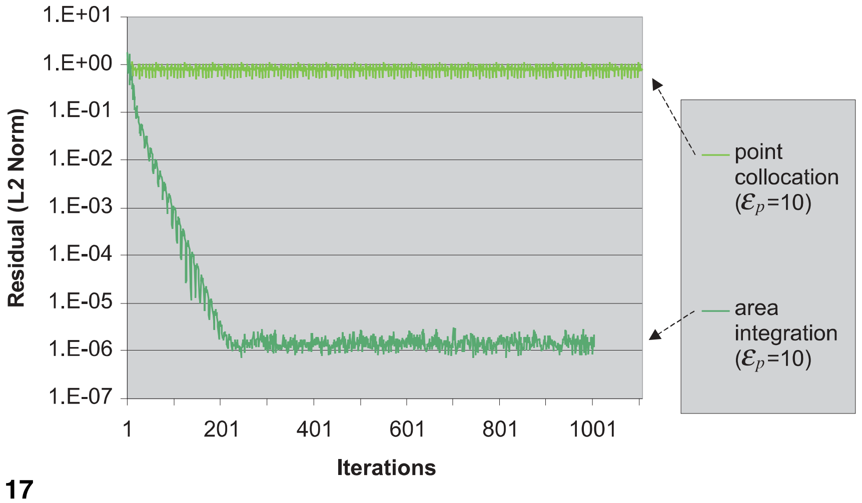
The tree structure is built by a top-down partitioning of elements in the direction of their greatest extent. We usually divide an "analysis session" of a phenomenon (e.g., flexion of a joint) into sequential simulation runs. Since we set up the simulation runs such that the mesh does not deform much in a single run, it is sufficient to build the tree structure 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 a rather minor part of the total computation time (see Sect. 6.3).

5 Numerical comparison of convergence

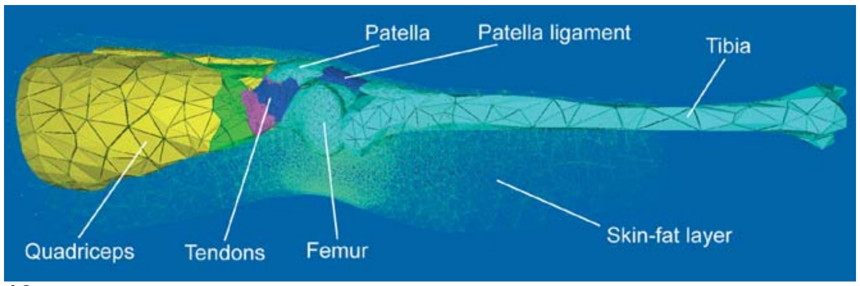
This section presents a numerical comparison of a point-collocation algorithm and the area-integration algorithm to illustrate the superior convergence characteristics of our method.

Figure 16 illustrates the test case used. In the initial configuration, two objects are intersecting. The pointy part of the object on the right is penetrating into the concave edge of the object on the left. This is a typical case that causes contact chatter.

The color encodes depth. The discontinuous borders of depth gradient are visible between the upper and lower parts of the objects. The directions of contact forces are the same as the gradient directions. As the position of sampling moves from the upper part to the lower part, the contact forces



17



18

Fig. 17. Comparison of convergence rates. The area-integration algorithm converges quickly while the point-collocation algorithm fails to converge

Fig. 18. Finite element model of human leg

change abruptly. But the area-integrated contact forces change continuously.

Figure 17 compares convergence rates of the point-collocation algorithm and the area-integration algorithm. The residual for the point-collocation algorithm stays just below 1.00E+00, while the residual for the area-integration algorithm goes down to a very low level until it hits a limit due to the floating-point truncation error.

6 Application to the human body

Accurate simulation, in the sense that one can confidently control the numerical error compared to real subjects, requires an accurate anatomical model. But building such a model is quite difficult because neither established tissue material models (constitutive laws) nor measurement technology exist. Thus the simulation results in this section are meant to give a glimpse of what our computational method can offer once the advancement of biomechanics enables the construction of accurate models of biological systems.

The goal of our algorithm is to handle contacts, especially complex self-contacts. Therefore, it is best tested on the skin folds around flexing joints. The wrinkles around joints are particularly important in realistic image synthesis.

6.1 The model

We simulated flexion of a human knee joint using a finite-element model of a right human leg (Fig. 18). To build the model, we first generated boundary polygons of all the organs from a man-

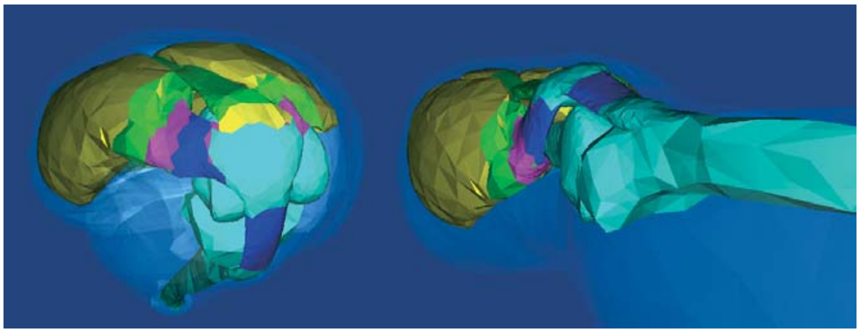


Fig. 19. Bent knee (*left*) and stretched (initial) position (*right*). The patella automatically slides over the femur as a result of the simulation

ually segmented mask image volume of the Visible Human Male [27]. Then a tetrahedral mesh was generated from the polygonal boundaries using SolidMesh [19]. 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 skin-fat layer. The model encompasses mechanically diverse tissues ranging from hard bones, ligaments, and tendons to softer muscles and a skin-fat layer. Therefore, a broad range of material properties are used. The Mooney–Rivlin and Veronda models were both applied.

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.

6.2 The result

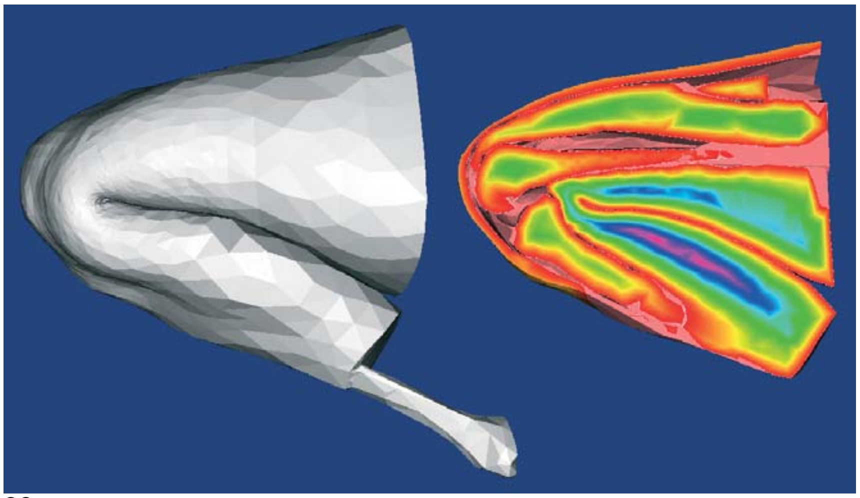
Anatomically expected movements were automatically generated by the simulation. The patella's movement is particularly noteworthy. Figure 19 shows the movement of the patella. Even though the movement of the patella is not prescribed, it slides over the head of the femur (between the lateral and medial condyles) as it is pulled by the rotating tibia via the patella ligament. Figure 20 shows the result at the maximum flexion. The left image shows the skin geometry. The right image shows the cut-away view. The colors encode the material-depth value. The folded skin-fat layer inside the knee joint is highly compressed, yet the amount of penetration is not visible. Figure 21 shows the complex wrinkling pattern of the skin. The curvature of the skin surface is extremely high where the folding occurs. Figure 22 shows cut-away views of the same part. Again, the penetration is visually undetectable.

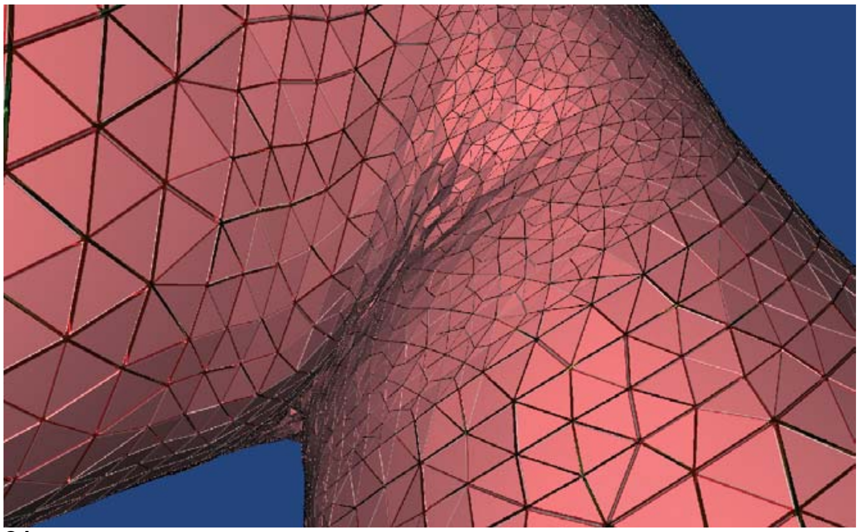
6.3 CPU time statistics

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

7 Other examples

Figure 23 shows another animation sequence. This animation was created by an old version of our algorithm.


20


21

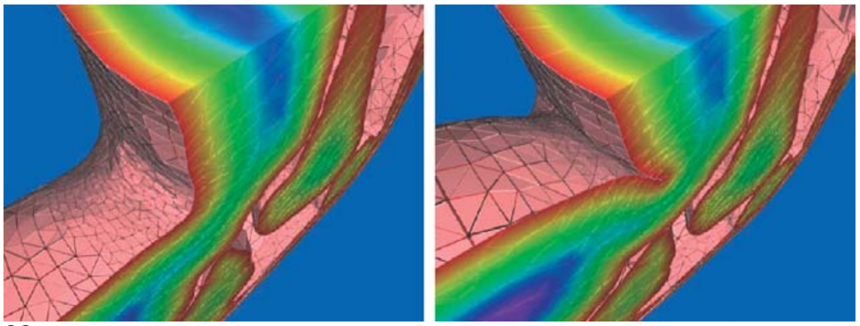

22

Fig. 20. Skin surface of the highly flexed knee (*left*); cut-away view of the same flexed knee (*right*)
Fig. 21. Close-up of knee, illustrating pattern of skin folding
Fig. 22. The complex self-contact of folding skin was handled without visible penetration

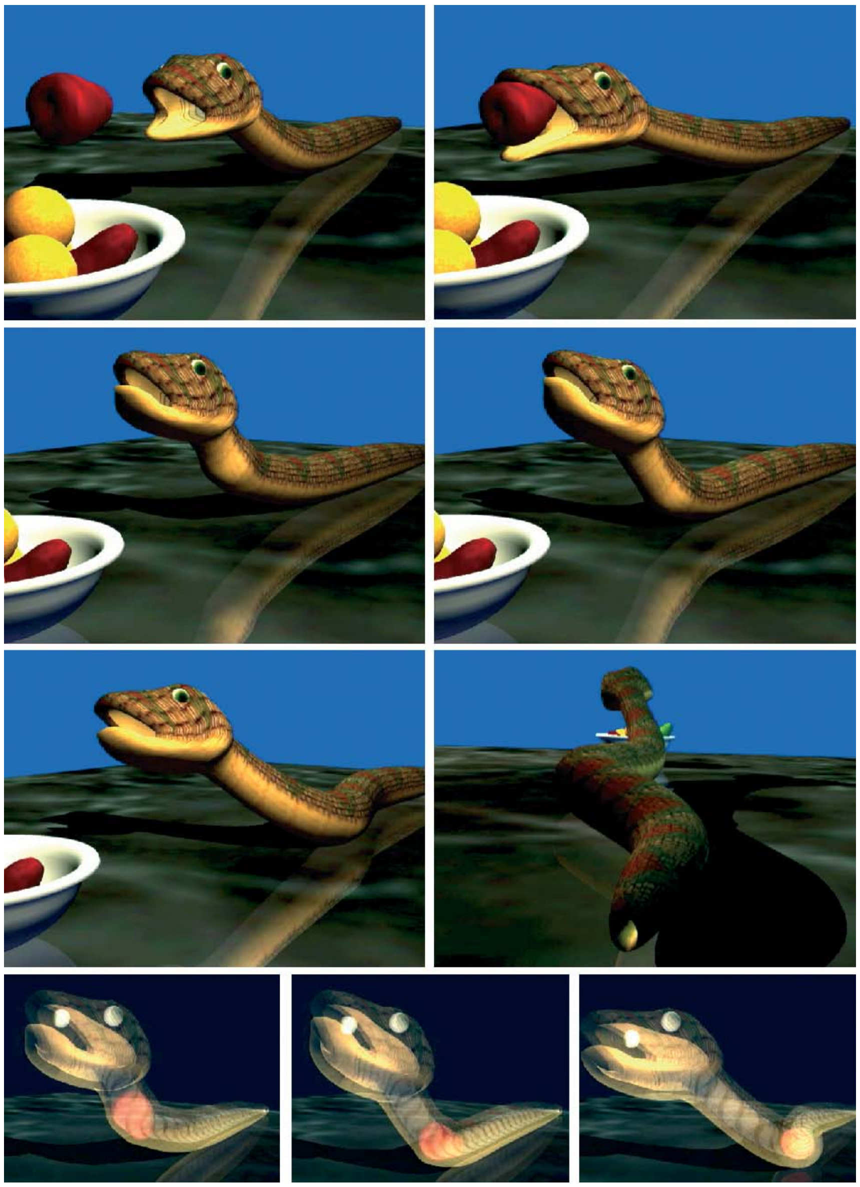


Fig. 23. Snake and apple. A snake swallows an apple. The *upper 3 rows* show the sequence in opaque rendering. The images in the *bottom row* use transparent rendering to show the internal movement of the apple

Figure 24 shows an example of dynamic simulation. There is a 7×5 array of rods mounted on a 45°-slanted foundation. At the beginning of the simulation, all the objects are stationary. Gravity then pulls the rods toward the ground and causes complex collisions and sliding contacts.

8 Discussion and future work

8.1 Trading accuracy for speed

In many applications such as video games or surgical simulators, interactive response is preferred to slow

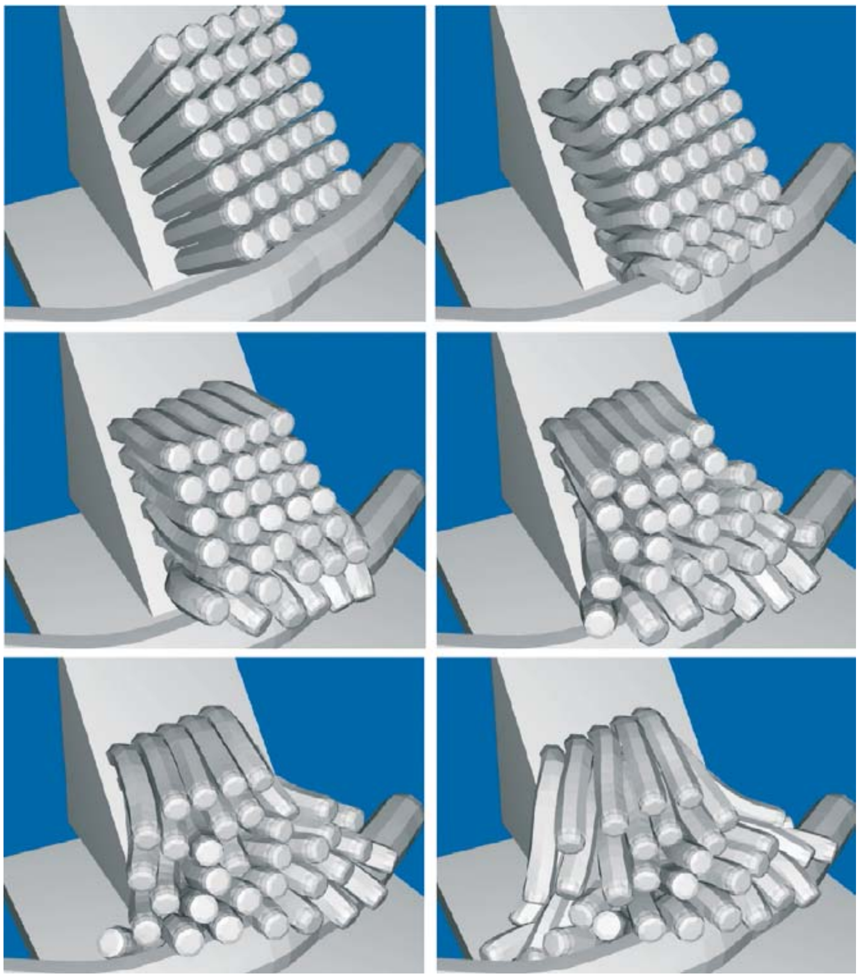


Fig. 24. A stack of rods deformed by gravity

and accurate simulations. The most decisive factor for computation time is the complexity of the problem. Fast simulations can be achieved by simplifying (hence reducing the fidelity of) the simulation models.

Reducing the number of elements is the simplest way to save computation time. The materials can be simplified, too. Using a simpler material model, for example the neo-Hookean material, instead of Mooney–Rivlin and Veronda materials would reduce the stiffness matrix computation by nearly an order of magnitude [7]. The movements of objects can be slowed down by increasing the inertial forces. Because of the sluggishness of the objects, larger time steps can be safely chosen [26]. Since such modification changes the objects' behavior, the fidelity of the simulation is compromised.

8.2 Thin objects

It is rare that flat objects are simulated as zero-thickness plates [12]. However, it is certainly an advantage for an algorithm if it can handle objects with smaller thickness efficiently. This can probably be achieved by truncating the length of time steps to the nearest contact time estimated by a method similar to the one proposed by Heinstein et al. [12, 13]. After the truncation, our algorithm can perform the rest of the contact handling.

8.3 Augmented Lagrangian method

The penalty method used in this paper seems to perform very well. Yet, a more sophisticated method can be constructed by representing pressure on contact surfaces (contact-force magnitude) by Lagrange multipliers. Lagrange multipliers would make the enforcement of the penetration limit more robust. The pressure field would be approximated by a linear interpolation of nodal pressure values and the contact force would be defined as a product of the interpolated pressure and the gap function gradient. By combining penalty methods and Lagrangian multiplier techniques, an augmented Lagrangian scheme can be devised. In this scheme, the penalty term is preserved from the current algorithm. The difference between the interpolated pressure and the magnitude of the penalty force is integrated and used to update the nodal pressures. The pressure update is repeated until the penetration becomes below the desired tolerance.

8.4 Friction

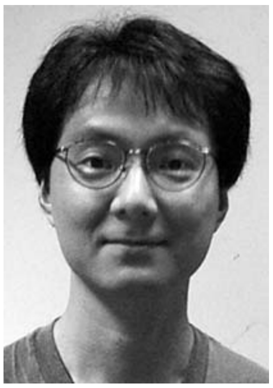
Our algorithm is limited to frictionless contact problems. This limitation is justified because the friction between lubricated organ surfaces inside bodies is known to be small [18, 21]. This is not the case for friction between (non-lubricated) skin surfaces. We expect that our algorithm can be extended easily to accommodate friction. Frictional forces are functions of normal forces and relative velocities on the contact surfaces. They can be integrated over the contact areas just as normal contact forces are integrated. Furthermore, continuous normal forces (which were realized by our algorithm) are helpful in performing more accurate simulations of frictional contacts [25].

Acknowledgements. We would like to thank Henry Fuchs, Sarang Joshi, Kenneth Hoff, David O'Brien, Paul Zimmons, Chris Lee, and Karl Reinig for their help.

References

1. Anderson P et al (1999) Using Maya. Alias|Wavefront
2. Baraff D, Witkin A (1992) Dynamic simulation of non-penetrating flexible bodies. In: Proceedings of the 19th annual conference on computer graphics and interactive techniques. ACM Press, New York
3. Baraff D, Witkin A (1998) Large steps in cloth simulation. In: Proceedings of the 25th annual conference on computer graphics and interactive techniques. ACM Press, New York
4. Belytschko T, Neal MO (1991) Contact impact by the pinball algorithm with penalty and Lagrangian methods. Int J Numer Methods Eng 31:547–572
5. Belytschko T, Yeh I-S (1992) The splitting pinball method for general contact. In: Glowinksi R (ed) Computing methods in applied science and engineering. Nova Science, New York
6. Benson DJ, Hallquist JO (1990) A single surface contact algorithm for the post-buckling analysis of shell structures. Comput Methods Appl Mech Eng 78:141–163
7. Bonet J, Wood RD (1997) Nonlinear continuum mechanics for finite element analysis. Cambridge University Press, New York
8. Bridson R, Fedkiw R, Anderson J (2002) Robust treatment of collisions, contact and friction for cloth animation. In: Proceedings of the 29th annual conference on computer graphics and interactive techniques. ACM Press, New York
9. Brown K, Attaway S, Plimpton S, Hendrickson B (2000) Parallel strategies for crash and impact simulations. Comput Methods Appl Mech Eng 184:375–390
10. Carstensen C, Scherf O, Wriggers P (1999) Adaptive finite elements for elastic bodies in contact. SIAM J Sci Comput 20(5):1605–1626.
11. Gourret J-P, Thalmann NM, Thalmann D (1989) Simulation of object and human skin deformations in a grasping task. In: Proceedings of the 16th annual conference on computer graphics and interactive techniques. ACM Press, New York
12. Heinstein MW, Attaway SW, Swegle JW, Mello FJ (1993) A general-purpose contact detection algorithm for nonlinear structural analysis codes. Sandia Report, SAND92-2141 UC-705. Sandia National Laboratories Technical Library, Albuquerque, New Mexico
13. Heinstein MW, Attaway SW, Swegle JW, Mello FJ (1993), A general contact detection algorithm for finite element analysis. In: Aliabadi MH, Brebbia C A (eds) Contact mechanics. Computational Mechanics Publications, Southampton, UK
14. Hirota G, Fisher S, State A, Fuchs H, Lee C (2001) Simulation of deforming elastic solids in contact. SIGGRAPH Technical Sketch. In: SIGGRAPH 2001 conference abstracts and applications. ACM Press, New York
15. Hirota G, Fisher S, State A, Fuchs H, Lee C (2001) An implicit finite element method for elastic solids in contact In: Computer Animation 2001, Seoul, Korea, 7–8 November 2001
16. Hirota G (2002) An improved finite element contact model for anatomical simulations. PhD thesis, Department of Computer Science, University of North Carolina at Chapel Hill
17. Lin JI (1998) DYNA3D: a nonlinear, explicit, three-dimensional finite element code for solid and structural mechanics. User manual, Methods Development Group, Lawrence Livermore National Laboratory
18. Malcolm LL (1976) An experimental investigation of the frictional and deformational response of articular cartilage interfaces to statistic and dynamic loading. PhD thesis, University of California San Diego
19. Marcum DL, Weatherill NP (1995) Unstructured grid generation using iterative point insertion and local reconnection. AIAA J 33(9):1619–1625

20. Miller K, Chinzei K, Orssengo G, Bednarz P (2000) Mechanical properties of brain tissue in-vivo: experiment and computer simulation. J Biomechanics 33:1369–1376
21. Moro-oka T, Miura H, Higaki H, Arimura S, Mawatari T, Murakami T, Iwamoto Y (1999) A new friction tester of the flexor tendon. J Biomechanics 32:1131–1134
22. O'Brien JF, Hodgins JK (1999) Graphical modeling and animation of brittle fracture. In: Proceedings of the 26th annual conference on computer graphics and interactive techniques. ACM Press, New York
23. Padmanabhan V, Laursen TA (1998) A new contact surface smoothing procedure for the implicit finite element analysis of frictional contact. In: Guran A (ed) Proceedings of the first international symposium on impact and friction of solids, structures and intelligent machines, Ottawa, Canada, 27–30 June 1998. Series on stability, vibration and control of systems, series B, vol 14. World Scientific, Singapore
24. Papadopoulos P, Taylor RL (1993) A simple algorithm for three-dimensional finite element analysis of contact problems. Comput Struct 46(6):1107–1118
25. Puso MA, Laursen TA (2001) A 3D contact smoothing method using Gregory patches. Int J Numer Methods Eng (under review)
26. Richtmyer RD, Morton LW (1967) Difference methods for initial-value problems. Interscience, New York
27. Spitzer V, Ackermann MJ, Scherzinger AL, Whitlock D (1996) The visible human male: A technical report. J Am Med Inform Assoc 3(2):118–130
28. Terzopoulos D, Platt J, Barr A, Fleischer K (1987) Elastically deformable models. In: . ACM Press, New York
29. Zhuang Y (2000) Real-time simulation of physically-realistic global deformations. PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley



GENTARO HIROTA is an architect at NVIDIA Corporation, Santa Clara, Calif. He received a master's degree in biophysical engineering from Osaka University (Japan) and a PhD degree in computer science from the University of North Carolina, Chapel Hill. His research interests include biomechanics, computational mechanics, and physics-based modeling.

SUSAN FISHER is a graphics software engineer at Pixar Animation Studios, Emeryville, Calif. She received a BSc degree in computer science from Georgia Tech in 1999, and a master's degree in computer science from the University of North Carolina, Chapel Hill, in 2001. Her research interests include the computation and use of distance fields in real-time applications, and physically-based modeling and rendering of hair.

ANDREI STATE is a senior research associate in the Department of Computer Science at the University of North Carolina at Chapel Hill. His research interests include augmented reality and realistic human body simulation. He received his Dipl.-Ing. aer. degree (aerospace engineering) from the University of Stuttgart (Germany) and his master's degree in computer science from the University of North Carolina at Chapel Hill.