Pergamon

Comput. & Graphics, Vol. 21, No. 3, pp. 335–346, 1997

Printed in Great Britain
0097–8493/97 $17.00 + 0.00

PII: S0097–8493(97)00011–3

Technical Section

A SPRING FORCE FORMULATION FOR ELASTICALLY DEFORMABLE MODELS

UĞUR GÜDÜKBAY[1,2†], BÜLENT ÖZGÜÇ[2] and YILMAZ TOKAD[3]

[1]University of Pennsylvania, Department of Computer and Information Science, 200 South 33rd Street, Philadelphia, PA 19104-6389, U.S.A.
e-mail: ugur@graphics.cis.upenn.edu
[2]Bilkent University, Department of Computer Engineering and Information Science, Bilkent, 06533 Ankara, Turkey
[3]Eastern Mediterranean University, G. Magusa, North Cyprus, Mersin 10, Turkey

Abstract—Continuous deformable models are generally represented using a grid of control points. The elastic properties are then modeled using the interactions between these points. The formulations based on elasticity theory express these interactions using stiffness matrices. These matrices store the elastic properties of the models and they should be evolved in time according to changing elastic properties of the models. However, forming the stiffness matrices at any step of an animation is very difficult and sometimes the differential equations that should be solved to produce animation become ill-conditioned. Instead of modeling the elasticities using stiffness matrices, the interactions between model points could be expressed in terms of external spring forces. In this paper, a spring force formulation for animating elastically deformable models is presented. In this formulation, elastic properties of the materials are represented as external spring forces as opposed to forming complicated stiffness matrices. © 1997 Elsevier Science Ltd

1. INTRODUCTION

An important aspect in realistic animation is modeling the behavior of deformable objects. To simulate the behavior of deformable objects, we should approximate a continuous model by using discretization methods, such as finite difference and finite element methods. For finite difference discretization, a deformable object could be approximated by using a grid of control points where the points are allowed to move in relation to one another. The manner in which the points are allowed to move determines the properties of the deformable object. Simulating the physical properties (such as tension and rigidity), static shapes exhibited by a wide range of deformable objects (including string, rubber, cloth, paper, and flexible metals) can be modeled. For example, to obtain the effect of an elastic surface, the grid points are connected by springs. The physical quantities, such as forces, torques, velocities, accelerations, kinetic and potential energies, should be used to simulate the dynamics of these objects.

1.1. *Previous work for deformable models*

There are some formulations which employ continuous elasticity theory to model the shapes and motions of deformable models. The primal [1] and hybrid [2] formulations are in this category. In these formulations, elastic properties of the materials are represented using potential energy functionals and stored in stiffness matrices. Potential energies of deformation are defined using the concepts from differential geometry and spline energies.

There are other approaches to model and animate deformable models. Some of these approaches are explained in the sequel.

Witkin *et al.* formulate a model for nonrigid dynamics based on global deformations with relatively few degrees of freedom [3]. This model is restricted to simple linear deformations that can be formulated by affine transformations. The use of deformations that are linear in the state of the system causes the constraint matrices in equations of motion to be constant. Hence, pre-inverting these matrices yields an enormous benefit in performance. In [4], Pentland and Williams describe the use of modal analysis to create simplified dynamic models of nonrigid objects. This approach breaks nonrigid dynamics down into the sum of independent vibration modes. This allows Pentland and Williams to achieve a level of control not possible with the massed equations normally used in dynamic simulation. This approach reduces the dimensionality and stiffness of the models by discarding high-frequency modes. High-frequency modes have no effect on linear deformations and rigid body dynamics. Both of these methods achieve large computational savings at the expense of limited deformations.

Another method, based on physics and optimization theory, uses mathematical constraint methods to

† Author for correspondence.

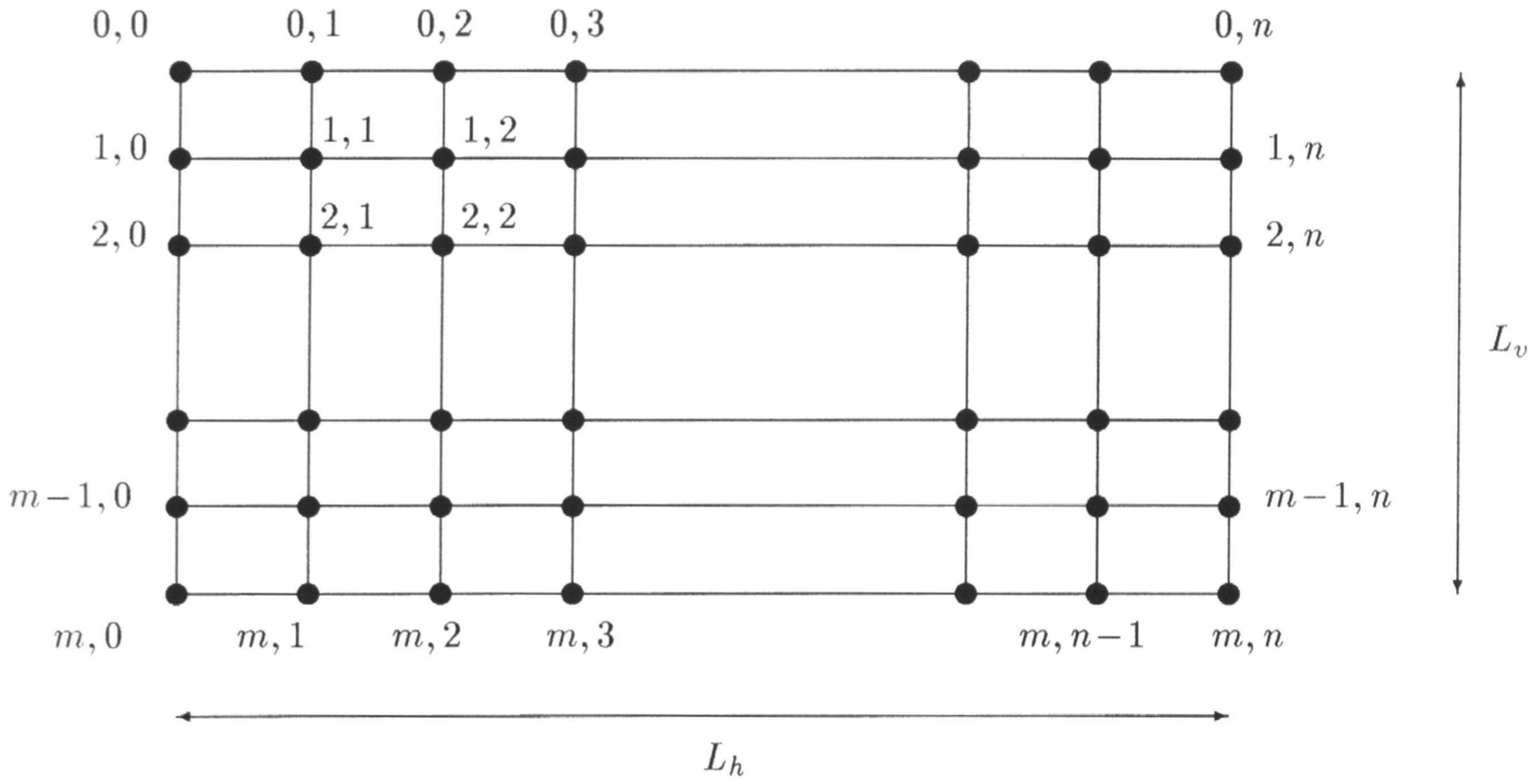


Fig. 1. Numbering of the grid.

create realistic animation of flexible models [5]. This method of Platt and Barr uses reaction constraints for fast computation of collisions of flexible models with polygonal models, and augmented Lagrangian constraints for creating animation effects, such as volume preserving squashing, and the molding of taffy-like substances. To model the flexible objects, the finite element method is used in Platt and Barr's work.

Thingvold and Cohen [6] define a model of elastic and plastic B-spline surfaces which supports both animation and design operations. They develop 'refinement' operations for spring and hinge B-spline models which are compatible with the physics and the mathematics of B-spline models. Their model can be viewed as a continuous physical representation of a physical model rather than the more standard discretized geometry point mass models. The motion of their models is controlled by assigning different physical properties and kinematic constraints on various portions of the surface.

In [7], an approach to imposing and solving geometric constraints on parameterized models is given. This approach is applicable to animation as well as model construction. Constraints are expressed as energy functions, and constraint satisfaction is achieved by solving energy minimization problems. Although this approach is not as realistic as the above three approaches because of the lack of physics, it is simple and general.

Metaxas and Terzopoulos [8] propose an approach for creating dynamic solid models capable of realistic physical behaviors starting from common solid primitives such as spheres, cylinders, cones, and superquadrics. Such primitives can 'deform' kinematically in simple ways. For example, a cylinder deforms as its radius (or height) is changed. To gain additional modeling power they allow the primitives to undergo parameterized global deformations (bends, tapers, twists, shears, *etc.*). Even though their models' kinematic behavior is stylized by the particular solid primitives used, the models behave in a physically correct way with prescribed mass distributions and elasticities. Metaxas and Terzopoulos also proposed efficient constraint methods for connecting the dynamic primitives together to make articulated models.

Gourret *et al.* [9] simulate deformations between objects and the hand of a synthetic human character during a grasping process. They use a numerical method based on finite element theory which allows them to take into account the active forces of the fingers on the object and the reactive forces of the object on the fingers. Their solution to the grasping problem is based on displacement commands instead of force commands used in robotics and human behavior. The human skin deformations and object deformations are modeled in the same way in their work. This improves the modeling of contacts between them and allows a realistic skin deformation of the human fingers.

Miller [10] propose a model for animating legless figures such as snakes and worms using mass-spring systems. Muscle contractions are simulated by animating the spring tensions in his work. He also includes directional friction due to the surface

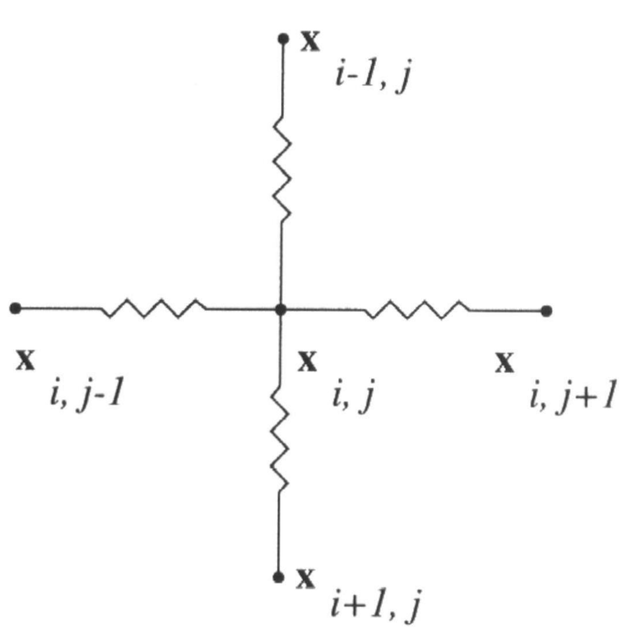
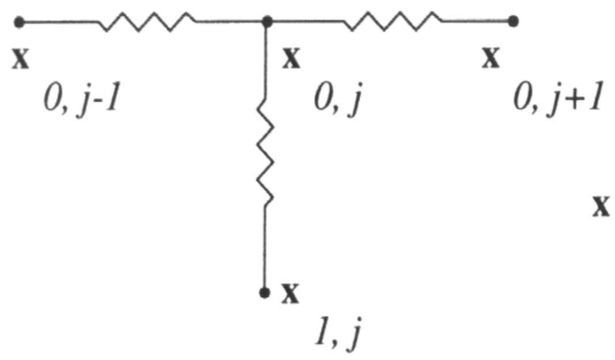
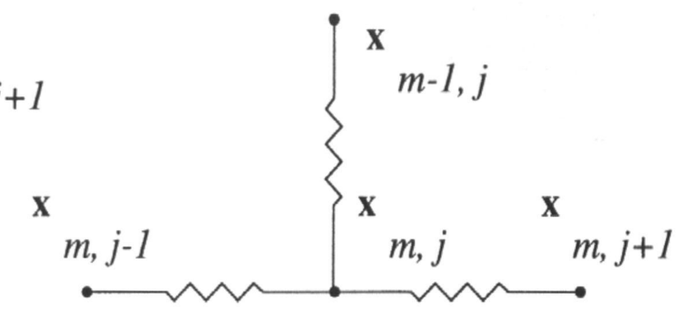


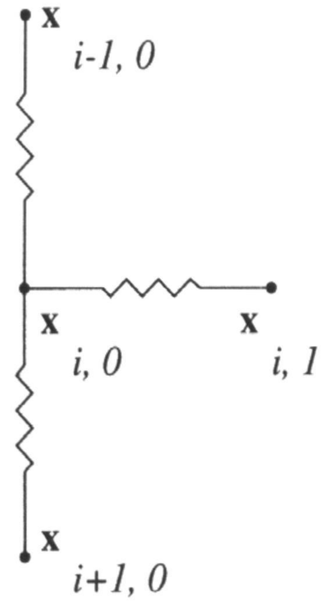
Fig. 2. Interactions (couplings) between grid points (general case).



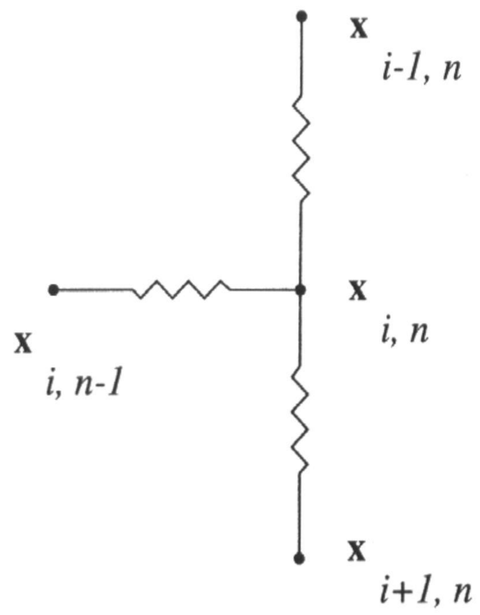
(a) Upper Boundary



(b) Lower Boundary



(c) Left Boundary

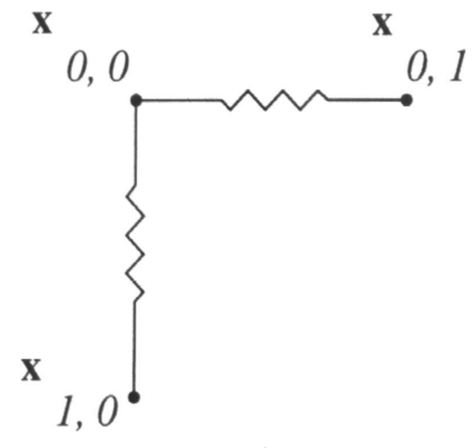


(d) Right Boundary

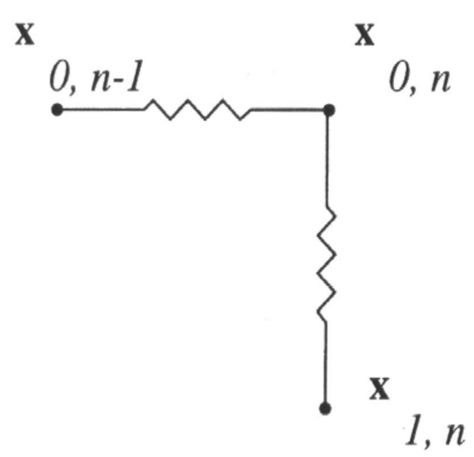
Fig. 3. Interactions (couplings) between grid points (boundaries).

structure in the dynamic model and legless figure locomotion results. Although real snakes and worms have complex internal structures, the simplified model proposed in Miller's work provides an elegant way to simulate the motion dynamics of these creatures.

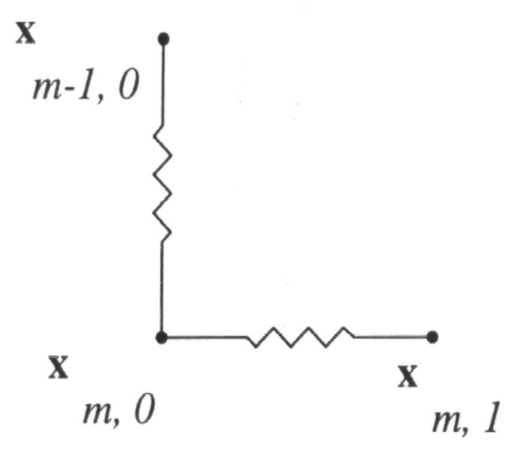
Szeliski and Tonnesen [11] propose a model of elastic surfaces based on interacting particle systems. This model has characteristics of both physically-based surface models and of particle systems. It can be used to model smooth, elastic, moldable surfaces



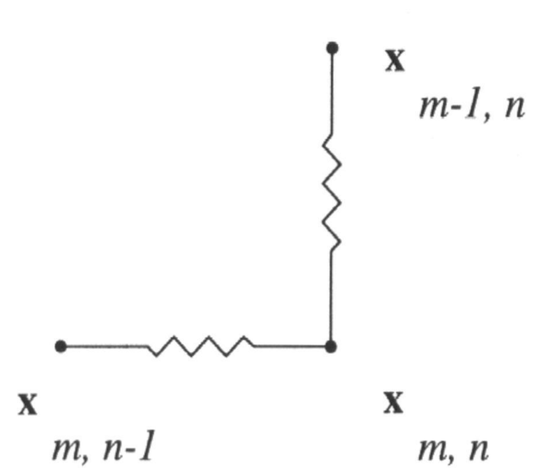
(a) Upper Left corner



(b) Upper Right Corner

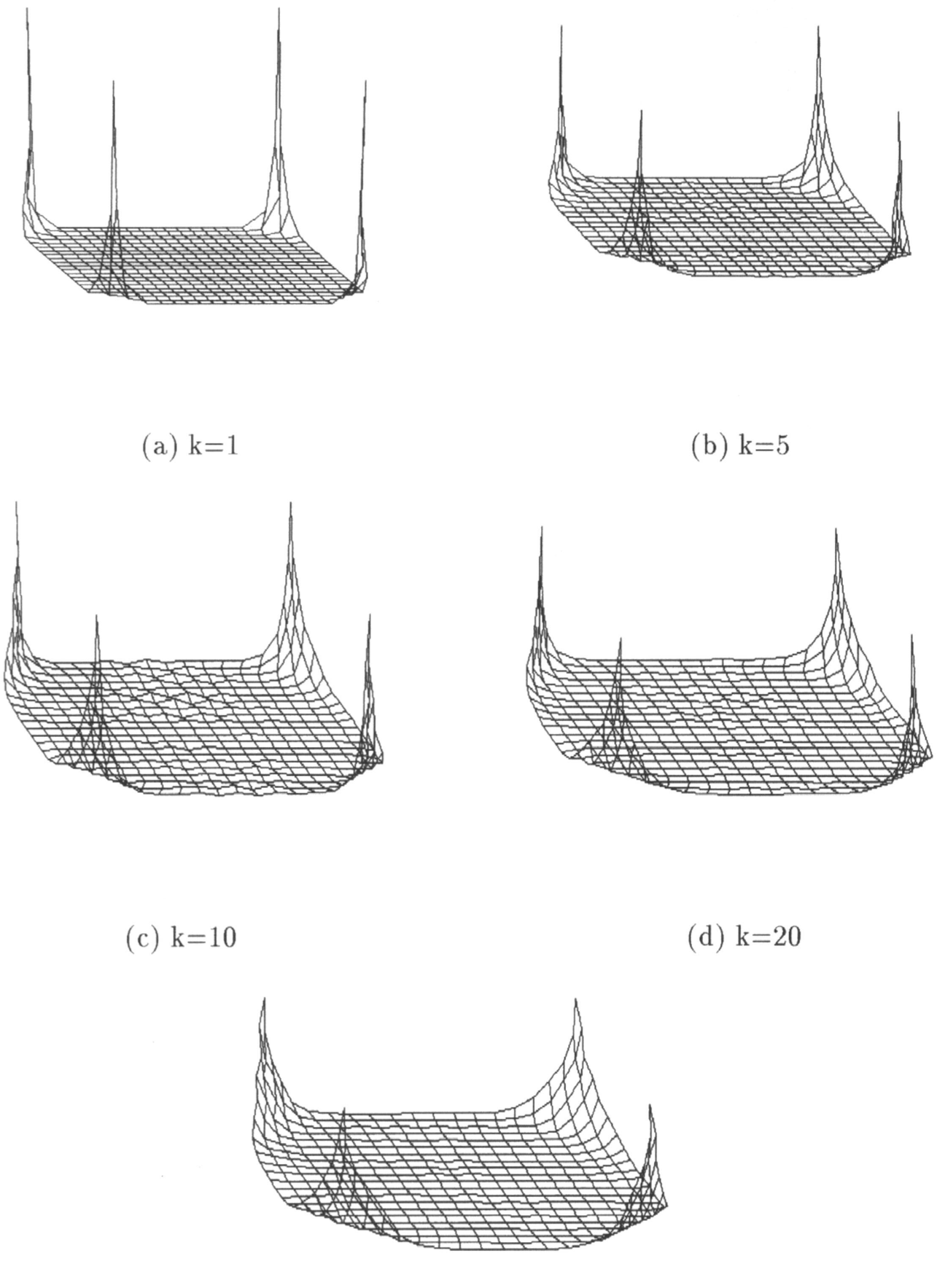


(c) Lower Left Corner



(d) Lower Right Corner

Fig. 4. Interactions (couplings) between grid points (corner points).)



(a) k=1 (b) k=5

(c) k=10 (d) k=20

Fig. 5. Different elastic surfaces, constrained from four corners, fall.

and it allows for arbitrary interactions and topologies. Their model also has the ability to grow new particles. This ability gives the model more fluid-like properties which extends the range of interactions. For example, the surfaces can be joined and cut at arbitrary locations. A potential drawback of their technique is the lack of precise control over the mathematical form of the surfaces.

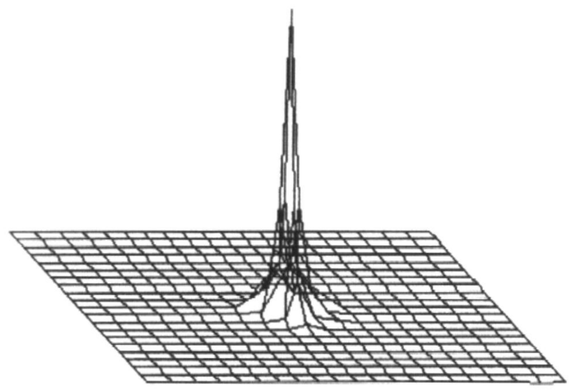
Breen *et al.* [12] propose a physically-based model and a simulation methodology, which when used together are able to reproduce many of the attributes of the characteristic behavior of cloth. Their model utilizes a microscopic particle representation that directly treats the mechanical constraints between the threads in a woven material rather than a macroscopic continuum approximation. Their simulation technique is hybrid, employing force methods for gross movement of the cloth and energy methods to enforce constraints within the material. Although limited only to cloth object behavior in scope, their approach is very realistic since a microscopic particle representation is utilized.

There are other physically-based models of flexible objects which are concerned only with the static shape. Weil [13] propose a geometric approach for interpolating surfaces to produce draped 'cloth' effects. The clothes synthesized with his model contain folds and appear very realistic. The cloth is assumed to be rectangular, and is represented as a grid of three-dimensional coordinates. He uses the catenary curves to define the positioning of the points along a given thread.

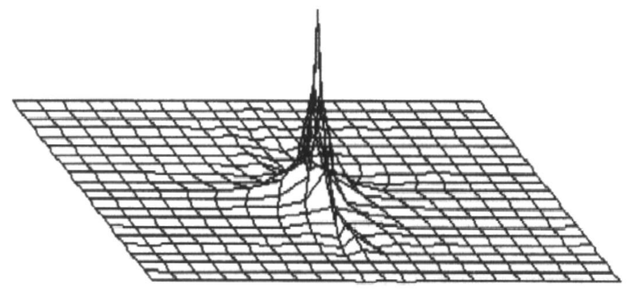
Feynman [14] described a technique for modeling the appearance of cloth. His computational framework minimizes energy functions defined over a grid of points. Feynman derives his functions from the theory of elasticity and from the assumption that cloth is a flexible shell.

1.2. *Organization of the paper*

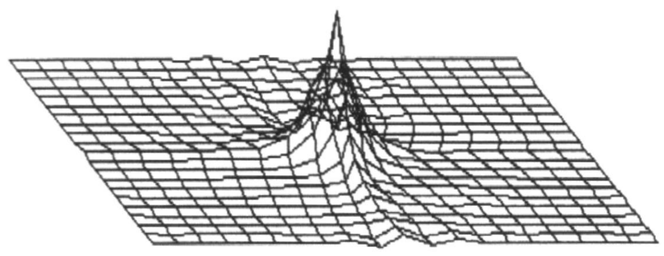
In Section 2, a spring force formulation for animating deformable models is explained together with its implementation details. In Section 3, some simulation results using the spring force formulation are given. Section 4 gives conclusions and suggestions for further research.



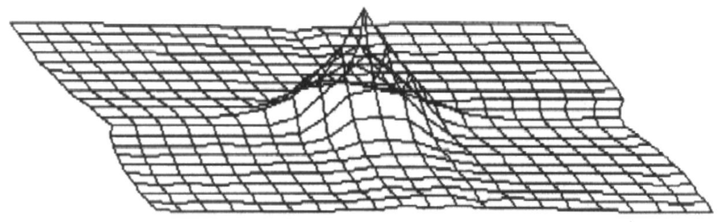
(a) k=1



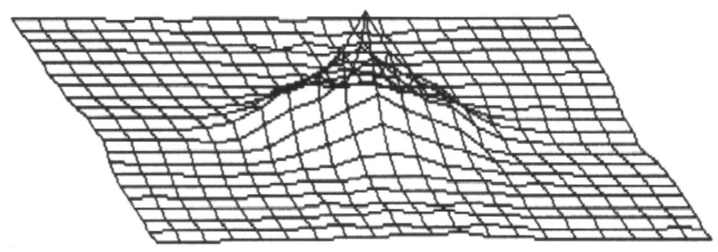
(b) k=5



(c) k=10



(d) k=20



(e) k=30

Fig. 6. Different elastic surfaces, constrained from the center of mass, fall.

2. SPRING FORCE FORMULATION FOR DEFORMABLE MODELS

In other formulations based on elasticity theory (primal [1] and hybrid [2] formulations), the elastic properties of the materials are stored in the stiffness matrix. However, formation of the stiffness matrix automatically is very difficult and sometimes it becomes impossible to solve the differential equations for animating the models because of numerical ill-conditioning problems. In this section, a new formulation for the animation of deformable models, called the spring force formulation, is presented. In this formulation, instead of forming the stiffness matrix automatically, elastic properties are represented as external spring forces. Although handling the elasticities using the stiffness matrix approach is elegant and the most suitable way, our approach is more effective and very fast.

The inter-node spacings on the grid are $h_1 = L_h/n$, $h_2 = L_v/m$ in the horizontal and vertical directions, respectively. Initially, we take $h_1 = h_2 = h$, for simplicity.

We can apply external forces to many of the grid points at the same time. One type of such external force can be the gravitational force. These external forces are known. Besides, if some of the grid points are constrained to fixed positions in space, then there will be some unknown spring (constraint) forces at these points.

The line segments in the grid (Fig. 1) will correspond to the spring elements. According to the initial positions of the grid points, there will be some spring forces on the model.

The equations of motion for a deformable model can be written in Lagrange's form as follows (this should hold for all grid points):

$$\mathbf{M}\frac{d^2}{dt^2}\mathbf{x} + \mathbf{C}\frac{d}{dt}\mathbf{x} + \mathbf{K}(\mathbf{x})\mathbf{x} = \mathbf{f}(\mathbf{x}) \qquad (1)$$

We can take the elastic force expression as an

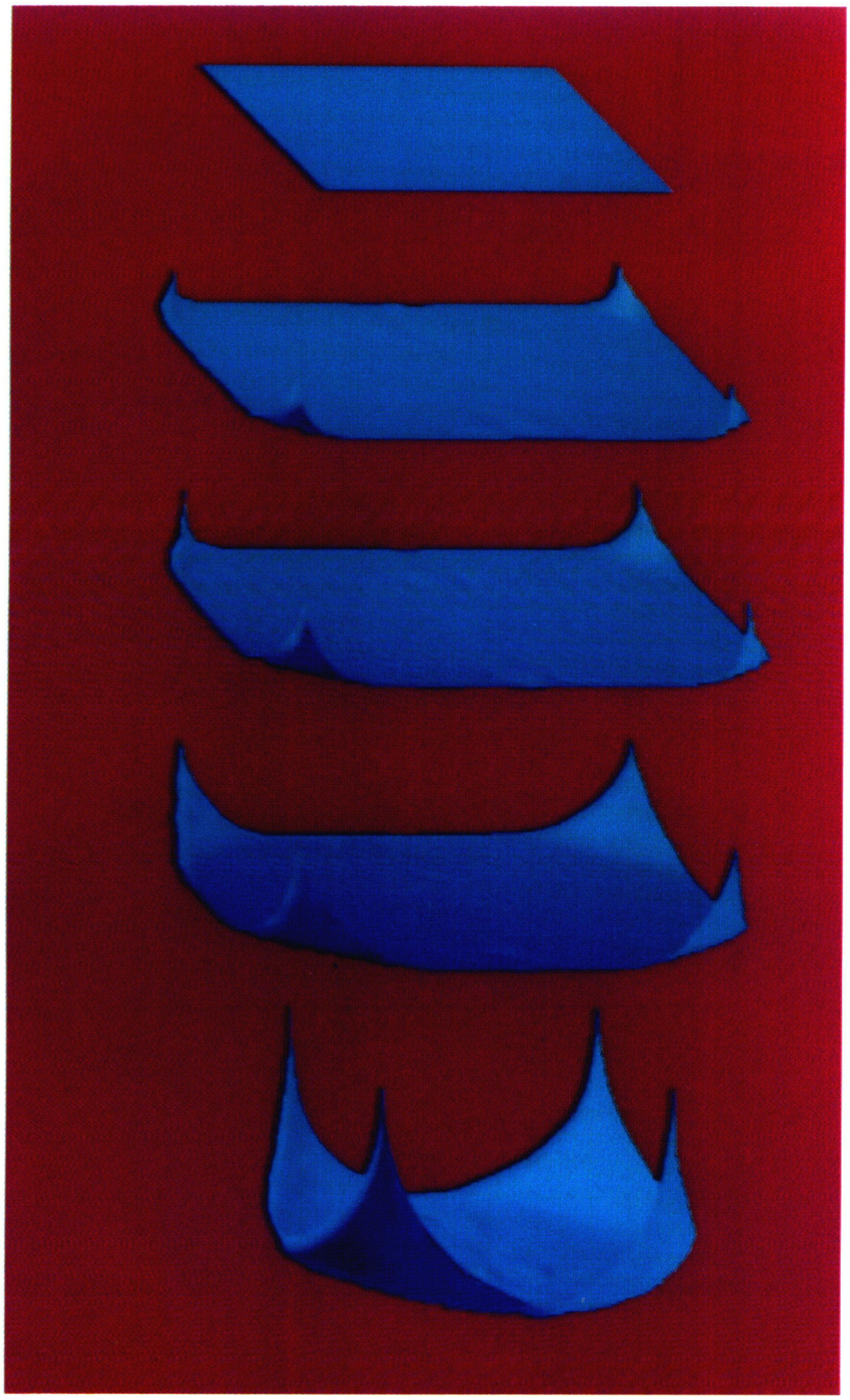


Fig. 7. A stretchy sheet, constrained from its four corners, falls.

external force $\mathbf{f}_{spr} = \mathbf{K}(\mathbf{x})\mathbf{x}$, and take $\mathbf{f}_{spr}$ to the right hand side of the Equation (1). This new form of the equation will simplify the formulation procedure.

The position vector $\mathbf{x}$ for the model points is as follows (T denotes the transpose of a matrix):

$$\mathbf{x}^{\mathbf{T}} = [\mathbf{x}_0^T \mathbf{x}_1^T \cdots \mathbf{x}_m^T] \tag{2}$$

where $\mathbf{x}_i$ represents all the position vectors of the grid points on the ith row, and

$$\mathbf{x}_i^T = \left[\mathbf{x}_{i,0}^T \mathbf{x}_{i,1}^T \cdots \mathbf{x}_{i,n}^T\right] \tag{3}$$

where $\mathbf{x}_{i,j}$ is the position vector of the grid point (i,j) $(i=0,1,\ldots,m;\ j=0,\ 1,\ldots,n)$. In Equation (1), $\mathbf{M}$ is the mass matrix, an $(m+1)(n+1)\times(m+1)(n+1)$ diagonal matrix which contains masses of the grid points as diagonal elements, and $\mathbf{C}$ is the damping matrix, an $(m+1)(n+1)\times(m+1)(n+1)$ diagonal matrix which contains dampers of the grid points as diagonal elements.

Note that Equation (1) can be rewritten as

$$\mathbf{M}\frac{d^2}{dt^2}\mathbf{x} + \mathbf{C}\frac{d}{dt}\mathbf{x} = \mathbf{f}(\mathbf{x}) - \mathbf{f}_{spr}(\mathbf{x}) \tag{4}$$

In this way, there will be no need for calculating the entries of the stiffness matrix. Instead of this, it is necessary to find the expressions for the column matrix f_{spr}^T (external spring forces representing elasticities). The spring force vector can also be partitioned as

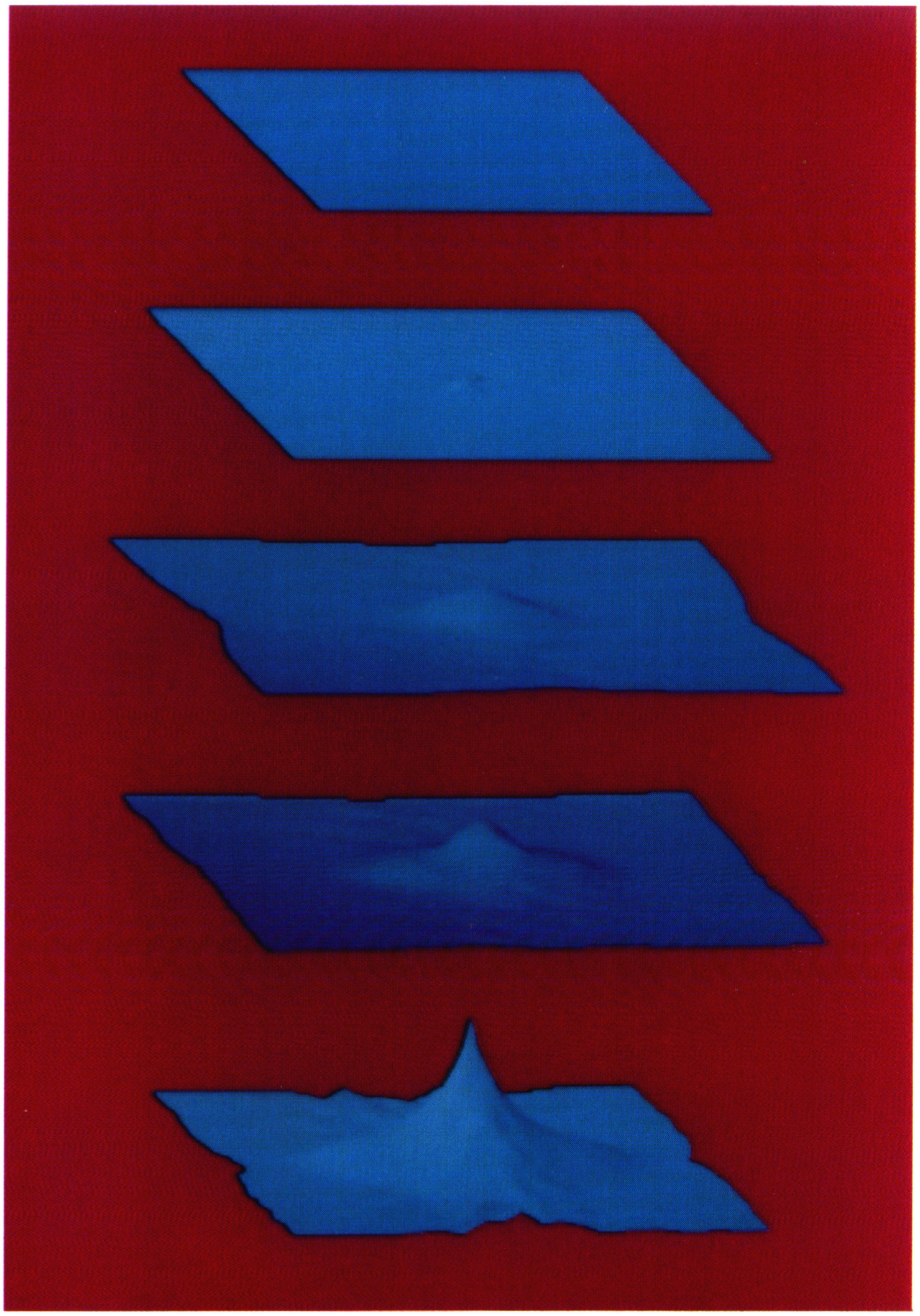


Fig. 8. A stretchy sheet, constrained from its center of mass, falls.

$$\mathbf{f}_{spr}^{T} = [\mathbf{f}_0^T \mathbf{f}_1^T \cdots \mathbf{f}_m^T] \tag{5}$$

where the entries in the vector $\mathbf{f}_i^T = [\mathbf{f}_{i,o}^T \mathbf{f}_{i,1}^T \cdots \mathbf{f}_{i,n}^T]$ correspond to the spring forces acting at the grid points.

Using the discussion in [15](pp. 359–362), the terminal equation of a two-terminal spring component of free length ℓ in three-dimensional space is given as

$$\mathbf{f}_\ell = k\left[(\mathbf{x}_1 - \mathbf{x}_2) - \ell \frac{\mathbf{x}_1 - \mathbf{x}_2}{\|\mathbf{x}_1 - \mathbf{x}_2\|}\right] \tag{6}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ are the position vectors of its terminal points. Note that calculation of the vector $(\mathbf{x}_1 - \mathbf{x}_2)$ is essential; it also appears in the second term of this expression. Equation (6) can be used to obtain expressions for the entries of $\mathbf{f}_{spr}$ in Equation (5).

For the grid points not on the boundaries, the elastic force is calculated by adding the spring forces applied to the grid point by its four neighbors.

If $i = 1,2,\ldots,m-1$ and $j = 1,2,\ldots,n-1$ then (see Fig. 2)

$$\begin{aligned}
\mathbf{f}_{ij} = {} & k\left[(\mathbf{x}_{i,j} - \mathbf{x}_{i,j-1}) - \ell \frac{\mathbf{x}_{i,j} - \mathbf{x}_{i,j-1}}{\|\mathbf{x}_{i,j} - \mathbf{x}_{i,j-1}\|}\right] \\
& + k\left[(\mathbf{x}_{i,j} - \mathbf{x}_{i-1,j}) - \ell \frac{\mathbf{x}_{i,j} - \mathbf{x}_{i-1,j}}{\|\mathbf{x}_{i,j} - \mathbf{x}_{i-1,j}\|}\right] \\
& + k\left[(\mathbf{x}_{i,j} - \mathbf{x}_{i,j+1}) - \ell \frac{\mathbf{x}_{i,j} - \mathbf{x}_{i,j+1}}{\|\mathbf{x}_{i,j} - \mathbf{x}_{i,j+1}\|}\right] \\
& + k\left[(\mathbf{x}_{i,j} - \mathbf{x}_{i+1,j}) - \ell \frac{\mathbf{x}_{i,j} - \mathbf{x}_{i+1,j}}{\|\mathbf{x}_{i,j} - \mathbf{x}_{i+1,j}\|}\right]
\end{aligned} \tag{7}$$

For the grid points on the boundaries, three

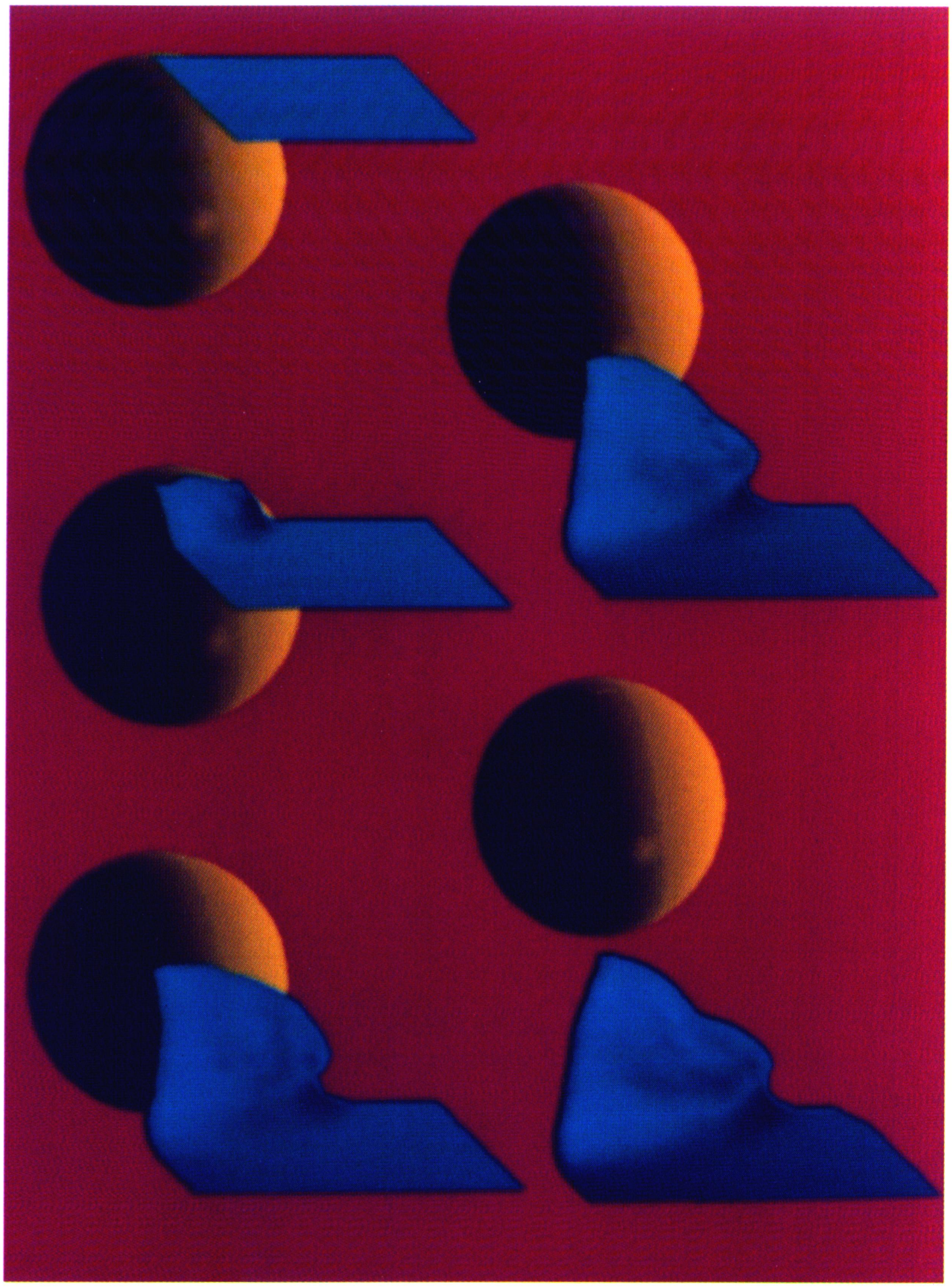


Fig. 9. A piece of cloth collides with an impenetrable ellipsoid.

neighbors have an effect on the elastic force (Fig. 3). For the grid points on the corners, only two neighbors have an effect on the elastic force (Fig. 4). The elastic force expressions for the grid points on the boundaries and corners can be easily derived in a similar way.

2.1. *Implementation of the spring force formulation*

- Since the initial position vectors of the grid points are known, the vector $\mathbf{f}_{spr}$ can be calculated from the external spring force equations.
- Then by solving the differential equation in Equation (4) at the first step, next values of the position vectors of the grid points are determined.
- The next value of the vector $\mathbf{f}_{spr}$ is calculated and the process is repeated.

As initial positions, we have $h \neq \ell$ in general. Therefore $\mathbf{f}_{spr} \neq 0$. In other words, there will be some internal stresses in the system. If $h = \ell$, then $\mathbf{f}_{spr} \equiv 0$. On the other hand, if $h_1 \neq h_2$, then $\mathbf{f}_{spr} \neq 0$ initially (assuming that all the springs have the same lengths). We may select the lengths of the horizontal springs as $\ell_1 = h_1$ and the lengths of the vertical springs as $\ell_2 = h_2$. In this case, $\mathbf{f}_{spr} = 0$ initially, and some of the ℓ factors will change to ℓ_1 and the remaining ones to ℓ_2 in the external spring force equations. Other modifications are also possible; *e.g.* on the spring coefficients (k).

3. SIMULATION EXAMPLES USING SPRING FORCE FORMULATION

In the spring force formulation, by setting the stiffness constants to different values it is possible to obtain different elastic properties. In Fig. 5, a flat surface, which is assigned different elastic properties by setting the spring constant k to different values and constrained from its four corners, is animated.

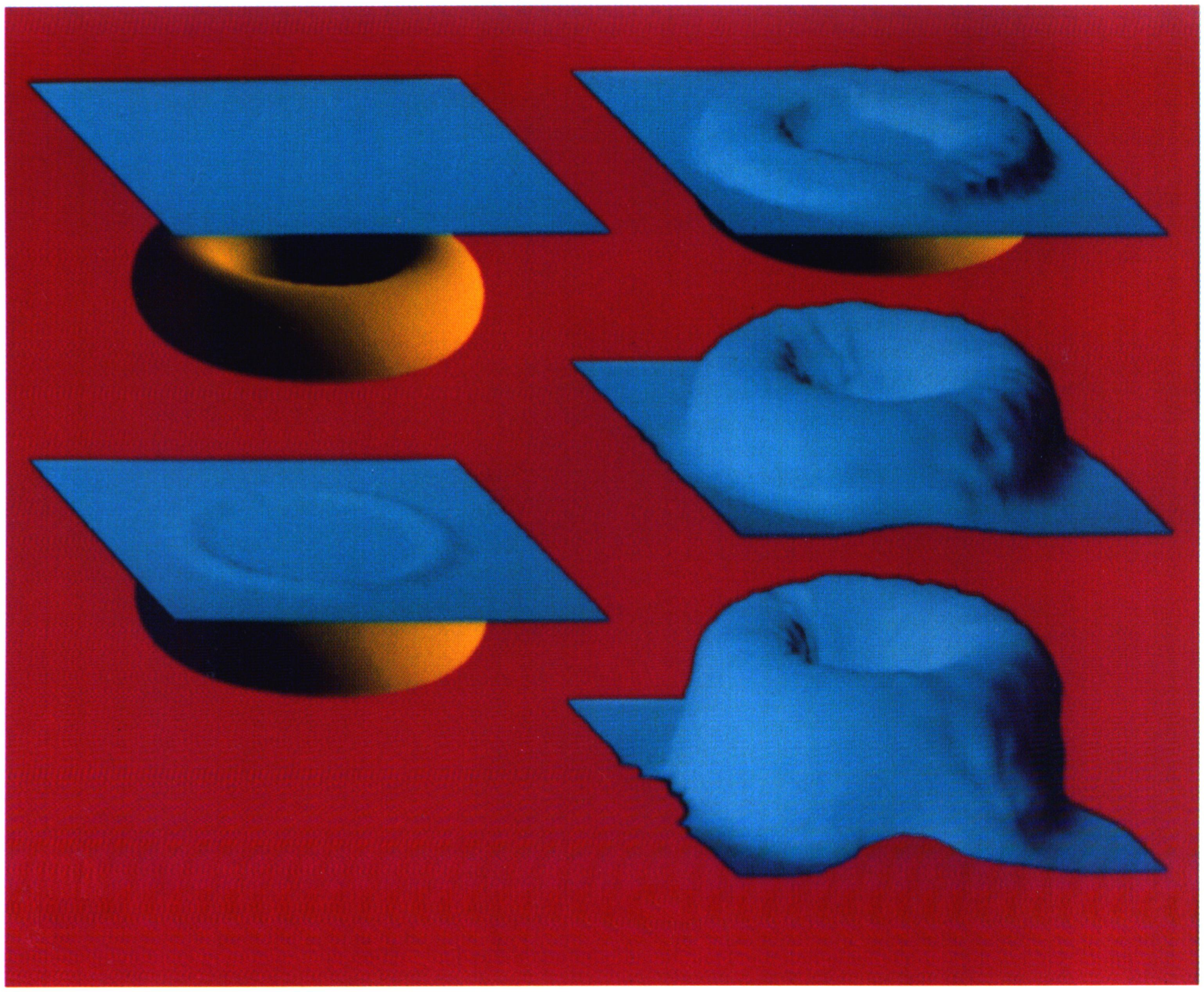


Fig. 10. A stretchy sheet drops over a toroid.

The surfaces in each part of the figure correspond to a different animation of the same surface with different elastic properties. Each part of the figure shows the surface after a specific number of animation frames, which is the same in Fig. 5(a), (b), (c), (d) and (e).

Figure 6 gives the animation frames for the same flat surface, which is assigned different elastic properties and constrained from its center of mass. This figure also shows how objects with different elastic properties can be modeled by setting spring constants to different values.

In Fig. 7, a stretchy sheet constrained from its four corners falls with the effect of gravity. In Fig. 8, a stretchy sheet constrained from its center of mass falls. In Fig. 9, a piece of cloth collides with an impenetrable obstacle, which is an ellipsoid. In Fig. 10, a stretchy sheet drops over a toroid. In Fig. 11, an elastic surface drops over a toroid with a very small hole. In Fig. 12, an elastic surface passes through a toroid.

Any point on a model could be constrained to a fixed location in space so that when the model is animated, the constrained points remain in their initial positions. The constraint forces are taken into account in the following way. When a constrained point tends to move, an opposite force for bringing it back to its original position is calculated and added to the total external force for that point. Each constrained point has an effect on the total external force for all points in the model depending on the difference between the body coordinates of the points. This coupling effect is taken into account automatically according to the elastic properties of the models. The constraint force that connects a material point u_0 on a deformable model to a point $\mathbf{p}_0$ in space by a spring is

$$\mathbf{f}_s(u,t) = k(\mathbf{p}_0 - \mathbf{x}(u_0,t))\delta(u-u_0) \tag{8}$$

where k is the spring constant and δ is the unit delta function [1].

The forces due to the collision of deformable models with impenetrable obstacles are calculated using the obstacle's implicit (inside–outside) function. The obstacle exerts a repulsive force on the deformable model which can be calculated as a function of the obstacle's implicit function such that the force grows quickly if the model attempts to penetrate the obstacle. This is achieved by creating a potential energy function $c\ exp(f(\mathbf{x})/\xi)$ around each obstacle, where f is the obstacle's implicit function, and c and ξ are constants determining the properties of the obstacle. The repulsive force due to an impenetrable obstacle (expressed using the gradient ∇ of the potential energy function) is

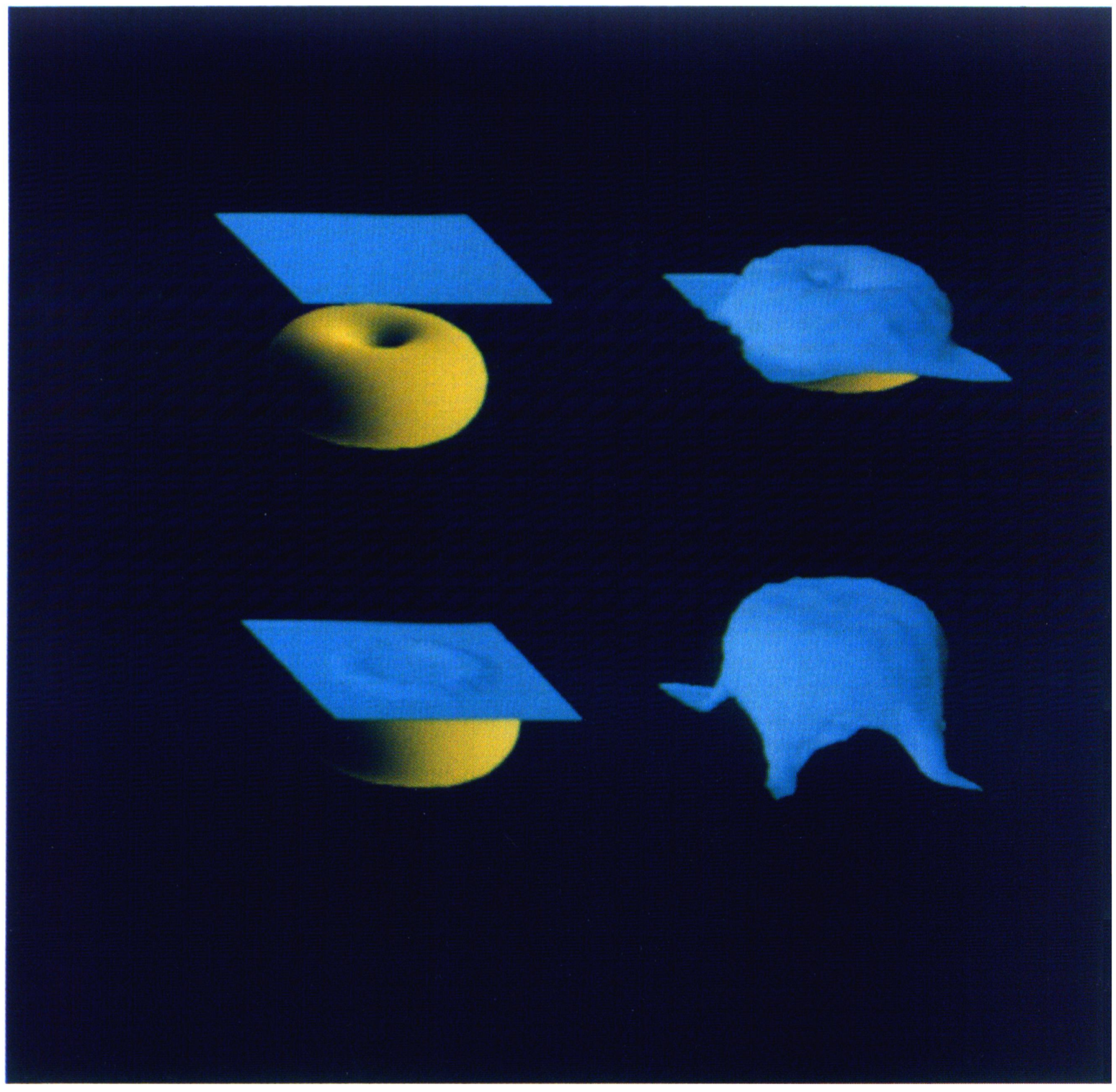


Fig. 11. An elastic surface drops over a toroid with a small hole.

$$\mathbf{f}_c(u,t) = -c((\nabla f(\mathbf{x})/\xi)\exp(-f(\mathbf{x})/\xi)\cdot\mathbf{n})\mathbf{n}, \quad (9)$$

where $\mathbf{n}(u,t)$ is the unit surface normal vector of the deformable body's surface [1].

4. PROCESSING TIMES FOR THE SPRING FORCE FORMULATION

The graph in Fig. 13 gives the processing times of the animations of the Bézier surfaces of different sizes, using the spring force formulation. Times are measured on a Sun/Sparc Server (Sparc Processor). The processing times for each frame given in the tables include

- the time for calculating the external forces (gravitational, constraint, and collision forces) for each model point,
- the time for calculating elastic forces (which are modeled as external spring forces) between model points,
- the time for calculating the 3-D positions of model points, and
- wireframe rendering time of the calculated frame.

Processing times for the models increase almost linearly with the number of model points since the equations to be solved to calculate the 3-D positions of model points contain only diagonal entries and can be solved in linear time.

Calculation of the external forces and elastic forces, and wireframe rendering of the calculated frames also increase linearly with the number of model points.

Although the proposed spring force formulation seems to be a drastic simplification, it provides a simple and very fast technique for the animation of deformable models. Deformable surfaces containing 1000 model points can be animated at interactive speeds. Since the deformation forces at one point effect only the neighbors of that point, the method provides a rough simulation model. Therefore, it can be used for applications not requiring a very high accuracy.

If the stiffness matrices had to be formed to model elastic properties, the processing times would increase quadratically with the number of model points.

5. CONCLUSION

A new formulation for animating deformable models, called the spring force formulation, is presented in this paper.

5.1. *Contributions of the paper*

Contributions of the paper could be summarized as follows:

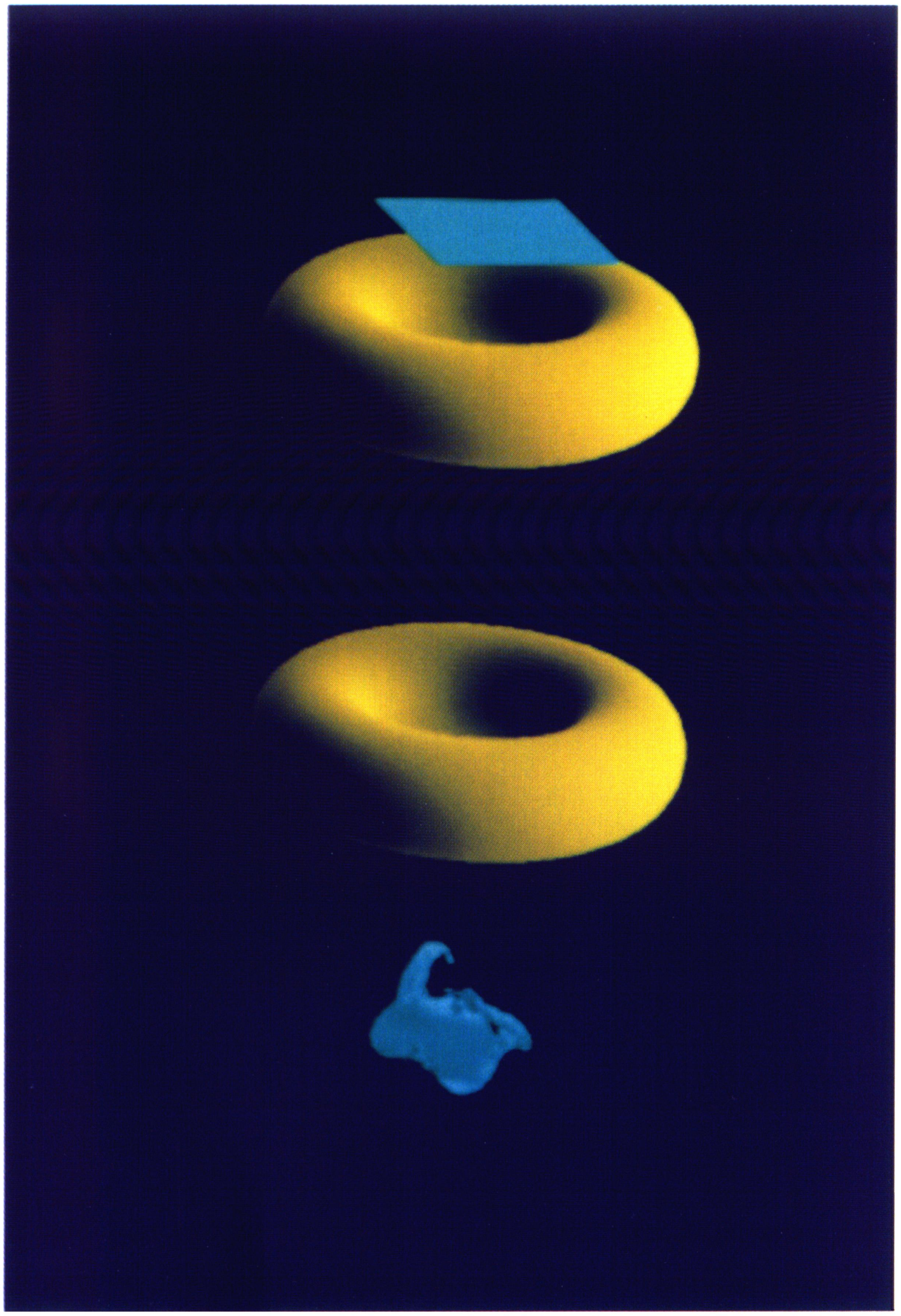


Fig. 12. A small elastic surface passes through a toroid.

- In the spring force formulation that is presented, the elastic properties of the materials are represented as external spring forces, instead of using the stiffness matrix approach. In this way, the problem of automatically constructing the stiffness matrix is avoided.
- Since the stiffness matrix is not formed, models could be animated faster than the other approaches. The linear system of equations that should be solved to compute animation frames contains only mass and damping values which are the diagonal entries. This allows us to use simple linear system solving methods.
- The elastic properties of the materials could be given by setting the spring constants to proper values.
- Since the formulation models a deformable object using a finite number of grid points, it is possible to give different elastic properties to different parts of a model.

5.2. *Future research directions*

Future extensions to the research explained in this paper could be summarized as follows:

- The equations of motion proposed for deformable models could be modified in such a way that new types of constraints will be taken into account by using external forces. This approach allows modeling and animating articulated bodies consisting of rigid and nonrigid parts by creating complex models from simpler primitives using

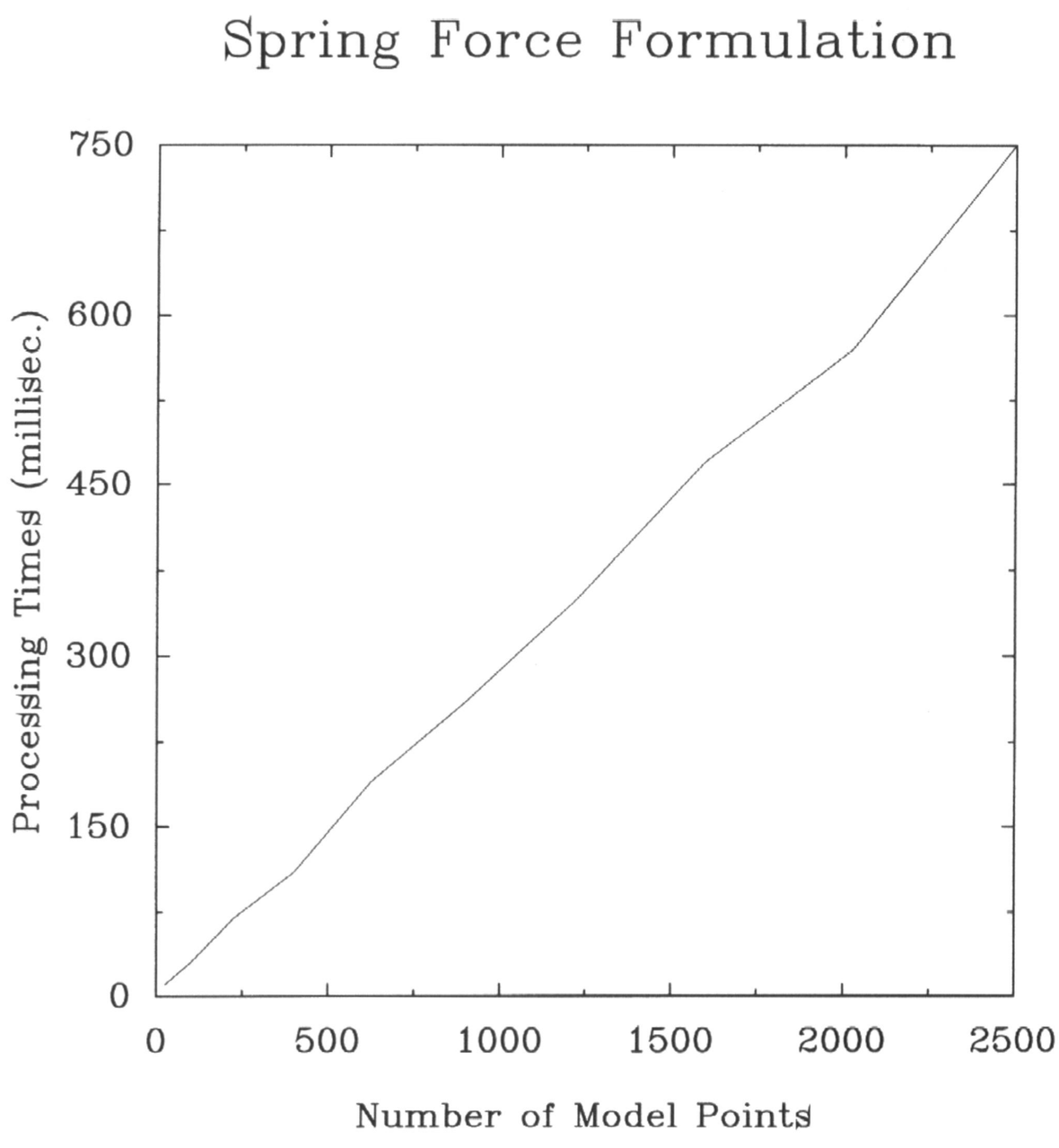


Fig. 13. Processing times using the spring force formulation.

point-to-point constraint. Also, other constraints, such as point-to-path and orientation, could be used to control the motion of the models.

- The current implementation of deformable models allows the animation of a single object. The implementation could be improved to animate more than one object at the same time. Parallelism could be utilized for this in the sense that different processors handle the motion of different objects and a host processor animate all the objects. Collisions of the objects could be detected by the host processor to prevent the objects sailing through each other.

REFERENCES

1. Terzopoulos, D., Platt, J., Barr, A. H. and Fleischer, K., Elastically deformable models. *ACM Computer Graphics (Proc. SIGGRAPH'87)*, 1987, **21,** 205–214.
2. Terzopoulos, D. and Fleischer, K., Modeling inelastic deformation: viscoelasticity, plasticity, fracture. *ACM Computer Graphics (Proc. SIGGRAPH'88)*, 1988, **22,** 269–278.
3. Witkin, A., Gleischer, M. and Welch, W., Interactive dynamics. *ACM Computer Graphics (Proc. SIGGRAPH'90)*, 1990, **24,** 11–22.
4. Pentland, A. and Williams, J., Good vibrations: model dynamics for graphics and animation. *ACM Computer Graphics (Proc. SIGGRAPH'89)*, 1989, **23,** 215–222.
5. Platt, J. and Barr, A. H., Constraint methods for flexible models. *ACM Computer Graphics (Proc. SIGGRAPH'88)*, 1988, **22,** 279–288.
6. Thingvold, J. A. and Cohen, E., Physical modeling with B-spline surfaces for interactive design and animation. *ACM Computer Graphics (Proc. SIGGRAPH'90)*, 1990, **24,** 129–137.
7. Witkin, A., Fleischer, K. and Barr, A. H., Energy constraints on parameterized models. *ACM Computer Graphics (Proc. SIGGRAPH'87)*, 1987, **21,** 225–232.
8. Metaxas, D. and Terzopoulos, D., Dynamic deformation of solid primitives with constraints. *ACM Computer Graphics (Proc. SIGGRAPH'92)*, 1992, **26,** 309–312.
9. Gourret, J.-P., Thalmann, N. M. and Thalmann, D., Simulation of object and human skin deformations in a grasping task. *ACM Computer Graphics (Proc. SIGGRAPH'89)*, 1989, **23,** 21–30.
10. Miller, G. S. P., The motion dynamics of snakes and worms. *ACM Computer Graphics (Proc. SIGGRAPH'88)*, 1988, **22,** 169–178.
11. Szeliski, R. and Tonnesen, D., Surface modeling with oriented particle systems. *ACM Computer Graphics (Proc. SIGGRAPH'92)*, 1992, **26,** 175–180.
12. Breen, D. E., House, D. H. and Getto, P. H., Physically-based particle model of woven cloth. *The Visual Computer*, 1992, **8,** 264–277.
13. Weil, J., The synthesis of cloth objects. *ACM Computer Graphics (Proc. SIGGRAPH'86)*, 1986, **20,** 49–54.
14. Feynman, C. R., Modeling the appearance of cloth. Master's thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, 1986.
15. Tokad, Y., *Analysis of Engineering Systems—Part III.* Bilkent University, Ankara, Turkey, 1990.