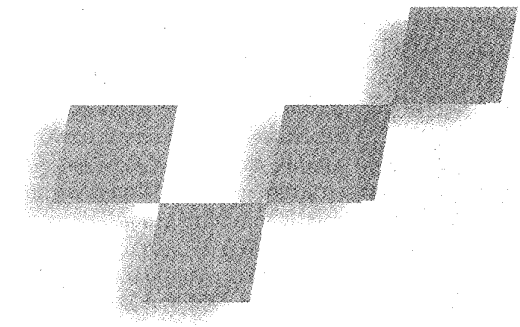


A Fast, Flexible, Particle-System Model for Cloth Draping



Bernhard Eberhardt, Andreas Weber,
and Wolfgang Strasser
University of Tübingen, Germany

For centuries people have admired the flowing robes in works by such artists as Raphael and Michelangelo. The sculpted or painted folds of cloth not only demonstrate the artist's technical skill but also reinforce the illusion of reality. The draping of virtual cloth is a challenge not only in art, but also in the generation of virtual worlds in computer science.

Engineers, mathematicians, and physicists have developed theories and accurate models to predict the behavior of stiff membranes made from materials such

as steel and wood. Good mechanical models for textiles, however, have only recently become available. One reason is that textiles undergo large deformations. To determine qualitative behavior, textile industries use standardized measurements like the Kawabata evaluation system.¹

In computer graphics, Weil² came up with an approach in 1986 that produced an image of "cloth draping" by interpolating surfaces between curves. More recently, computer scientists and engineers have used finite elements to model cloth.³⁻⁵

Terzopoulos⁶ tried to establish a physical basis for the behavior of deformable materials. He solved problems numerically by describing

large deformations as the result of small forces and by calculating the deformation energy of the material only from the given geometrical data. His model describes the physical behavior of cloth exactly and distinguishes it from any stiff material such as iron, plastic, or aluminum.

Breen, House, and Wozny⁷ proposed an approach that used coupled particles to model the cloth through a minimized energy function. Their work motivated the work presented here. However, we use a different, faster technique to calculate the exact particle trajectories. Our

model incorporates cloth-specific properties, such as hysteresis of the forces, as well as measured experimental data. Moreover, flexibility in the energy functions supports the simulation of effects like the crease in trousers.

Particle systems

Coupled particle systems first appeared in computer graphics in the work of Reynolds.⁸ He described a particle system as a set of particles interacting with each other according to certain laws imposed by the problem. A particle system offers a viable solution to a problem if you know how the position of one element changes over time depending on the properties of the other elements. The power of this idea is evident in the many recent applications of these systems, ranging from molecular dynamics to the calculation of colliding galaxies.

In their significant paper on the use of particle systems to model the drape of woven cloth, Breen, House, and Wozny⁷ described woven fabrics as a set of mass points falling in a gravitational field according to physical laws. This description is suitable for obtaining realistic predictions of the interactions of draping cloth with various environments.

The system described here extends the model of Breen, House, and Wozny. We introduce techniques to model measured force data exactly and thus cloth-specific properties such as anisotropic behavior and hysteresis. Moreover, we extended the particle system to model air resistance. Our system allows a dynamic simulation that shows the effects on falling cloth from air resistance, wind, moving bodies, and surface friction.

The physical basis

The starting point of modeling the drape of cloth via particle systems is the simulation of the fall or draping motion of a cloth in some environment such as air, water, or even oil. This dynamical simulation requires calculation of the exact trajectories of a falling particle. The trajectory is described through a physical model of the cloth and must include all forces acting on a single particle. The superposition of these forces gives the direction and acceleration of the moving particle.

Animating the drape of
different cloths must
address complex physical
behaviors. This particle
approach uses optimizations
that make it faster than
earlier implementations and
allow it to simulate behavior
over time.

The weft and warp directions give the internal structure of a woven cloth. A rectangular grid can model this structure in a natural way.⁷ For a physical model of cloth, we consider the following “internal forces”:

- tension
- shearing
- bending

All three forces depend strongly on the type of cloth to be modeled. Kawabata investigated these forces in detail.¹ Figure 1 presents an example of a Kawabata plot for the experimental data of a cotton shearing test. His experiments showed that

- the forces are linear only for small elongations (deformations);
- linear functions can approximate the shearing and bending forces for smaller angles quite well;
- over the full range of possible bend and shear angles, shearing and bending forces are nonlinear and show hysteresis effects;
- the forces are different in the weft and warp directions; and
- there are maximum forces in the sense that larger forces will tear the cloth.

The trajectory of a particle depends not only on the internal structure of the cloth, but also on the surrounding environment. We must therefore also consider “external forces” such as

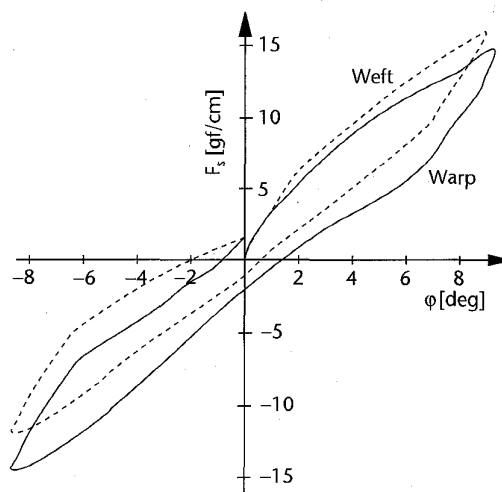
- gravitation;
- air resistance;
- interactions with solid bodies, specifically, friction at the surface and reactive forces prohibiting penetration of solid bodies and penetration of the cloth with itself; and
- blowing of air, moving bodies.

The latter two forces can be modeled easily because they depend only on local properties. Air resistance, however, depends on the global shape of the falling cloth. The air resistance coefficient c_w varies from 0.34 for a convex hemisphere to 1.1 for a square and 1.33 for a concave hemisphere.⁹ Ideally, an individual parameter should be set for each particle.

Mathematical aspects

Consider a set of n particles. Given the masses of the particles, we can describe the state of our particle system at time t by the sets $x_0, x_1, x_2, x_3, \dots, x_{3n-1}$ and $v_0, v_1, v_2, v_3, \dots, v_{3n-1}$, where the vectors (x_0, x_1, x_2) , (x_3, x_4, x_5) , and so forth are the locations of our particles, and (v_0, v_1, v_2) , (v_3, v_4, v_5) , ... are the velocities respectively at time t .

The forces described above are pushing the particles through space. Using forces to calculate the trajectory of a particle is difficult, since we must deal with the superposition of many forces. In many cases, however, forces are given as the gradient of a potential or energy function V . D'Alembert and Lagrange elucidated the advantages of using potentials two centuries ago. The



1 A Kawabata plot of a cotton shearing test.

Lagrange equation

$$\frac{d}{dt} \frac{\partial L}{\partial v_i} = \frac{\partial L}{\partial x_i} \quad (1)$$

gives a second-order differential equation for each coordinate of a moving particle, where $L = E_{\text{kin}} - V$ is the Lagrange function with V being a cloth-specific potential and E_{kin} the kinetic energy of the system (for example, see Arnold¹⁰). The Lagrange equation fully describes the trajectory of the moving particle.

What does the potential V look like for our cloth model? As described above we consider tension, shearing, and bending as internal forces and therefore get the respective energy functions E_{t_i} , E_{s_i} , and E_{b_i} for a particle i . In addition, each particle has a potential energy E_{pot_i} which is the usual energy depending on the height of the particle. Adding up all energies of all particles we get the Lagrange function

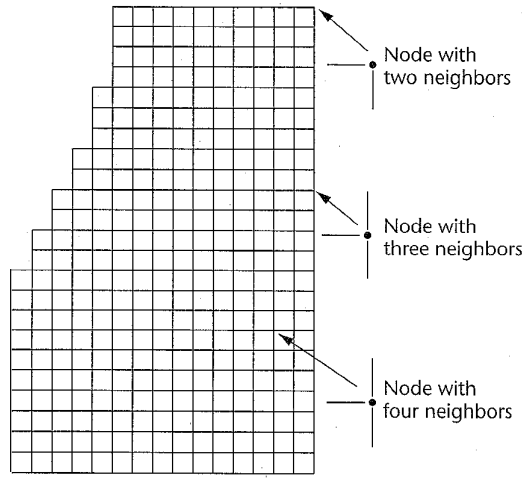
$$L = \sum_{i=0}^{n-1} E_{\text{kin}_i} - \left(\sum_{i=0}^{n-1} E_{\text{pot}_i} + \sum_{i=0}^{n-1} E_{t_i} + \sum_{i=0}^{n-1} E_{s_i} + \sum_{i=0}^{n-1} E_{b_i} \right) \quad (2)$$

where n is the number of particles. As we will see later, our potential V depends only on the location $x_0, x_1, x_2, x_3, \dots, x_{3n-1}$ of the particles, whereas the kinetic energy E_{kin} is a function of $v_0, v_1, v_2, \dots, v_{3n-1}$. Thus, L is now a function of dimension 6 times the number of particles.

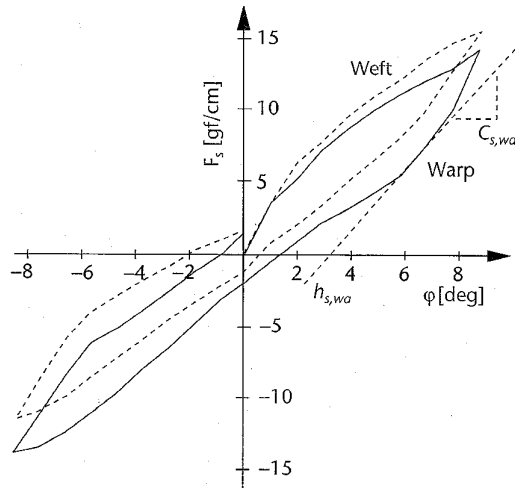
The dimension of the function is not a serious problem, since we now take the partial derivative of L on the right side in Equation 1 with respect to the coordinate of a single particle. We obtain a “simple” function for the differential equation, because most of the variables of L vanish. The only variables remaining are the coordinates of coupled particles. Moreover, because of the symmetry of our particle grid, this function has the same form for most of the particles, differing only in the indices. We only need to consider the three cases shown in Figure 2 (next page), depending on the number of neighbors, that is, nodes with two, three, and four neighbors.

The partial differential equation is now reduced to a

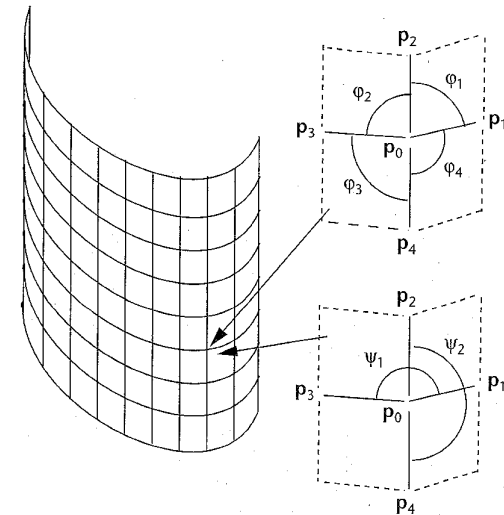
2 Three different cases of nodes.



3 Piecewise linear Kawabata plot of shearing force in Figure 1.



4 Particle grid.



system of ordinary differential equations, one for each coordinate of each particle. Several options exist for solving this set of differential equations.¹¹ We have tried

several techniques such as the Cash-Karp, Runge-Kutta, and Bulirsch-Stoer methods (for example, see below, "Optimizations for numerical methods").

Our cloth model

Since cloth has a weft and a warp direction, a rectangular grid immediately models these two directions (see Figure 2) and thus the anisotropic structure of textiles. We first have to find a suitable description of the so-called internal forces. For this we consider a single particle and investigate the forces acting on it, following the work of Kawabata¹ and Breen, House, and Wozny.⁷

Internal forces

We get empirical cloth data from the Kawabata experiments in the form of force plots of tension, shearing, and bending. Our aim is to model these measured forces as exactly as possible. One way of doing this is to approximate these plots by piecewise linear functions (see Figure 3). We consider now a particle \mathbf{p}_0 at location $\mathbf{p}_0 = (x_0, y_0, z_0) \in \mathbb{R}^3$ with four neighbors $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4 \in \mathbb{R}^3$. We start with shearing and bending energies and assume

$$E_s = \sum_{i=1}^4 \frac{1}{2} C_s (\varphi_i - \frac{\pi}{2} - h_{s_i})^2 \quad (3)$$

$$E_b = \sum_{i=1}^2 \frac{1}{2} C_b (\psi_i - \pi - h_b)^2 \quad (4)$$

The resulting forces for these energy functions are linear, having slopes C_s, C_b . We determine C_s, C_b and h_s, h_b as shown in Figure 3. Thus, for small neighborhoods of our Kawabata plots, we can define the piecewise linear approximations through energies as defined in Equations 3 and 4, modifying C_s, C_b and h_s, h_b , respectively. In addition, these parameters C and h also depend on weft and warp direction (indicated in Figure 3 as $C_{s,wa}$ and $h_{s,wa}$ for a shearing-warp approximation). In our implementation we have stored different values of C and h for every single degree, which gives a quite good approximation of the force plots (compare Figures 1 and 3). Once we know the history of φ and ψ (an easy task in our dynamic model), we can substitute the appropriate values into the energies and hence later in the calculations. Moreover, if we substitute π in the bending energy E_b by a different value, we can model, for instance, the crease in trousers.

The tension energy is implemented as follows:

$$E_t = \begin{cases} \sum_{i=1}^4 \frac{1}{2} C_{t,i} ((\mathbf{p}_0 - \mathbf{p}_i) - d_i - h_{t_i})^3 & \text{if } \mathbf{p}_0 - \mathbf{p}_i \geq d_i \\ \sum_{i=1}^4 \frac{1}{2} C_{t,i} ((\mathbf{p}_0 - \mathbf{p}_i) - d_i - h_{t_i})^5 & \text{if } \mathbf{p}_0 - \mathbf{p}_i < d_i \end{cases} \quad (5)$$

Here d_i denotes the unstretched distance between par-

ticle \mathbf{p}_0 and \mathbf{p}_i . It lets us distinguish between repelling and stretching forces on the textiles. Again C_{i,t_1} and C_{i,t_2} depend on weft and warp direction. Breen, House, and Wozny⁷ took slightly different polynomials of degree 4 and 5. Their polynomials and the ones we use give quite similar results for the tension energy, if the constants are set correctly. These tension energies have proved appropriate for modeling woven cloth.⁷

For the Lagrange function we also need the following energies:

$$E_{\text{kin}} = \frac{1}{2} m_i \cdot \mathbf{v}_i^2 \quad (6)$$

and the potential energies

$$E_{\text{pot}} = m_i \cdot g \cdot z_0 \quad (7)$$

With Equations 3 through 7 we have defined the energies for particles with four neighbors. To get the Lagrange function, we must add all the energies for the n particles and eliminate the redundant energies (for particles with fewer than four neighbors).

The resulting Lagrange function is then differentiated properly with respect to the coordinates. We do this symbolically using the computer algebra system Maple,¹² which allowed us to generate optimized “C code” for the resulting functions of the differential equation (see also below, “Optimizations by computer algebra techniques”).

Air resistance and external forces

Air resistance is calculated by

$$F_{\text{air}} = \frac{1}{2} \cdot \rho \cdot c_w \cdot A \cdot \mathbf{v}_i^2 \quad (8)$$

where ρ is the specific weight of air, c_w is the resistance coefficient, and A is the surface perpendicular to the velocity \mathbf{v}_i of particle i .

It is not easy to determine the exact value for c_w for each particle, since it depends on the location of all particles of the cloth. Therefore, we assume in our model the same value for c_w for the particles with four neighbors, since they are “inside” the cloth. At the border we have two cases. Particles with three neighbors are part of an edge and those with two neighbors create an edge (see Figure 2). Thus the air may flow around them. From aerodynamics we know that this “flowing around” an edge or corner causes strong turbulences on the backside of the cloth. Hence the friction, and the c_w is higher than in the middle of the cloth.

This model of air resistance is very simple and could be improved, but calculating an individual c_w for each particle is a difficult and expensive task. We have achieved good results using this model, which demonstrates interesting dynamical effects (see Figure 8).

Adding the computations for the air resistance to our model is not so expensive as you might expect. Since we have already calculated angles and distances in our Maple procedure for the energies, it is easy to calculate the area A perpendicular to the velocity \mathbf{v}_i . Therefore,

we calculate the air resistance force in Maple and add it to the second derivative of L .

In addition to air resistance, we can add arbitrary forces at this point to model blowing air, moving bodies, strokes on the cloth, and so on.

Interaction with environment

An important issue in modeling cloth is to determine its interaction or penetration with itself and its surroundings. In fact, you have only half the story without it. As described earlier, we calculate the trajectory of a particle via an integration of the Lagrange differential equation. To be more precise, we solve an *initial value problem*. The initial values—that is, the locations and velocities of the particles—are well defined before we start a simulation via a Runge-Kutta or other integration method.

Between two simulation steps, we detect penetration and adjust the locations and velocities—the new initial values—according to physics. At this point we can distinguish between interaction of the cloth with itself or with any other objects. This distinction is necessary because the way we calculate the reflected position is different if different materials are coming together.

Given the initial position \mathbf{p}_0 of a particle, we start an interpolation of the differential equation. The new position of the particle is then \mathbf{p}_1 . We send a ray R from \mathbf{p}_0 to \mathbf{p}_1 . If this ray intersects an obstacle at point P , we calculate and store the surface normal N at the point together with the material. A flag is set to indicate the detection of an intersection. This flag is used to modify the velocities and accelerations in the following calculations. We allow only velocities, with no direction component opposite to the normal N at P . This prevents further penetration. In addition we adjust the location of the particle to

$$\mathbf{p}_{\text{new}} = P + d_{\min} \cdot n \quad (9)$$

where $d_{\min} > 0$ is the minimal ray length of R .

To be more precise, accelerations and velocities, in case of penetration, are modified as follows: Assume we have a velocity $\mathbf{v}_{\text{old}} = \mathbf{v}_{\text{normal}} + \mathbf{v}_{\text{hor}}$, where $\mathbf{v}_{\text{normal}}$ is the component of \mathbf{v}_{old} parallel to the intersection normal I_n and \mathbf{v}_{hor} is the perpendicular component to I_n . Then we calculate

$$\mathbf{v}_{\text{new}} = \text{fric_coeff} \cdot \mathbf{v}_{\text{hor}} + \text{fric_coeff} \cdot \|\mathbf{v}_{\text{normal}}\| \cdot I_n \quad (10)$$

Both coefficients, fric_coeff and refl_coeff , depend strongly on material properties. In the case of interacting textiles, we can get some clues from Kawabata experiments on compression and surface.

Optimizations by computer algebra

We used the Maple computer algebra system¹² to compute the forces and thus the resulting accelerations defining particle movement in symbolic form. The symmetry of the particle grid (see Figure 4) lets us use Maple’s C-code generator to obtain the code for the acceleration, that is, the flow function of the differential

Table 1. Computational costs of simulation of forces for one particle.*

	Additions	Multiplications	Divisions	Functions	Subscripts	Assignments
Optimized	1,243	2,346	40	42	645	628
Nonoptimized	26,079	49,397	483	6,594	35,265	15

*These values are for a particle in a general position (that is, having four neighbors). The values are smaller for particles having two or three neighbors. The function calls are calls to square root, signum, and arccosine functions.

Table 2. Time steps taken in simulation (in milliseconds of simulated time).

Scenario	900 Particles	2,704 Particles
Adaptive Runge-Kutta Method:		
Table	0.27 – 5.8	0.22 – 2.1
Round table	0.18 – 25.0	0.13 – 6.0
Sphere	0.19 – 9.0	0.12 – 2.1
Bulirsch-Stoer Method:		
Table	0.018 – 0.92	0.00008 – 6.1

Table 3. CPU times in seconds of various function calls.*

Function	900 Particles	2,704 Particles
Accelerations	0.26	0.86
Adjusting bend- and shear-coefficients	0.13	0.43
Computing interactions:		
Table	0.18	0.44
Round table	0.84	2.20
Sphere	0.06	0.14

*Measured on a Silicon Graphics R8000 workstation with R8010 floating-point coprocessor.

Table 4. Computation times in minutes to simulate one frame of an animation (about 40 ms).*

Example	30 × 30 Grid 900 Particles			52 × 52 Grid 2,704 Particles		
	min.	max.	mean	min.	max.	mean
Table	0.3	7	5	7	67	21
Round table	3	8	6	10	94	24
Sphere	1	9	6	4	50	27

*The times were measured on a Silicon Graphics Indigo2 R8000 workstation with R8010 floating-point coprocessor.

equation of a single particle and its neighbors. The flow functions of all other particles are then obtained by substituting the appropriate indices for these particles.

This method of obtaining the code for calculating the differential equation's flow function has several advantages:

- The code development is supported by the rich library of a large computer algebra system; it can be related closely to mathematical formulas describing the physical laws. Also, the system performs all standard calculations like integration and partial derivatives.

■ The optimizer generating the C-code finds all occurring subexpressions and can store their values in auxiliary variables. This saves many computations (see Table 1). Therefore the system generates more efficient code than that generated by hand, and it contains fewer errors.

■ The development is modular but ultimately produces highly efficient code that could otherwise result only from an error-prone nonmodular approach.

Table 1 gives the computational costs of the C code generated by Maple for calculating the accelerations of one particle.

As Table 1 shows, the optimization improves the code efficiency considerably. For the most costly calls of floating-point functions, the optimization achieves an improvement factor of approximately 150!

Optimizations by numerical methods

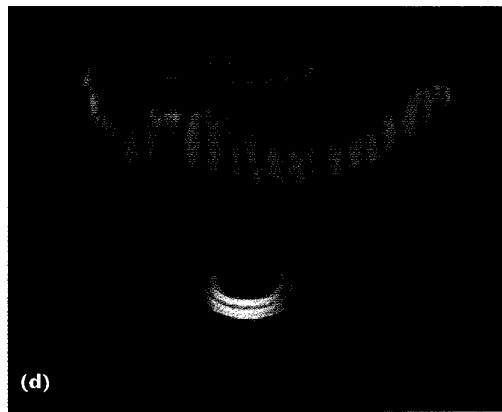
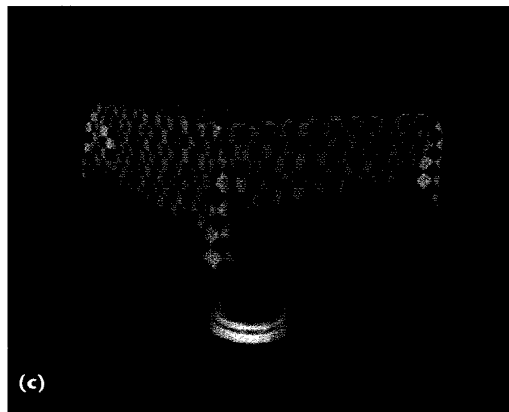
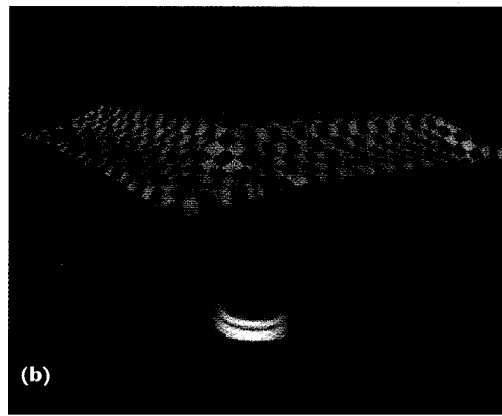
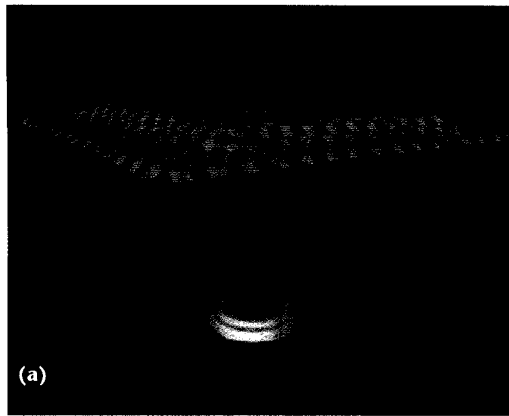
We have used a Runge-Kutta method with adaptive step-size control and the Bulirsch-Stoer method¹¹ as a numerical solver for the differential equations.

As Table 2 shows, the step-sizes taken by the adaptive methods vary considerably. Obviously the right choice of the solver has an immense impact on the time steps and thus reduces the number of necessary steps to simulate the behavior of cloth for a certain time.

We have found the Runge-Kutta method with adaptive step-size to be much faster than the Bulirsch-Stoer method in our examples. It may be possible to improve our system's performance even more by using one of the more specialized algorithms for the numerical integration of ordinary differential equations (see Press et al.¹¹).

Algorithm complexity and CPU requirements

The function calculating the derivatives is linear in the number of particles. Table 3 gives the CPU time for one call of the function on a Silicon Graphics Indigo2 R8000.



5 Sequence of a simulated falling tablecloth at (a) 0.16, (b) 0.28, and (c) 1.08 seconds. In (d), the tablecloth in (c) is shown with a different texture.

The function adjusting the bend and shear coefficient according to the hysteresis curve of a specific material is also linear in the number of particles. It is called less frequently than the function calculating the accelerations. Using the adaptive Runge-Kutta method, the number of calls to accelerations is about 6 to 10 times higher (which varies depending on the necessary corrections of the step sizes and is hard to predict, see below).

The complexity of the computation of interactions with the environment and possible penetrations of the cloth with itself is proportional to the product of the number of particles and the number of surfaces of the interacting environment. Although the theoretical complexity is not affected, the necessary computations can be reduced considerably by standard optimization techniques used in ray tracing (3D grids or hierarchical structures).

Because of the self-correcting nature of the adaptive step-size methods for solving an ordinary differential equation, the number of needed calls to the acceleration function and other auxiliary functions varies considerably. A theoretical prediction of them for a differential equation involving several thousand variables, as in our examples, is extremely difficult and well beyond the scope of this article.

Table 4 summarizes the needed computation times for several of our examples. The step sizes—that is, the time that could be simulated by one call to an adaptive step-size solver—varied considerably. Table 2 includes

results for some of our examples.

The function calculating the acceleration takes between 77 and 86 percent of the total computation time. The computation of the interactions takes about 6 to 15 percent of the computation time (depending on the example). The adjustment of the bend and shear coefficients takes about 6 to 8 percent. All other computations require less than 2 percent of the total CPU time.

Examples

Figure 5 shows a falling tablecloth. The simulation was performed on a Silicon Graphics Indigo2 R8000. We used a 52×52 particle grid and simulated a cotton polyester blend with different bend coefficients C_b for the weft and warp directions, taking the values from Breen, House, and Wozny.⁷ Output is an Inventor ASCII file rendered using the Davis-Software at the Wilhelm-Schickard-Institut für Informatik/Graphisch Interaktive Systeme (WSI/GRIS).¹³

As another example, we calculated a 30×30 and a 52×52 particle grid and draped the cloth over a round table. The resulting images, in Figures 6a and 6b respectively (next page), do not justify the higher particle number. Therefore, it might be possible to calculate an approximate location of the cloth with a low resolution and the final position with a refinement where necessary.

Yet another computation can be seen in Figure 7, where we have draped the cloth over a sphere.

Castel del Monte, a castle in the south of Italy, was

used to demonstrate the capabilities of the intersection part of the implementation. The cloth falls over a complete model of Castel del Monte, which consists of more than 4,000 faces and 120,000 vertices. Figure 8 shows the interesting effects of the cloth's vibrating parts in the middle of the castle.

Conclusion

The modeling system presented here computes the full trajectories of particles and not just the "final positions," as described in Breen, House, and Wozny.⁷ This offers several important advantages:

- Since the full "history" of each particle is known, hysteresis effects can be modeled accurately.
- The Kawabata experimental data for different textiles can be input directly to the model.
- The effects of external forces, especially those produced by wind or moving solid bodies, can be modeled accurately.

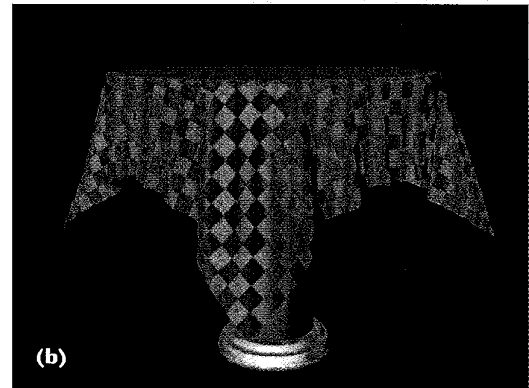
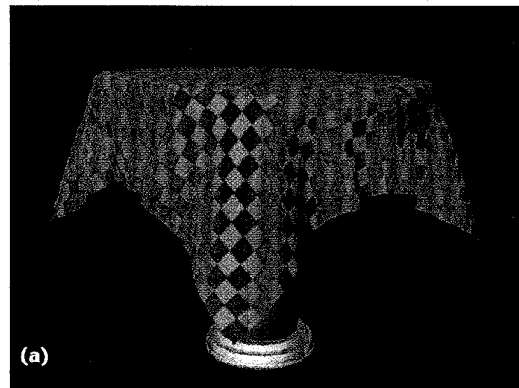
Despite this extra dimension of detail, our system computes "final positions" considerably faster than the times given in Breen, House, and Wozny.⁷

Our model can be easily extended to simulate the effects of manufacturing processes or interacting bodies. In particular, high stresses of the kind that occur in manufacturing can only be modeled if the full trajectory of each particle is known.

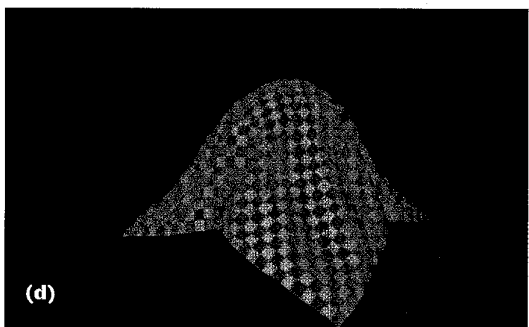
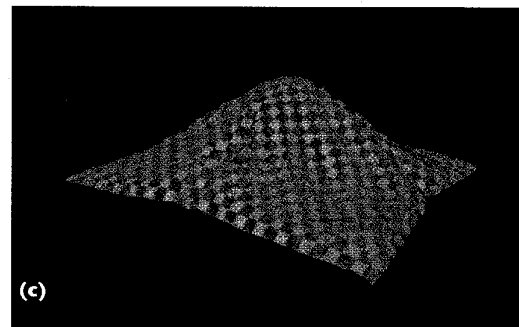
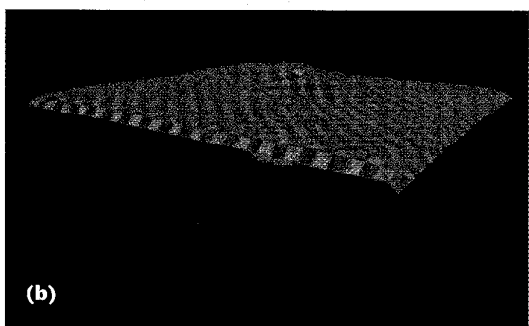
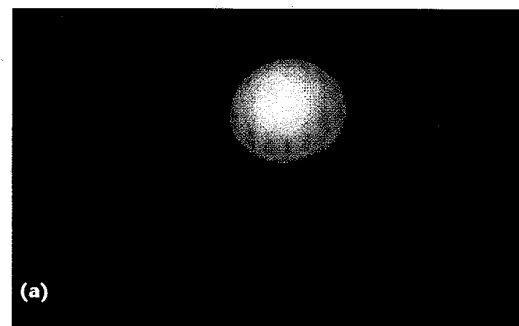
We have implemented our model as a C++ class library. As Breen, House, and Wozny argued, particle systems are more flexible than approaches using continuum mechanics.

Our system's fast computation times, mainly due to the numerical solution of ordinary differential equations, compare favorably to approaches using a finite-element method. Therefore, our approach might be an interesting alternative for other engineering problems currently solved by a finite-element method, for example, the computation of minimal surfaces, heavy membranes, vibrating membranes, and population dynamics.

6 (a) a 30×30 particle grid.
(b) a 52×52 particle grid.



7 A sphere and cloth.



Movies of the falling table cloths and the Castel del Monte examples described here are available on the World Wide Web at URL <http://www.gris.informatik.unituebingen.de/gris/proj/hc.html>. ■

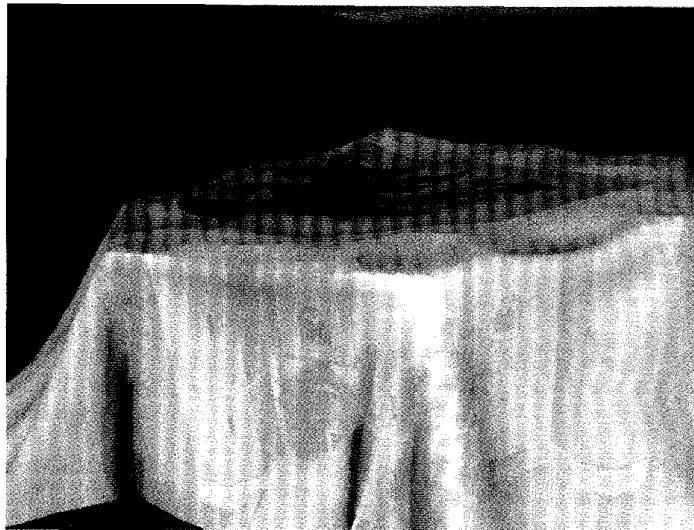
Acknowledgments

We would like to thank R. Sonntag for a version of his RadioLab illumination system, the focus of his 1996 doctoral dissertation at the University of Tübingen. We used this system to calculate collisions and intersections. RadioLab is a C++ class library for calculating photorealistic images using raytracing and radiosity.

Many thanks go to the anonymous reviewers for their very detailed comments and suggestions.

References

1. S. Kawabata, *The Standardization and Analysis of Hand Evaluation*, Textile Machinery Soc. of Japan, Osaka, 1980.
2. J. Weil, "The Synthesis of Cloth Objects," *Computer Graphics (Proc. Siggraph)*, Vol. 20, No. 4, Aug. 1986, pp. 49-54.
3. P. Volino, M. Courchese, and N. Magnenat-Thalmann, "Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects," *Computer Graphics (Proc. Siggraph)*, Aug. 1995, pp. 137-144.
4. J.R. Collier et al., "Drape Prediction by Means of Finite Element Analysis," *J. Textile Institute*, Vol. 82, No. 1991, pp. 96-107.
5. B. Chen and M. Govindaraj, "A Physically Based Model for Fabric Drape Using Flexible Shell Theory," *Textile Research J.*, Vol. 65, No. 6, June 1995, p. 324.
6. D. Terzopoulos and K. Fleischer, "Deformable Models," *Visual Computer*, Vol. 4, No. 6, Dec. 1988, pp. 306-331.
7. D.E. Breen, D.H. House, and M.J. Wozny, "A Particle-Based Model for Simulating the Draping Behavior of Woven Cloth," *Textile Research J.*, Vol. 64, No. 11, Nov. 1994, pp. 663-685.
8. C.W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," *Computer Graphics (Proc. Siggraph)*, Vol. 21, No. 4, July 1987, pp. 25-34.
9. H. Franke, ed., *Lexikon der Physik*, Franck'sche Verlagshandlung, Stuttgart, Germany, 1969.
10. V.I. Arnold, *Mathematical Methods of Classical Mechanics*, 2nd edition, Springer-Verlag, New York, 1989.
11. W.H. Press et al., *Numerical Recipes in C: The Art of Scientific Computing*, 2nd edition, Cambridge Univ. Press, New York, 1992.
12. B.W. Char et al., *Maple V Language Reference Manual*, Springer-Verlag, New York, 1991.
13. A. Groene, *Entwurf Eines Objektorientierten Visualisierungssystems auf der Basis von Raytracing*, doctoral dissertation, Universität Tübingen, 1996.



8 Wrapping
Castel del
Monte.



Bernhard Eberhardt studied math at the University of Massachusetts at Amherst (MSc 1988) and at the University of Tübingen (Diplom 1991). He received a PhD in math in 1994 at the University of Tübingen. In 1994 he joined the Wilhelm Schickard Institut für Informatik/Graphisch Interaktive Systeme (WSI/GRIS), where he works on textile modeling in a joint project between the University of Tübingen and industry. His research interests include computer animation, physically based modeling, parallelization, and numerics.



Andreas Weber studied math at the University of Tübingen, Germany, and the University of Colorado in Boulder. He received an MS (Diplom) in math in 1990 from the University of Tübingen and a PhD in computer science in 1993. His main area of research is computer algebra and its applications. Currently, he is working as a postdoctoral fellow at the Computer Science Department of Cornell University with a scholarship of the German Science Foundation (DFG).



Wolfgang Strasser is full professor of computer science and adjunct professor of math at the University of Tübingen. His graphics group (WSI/GRIS) consists of about 20 researchers working in graphics system design, graphics hardware, visualization, rendering, and geometric modeling. He is a member of IEEE, Eurographics, and the Gesellschaft für Informatik.

Readers may contact the authors at the University of Tübingen, WSI/GRIS, Auf der Morgenstelle 10/C9, D-72076 Tübingen, Germany.