Discrete Shells

Eitan Grinspun
Caltech

Anil N. Hirani
Caltech

Mathieu Desbrun
USC

Peter Schröder
Caltech

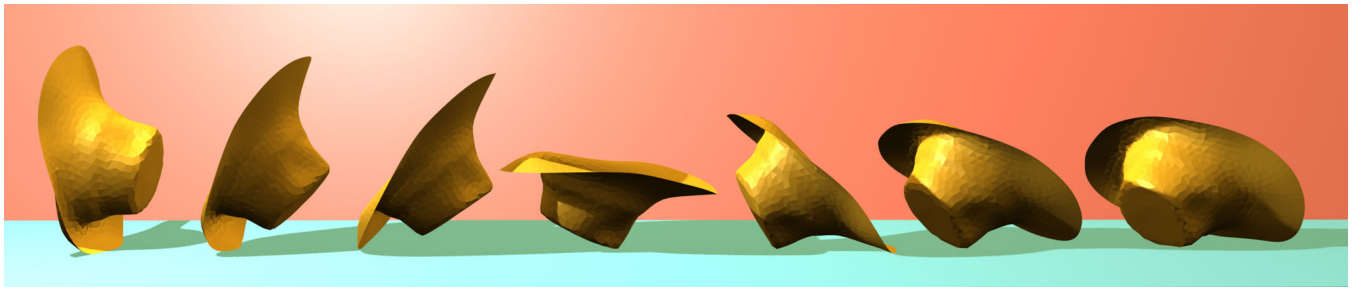


Figure 1: *Composite of 7 frames from a simulation with our thin-shell simulator as a hat is hitting the floor and tumbling to the right.*

Abstract

In this paper we introduce a *discrete shell* model describing the behavior of thin flexible structures whose rest configuration is non-flat. Previously such models required complex continuum mechanics formulations and correspondingly expensive algorithms. We show that a straightforward shell model can be derived in the discrete setting of triangle meshes and implemented through a simple modification to standard cloth simulation algorithms. The resulting technique convincingly simulates a variety of thin shell models ranging from cloth to thin metal like materials. We show the importance of non-flat rest configurations with a number of examples and demonstrate the quality of our results by comparing a simulation of a falling hat with real video footage.

1 Introduction

The pioneering work of Baraff and Witkin [1998] on cloth modeling has greatly impacted the realism of many physical modeling techniques used in computer graphics. Previous animation techniques suffered from restrictive time stepping criteria, remedied mostly through exaggerated damping or overly "springy" materials. Much progress has occurred since Baraff and Witkin demonstrated the pay-off of implicit integration for visual simulation: examples include deformable bodies [James and Pai 1999; Debunne et al. 2001], fluids [Stam 1999; Foster and Fedkiw 2001; Fedkiw et al. 2001; Enright et al. 2002], and cloth [Baraff and Witkin 1998; Choi and Ko 2002], in each case bringing us closer to realistic dynamics.

An important class of objects has not received much attention until recently: *shells*. The term shell (or thin shell) refers to thin flexible structures with a high ratio of width to thickness (> 100). They are distinguished from the more well known *thin plate*. Plates have a flat rest position, while shells have a curved rest position. Cloth, for instance, typically has a flat rest position. All other thin-walled objects which are either naturally curved, or put into that shape through plastic deformation fall into the shell category. These are all around us: hats, cans, carton boxes, pans, car bodies, detergent bottles, *etc.*

Thin shell materials are difficult to simulate. Because of their relative degeneracy in one dimension (their "thinness") they do not admit to straightforward tessellation and treatment as a three-dimensional solid. The numerics of such approaches become catastrophically ill-conditioned destroying convergence or, worse, converging to entirely erroneous configurations. Cloth, and its relatively much more successful numerical treatment, is closely related though. The main difference is that it is a plate: its reference configuration ("rest shape") is flat. Thus it cannot account for the *structural rigidity* that arises from curvature effects.

Full blown numerical treatments which can capture these effects and are significantly more robust than classical approaches are now available in graphics [Cirak et al. 2000; Green et al. 2002; Grinspun et al. 2002]. These use the Kirchoff-Love model as the constitutive equation of shells. The second-order nature of such a continuum-based energy takes into account curvature effects in curved coordinate frames, *i.e.*, with respect to a deformed rest configuration. They can thus model a rich variety of materials. However, implementation of the appropriate material models is still quite complex and leads to long run times unless one returns to excessive damping or overly "squishy" materials. Therefore, shells made of stiff materials remain challenging and costly to simulate.

Contribution We revisit the existing models of shells in order to provide a simple, fast, and realistic technique for simulating shells in graphics. In particular, we show that a *discrete model* of shells leads to all the essential behaviors captured by more complex models, but with a surprisingly simple implementation: *a small change in the existing cloth animation algorithms allows for shell simulation at almost no extra computational cost*. We demonstrate the realism of our approach through different examples and, in particular, through comparisons with real world footage (see Figure 2).

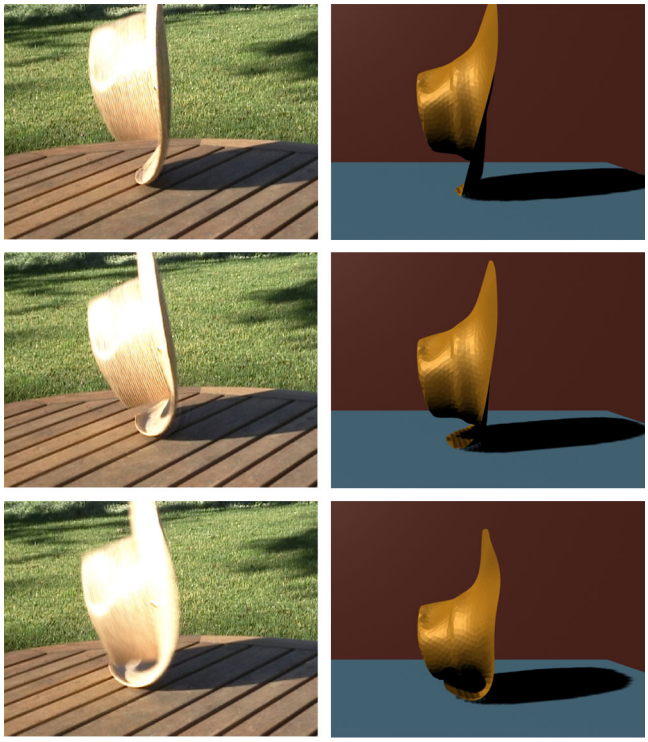


Figure 2: *Real footage vs. Simulation: left, a real hat is dropped on a table; right, our shell simulation captures the bending of the brim appropriately. Notice that traditional volumetric or cloth simulation could not provide this behavior.*

2 A Discrete Shell Model

We model thin surface-like objects governed by non-linear *membrane* and *flexural* energies. The energies measure differences between the undeformed (rest) configuration $\bar{\Omega}$ and deformed (current) configuration Ω. We take measurements which are invariant to rigid body motion of the undeformed and/or deformed configurations: this ensures conservation of linear and angular momentum. Furthermore our measurements are not explicit functions of time, ensuring conservation of total energy.

We use an arbitrary 2-manifold triangle mesh to describe the shell geometry, and denote a mesh edge (resp. face) with the letter e (resp. f). Let $\varphi : \bar{\Omega} \mapsto \Omega$ be the piecewise-affine deformation map from every vertex (resp. edge, face) of the undeformed mesh to the corresponding vertex (resp. edge, face) of the deformed mesh.

Membrane Consider a one dimensional spring defined by two endpoints joined by a straight segment. Assuming uniform strain and Hooke's Law (stress proportional to strain), the elastic energy is proportional to the normalized change in length $(\|e\| - \|\bar{e}\|)^2(1/\|\bar{e}\|)$, where $\|\bar{e}\|$ and $\|e\| = \|\varphi(\bar{e})\|$ are the lengths measured in the undeformed and deformed configurations respectively. This corresponds to the commonly accepted model for the energy of a discrete spring, resisting change in length. Notice that the elastic energy of a series of springs is simply the sum of each individual energy.

In two dimensions, an elastic surface resists change in area *and* shearing. Baraff and Witkin [1998] propose a discrete model to simulate this behavior by summing stretch and shear constraint-based energies of every triangle, extending the 1D case. Other discrete membrane models can also be constructed. For instance, we can extend the 1D spring model to 2D in the following natural way: the energy stored by mesh edges will be defined as $W_L(\mathbf{x}) = \frac{1}{2}\sum_{\bar{e}}(\|e\| - \|\bar{e}\|)^2/\|\bar{e}\|$, and the energy stored by mesh faces will be $W_A(\mathbf{x}) = \frac{1}{2}\sum_{\bar{f}}(\|f\| - \|\bar{f}\|)^2(1/\|\bar{f}\|)$, where $\|\bar{f}\|$ and $\|f\|$ are the areas of corresponding faces in the undeformed and deformed configurations respectively. This energy measures local changes to both length and area, and the associated proportionality constants determine the material resistance to stretching and area change. Another model commonly used for a membrane is a grid of springs with additional cross springs to, again, oppose shearing and change in area. We do not aim to evaluate the physical accuracy of these different models: for high membrane stiffnesses they all behave reasonable well.

So far we have only discussed energies that measure in-plane (*intrinsic* or *membrane*) deformations. However, when a surface bends (an *extrinsic* deformation), *flexural energy* comes into play.

Flexure We express flexural energy as a function of curvature, guided by Ciarlet [2000]:

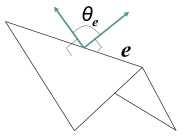
> *The stored energy function of a nonlinearly elastic flexural shell is thus remarkably simple: it is a quadratic and positive definite expression in terms of the exact difference between the curvature tensor of the deformed middle surface and that of the undeformed one ...*

A simple expression motivated by the above desiderata is the area-integrated, squared difference of mean curvature. Informally this can be derived from

$$[\mathrm{Tr}(\varphi^*\mathrm{S}) - \mathrm{Tr}(\bar{\mathrm{S}})]^2 = (2H - 2\bar{H})^2 \,, \tag{1}$$

where $\bar{\mathrm{S}}$ and S are the shape operators [Gray 1998] evaluated over the undeformed and deformed configurations respectively; likewise $\bar{H}$ and H are the mean curvatures; $\varphi^*\mathrm{S}$ is the pull-back of S onto $\bar{\Omega}$, and we use commutativity of the trace with the pull-back.

To use this formulation we must discretize the area integral $\int_{\bar{\Omega}}(2H - 2\bar{H})^2 d\bar{A}$ over the reference domain $\bar{\Omega}$. Fortunately, since our surface model is piecewise linear, the expression for the discretized energy is simple. Consider two adjacent triangles. At every point on the shared edge the principal curvatures are equal to zero (along the edge) respectively the complement of dihedral angle θ_e between incident surface normals (across the edge). Away from the edge the curvature vanishes everywhere. For any particular mesh edge we have pointwise $(2H - 2\bar{H})^2 = (\theta_e - \bar{\theta}_E)^2$; integration over the surface becomes a summation over edges giving the *discrete flexural energy*:



$$W_B(\mathbf{x}) = \sum_{\bar{e}}(\theta_e - \bar{\theta}_e)^2\|\bar{e}\|, \tag{2}$$

where θ_e and $\bar{\theta}_e$ are corresponding complements of the dihedral angle of edge e measured in the deformed and undeformed configuration respectively.

Following the reasoning for Eq. 1, we could have formed a second energy term taking the determinant instead of the trace of S. This would lead to a difference of Gaussian curvatures, but this is always zero under isometric deformations (pure bending). This is not surprising, as Gaussian curvature is an *intrinsic* quantity, whereas flexural energy measures *extrinsic* deformations [Gray 1998].

Dynamics A simple, physically-motivated shell model can therefore be expressed by the following sum of membrane and flexural energies:

$$W = K_L W_L + K_A W_A + K_B W_B \tag{3}$$

where K_L and K_A are membrane stiffnesses related to Lamé constants, while K_B is the bending or flexural stiffness. Tuning

these parameters allows us to go from elastic, rubber-like sheets to aluminium-like shells. Our dynamic system is governed by the ordinary differential equation of motion $\ddot{\mathbf{x}} = -\mathbf{M}^{-1} \nabla W(\mathbf{x})$ where $\mathbf{x}$ is the vector of unknown DOFs (*i.e.*, the vertices of the current geometry) and $\mathbf{M}$ is the mass matrix. We use the conventional simplifying hypothesis that the mass distribution is lumped at vertices: the matrix M is then diagonal, and the mass assigned to a vertex is the area of its Voronoi region in the rest state times the mass density of our material (the Voronoi area is computed as per [Meyer et al. 2002]).

Discussion Our proposed discrete flexural energy (Eq. 2) generalizes on published energies for *flat rest-state* plates both continuous and discrete: (1) Ge *et al.* presented a geometric argument that the stored energy of a continuous inextensible plate has the form $\int_{\bar{\Omega}} c_H H^2 + c_K K dA$ for material-specific coefficients c_H and c_K [1996]; (2) Haumann [1987] used a discrete hinge energy, similarly Baraff and Witkin [1998] used a discrete constraint-based energy, of the form $W_B(\mathbf{x}) = \sum_{\bar{e}} (\theta_e)^2$. Our approach generalizes both (1) and (2); this is verified by substituting $\bar{H} = 0$ (resp. $\bar{\theta} = 0$) in Eq. 1 (resp. Eq. 2). This simple generalization produces convincing simulations far beyond the domain of thin plate models (see section 4).

Our novel formulation has three salient features: (1) the flexural energy is invariant under geometry-preserving refinement of mesh connectivity: our energy is therefore dependent on geometry, *not* on parameterization; (2) the energy is invariant to rigid body transformation of both the undeformed and the deformed shape: our system conserves linear and angular momenta; (3) the piecewise nature of our geometry description is fully captured by the edge-*length* preservation term, the triangle-*area* preservation term, and the (purely extrinsic) dihedral-*angle* preservation term.

Many materials dissipate energy via flexural oscillations. To that end we complete our model with an *optional* damping force proportional to $(\dot{\theta} - \dot{\bar{\theta}})\nabla\theta$ where $\dot{\bar{\theta}} = 0$ for elastic deformations but is in general non-zero for elastoplastic deformations. This is consistent with standard derivations of "Rayleigh-type" damping forces using the strain rate tensor (see for example [Baraff and Witkin 1998]).

3 Implementation

Newmark Time Stepping We adopt the Newmark scheme for ODE integration; West *et al.* demonstrate the numerical advantages of this scheme [2000]. The Newmark scheme can be used as either an explicit or implicit integrator by adjusting the parameter β; we used $\beta = 1/4$ (implicit integration) during production, and $\beta = 0$ (explicit integration) to aid in debugging. Newmark gives control over numerical damping via its second parameter Γ. We obtained the best results by minimizing numerical damping ($\Gamma = 1/2$); this is consistent with Baraff and Witkin's observation that numerical damping causes undesirable effects to rigid body motions. Note however that our flexural energy can be used with other ODE integrators such as the implicit scheme described in [Baraff and Witkin 1998].

Automatic Differentiation The use of an explicit integrator necessitates the evaluation of energy gradients, or forces, with respect to vertex DOFs. Formulae for the gradients of edge-length and area easily found in the literature [Desbrun et al. 2002]; the gradient of the dihedral angle requires more work, but can still be derived by hand. Alternatively they may be found automatically as described below.

The use of an *implicit* integrator necessitates evaluation of *force* gradients with respect to vertex DOFs (*i.e.*, we need formulae for *second* derivatives of energy). Deriving such formulae is cumbersome and error-prone. Consequently we implemented an automatic differentiation (AD) code. AD is a technique for augmenting software with derivative computations (for earlier uses of AD in graphics see [Gleicher 1994; Kass 1992]). The technique is based on the observation that every computational algorithm can be written as the composition of simple–and easily differentiable–steps to which the chain rule can be applied.

The salient features of our implementation are: (1) it differentiates directly with respect to vector (not just scalar) unknowns; (2) it harnesses C++ type-checking to ensure efficiency and completeness of differentiation. We define two classes, Scalar and Vector, to represent *independent* scalar and vector values respectively. The related classes DScalar and DVector represent *dependent* quantities; these carry a tuple *(scalar value, vector-valued derivative)* and *(vector value, matrix-valued derivative)* respectively. All the standard algebraic operators are overloaded to inter-operate between the classes, with a special restriction on assignment: dependent quantities may not be assigned to independent variables, and vice-versa. This condition ensures correctness (no dependent quantity is overlooked) and ensures efficiency (no independent or constant quantity carries needless derivative computation). Although there are several good AD libraries publicly available (*e.g.*, [ADIFOR 2.0 2002]), we opted for implementing this small, efficient set of classes particularly tailored to differentiation with respect to vector variables. Our AD classes will be made available online.

Damping We approximate the $\dot{\theta}_e$ terms through the usual forward finite difference: $(\theta_e^{t+dt} - \theta_e^t)/dt$. The damping is therefore automatically coherent to the motion happening *during* the simulation time step, and not simply to the initial derivative value.

4 Results

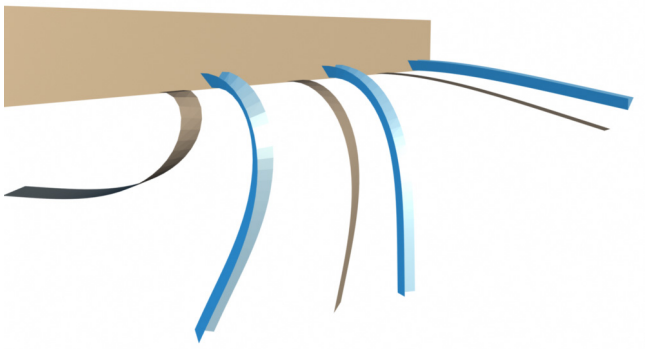


Figure 3: *Three pairs of flat- and v-beams with increasing flexural stiffnesses (left to right) of 100, 1000, and 10000. The flexural energy coefficient has a high dynamic range; extreme values (from pure-membrane to near-rigid) remain numerically and physically well-behaved. Observe that increasing flexural stiffness gives increasing structural rigidity. Compare the behavior of beams with flat- and v-shaped cross sections: the non-flat cross-section of the v-beam contributes to structural rigidity, especially for low flexural stiffeness.*

Beams We computed a series of proof-of-concept tests and full-blown simulations. Figure 3 demonstrates how varying flexural stiffness affects the behavior of thin plates (flat beams) and shells (v-beams). Observe the increasing structural rigidity (from left to right) as the flexural stiffness of the material is increased. The non-flat rest-state of a v-beam gives qualitatively and quantitatively different behavior than a flat beam (see also the accompanying movies). The reader is invited to experience similar behavior on a simple paper strip, before and after folding a v-shaped cross-section. By varying the undeformed dihedral angles and the flexural

rigidity, our technique can seamlessly simulate materials ranging from cloth to aluminium car bodies.

Elastic hats We also use a hat model to provide a comparison with real footage. As can be seen from the sequence of pictures in Figure 2, the behavior is qualitatively the same: we observe a similar behavior during the impact with the ground where the brim gets significantly bent before springing back and reshaping; the same type of vibrations of the hat brim after impact are also visible.

Figure 4: *Modeling a curled, creased, and pinned sheet of paper: by altering dihedral angles of the reference configuration, we effect plastic deformation. While the rendering is texture-mapped we kept flat-shaded triangles to show the underlying mesh structure.*

Plastoelastic paper As proof of concept for plastoelastic applications, we demonstrate the modelling of a curled, creased, and pinned sheet of paper using plastic as well as elastic deformations (see Figure 4). We begin with a flat rectangular surface, and effect *plastic* deformations by modifying the angles $\bar{\theta}_e$ of the *undeformed* configuration. To demonstrate plastic deformations we modeled the characterisic curling of rolled paper and the creases of bent paper. The paper is simultaneously elastically deformed by pin constraints on the *deformed* configuration.

5 Conclusions

We have presented a model of shells for computer animation. A simple discrete bending energy was presented, extending the plate case to arbitrarily curved thin objects. An implicit integration model along with an automatic force derivation is used to alleviate strong time step limitations, resulting in fast computations. This straightforward model captures the expected behavior of shells such as flexural rigidity, crumpling, ... In fact, the results offer a realism and a visual complexity comparable to sophisticated shell models requiring second-order finite element computations. The most compelling feature of our model is its simplicity of implementation: a cloth animation algorithm only needs ten additional lines to become a full-blown shell simulator.

This work was supported in part by NSF (DMS-0220905, DMS-0138458, DMS-0221666, DMS-0221669, CCR-0133983, EEC-9529152, ACI-0219979) the DOE (W-7405-ENG-48/B341492), nVidia, the Center for Integrated Multiscale Modeling and Simulation, Intel, Alias|Wavefront, Pixar, Microsoft, and the Packard Foundation. We thank Jerry Marsden and Tom Duchamp for insightful discussions, and our colleagues in the multi-res group for their support. We are indebted to Steven Schkolne for his production assistance.

References

ADIFOR 2.0. 2002. http://www-unix.mcs.anl.gov/autodiff/ADIFOR/. In *Argonne National Laboratory / Rice University*.

BARAFF, D., AND WITKIN, A. 1998. "Large Steps in Cloth Simulation". In *Proceedings of SIGGRAPH*, 43–54.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but Responsive Cloth. *ACM Transactions on Graphics 21*, 3 (July), 604–611.

CIARLET, P. 2000. *Mathematical Elasticity. Vol. III*, vol. 29 of *Studies in Mathematics and its Applications*. Amsterdam. Theory of shells.

CIRAK, F., ORTIZ, M., AND SCHRÖDER, P. 2000. Subdivision Surfaces: A New Paradigm for Thin-Shell Finite-Element Analysis. *Internat. J. Numer. Methods Engrg. 47*, 12, 2039–2072.

DEBUNNE, G., DESBRUN, M., CANI, M.-P., AND BARR, A. H. 2001. "Dynamic Real-Time Deformations Using Space & Time Adaptive Sampling". *Proceedings of SIGGRAPH 2001*, 31–36.

DESBRUN, M., MEYER, M., AND ALLIEZ, P. 2002. Intrinsic Parameterizations of Surface Meshes. In *Proceedings of Eurographics*, 209–218.

ENRIGHT, D. P., MARSCHNER, S. R., AND FEDKIW, R. P. 2002. Animation and Rendering of Complex Water Surfaces. *ACM Transactions on Graphics 21*, 3 (July), 736–744.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual Simulation of Smoke. In *Proceedings of ACM SIGGRAPH 2001*, 15–22.

FOSTER, N., AND FEDKIW, R. 2001. "Practical Animation of Liquids". *Proceedings of SIGGRAPH 2001*, 23–30.

GE, Z., KRUSE, H. P., AND MARSDEN, J. E. 1996. The Limits of Hamiltonian Structures in Three-Dimensional Elasticity, Shells, and Rods. *Journal of Nonlinear Science 6*, 19–57.

GLEICHER, M. 1994. *A Differential Approach to Graphical Manipulation (Chapter 5)*. PhD thesis.

GRAY, A., Ed. 1998. *Modern Differential Geometry of Curves and Surfaces*. Second edition. CRC Press.

GREEN, S., TURKIYYAH, G., AND STORTI, D. 2002. "Subdivision-Based Multilevel Methods for Large Scale Engineering Simulation of Thin Shells". In *Proceedings of ACM Solid Modeling*.

GRINSPUN, E., KRYSL, P., AND SCHRÖDER, P. 2002. CHARMS: A Simple Framework for Adaptive Simulation. *ACM Transactions on Graphics 21*, 3 (July), 281–290.

HAUMANN, R. 1987. Modeling the Physical Behavior of Flexible Objects. In *Topics in Physically-based Modeling*, Eds. Barr, Barrel, Haumann, Kass, Platt, Terzopoulos, and Witkin, *SIGGRAPH Course Notes*.

JAMES, D. L., AND PAI, D. K. 1999. ArtDefo - Accurate Real Time Deformable Objects. In *Proceedings of SIGGRAPH*, 65–72.

KASS, M. 1992. CONDOR: Constraint-based Dataflow. In *Proceedings of SIGGRAPH*, 321–330.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H., 2002. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. Proceedings of VisMath.

STAM, J. 1999. Stable Fluids. In *Proceedings of SIGGRAPH*, 121–128.

WEST, M., KANE, C., MARSDEN, J. E., AND ORTIZ, M. 2000. Variational integrators, the Newmark scheme, and dissipative systems. In *International Conference on Differential Equations 1999*, World Scientific, Berlin, 1009 – 1011.